Geometric View of GAN and Visualization

Anonymous Submission
Paper ID: 1155070507

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 toy experiments.

1 Introduction

Generative Adversarial Networks (GAN) [7] as generative models have been actively studied and developed [2–6, 9, 10, 14–17, 19, 20, 24–27] in the last few years. There are theoretical discussions [2, 16, 25], various extensions [3, 4, 6, 9, 14, 15, 20], exploring effective network design and training methods [1, 19], and applications in image generation [5, 17, 24], manipulation [18, 26], and cross domain transfer [10, 27], and many others. Compared to other generative models, like Restricted Boltzmann Machine [8], Variational Auto-Encoders [11], etc., GAN is reported to be able to generate higher quality examples. For example, in applications in super-resolution [12], the generated images are visually more sharp with adversarial losses. This is one of the advantages of the GAN formulation. However, in general, GAN also faces some well known problems: hard to train, mode collapse, lack of systematic evaluations methods, to name a few.

GAN is formulated as a two-player game that involves a generator and a discriminator. 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. This is a dynamic process. In each training cycle, both the generator and the discriminator have to adjust themselves to cope with the changes in their opponents. In 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 distribution. Formally, GAN is formulated [7] as the following problem,

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))], \tag{1}$$

where G is the generator neural network, D is the discriminator neural network, x is a sample from data distribution, and z is a sample from noise distribution. If the discriminator is certain that a sample x comes from data distribution, then the output should be 1. If it is certain that a fake sample comes, the output of D is 0. In the equilibrium state, when the discriminator D cannot decide where the sample comes from, the output is $1/2$. It can be shown that if in each step, the discriminator is at its optimum, then the objective of minimizing generator G in Equation (1) is equivalent to minimizing the Jensen-Shannon divergence between the real data distribution and the generated distribution. Note that the theoretical justification of GAN only considers an idealized setting, in which the discriminator is able to achieve the optimum, and the capacities of networks are unlimited. These assumptions never hold in real experiments. So in contrast to the utopian world that the theory demonstrates, neither are we guaranteed that the perfect distribution can be generated in the end, nor can we be confident to say that the algorithm would converge.

Under review in the final project for ELEG5491 [22]. Do not distribute.

Although the original formulation of GAN adopts a game theory and probabilistic point of view, it is also natural to interpret the training dynamics of GAN in the context of manifold learning. In most real applications, we are dealing with structured data, such as natural images. The distribution of such data may lie on a low dimensional manifold instead of distributing all over the space. 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] use this point of view to explain why GANs are so hard to train in practice. The core idea in their argument is that, since the dimension of the latent space and the dimension of the data manifold are usually smaller than the data space, we are trying to matching two low dimensional manifolds in a high dimensional ambient space. The chance of two such manifolds have sufficient overlap is zero. In this case, the Jensen Shannon divergence do not even have a valid definition. This explains why training GAN is so difficult.

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, such as dimensionality, connectivity, and the topology. All those aspects lack sufficient discussion in current literature. In this report, we study the effect of these geometric properties on the training of GAN.

1.1 Related Work

The geometric viewpoint appears in many works on GAN [1–3, 25–27]. As discussed above, the geometric view can be used to explain the difficulties encountered in training GAN [1]. WGAN [2] exploits the good behavior of Wasserstein distance in measuring two distributions concentrated on low-dimensional manifolds. In [23], the author try to convince us that a spherical latent space is better than a cube. And we should notice the topological difference of these two latent space structures. Mode regularized GAN [3] introduces mode regularizer and manifold-diffusion training based on the geometric intuition. The geometric viewpoint is also implicitly referred in many other works. Except for a few papers that pursue a mathematically rigorous style, most of them use terminologies such as manifold, in a casual way. Nevertheless, the geometric viewpoint gives a good intuition for us to understand what GAN is doing and facilitates inspirations for new structures and algorithms.

2 Geometric View of GAN

A *manifold* (without boundary) is a set of points that locally resembles Euclidean space near each point. Specifically, near 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; *e.g.*, a *point* in a line, a *circle* in a plane, and a *ball* in the 3-dimensional space that we live in.

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

$$\varphi : \Omega \to M \subset \mathbb{R}^N, \tag{2}$$

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 generate samples in the right probability. So, we can decouple the generation task as two subtasks: generating the right geometry, and the right distribution. Following this view, we can first aim to generate the correct data manifold without worrying about the distribution. For example, we can exploit various sampling tricks to aid this geometry learning process. Once we can generate the right geometry, we may fix the generator and attach another network before the latent space to learn the right distribution. The latter task would be easier since the dimension of latent space is lower than the dimension of the space that data live in.

Inverse Mapping The coordinate mapping φ is a homeomorphism, and thus a bijection. The necessary and sufficient condition for φ being a bijection is that there exists a mapping ψ,

$$\psi : M \to \Omega \subset \mathbb{R}^n, \tag{3}$$

such that,

$$\psi \circ \varphi = \mathrm{id}_\Omega \quad (\varphi \text{ is injective}), \quad \varphi \circ \psi = \mathrm{id}_\mathcal{M} \quad (\varphi \text{ is surjective}). \tag{4}$$

If we impose the bijection as an extra regularization for GAN, then the mode collapse problem might be mitigated. In fact, many existing works incorporate the idea of inverse mapping into their formulation. Some works [3, 9, 10, 18] view this inverse mapping as an encoder using the variational auto-encoder language. Some works [26, 27] also motivate this from geometric intuition.

Learning Mapping as Graph In conditional GAN, we want to control the generated sample x through some condition c; *i.e.*, we want to learn 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.

These observations lead to some useful insights into the problem of GAN. As we have seen, the inverse mapping and learning mapping as graph have been extensively studied recently, although their starting point might be quite different.

In this report, we focus on studying the impact of the geometric and topological properties of the latent space and the data manifold on the training of GAN. Specifically, we are interested in the following three properties:

- *Dimensionality*. We know that two manifolds would not match if their dimensions are different. However, in real applications, the dimension of data manifold is unknown. So it is interesting to see what would happen to GAN if the dimension of latent space and data manifold do not match.
- *Connectivity*. Natural images are clustered into distinct classes naturally. This means that the data manifold may have several components that are disconnected with each other. The question is, do we have to require the latent space to be disconnected at the same time? What if the latent space is connected while the data distribution is connected?

3 Experiments and Visualization

To answer the questions raised in previous section, we specially designed toy problems in low-dimensions, so that we can visualize what happens in the training dynamics of GAN. We use *fully connected* layers as the building blocks for both the generator and discriminator. After each fully connected layer, we add a *batch normalization* layer and a *Elu* activation layer. Both the latent space and the data space is restricted to 1-dimension or 2-dimension. Across all experiments, we use fc networks with $20-40-100-200-200-100-40-20$ hidden neuron numbers as the structure for generator. The same hidden neuron setting is also applied to the discriminator. The capacity of these networks is large enough for our experiments. We use *rmsprop* with initial lr 0.001 as the optimization method. learning rate is multiplied by 0.98 after every 300 iterations. The batch size is set to 256, and we update generator and discriminator alternately, with each processes 1 batch of data.

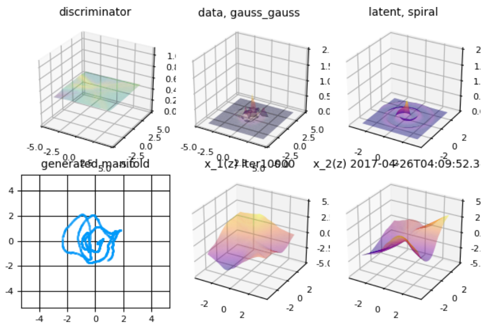
We use *Parrots* [21] as the deep learning framework and use the the Julia port *Parrots.jl* [13] as the working language. All codes were implemented from scratch based on the algorithm described in [7]. Codes and experiment results are available at `https://github.com/innerlee/ELEG5491`.

3.1 Dimensionality

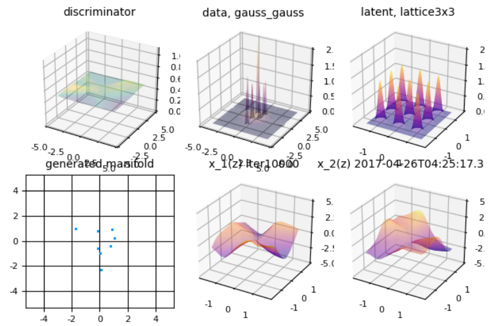
We use three manifolds: 1) A 3×3 lattice. It is 0-dimensional manifold in $\mathbb{R}^2$. 2) A 1-dimensional spiral curve in $\mathbb{R}^2$. 3) the whole 2-dimensional plane with Gaussian distribution. As shown in Figure 1, we can see that 1) If the dimension of latent manifold is lower than that of the data manifold (Figure 1 (a, b)), the generated low dimensional manifold try to cover the larger dimensional data manifold as possible. 2) 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.

3.2 Connectivity

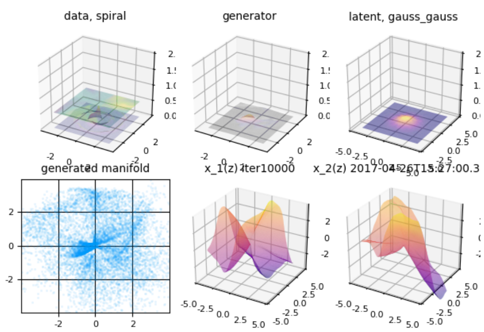
We have two settings: 1) A two dimensional square with uniform distribution. It is a connected manifold. 2) A disconnected manifold that consists 9 small squares as components. From Figure 2,



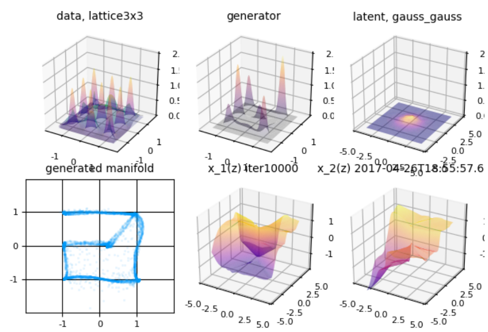
(a) z: 1-dim, spiral. x: 2-dim, Gaussian.



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

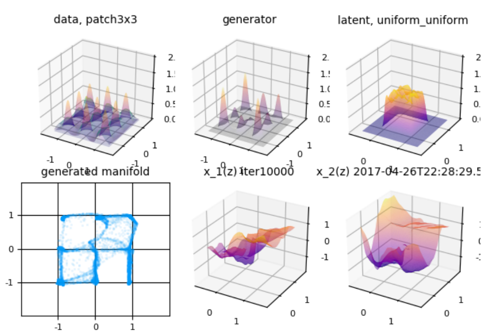


(c) z: 2-dim, Gaussian. x: 1-dim, spiral.

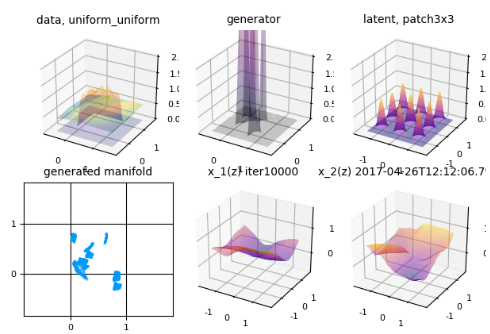


(d) z: 2-dim, Gaussian. x: 0-dim, lattice.

Figure 1: GAN in different dimensions. The snapshots are taken at iteration 10000.



(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.

Figure 2: GAN in connected/disconnected manifolds. The snapshots are taken at iteration 10000.

we can observe that, 1) The connected latent space is able to approximate the disconnected data manifold (Figure 2 (a)). 2) The disconnected latent space has difficulty in covering the data manifold although the dimensions are the same (Figure 2 (b)). This may be caused by the continuity in the learned mapping of the generator.

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*, volume 2016, 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] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[12] 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.

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

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

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

[16] 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.

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

[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] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[20] 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.

[21] P. Team. Parrots deep learning platform. http://www.parrotsdnn.org/. Accessed: 2017.

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

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

[24] 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.

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

[26] 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.

[27] 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.