

Statistical Models & Computing Methods

Lecture 19: Generative Adversarial Nets



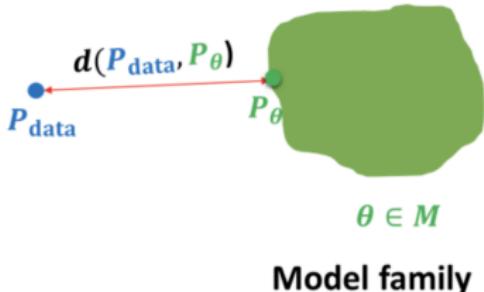
Cheng Zhang

School of Mathematical Sciences, Peking University

November 29, 2022



$$\mathbf{x}_i \sim P_{\text{data}} \\ i = 1, 2, \dots, n$$



► Model families

- Autoregressive Models: $p_\theta(x) = \prod_{i=1}^n p_\theta(x_i | x_{<i})$
- Variational Autoencoders: $p_\theta(x) = \int_z p_\theta(x, z) dz$
- Normalizing Flow Models:

$$p_X(x; \theta) = p_Z(f_\theta^{-1}(x)) \left| \det \left(\frac{\partial f_\theta^{-1}(x)}{\partial x} \right) \right|$$

- All the above families are based on maximizing likelihoods (or approximations, e.g., lower bound)
- Is the likelihood a good indicator of the quality of samples generated by the model?

- ▶ Optimal generative model will give best sample quality and highest test log-likelihood. However, in practice, **high log-likelihoods \neq good sample quality** (Theis et al., 2016)
- ▶ **Case 1:** great test log-likelihoods, poor samples. Consider a mixture model $p_\theta(x) = 0.01p_{\text{data}}(x) + 0.99p_{\text{noise}}(x)$, we have

$$\mathbb{E}_{p_{\text{data}}} \log p_{\text{data}}(x) \geq \mathbb{E}_{p_{\text{data}}} \log p_\theta(x) \geq \mathbb{E}_{p_{\text{data}}} \log p_{\text{data}}(x) - \log 100$$

This means $\mathbb{E}_{p_{\text{data}}} \log p_\theta(x) \approx \mathbb{E}_{p_{\text{data}}} \log p_{\text{data}}(x)$ when the dimension of x is large.

- ▶ **Case 2:** great samples, poor test log-likelihoods. E.g., memorizing training set: samples look exactly like the training set; test set will have zero probability
- ▶ The above cases suggest that it might be useful to disentangle likelihoods and samples \Rightarrow **likelihood-free learning!**



vs.



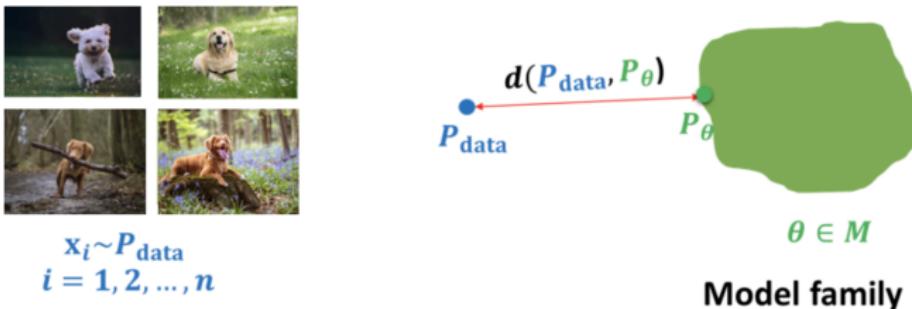
$$S_1 = \{\mathbf{x} \sim P\}$$

$$S_2 = \{\mathbf{x} \sim Q\}$$

Given samples from two distributions $S_1 = \{x \sim P\}$ and $S_2 = \{x \sim Q\}$, how can we tell if these samples are from the same distribution? (i.e., $P = Q$?)

- ▶ Given $S_1 = \{x \sim P\}$ and $S_2 = \{x \sim Q\}$, a **two-sample test** considers the following hypotheses
 - ▶ Null hypothesis $H_0 : P = Q$
 - ▶ Alternative hypothesis $H_1 : p \neq Q$
- ▶ Test statistic T compares S_1 and S_2 , e.g., difference in means, variances of the two sets of samples
- ▶ If T is less than a threshold α , accept H_0 else reject it
- ▶ **Key observation:** Test statistics is likelihood-free since it does not involve the densities P or Q (only samples)

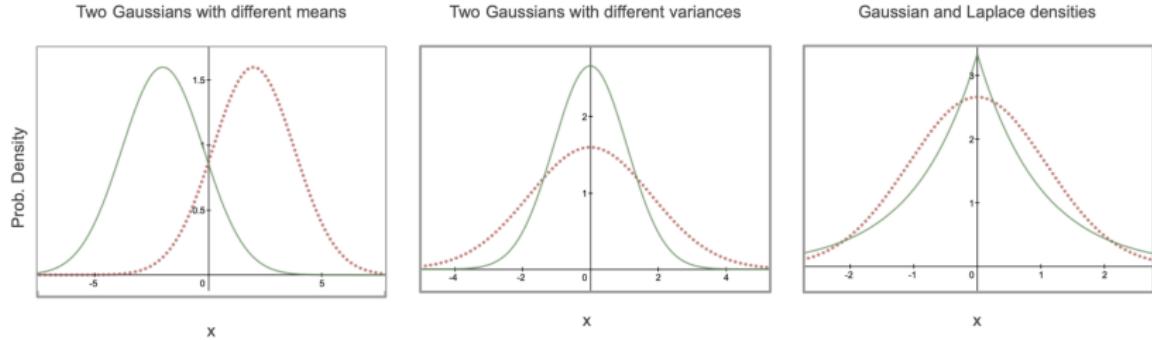




- ▶ Suppose we have direct access to the data set
 $S_1 = \mathcal{D} = \{x \sim p_{\text{data}}\}$
- ▶ Now assume that the model distribution p_θ permits efficient sampling (e.g., directed models). Let $S_2 = \{x \sim p_\theta\}$
- ▶ Use a two-sample test objective to measure the distance between distributions and train the generative model p_θ to minimize this distance between S_1 and S_2

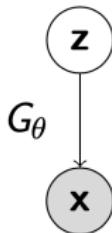
Two-Sample Test via a Discriminator

7/37



- ▶ Finding a two-sample test objective in high dimensions is non-trivial
- ▶ In the generative model setup, we know that S_1 and S_2 come from different distributions p_{data} and p_{θ} respectively
- ▶ **Key idea:** Learn a statistic that maximizes a suitable notion of distance between the two sets of samples S_1 and S_2

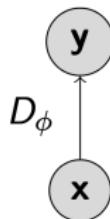
The **generator** and **discriminator** play a minimax game!



Generator

- ▶ Directed, latent variable model with a deterministic mapping between z and x given by G_θ
- ▶ Minimizes a two-sample test objective (in support of the null hypothesis $p_{\text{data}} = p_\theta$)

The **generator** and **discriminator** play a minimax game!



Discriminator

- ▶ Any function (e.g., neural network) which tries to distinguish “real” samples from the dataset and “fake” samples generated from the model
- ▶ Maximizes the two-sample test objectivee (in support of the alternative hypothesis $p_{\text{data}} \neq p_\theta$)

- ▶ Training objective for discriminator:

$$\max_D V(G, D) = \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) + \mathbb{E}_{x \sim p_G} \log(1 - D(x))$$

- ▶ For a fixed generator G , the discriminator is performing binary classification with the cross entropy objective
 - ▶ Assign probability 1 to true data points $x \sim p_{\text{data}}$
 - ▶ Assign probability 0 to fake samples $x \sim p_G$
- ▶ Optimal discriminator

$$D_G^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}$$



- ▶ Training Objective for generator:

$$\min_G V(G, D) = \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) + \mathbb{E}_{x \sim p_G} \log(1 - D(x))$$

- ▶ For the optimal discriminator $D_G^*(\cdot)$, we have

$$\begin{aligned} V(G, D_G^*) &= \mathbb{E}_{x \sim p_{\text{data}}} \log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)} + \mathbb{E}_{x \sim p_G} \log \frac{p_G(x)}{p_{\text{data}}(x) + p_G(x)} \\ &= \mathbb{E}_{x \sim p_{\text{data}}} \log \frac{p_{\text{data}}(x)}{\frac{p_{\text{data}}(x) + p_G(x)}{2}} + \mathbb{E}_{x \sim p_G} \log \frac{p_G(x)}{\frac{p_{\text{data}}(x) + p_G(x)}{2}} - \log 4 \\ &= \text{KL} \left(p_{\text{data}} \middle\| \frac{p_{\text{data}} + p_G}{2} \right) + \text{KL} \left(p_G \middle\| \frac{p_{\text{data}} + p_G}{2} \right) - \log 4 \end{aligned}$$

- ▶ The sum of KL in the above equation is known as Jensen-Shannon divergence (JSD)

$$\text{JSD}(p, q) = \text{KL} \left(p \left\| \frac{p+q}{2} \right. \right) + \text{KL} \left(q \left\| \frac{p+q}{2} \right. \right)$$

- ▶ Properties
 - ▶ $\text{JSD}(p, q) \geq 0$
 - ▶ $\text{JSD}(p, q) = 0$ iff $p = q$
 - ▶ $\text{JSD}(p, q) = \text{JSD}(q, p)$
 - ▶ $\sqrt{\text{JSD}(p, q)}$ satisfies triangle inequality
- ▶ Optimal generator for the JSD GAN

$$p_G = p_{\text{data}}$$

- ▶ For the optimal discriminator $D_{G^*}^*(\cdot)$ and generator $G^*(\cdot)$, we have

$$V(G^*, D_{G^*}^*(x)) = -\log 4$$



$$\min_{\theta} \max_{\phi} V(G_{\theta}, D_{\phi}) = \mathbb{E}_{x \sim p_{\text{data}}} \log D_{\phi}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\phi}(G_{\theta}(z)))$$

- ▶ sample m training points $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ from \mathcal{D}
- ▶ sample m noise vectors $z^{(1)}, z^{(2)}, \dots, z^{(m)}$ from p_z
- ▶ generator parameters θ update: stochastic gradient **descent**

$$\nabla_{\theta} V(G_{\theta}, D_{\phi}) = \frac{1}{m} \nabla_{\theta} \sum_{i=1}^m \log(1 - D_{\phi}(G_{\theta}(z^{(i)})))$$

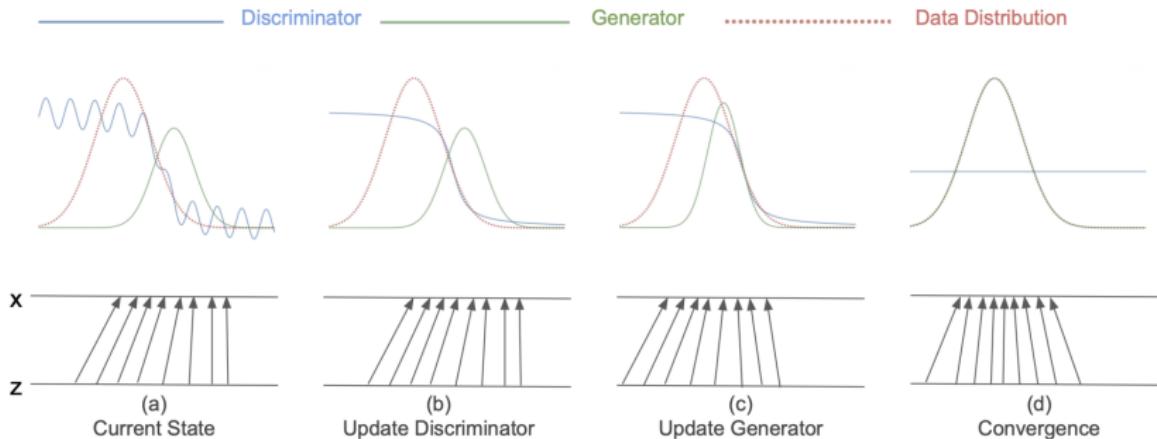
- ▶ discriminator parameters ϕ update: stochastic gradient **ascent**

$$\nabla_{\phi} V(G_{\theta}, D_{\phi}) = \frac{1}{m} \nabla_{\phi} \sum_{i=1}^m \log D_{\phi}(x^{(i)}) + \log(1 - D_{\phi}(G_{\theta}(z^{(i)})))$$

- ▶ Repeat for fixed number of epochs

A Toy Example

14/37



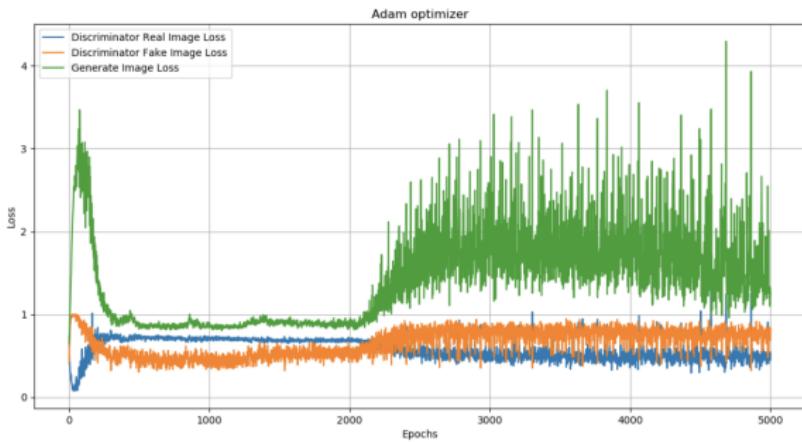
Adapted from Goodfellow, 2014



- ▶ GANs have been successfully applied to several domains and tasks
- ▶ However, working with GANs can be very challenging in practice: **unstable optimization/mode collapse/evaluation**
- ▶ Many bag of tricks applied to train GANs successfully

Image source: Ian Goodfellow. Samples from Goodfellow et al., 2014, Radford et al., 2015, Liu et al., 2016, Karras et al., 2017, Karras et al., 2018

- **Theorem:** If the generator updates are made in function space and discriminator is optimal at every step, then the generator is guaranteed to converge to the data distribution
- **Unrealistic assumptions!** In practice, the generator and discriminator loss keeps oscillating during GAN training



- No robust stopping criteria in practice (unlike MLE)

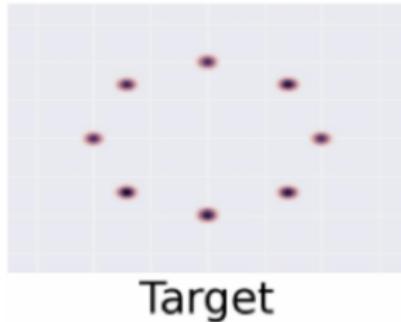
- ▶ GANs are notorious for suffering from **mode collapse**
- ▶ Intuitively, this refers to the phenomena where the generator of a GAN collapse to one or few samples (i.e., “modes”)



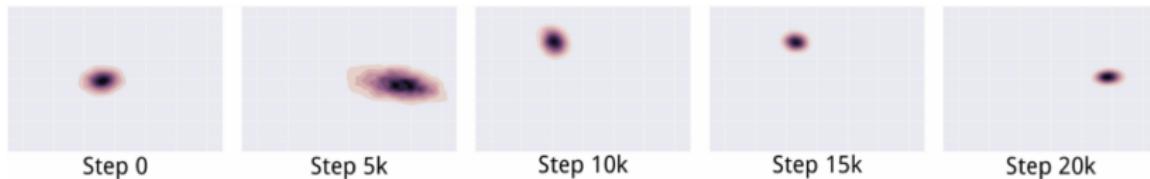
Arjovsky et al., 2017



北京大学
PEKING UNIVERSITY

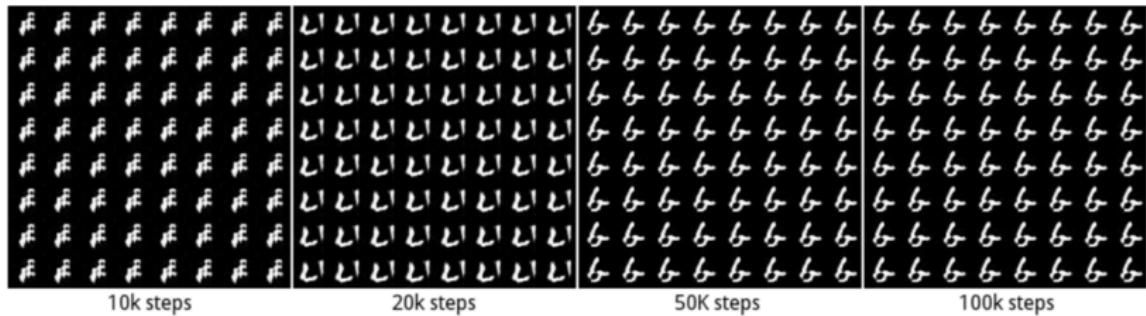


- ▶ True distribution is a mixture of Gaussians



Source: Metz et al., 2017

- ▶ The generator distribution keeps oscillating between different models



Source: Metz et al., 2017

- ▶ Fixes to mode collapse are mostly empirically driven: alternate architectures, adding regularization terms, injecting small noise perturbations etc.
- ▶ Tips and tricks to make GAN work by Soumith Chintala:
<https://github.com/soumith/ganhacks>



Source: Robbie Barrat, Obvious

GAN generated art auctioned at Christie's.

Expected Price: \$7,000 – \$10,000

True Price: \$432,500



北京大学
PEKING UNIVERSITY

- ▶ The GAN Zoo:
<https://github.com/hindupuravinash/the-gan-zoo>
- ▶ Examples
 - ▶ Rich class of likelihood-free objectives
 - ▶ Combination with latent representations
 - ▶ Application: Image-to-image translation, etc.

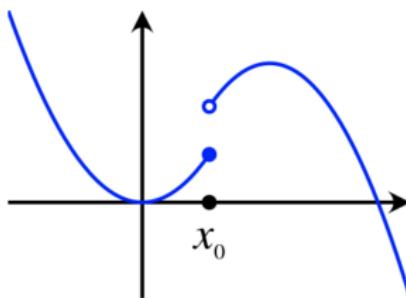


- Given two densities p and q , the f -divergence is given by

$$D_f(p\|q) = \mathbb{E}_{x \sim q} f\left(\frac{p(x)}{q(x)}\right)$$

where f is any convex, lower-semicontinuous function with $f(1) = 0$

- Lower-semicontinuous: function value at any point x_0 is close to $f(x_0)$ or greater than $f(x_0)$



- Example: KL divergence with $f(u) = u \log u$

Many more f -divergence!

Name	$D_f(P\ Q)$	Generator $f(u)$
Total variation	$\frac{1}{2} \int p(x) - q(x) dx$	$\frac{1}{2} u - 1 $
Kullback-Leibler	$\int p(x) \log \frac{p(x)}{q(x)} dx$	$u \log u$
Reverse Kullback-Leibler	$\int q(x) \log \frac{q(x)}{p(x)} dx$	$-\log u$
Pearson χ^2	$\int \frac{(q(x) - p(x))^2}{p(x)} dx$	$(u - 1)^2$
Neyman χ^2	$\int \frac{(p(x) - q(x))^2}{q(x)} dx$	$\frac{(1-u)^2}{u}$
Squared Hellinger	$\int \left(\sqrt{p(x)} - \sqrt{q(x)} \right)^2 dx$	$(\sqrt{u} - 1)^2$
Jeffrey	$\int (p(x) - q(x)) \log \left(\frac{p(x)}{q(x)} \right) dx$	$(u - 1) \log u$
Jensen-Shannon	$\frac{1}{2} \int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} dx$	$-(u+1) \log \frac{1+u}{2} + u \log u$
Jensen-Shannon-weighted	$\int p(x)\pi \log \frac{2p(x)}{\pi p(x)+(1-\pi)q(x)} + (1-\pi)q(x) \log \frac{2q(x)}{\pi p(x)+(1-\pi)q(x)} dx$	$\pi u \log u - (1 - \pi + \pi u) \log(1 - \pi + \pi u)$
GAN	$\int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} dx - \log(4)$	$u \log u - (u+1) \log(u+1)$
α -divergence ($\alpha \notin \{0, 1\}$)	$\frac{1}{\alpha(\alpha-1)} \int \left(p(x) \left[\left(\frac{q(x)}{p(x)} \right)^\alpha - 1 \right] - \alpha(q(x) - p(x)) \right) dx$	$\frac{1}{\alpha(\alpha-1)} (u^\alpha - 1 - \alpha(u-1))$

Source: Nowozin et al., 2016



- ▶ To use f -divergences as a two-sample test objective for likelihood-free learning, we need to be able to estimate it only via samples
- ▶ Fenchel conjugate: For any function $f(\cdot)$, its convex conjugate is defined as

$$f^*(t) = \sup_{u \in \text{dom}_f} ut - f(u)$$

- ▶ Duality: $f^{**} = f$. When $f(\cdot)$ is convex, lower semicontinuous, so is $f^*(\cdot)$

$$f(u) = \sup_{t \in \text{dom}_{f^*}} tu - f^*(t)$$

- We can obtain a lower bound to any f -divergence via its Fenchel conjugate

$$\begin{aligned}D_f(p\|q) &= \mathbb{E}_{x \sim q} f\left(\frac{p(x)}{q(x)}\right) \\&= \mathbb{E}_{x \sim q} \sup_{t \in \text{dom}_{f^*}} \left(t \frac{p(x)}{q(x)} - f^*(t) \right) \\&\geq \mathbb{E}_{x \sim q} t(x) \frac{p(x)}{q(x)} - f^*(t(x)) \\&= \int_{\mathcal{X}} t(x)p(x) - f^*(t(x))q(x)dx \\&= \mathbb{E}_{x \sim p} t(x) - \mathbb{E}_{x \sim q} f^*(t(x))\end{aligned}$$

for any function $t : \mathcal{X} \mapsto \text{dom}_{f^*}$



- Variational lower bound

$$D_f(p\|q) \geq \sup_{t \in \mathcal{T}} (\mathbb{E}_{x \sim p} t(x) - \mathbb{E}_{x \sim q} f^*(t(x)))$$

- Choose any f -divergence
- Let $p = p_{\text{data}}$ and $q = p_G$
- Parameterize t by ϕ and G by θ
- Consider the following f -GAN objective

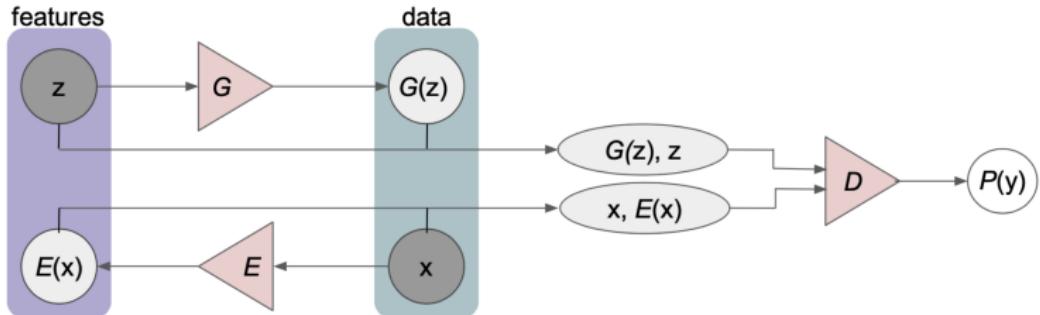
$$\min_{\theta} \max_{\phi} F(\theta, \phi) = \mathbb{E}_{x \sim p_{\text{data}}} t_{\phi}(x) - \mathbb{E}_{x \sim p_{G_{\theta}}} f^*(t_{\phi}(x))$$

- Generator G_{θ} tries to minimize the divergence estimate and discriminator t_{ϕ} tries to tighten the lower bound

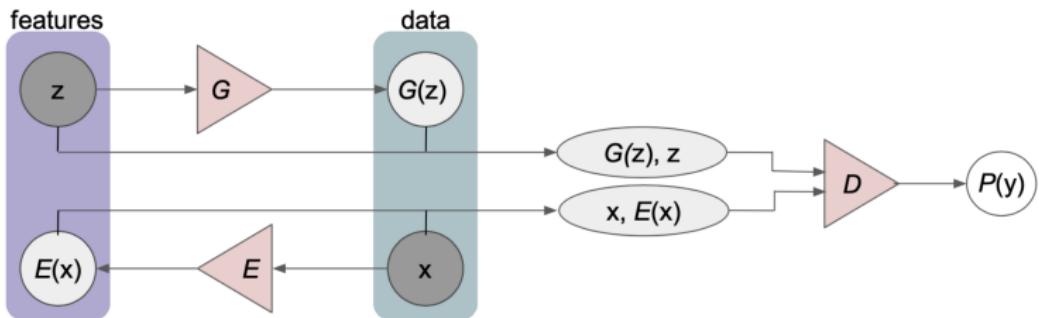
- ▶ The generator of a GAN is typically a directed, latent variable model with latent variable z and observed variables x . How can we infer the latent feature representations in a GAN?
- ▶ Unlike a normalizing flow model, the mapping $G : z \mapsto x$ need not to be invertible
- ▶ Unlike a variational autoencoder, there is no inference network $q(\cdot)$ which can learn a variational posterior over latent variables
- ▶ **Solution 1:** For any point x , use the activations of the prefinal layer of a discriminator as a feature representation
- ▶ Intuition: similar to supervised deep neural networks, the discriminator would have learned useful representations for x while distinguishing real and fake x

- ▶ If we want to directly learn the latent representation of x , we need a different learning algorithm
- ▶ A regular GAN optimizes a two-sample test objective that compares samples of x from the generator and the data distribution
- ▶ **Solution 2:** To infer latent representations, we will compare samples of x, z from joint distributions of observed and latent variables as per the model and the data distribution
- ▶ For any x generated via the model, we have access to z (sampled from a simple prior $p(z)$)
- ▶ For any x from the data distribution, the z is however unobserved (latent)



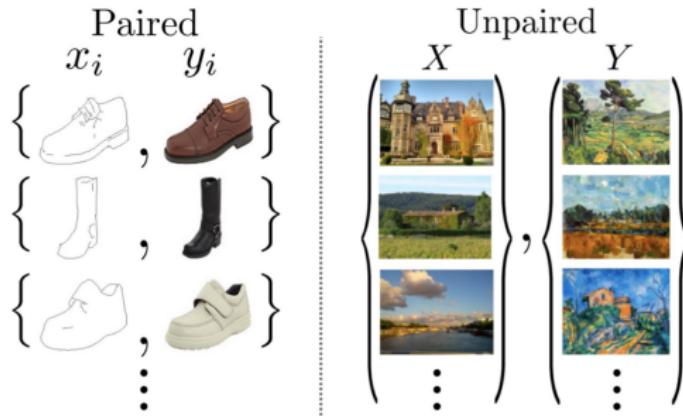


- ▶ In a BiGAN, we have an **encoder network** E in addition to the generator network G
- ▶ The encoder network only observes $x \sim p_{\text{data}}(x)$ during training to learn a mapping $E : x \mapsto z$
- ▶ As before, the generator network only observes the samples from the prior $z \sim p(z)$ during training to learn a mapping $G : z \mapsto x$



- ▶ The discriminator D observes samples from the generative model $z, G(z)$ and encoding distribution $E(x), x$
- ▶ The goal of the discriminator is to maximize the two-sample test objective between $z, G(z)$ and $E(x), x$
- ▶ After training is complete, new samples are generated via G and latent representations are inferred via E

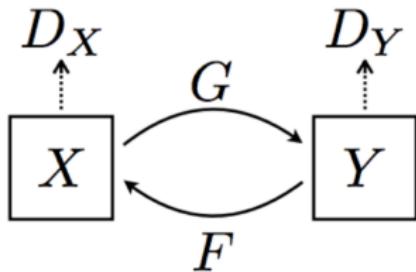
- ▶ Image-to-image translation: we are given image from two domains, \mathcal{X} and \mathcal{Y}
- ▶ Paired vs. unpaired examples



Source: Zhu et al., 2016

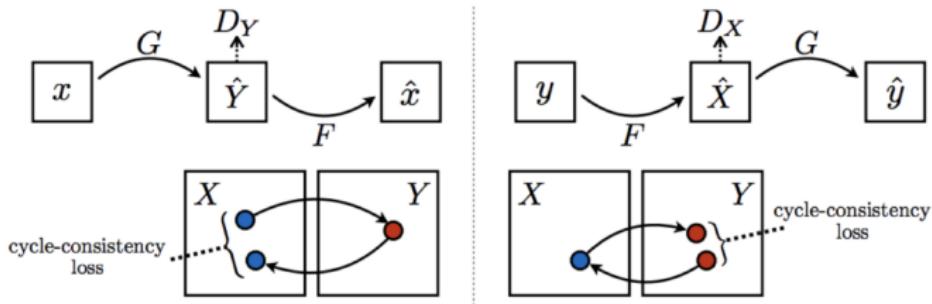
- ▶ Paired examples can be expensive to obtain. Can we translate from $\mathcal{X} \Leftrightarrow \mathcal{Y}$ in an unsupervised manner?

- ▶ To match the two distributions, we learn two parameterized conditional generative models $G : \mathcal{X} \mapsto \mathcal{Y}$ and $F : \mathcal{Y} \mapsto \mathcal{X}$
- ▶ G maps an element of \mathcal{X} to an element of \mathcal{Y} . A discriminator D_Y compares the observed dataset Y and the generated samples $\hat{Y} = G(X)$
- ▶ Similarly, F maps an element of \mathcal{Y} to an element of \mathcal{X} . A discriminator D_X compares the observed dataset X and the generated samples $\hat{X} = F(Y)$



Source: Zhu et al., 2016

- ▶ **Cycle consistency:** If we can go from X to \hat{Y} via G , then it should also be possible to go from \hat{Y} back to X via F
 - ▶ $F(G(X)) \approx X$
 - ▶ Similarly, vice versa: $G(F(Y)) \approx Y$



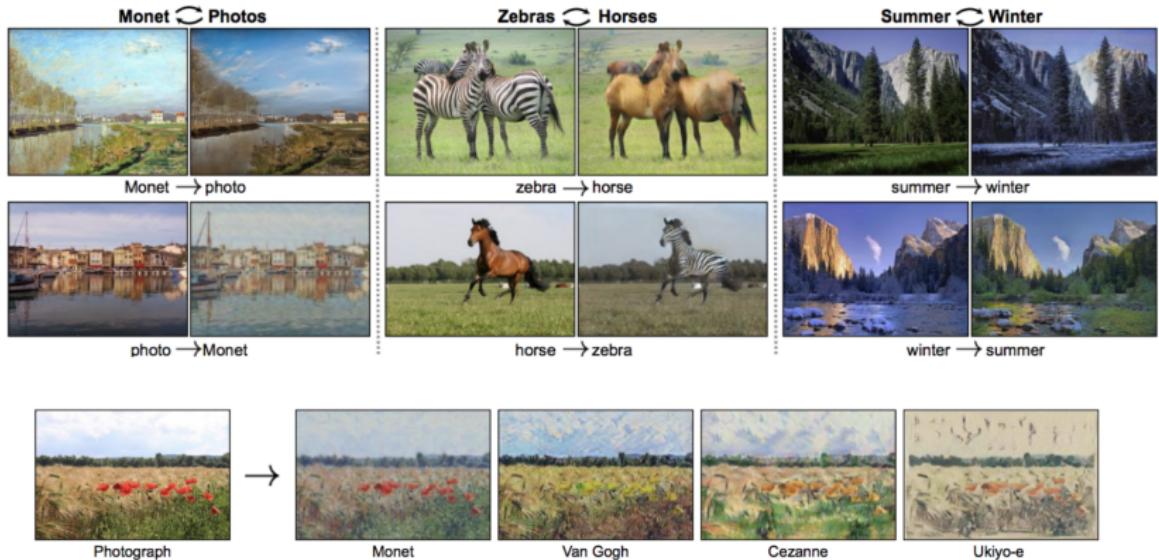
Source: Zhu et al., 2016

- ▶ Overall loss function

$$\begin{aligned} & \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, X, Y) \\ & + \lambda (\mathbb{E}_X \|F(G(X)) - X\|_1 + \mathbb{E}_Y \|G(F(Y)) - Y\|_1) \end{aligned}$$

CycleGAN in Practice

34/37



Source: Zhu et al., 2016



北京大学
PEKING UNIVERSITY

- ▶ Key observation: Samples and likelihoods are not correlated in practice
- ▶ Two-sample test objectives allow for learning generative models only via samples (likelihood-free)
- ▶ Wide range of two-sample test objectives covering f -divergences (and more)
- ▶ Latent representations can be inferred via BiGAN (and other GANs with similar autoencoder structures)
- ▶ Cycle-consistent domain translations via CycleGAN and other variants



- ▶ L. Theis, A. van den Oord, and M. Bethge. A note on the evaluation of generative models. In ICLR, 2016
- ▶ Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.
- ▶ Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.
- ▶ Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. arXiv preprint arXiv:1701.07875, 2017.

- ▶ L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled generative adversarial networks. arXiv preprint arXiv:1611.02163, 2016.
- ▶ 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.
- ▶ Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. arXiv preprint arXiv:1605.09782, 2016.
- ▶ Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In ICCV, 2017.

