Generative Adversarial Networks: A Review

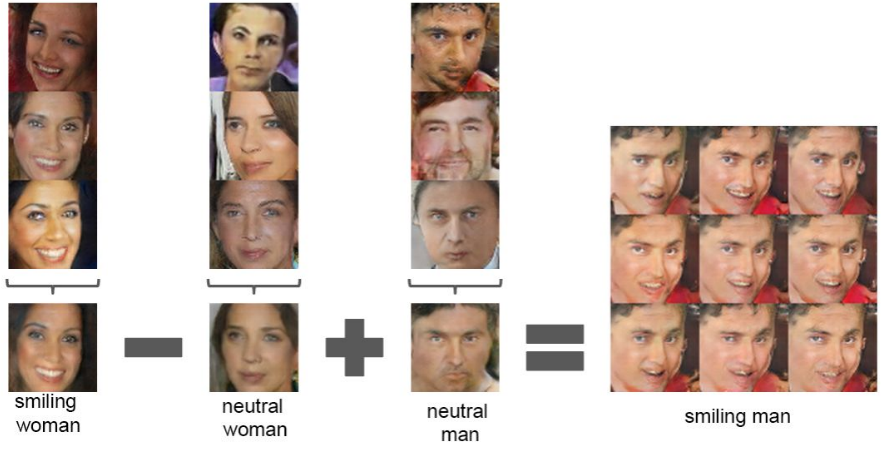
CSIC 5011: Topological and Geometrical Data Reduction and Visualization

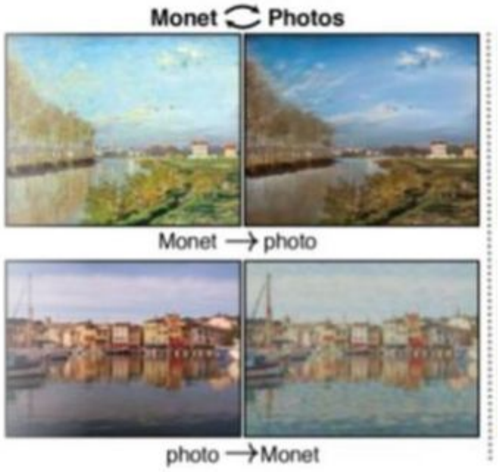
Spring 2020

Jose Vinicius de Miranda Cardoso, Yixin Men, Shunkang Zhang

Applications

- High-resolution Image Synthesis, Image Inpainting, Image Editing, Image to image translation
- Object Detection
- Anomaly Detection in Medical Field
- Music and Video Generation
- Attention Prediction
- Financial Time-series Modeling





Introduction

- Generative adversarial networks (GANs) and other adversarial methods are based on a game-theoretical perspective on joint optimization of two neural networks as players in a game.
- Generative Adversarial Networks (GAN) have received wide attention in the machine learning field for their potential to learn high-dimensional, complex real data distribution.
- In contrast to discriminative models, *generative models* aim to learn the underlying distribution of the data and the generative process that creates them.
- D is trained to maximize the probability of assigning the correct label to both the training samples and samples generated by G. Simultaneously, G is trained to minimize the $\log(1-D(G(z)))$.

GAN Structure

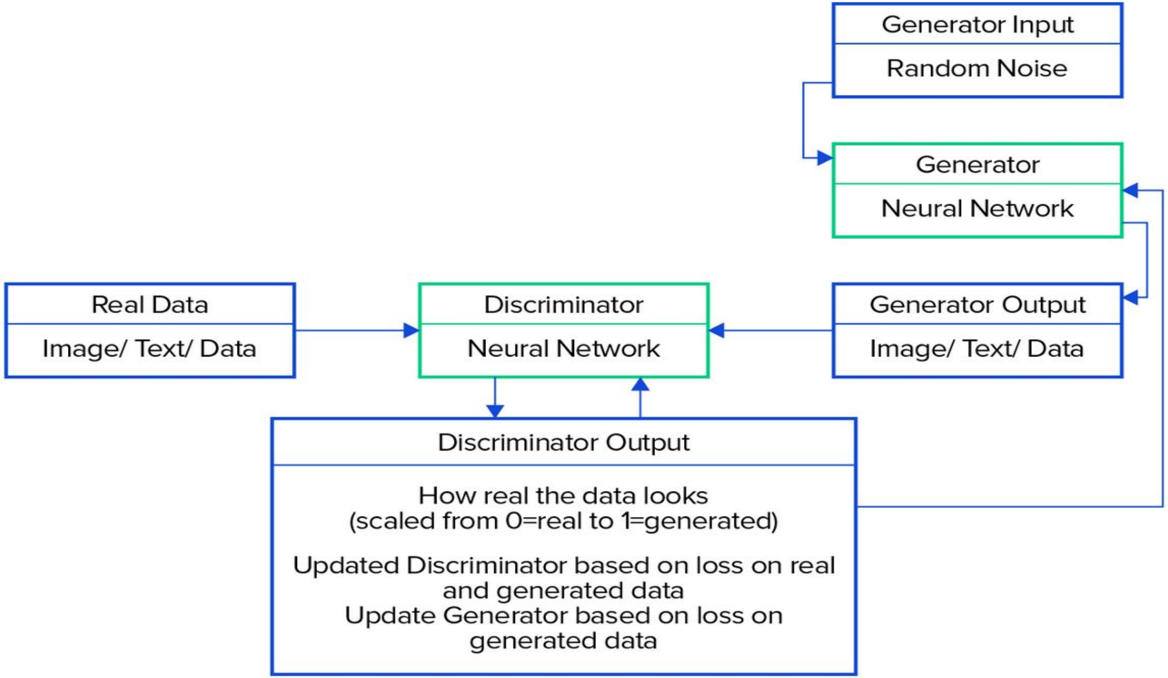


Figure 1. Generative adversarial network. The generator *G* takes a noise vector z sampled from a distribution p_z as input and uses fully connected or convolutional layers to transform this vector into a sample x. The discriminator *D* tries to distinguish these samples from samples drawn from the real data distribution p_{data}.

Generative adversarial networks consist of two neural networks.
The first network, the *generator*, tries to generate synthetic but perceptually convincing samples $x \in p_{fake}$ that appear to have been drawn from a real data distribution p_{data}. It transforms noise vectors z drawn from a distribution p_z into new samples, i.e., $x=G(z)$.
The second network, the *discriminator*, has access to real samples from p_{data} and to the samples generated by *G*, and tries to discriminate between these two. GANs are trained by solving the following optimization problem that the discriminator is trying to maximize and the generator is trying to minimize:

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

where *G* is the generator, *D* is the discriminator, $V(D,G)$ is the objective function, p_{data} is the distribution of real samples, and p_z is a distribution from which noise vectors are drawn, e.g., a uniform distribution or spherical Gaussian. The final layer of the discriminator network contains a sigmoid activation function, so that $D(x), D(G(z)) \in [0,1]$.

Basic training algorithm

Algorithm 1: Minibatch stochastic gradient descent training of generative adversarial nets.

for *number of training iterations* **do**

$\triangleright$ Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.

$\triangleright$ Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.

$\triangleright$ Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

$\triangleright$ Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.

$\triangleright$ Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end

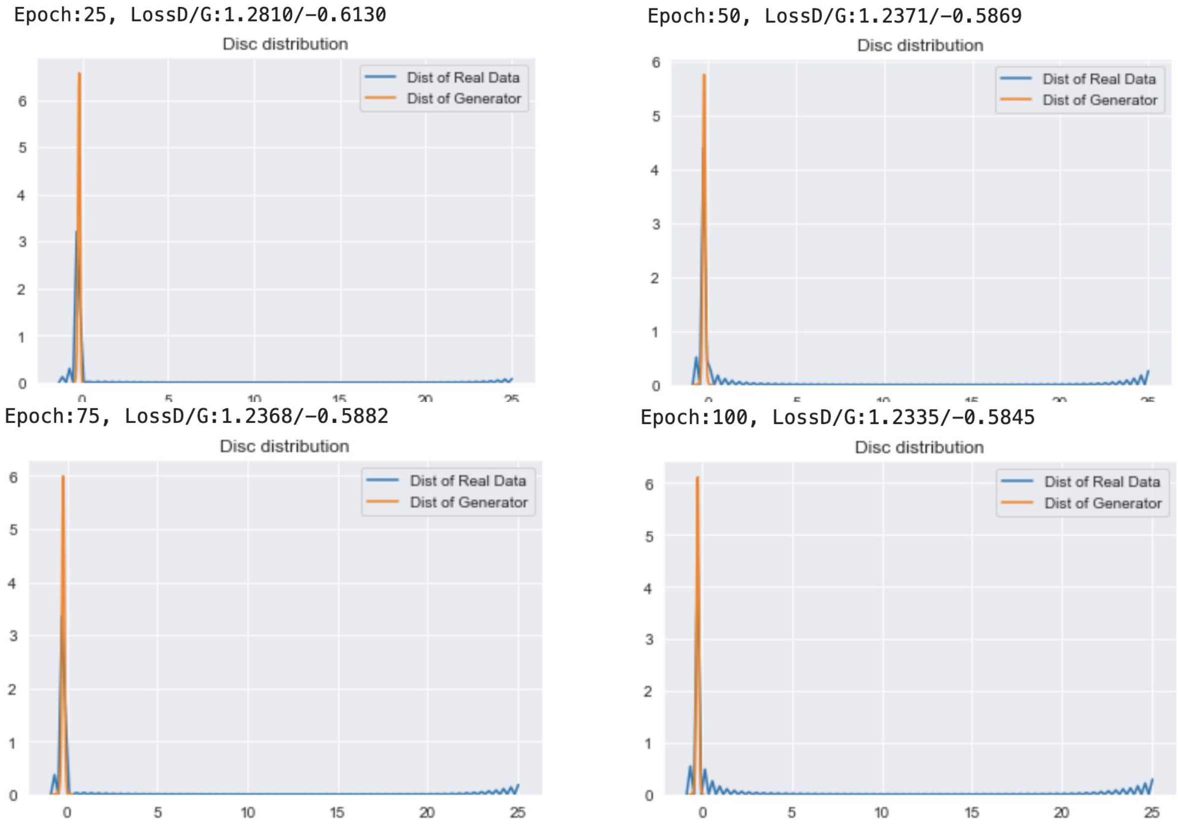
Experiments with synthetic data

We generate data under Huber Contamination model:

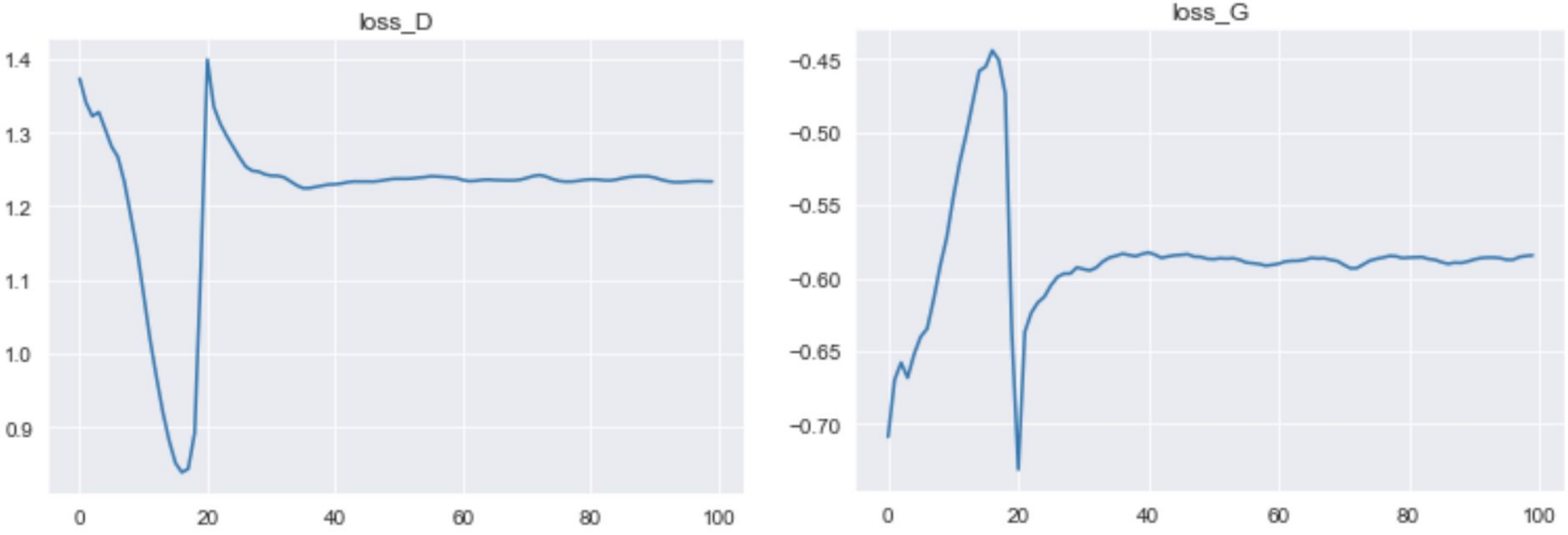
$$x_i \sim 0.8\mathcal{N}(0, I_p) + 0.2\mathcal{N}(5 * 1_p, 2 * I_p),$$

with $N = 50000$ samples and dimension $p = 50$.

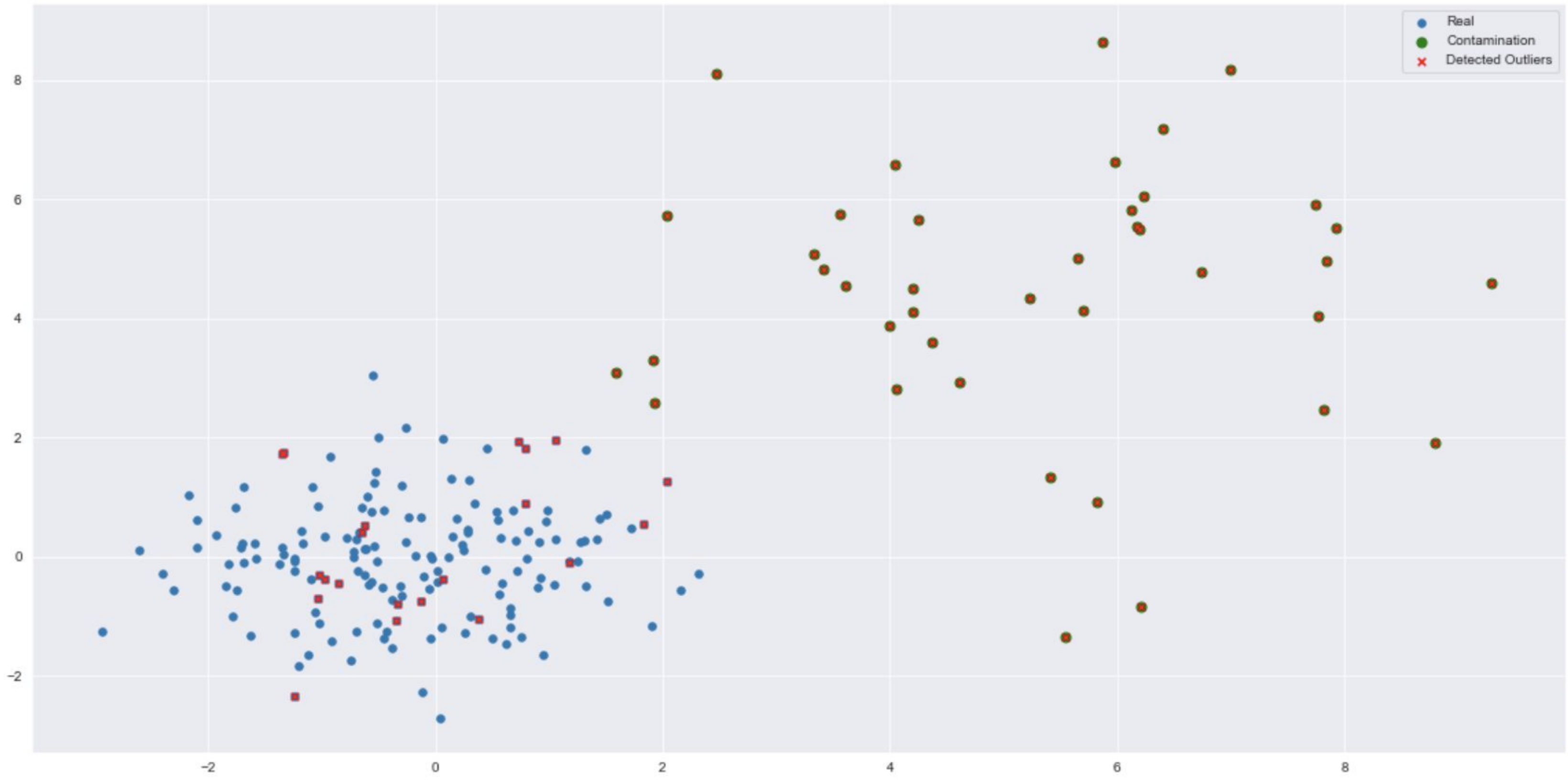
Experiments with synthetic data



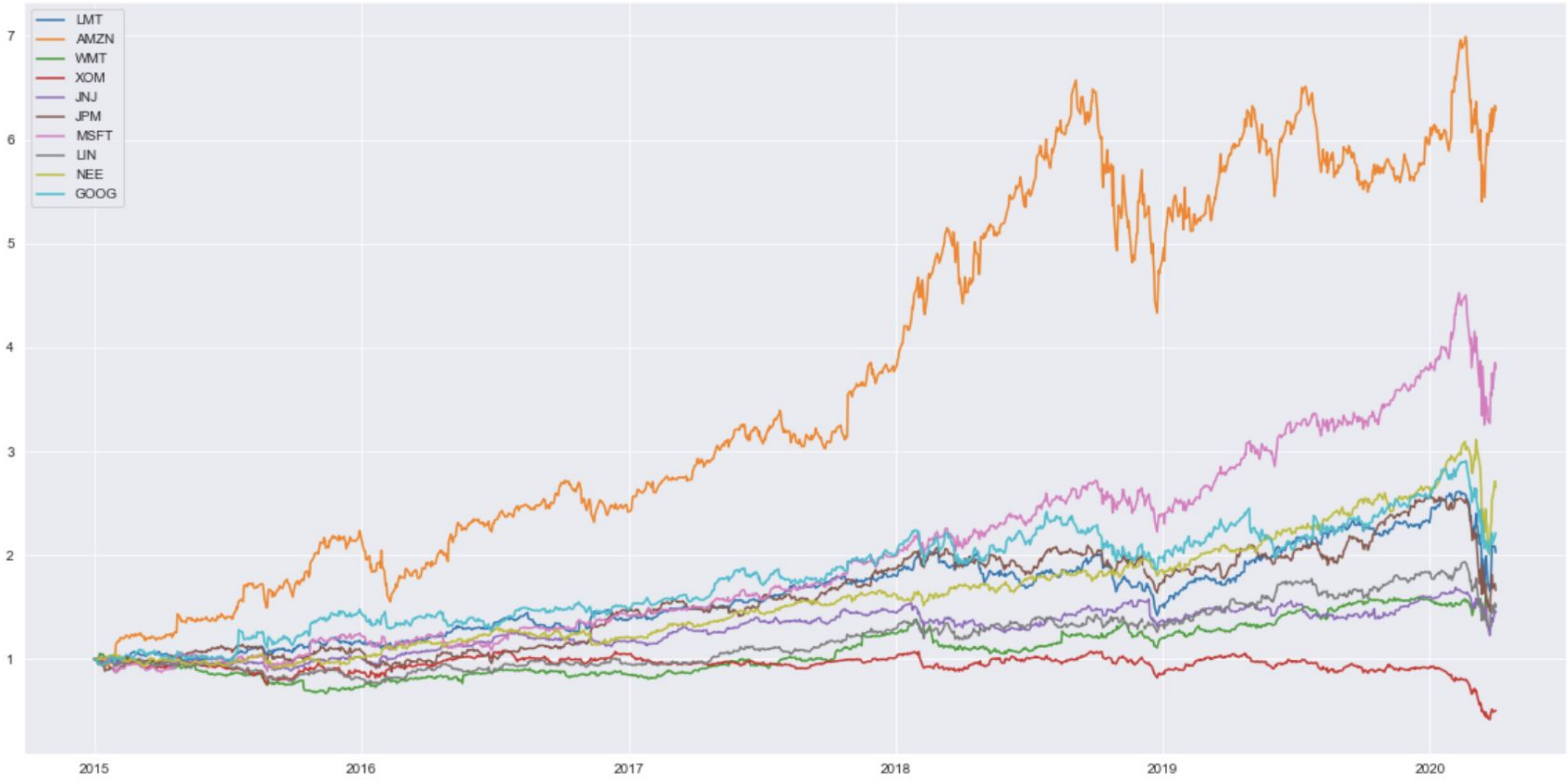
Experiments with synthetic data



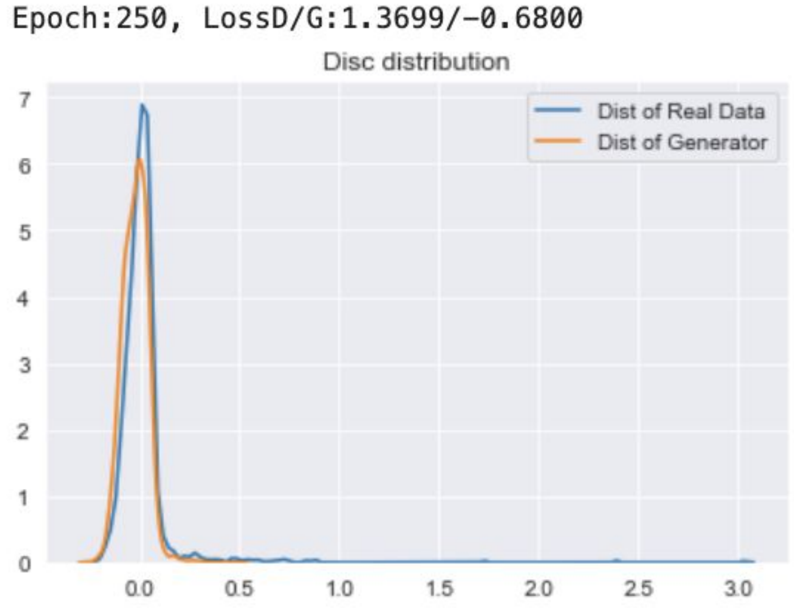
Experiments with synthetic data: outlier detection

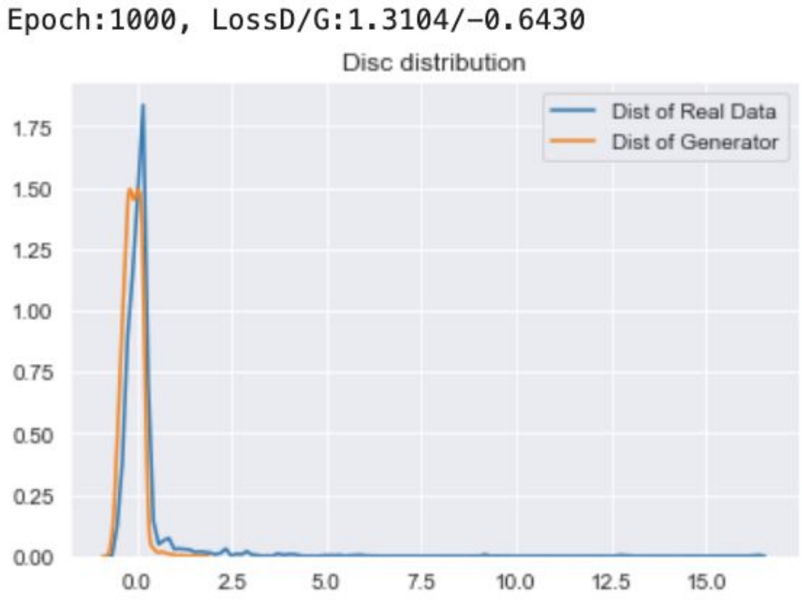


Stock data experiment



Stock data experiment





More applications in stock markets

- **CorrGAN: Sampling Realistic Financial Correlation Matrices Using Generative Adversarial Networks**
  https://arxiv.org/abs/1910.09504

- **Quant GANs: Deep Generation of Financial Time Series**

  https://arxiv.org/abs/1907.06673

Reformulate GAN in function space

Given $\mu \in \mathcal{P}(\mathbb{R}^d)$ with the density q, we use the f-divergence to measure the discrepancy between μ and ν which is defined as

$$\mathbb{D}_f(q\|p) = \int p(\mathbf{x}) f\left(\frac{q(\mathbf{x})}{p(\mathbf{x})}\right) d\mathbf{x}, \tag{1}$$

where $f : \mathbb{R}^+ \to \mathbb{R}$ is a convex and continuous function satisfying $f(1) = 0$. We also require $f(\cdot)$ is twice-differentiable. Let $\mathcal{F}[q]$ denote the energy functional $\mathbb{D}_f(\cdot\|p) : \mathcal{P}(\mathbb{R}^d) \to \mathbb{R}^+ \cup \{0\}$ for simplicity. We consider a curve $\mu_t : \mathbb{R}^+ \to \mathcal{P}(\mathbb{R}^d)$ and μ_t admits the density q_t. Let $\mathbf{v}_t = -\nabla\left(\frac{\delta\mathcal{F}}{\delta q_t}(q_t)\right) : \mathbb{R}^+ \to (\mathbb{R}^d \to \mathbb{R}^d)$ be the velocity vector field with $r_t(\mathbf{x}) = \frac{q_t(\mathbf{x})}{p(\mathbf{x})}$.

Definition 1 *We call μ_t a variational gradient flow of the energy functional $\mathcal{F}[\cdot]$ governed by the velocity vector field $\boldsymbol{v}_t$ if the following Vlasov-Fokker-Planck equation holds*

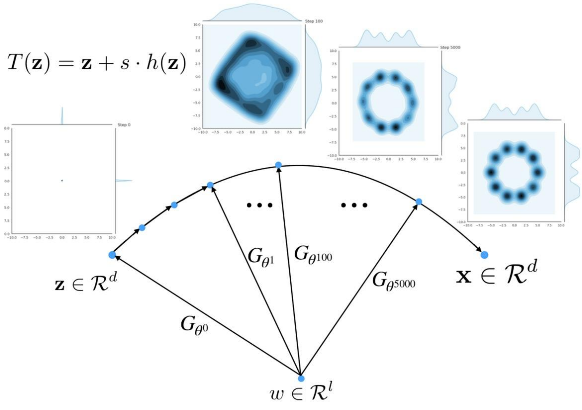
$$\frac{d}{dt} q_t = -\nabla \cdot (q_t \boldsymbol{v}_t) \quad \text{in} \quad \mathbb{R}^+ \times \mathbb{R}^d. \tag{2}$$

Theorem 1 *For any $\boldsymbol{g} \in \mathcal{H}(q_t)$, if the vanishing condition $\lim_{\|\boldsymbol{x}\|\to\infty} \|f'(r_t(\mathbf{x}))q_t(\mathbf{x})\mathbf{g}(\mathbf{x})\| = 0$ is satisfied, then*

$$\left\langle \frac{\delta\mathcal{L}}{\delta\mathbf{h}}[\mathbf{0}], \boldsymbol{g} \right\rangle_{\mathcal{H}(q_t)} = \langle f''(r_t)\nabla r_t, \boldsymbol{g}\rangle_{\mathcal{H}(q_t)}.$$

Theorem 2 *The evolving distribution q_t under the infinitesimal pushforward map $\mathbb{T}_{s,\mathbf{v}_t}$ satisfies the Vlasov-Fokker-Planck equation (2).*

Theorem 3 *At the population level, the logD-trick GAN minimizes the "logD" divergence $\mathbb{D}_f(q(\mathbf{x})\|p(\mathbf{x}))$, with $f(u) = (u+1)\log(u+1) - 2\log 2$, where $q(\mathbf{x})$ is the distribution of generated data.*

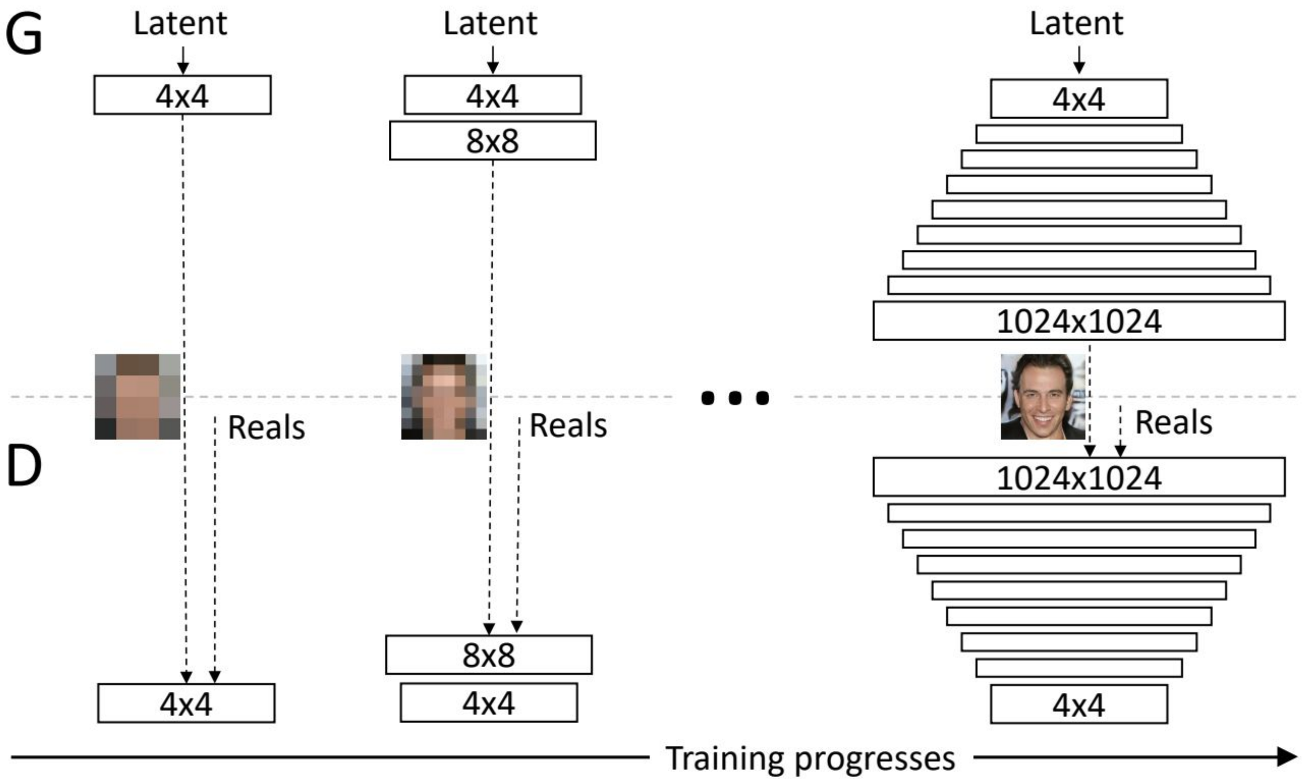


Yuan, G., Yuling, J., Yang, W., Yao, W., Can, Y., & Shunkang, Z. (2019). Deep generative learning via variational gradient flow. *arXiv preprint arXiv:1901.08469*.

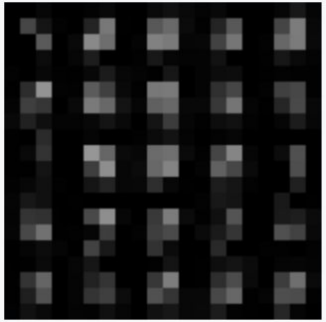
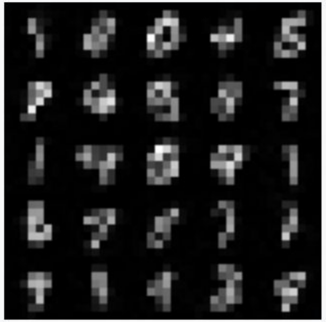
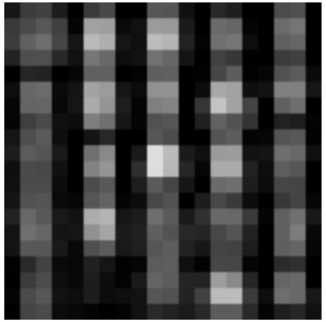
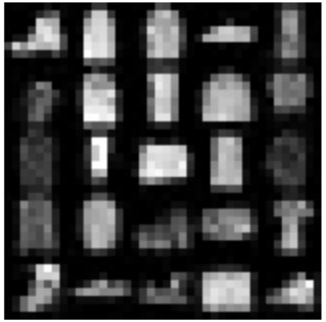
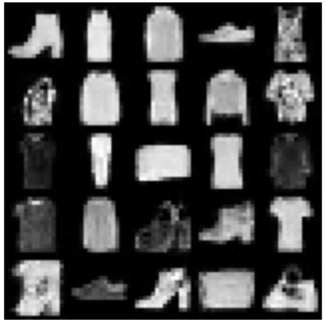
Reformulate GAN in function space

MNIST Fashion-MNIST CIFAR10 CelebA

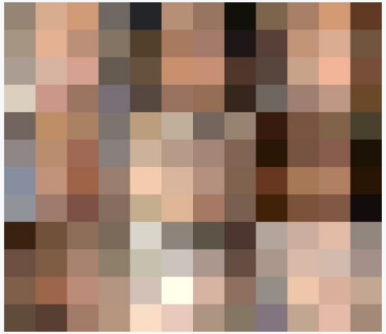
GAN with progressive training



GAN with progressive training

Resolution	4x4	8x8	16x16	32x32
MNIST				
Fashion-MNIST				

GAN with progressive training

Resolution	4x4	8x8	16x16
CelebA			
Resolution	32x32	64x64	128x128
CelebA			

Generative Adversarial Networks: A Review

CSIC 5011: Topological and Geometrical Data Reduction and Visualization

Spring 2020

Jose Vinicius de Miranda Cardoso, Yixin Men, Shunkang Zhang