EUROGRAPHICS 2018

Deep Learning for Graphics

Unsupervised Learning

Niloy Mitra
UCL

Iasonas Kokkinos
UCL/Facebook

Paul Guerrero
UCL

Vladimir Kim
Adobe Research

Kostas Rematas
U Washington

Tobias Ritschel
UCL

Timetable

	Niloy	Iasonas	Paul	Vova	Kostas	Tobias
Introduction	X	X	X			X
Theory	X					
NN Basics		X				X
Supervised Applications		X	X			
Data						X
Unsupervised Applications			X			
Beyond 2D	X			X		
Outlook	X	X	X	X	X	X

Unsupervised Learning

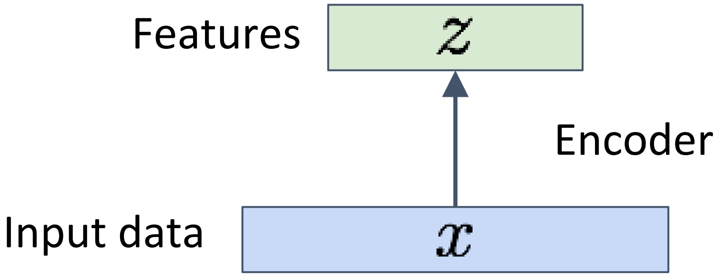
- There is no direct ground truth for the quantity of interest

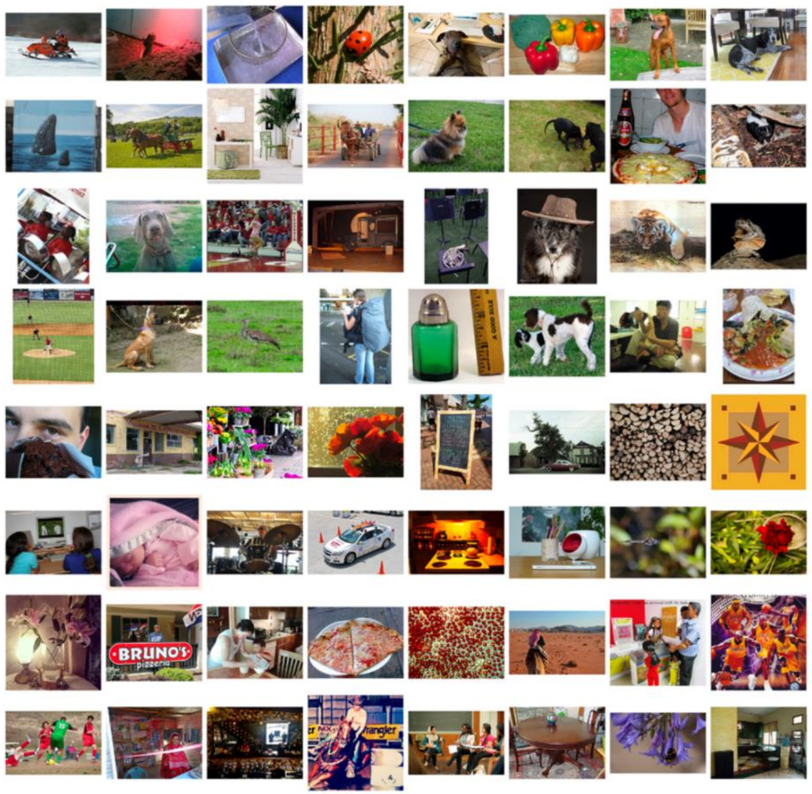
- Autoencoders
- Variational Autoencoders (VAEs)
- Generative Adversarial Networks (GANs)

Autoencoders

Goal: Meaningful features that capture the main factors of variation in the dataset

- These are good for classification, clustering, exploration, generation, …
- We have no ground truth for them



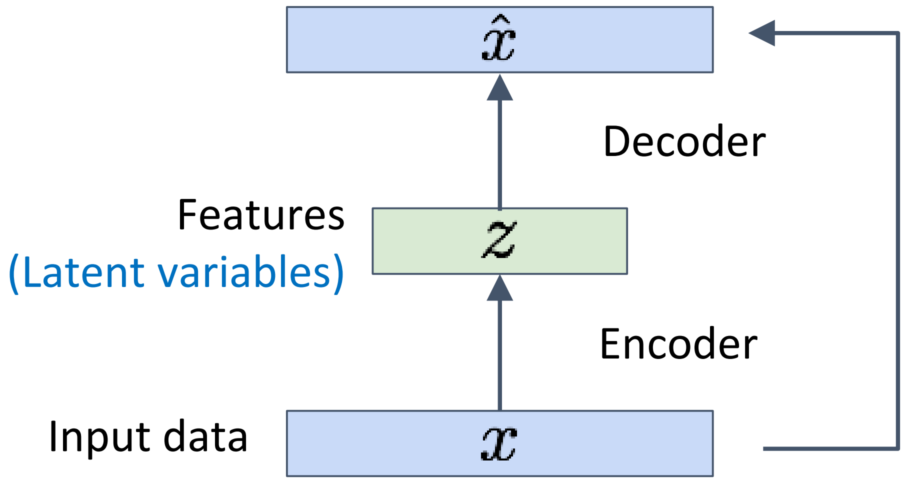


Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Autoencoders

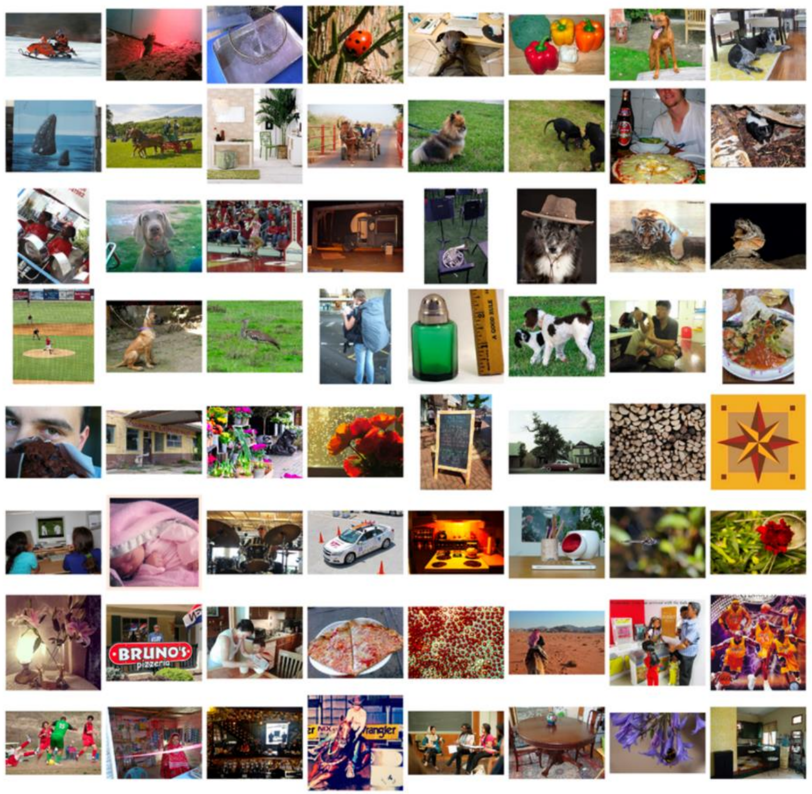
Goal: Meaningful features that capture the main factors of variation

Features that can be used to reconstruct the image



L2 Loss function:

$$||x - \hat{x}||^2$$

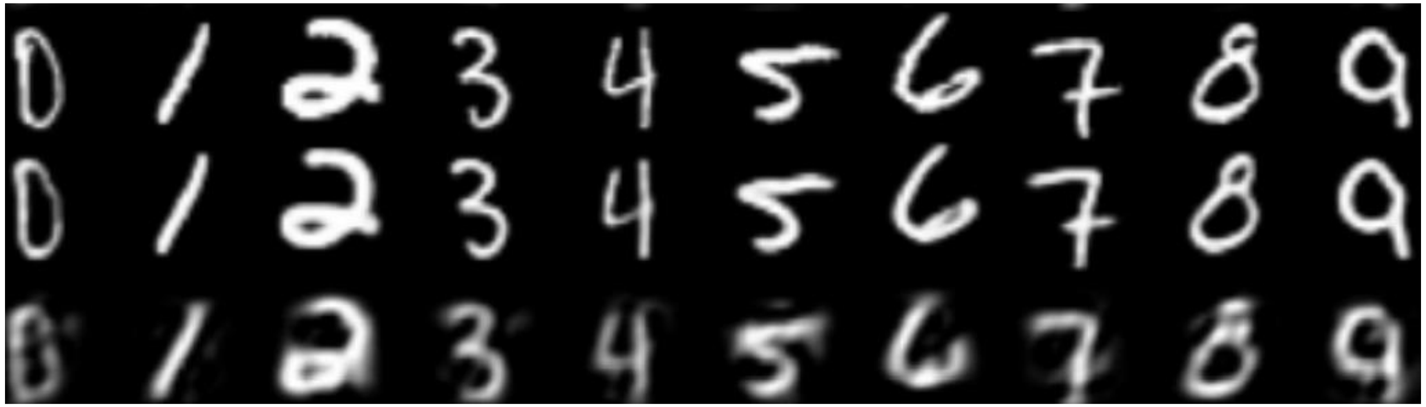


Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Autoencoders

Linear Transformation for Encoder and Decoder give result close to PCA

Deeper networks give better reconstructions, since basis can be non-linear



Original

Autoencoder

PCA

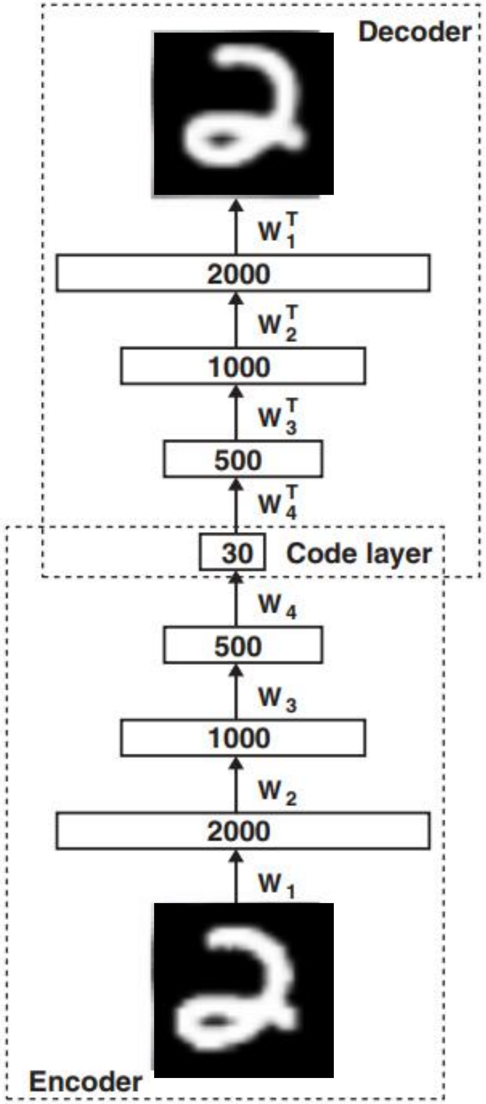
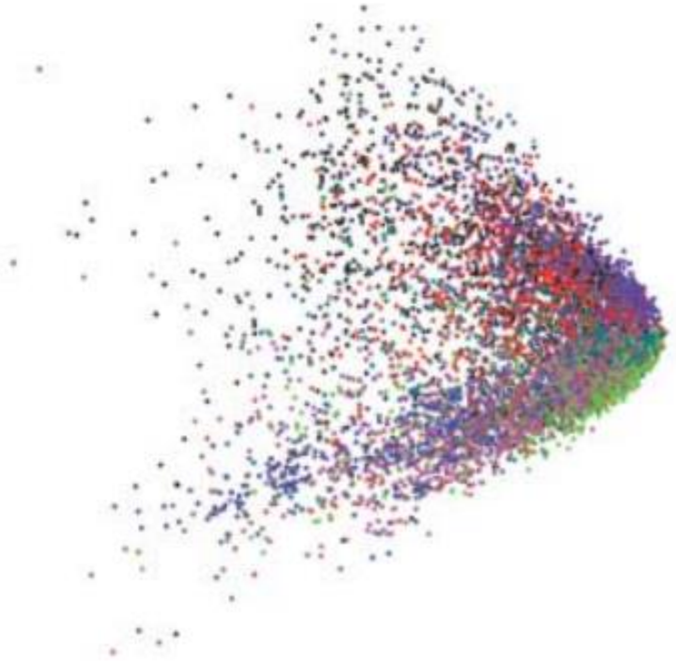


Image Credit: Reducing the Dimensionality of Data with Neural Networks, . Hinton and Salakhutdinov

Example: Document Word Prob. → 2D Code

LSA (based on PCA)



Autoencoder

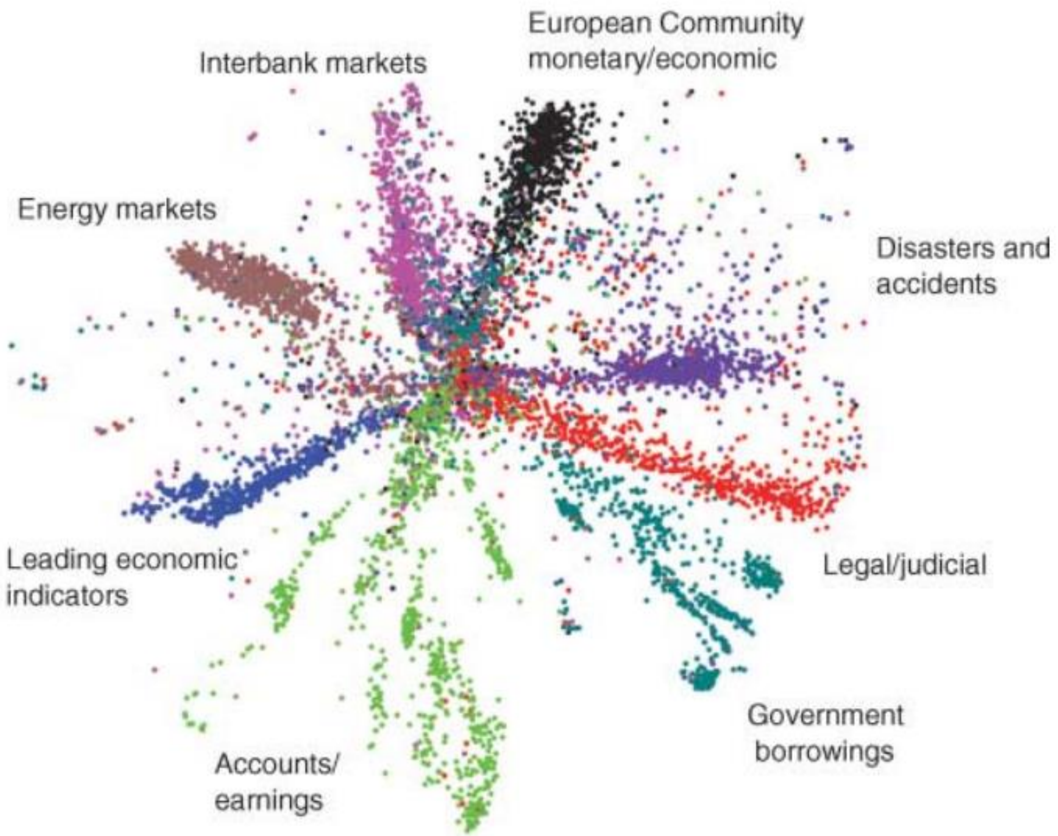
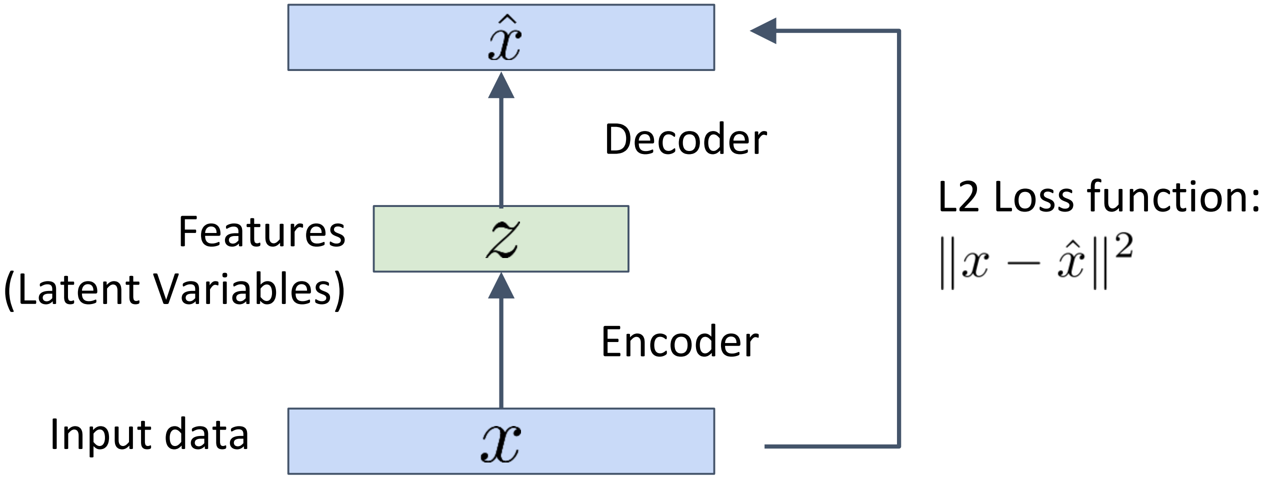


Image Credit: Reducing the Dimensionality of Data with Neural Networks, Hinton and Salakhutdinov

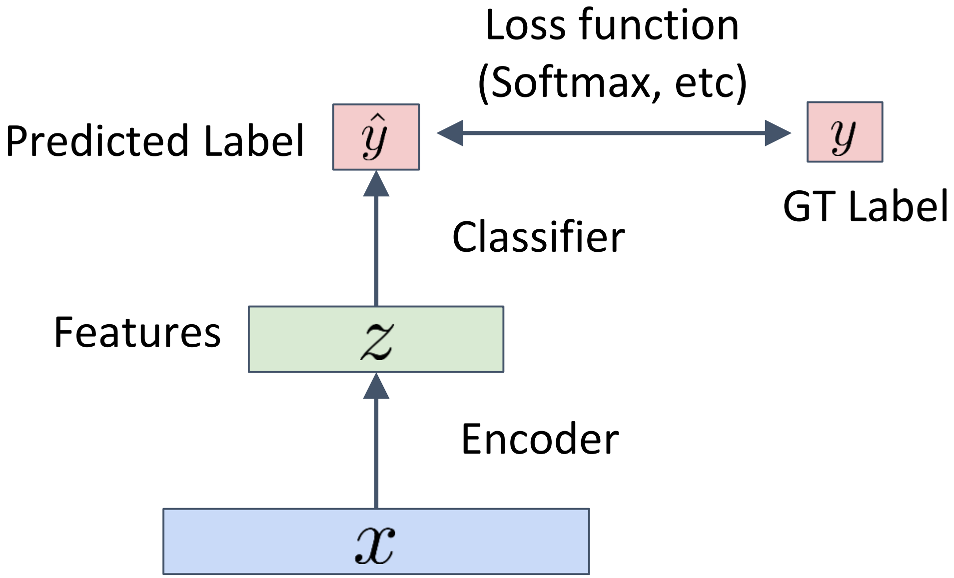
Example: Semi-Supervised Classification

- Many images, but few ground truth labels

start unsupervised
train autoencoder on many images



supervised fine-tuning
train classification network on labeled images



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Code example

Autoencoder
(autoencoder.ipynb)

Generative Models

- Assumption: the dataset are samples from an unknown distribution $p_{\mathrm{data}}(x)$
- Goal: create a new sample from $p_{\mathrm{data}}(x)$ that is not in the dataset

Dataset

Generated

Image credit: *Progressive Growing of GANs for Improved Quality, Stability, and Variation,* Karras et al.

Generative Models

- Assumption: the dataset are samples from an unknown distribution $p_{\text{data}}(x)$
- Goal: create a new sample from $p_{\text{data}}(x)$ that is not in the dataset

Dataset

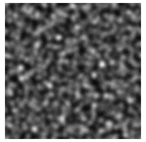
Generated

Image credit: *Progressive Growing of GANs for Improved Quality, Stability, and Variation*, Karras et al.

Generative Models

$$\boxed{p_\theta(x)} \approx p_{\text{data}}(x)$$

$\uparrow$ Generator with parameters θ



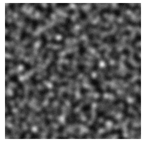
$$\boxed{p(z)}$$

known and
easy to sample from

Generative Models

$p_\theta(x) \approx p_{\text{data}}(x)$

Generator with parameters θ



$p(z)$

known and
easy to sample from

How to measure similarity of $p_\theta(x)$ and $p_{\text{data}}(x)$?

1) Likelihood of data in $p_\theta(x)$

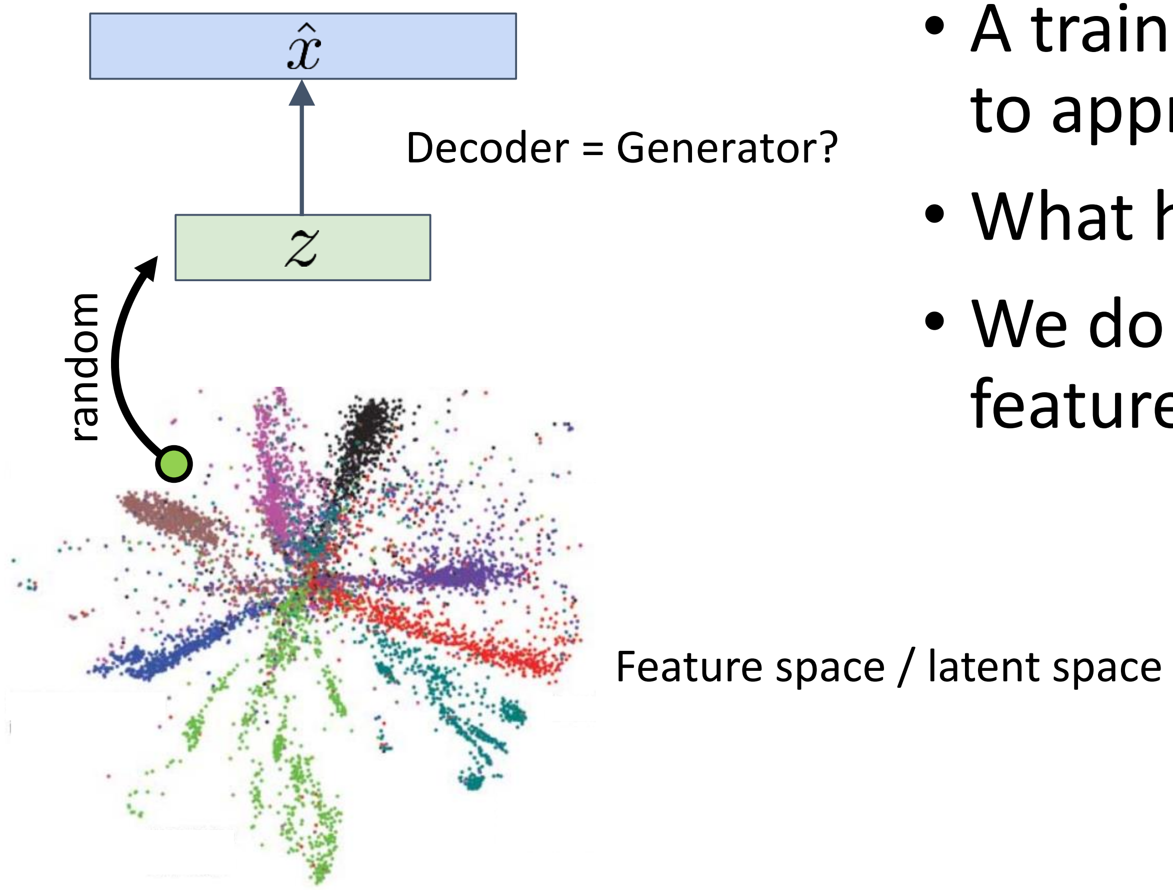
Variational Autoencoders (VAEs)

2) Adversarial game:

Discriminator distinguishes $p_\theta(x)$ and $p_{\text{data}}(x)$ vs *Generator* makes it hard to distinguish

Generative Adversarial Networks (GANs)

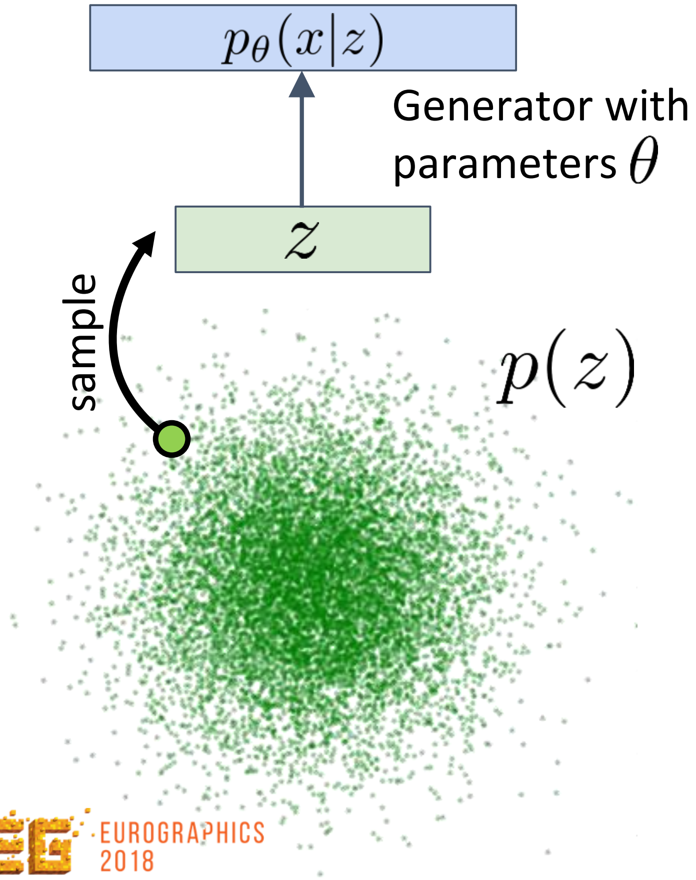
Autoencoders as Generative Models?



- A trained decoder transforms some features z to approximate samples from $p_{\text{data}}(x)$
- What happens if we pick a random z?
- We do not know the distribution $p(z)$ of features that decode to likely samples

Image Credit: *Reducing the Dimensionality of Data with Neural Networks*, Hinton and Salakhutdinov

Variational Autoencoders (VAEs)



- Pick a parametric distribution $p(z)$ for features
- The generator maps $p(z)$ to an image distribution $p_\theta(x)$ (where θ are parameters)

$$p_\theta(x) = \int p_\theta(x|z)\ p(z)\ dz$$

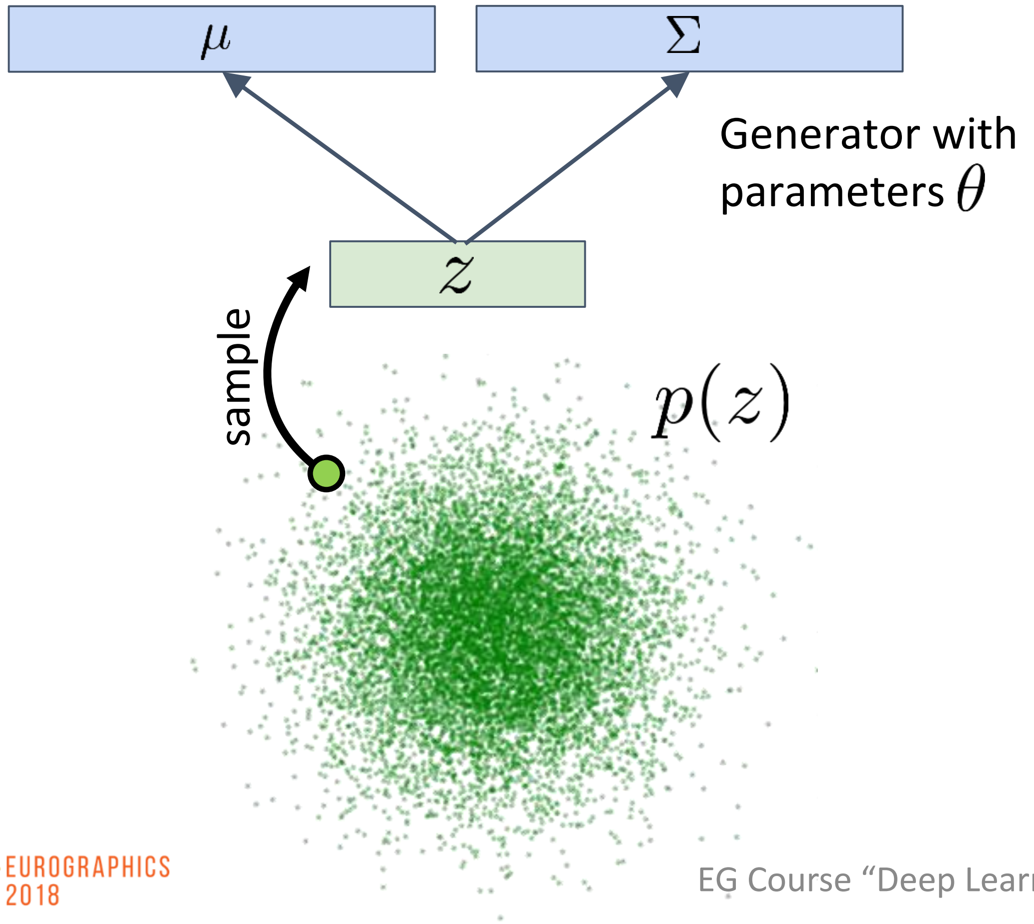
- Train the generator to maximize the likelihood of the data in $p_\theta(x)$:

$$\max_\theta \sum_{x \in \text{data}} \log p_\theta(x)$$

Outputting a Distribution

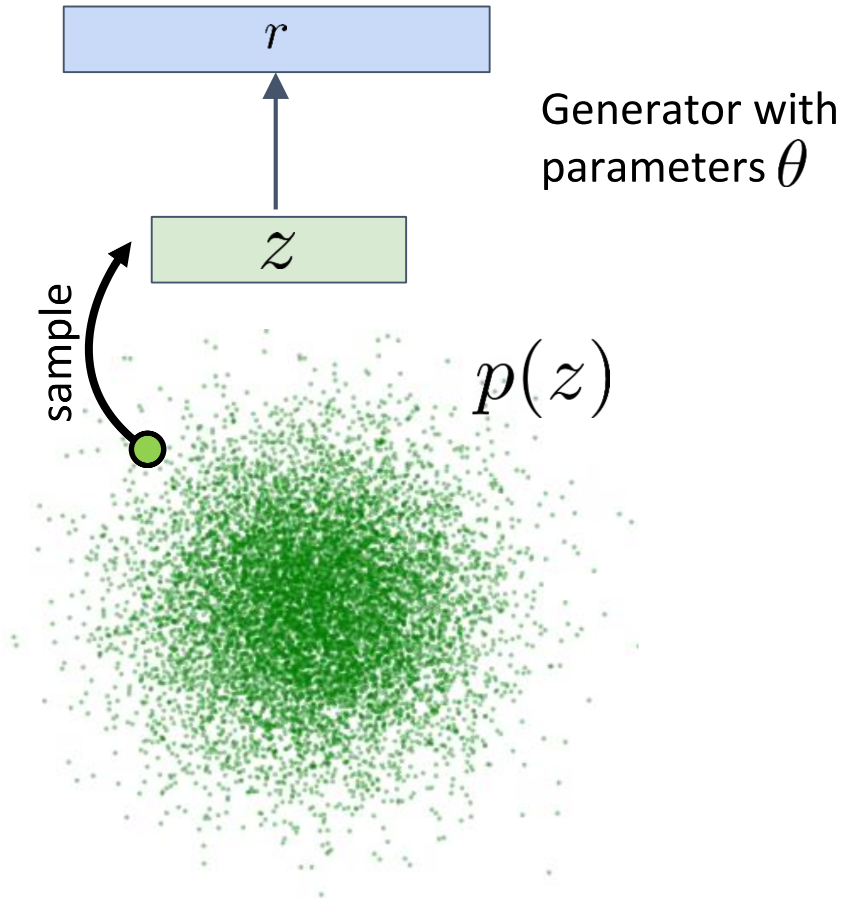
Normal distribution

$$p_\theta(x|z) = N(x; \mu(z), \Sigma(z))$$



Bernoulli distribution

$$p_\theta(x|z) = Bern(x; r(z))$$



Variational Autoencoders (VAEs): Naïve Sampling (Monte-Carlo)

$$p_\theta(x) = \int p_\theta(x|z)\ p(z)\ dz$$

$$\max_\theta \sum_{x \in \text{data}} \log p_\theta(x)$$

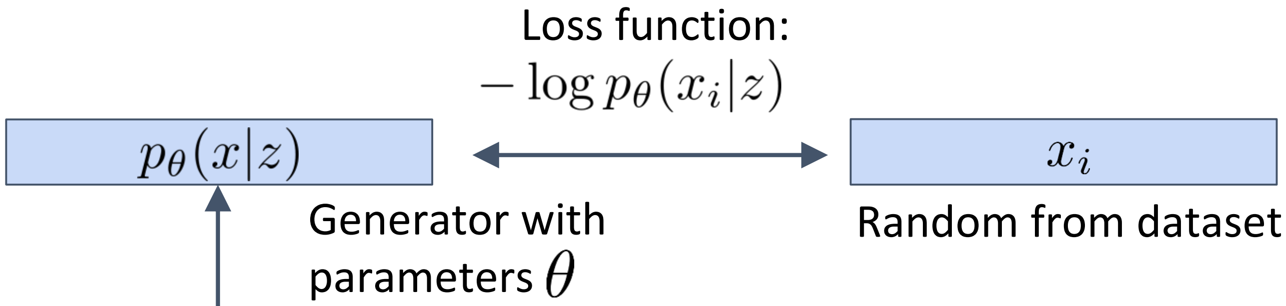
$$\theta^* = \arg\max_\theta \sum_{x \in \text{data}} \log \int p_\theta(x|z)\ p(z)\ dz$$

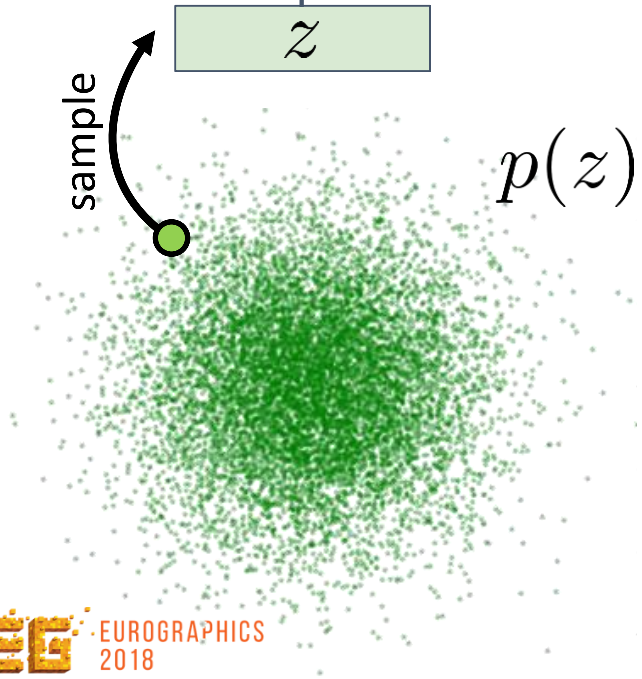
$$\theta^* \approx \arg\max_\theta \mathbb{E}_{x_i \sim p_{\text{data}}(x)} \mathbb{E}_{z \sim p(z)} \log p_\theta(x_i|z)$$

- SGD approximates the expected values over (z, x_i) samples
- In each training iteration, sample z from $p(z)$ …
- … and x_i randomly from the dataset, and maximize:

$$\max_\theta \log p_\theta(x_i|z)$$

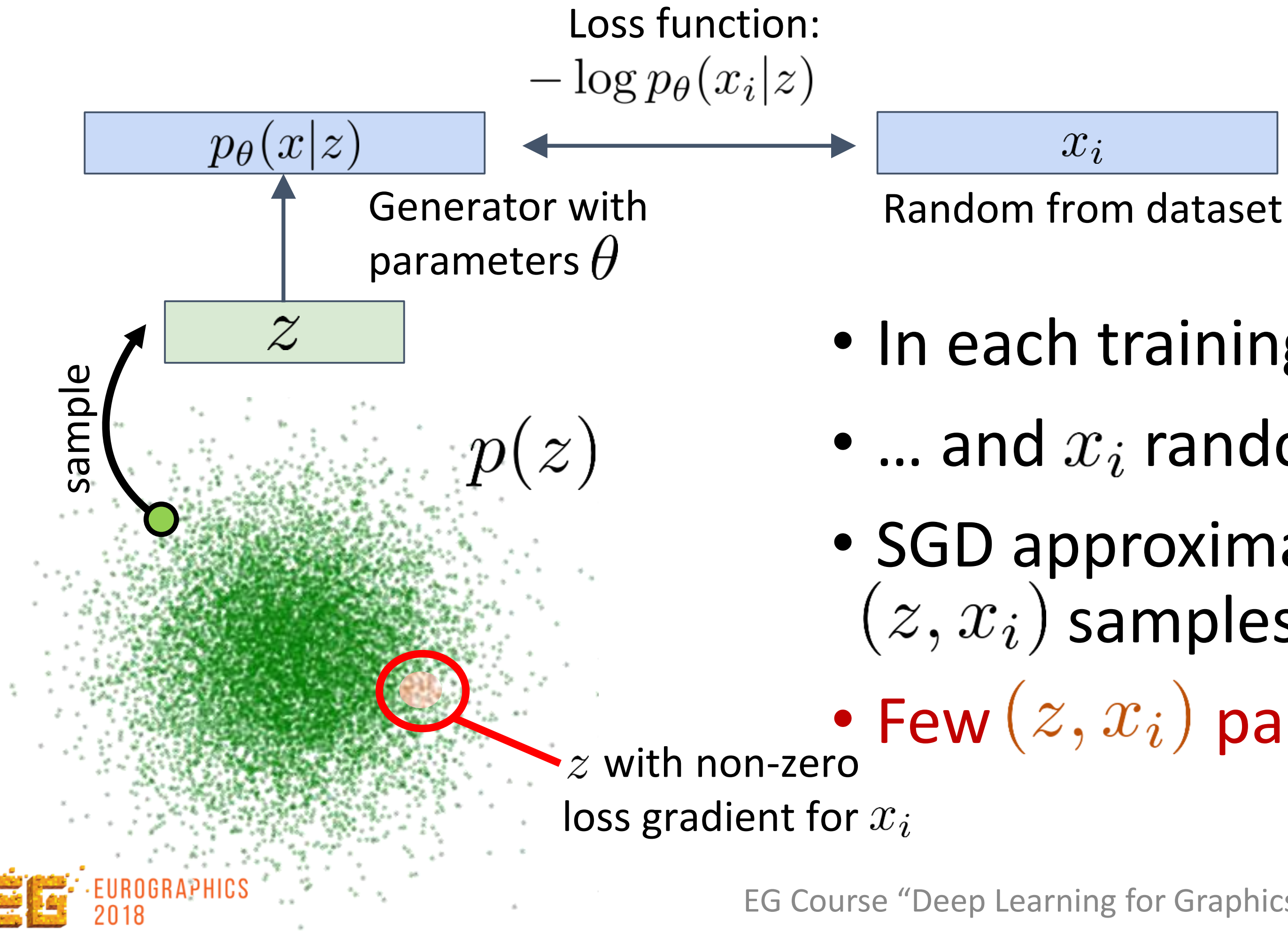
Variational Autoencoders (VAEs): Naïve Sampling (Monte-Carlo)





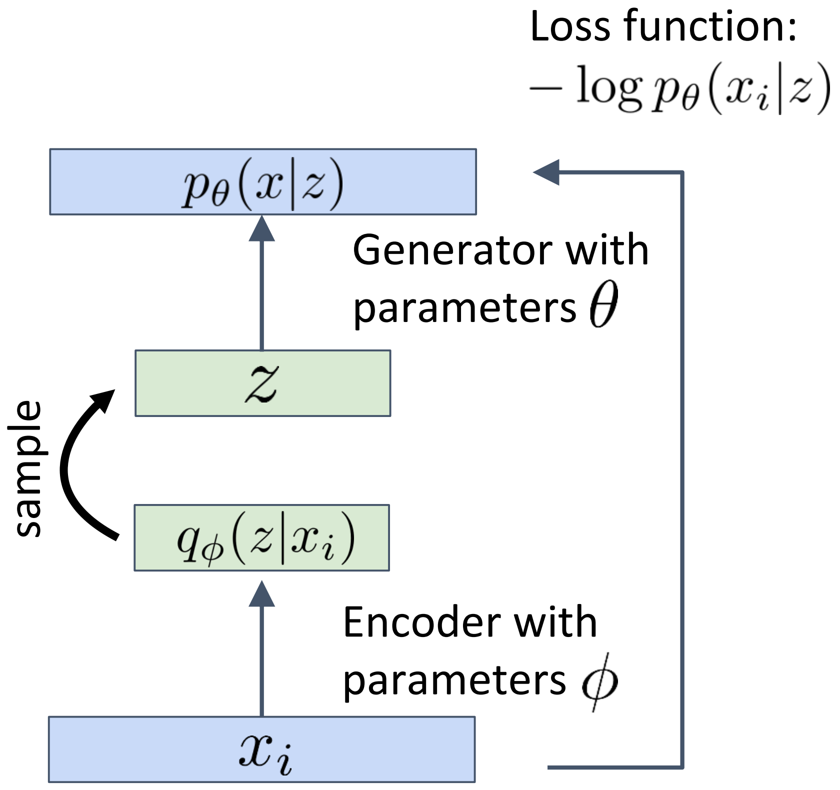
- In each training iteration, sample z from $p(z)$ …
- … and x_i randomly from the dataset
- SGD approximates the expected values over (z, x_i) samples

Variational Autoencoders (VAEs): Naïve Sampling (Monte-Carlo)



- In each training iteration, sample z from $p(z)$ …
- … and x_i randomly from the dataset
- SGD approximates the expected values over (z, x_i) samples
- Few (z, x_i) pairs have non-zero gradients

Variational Autoencoders (VAEs): The Encoder



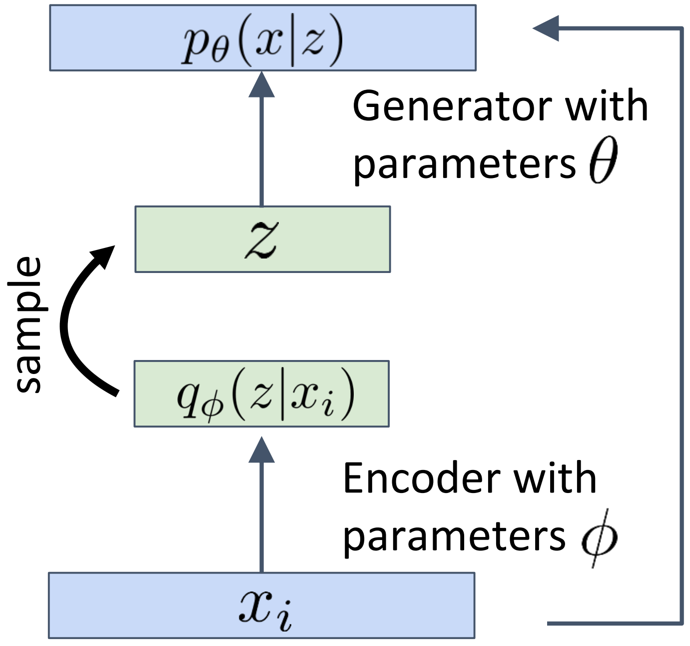
$$p_\theta(x) = \int p_\theta(x|z)\ p(z)\ dz$$

- During training, another network can guess a good z for a given x_i
- $q_\phi(z|x_i)$ should be much smaller than $p(z)$
- This also gives us the data point x_i

Variational Autoencoders (VAEs): The Encoder

Loss function:

$$-\log p_\theta(x_i|z) + KL(\ q_\phi(z|x_i)\ \|\ p(z)\ )$$



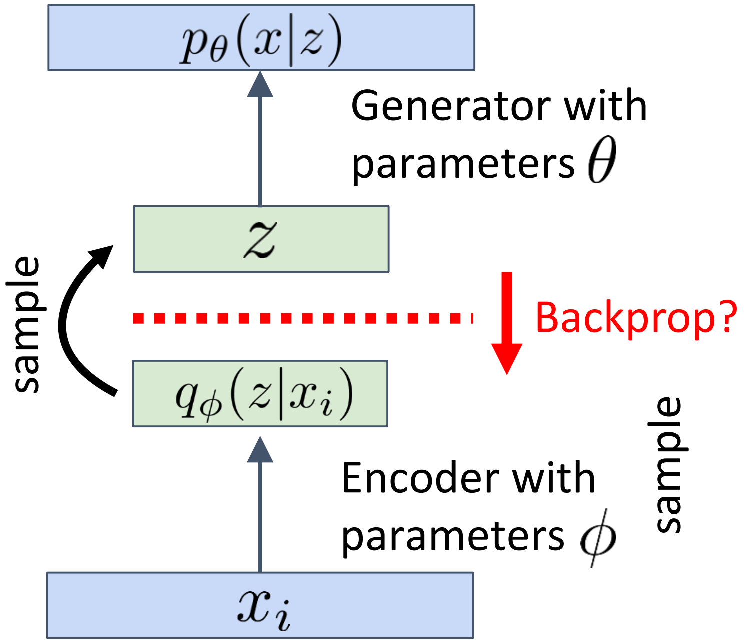
- Can we still easily sample a new z?
- Need to make sure $q_\phi(z|x_i)$ approximates $p(z)$
- Regularize with KL-divergence
- Negative loss can be shown to be a lower bound for the likelihood, and equivalent if $q_\phi(z|x) = p_\theta(z|x)$

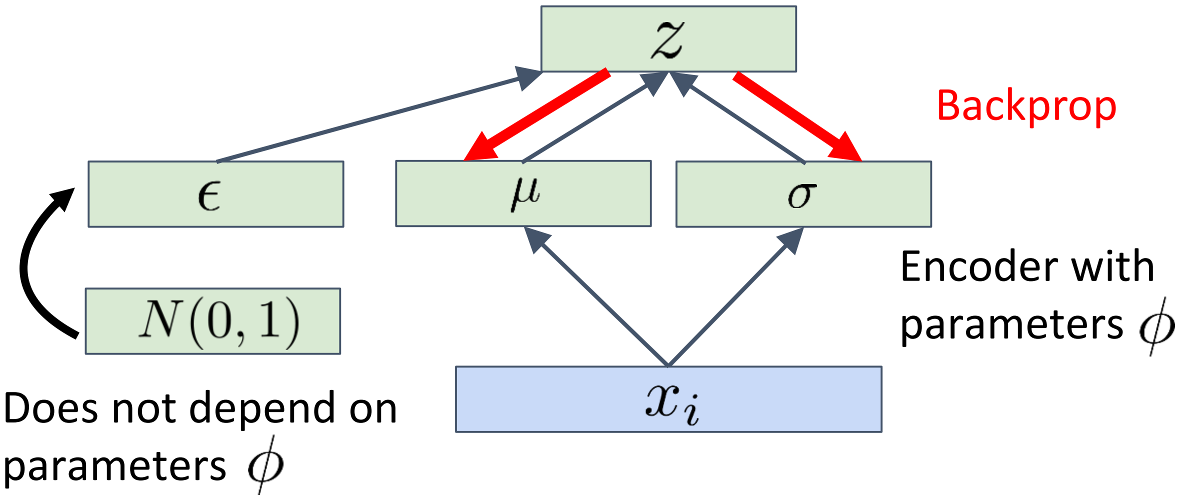
Reparameterization Trick

Example when $q_\phi(z|x_i) = N(z; \mu(x_i), \sigma(x_i))$:

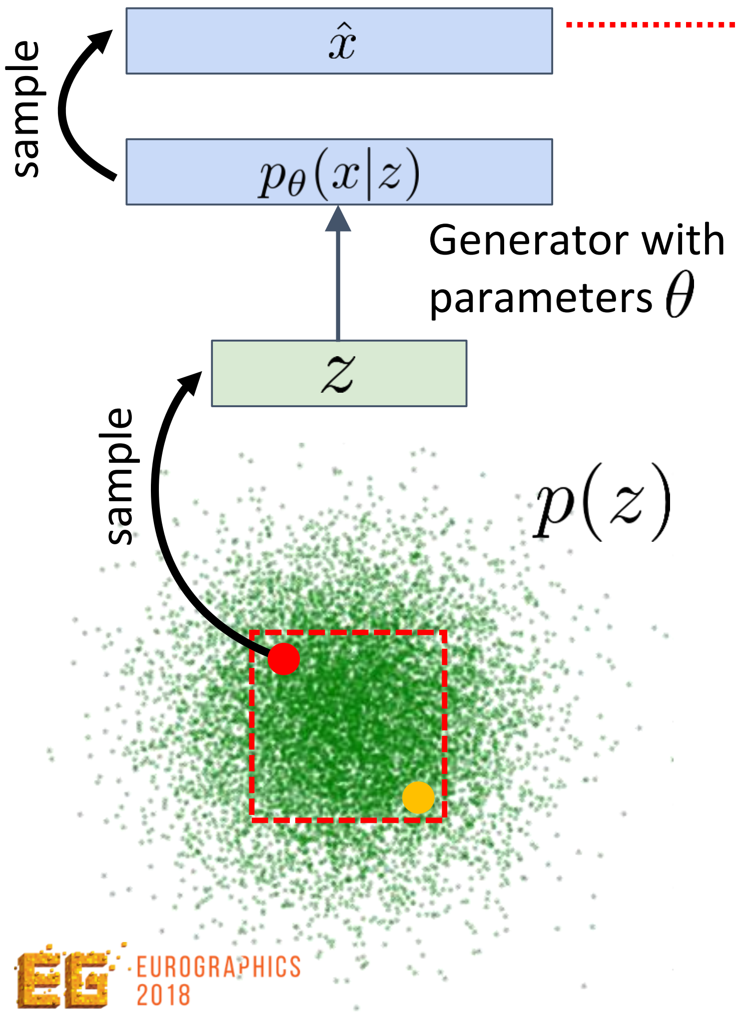
$z = \sigma + \mu \cdot \epsilon$, where $\epsilon \sim N(0,1)$

$$\frac{\partial z}{\partial \phi} = \frac{\partial \mu}{\partial \phi} + \frac{\partial \sigma}{\partial \phi} \cdot \epsilon$$

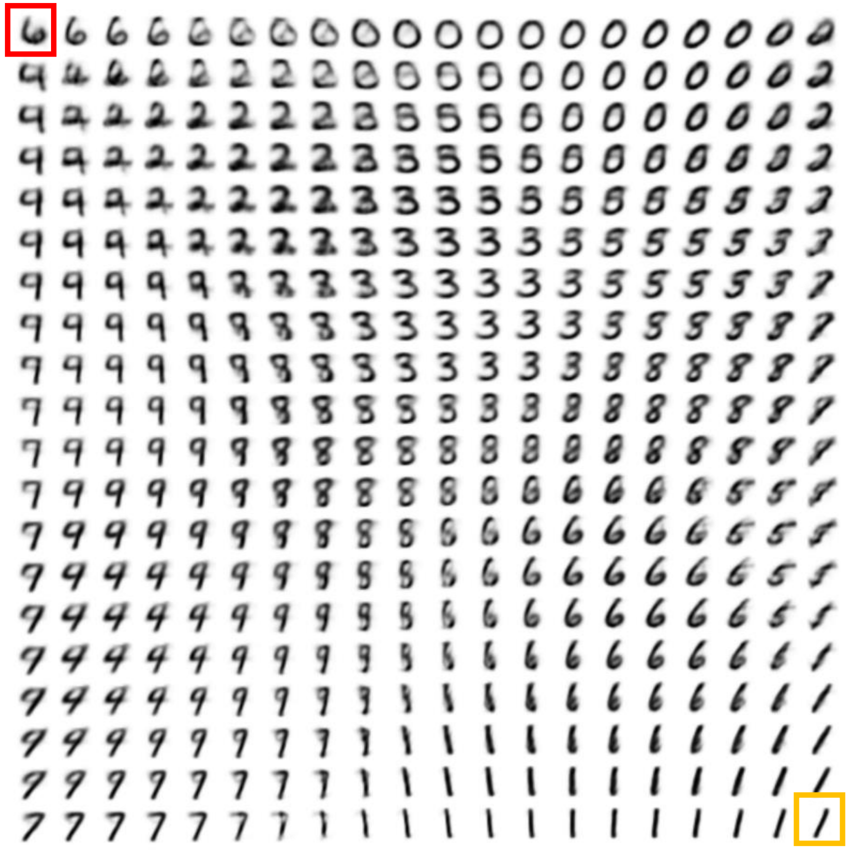




Generating Data



MNIST



Frey Faces

Image Credit: *Auto-Encoding Variational Bayes*, Kingma and Welling

Demos

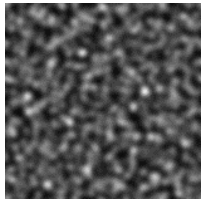
VAE on MNIST
http://dpkingma.com/sgvb_mnist_demo/demo.html

VAE on Faces
http://vdumoulin.github.io/morphing_faces/online_demo.html

Code example

Variational Autoencoder
(variational_autoencoder.ipynb)

Generative Adversarial Networks

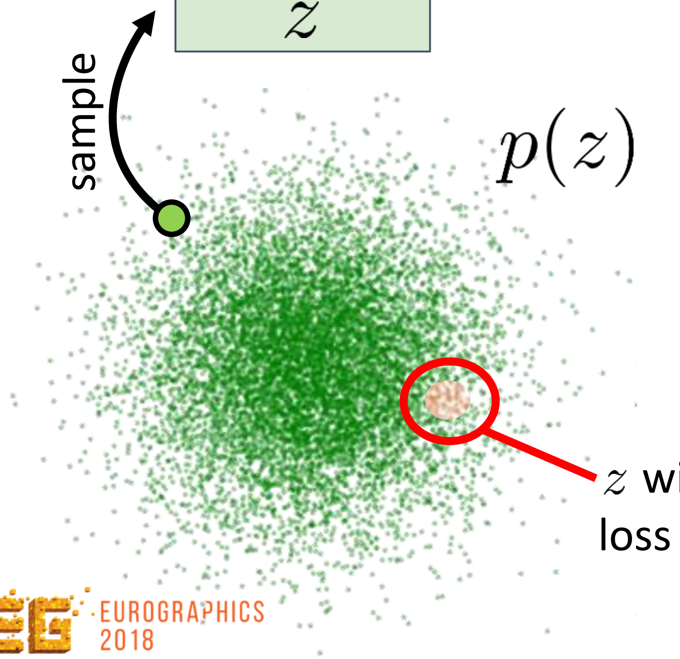


Player 1: generator
Scores if discriminator can't distinguish output from real image

from dataset

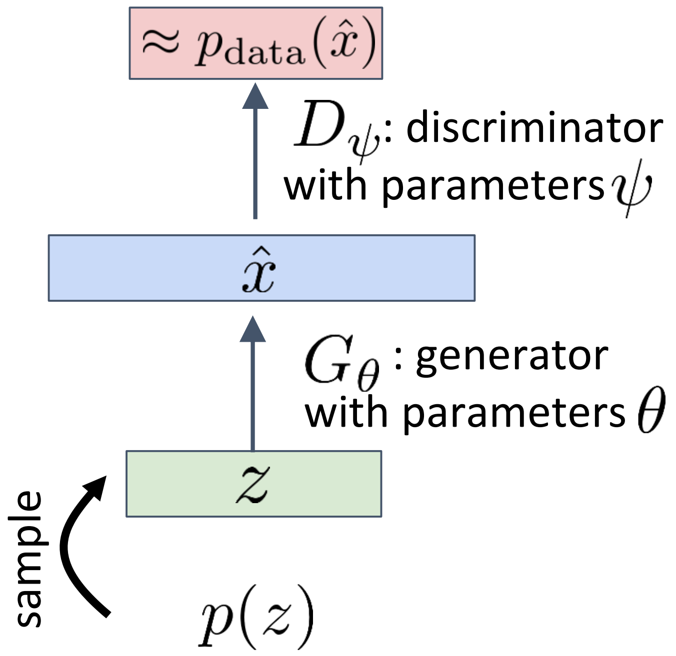
Player 2: discriminator → real/fake
Scores if it can distinguish between real and fake

Naïve Sampling Revisited



- Few (z, x_i) pairs have non-zero gradients
- This is a problem of the maximum likelihood
- Use a different loss: Train a discriminator network to measure similarity $p_\theta(x) \approx p_{\text{data}}(x)$

Why Adversarial?



- If discriminator approximates $p_{\text{data}}(x)$:
- x^*at maximum of $p_{\text{data}}(x)$ has lowest loss
- Optimal $p_\theta(x)$ has single mode at x^*, small variance

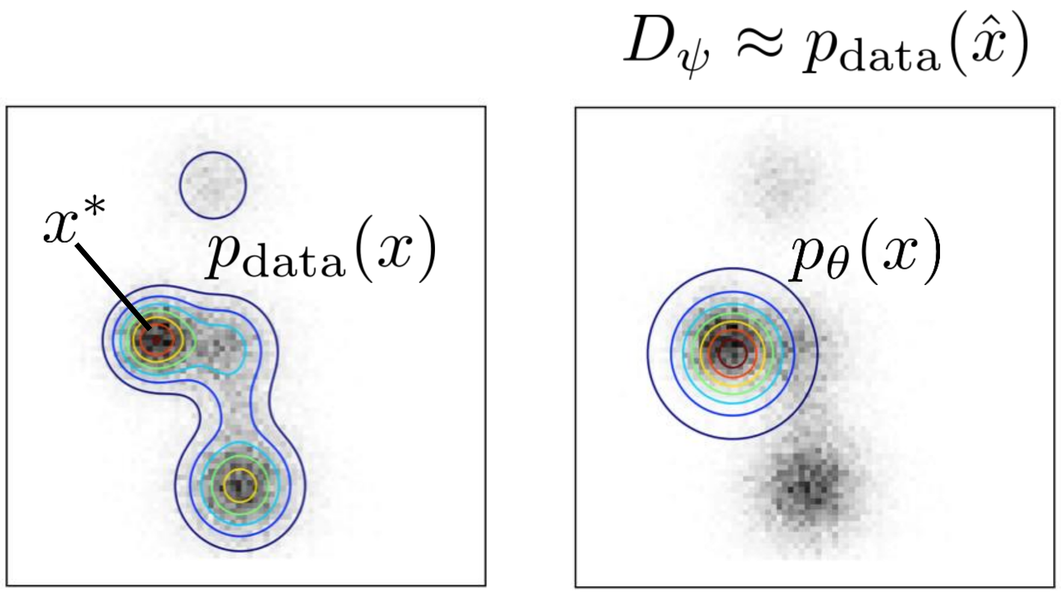
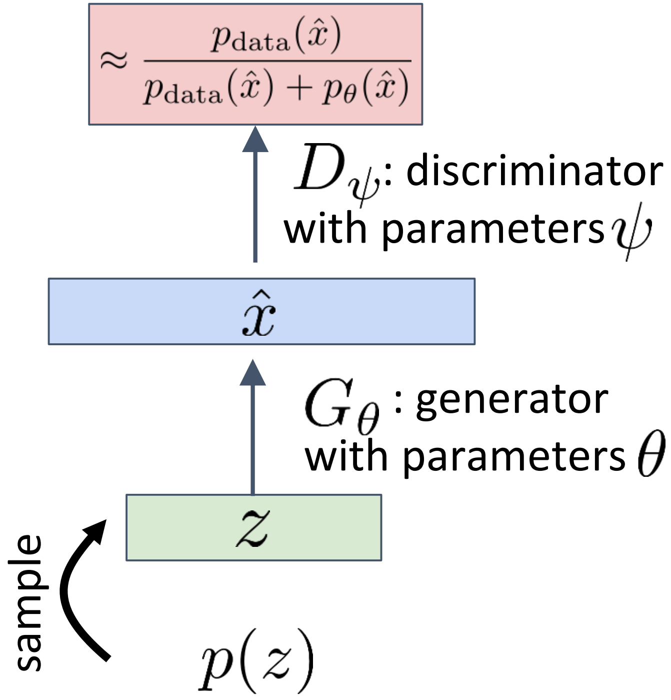


Image Credit: *How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary?*, Ferenc Huszár

Why Adversarial?



- For GANs, the discriminator instead approximates:

$$\frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_\theta(x)}$$

depends on the generator

$D_\psi \approx p_{\text{data}}(\hat{x})$ $\quad D_\psi \approx \frac{p_{\text{data}}(x)}{p_{\text{data}}(x)+p_\theta(x)}$

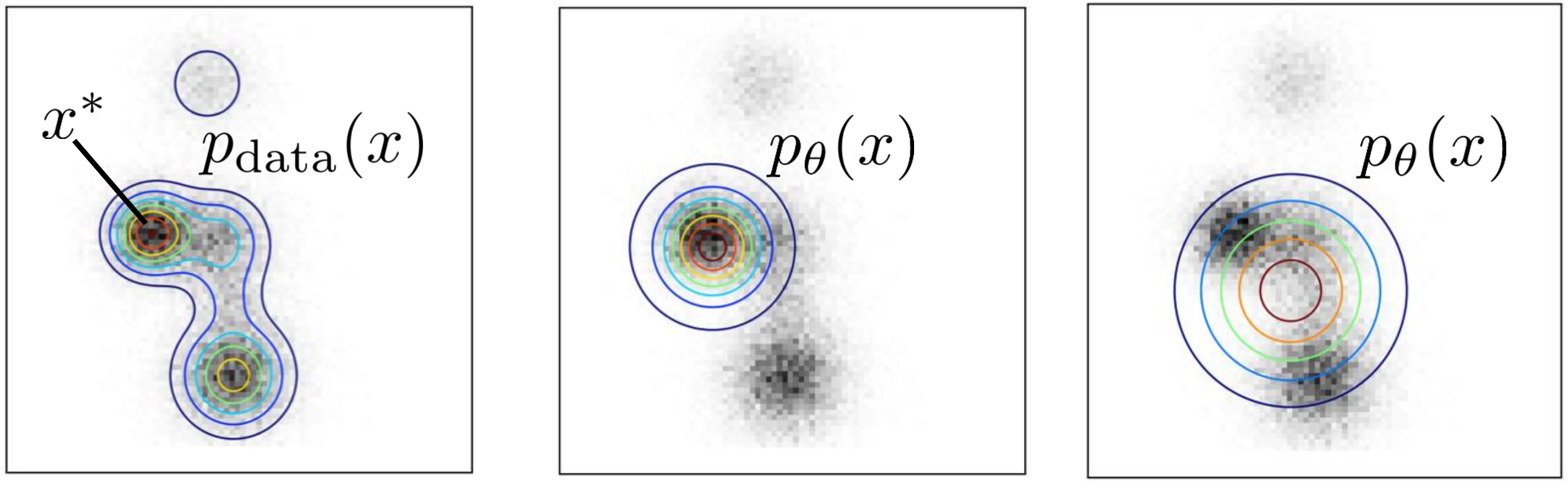
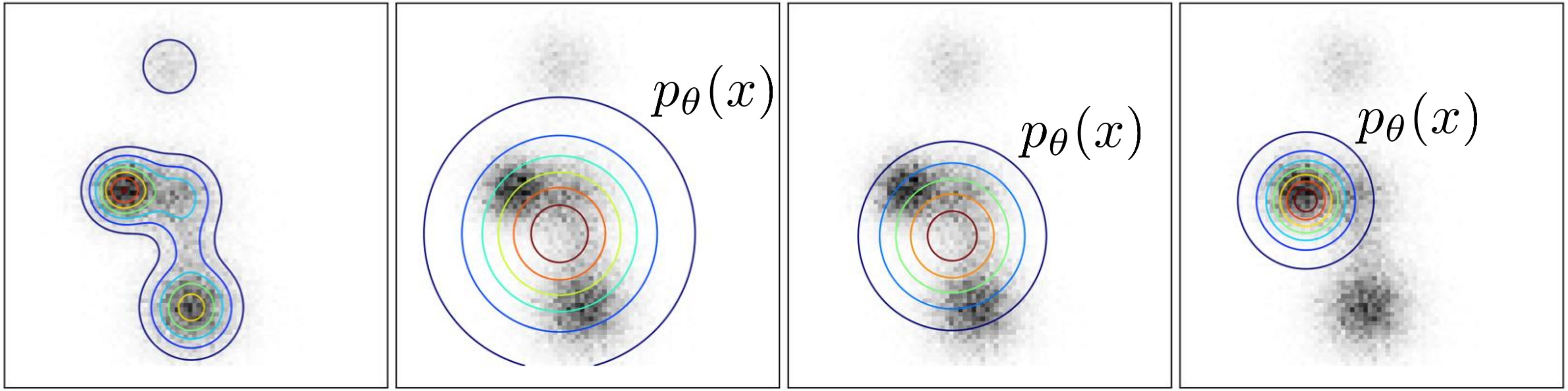


Image Credit: *How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary?*, Ferenc Huszár

Why Adversarial?



$p_{\text{data}}(x)$

VAEs:
Maximize likelihood of **data samples** in $p_\theta(x)$

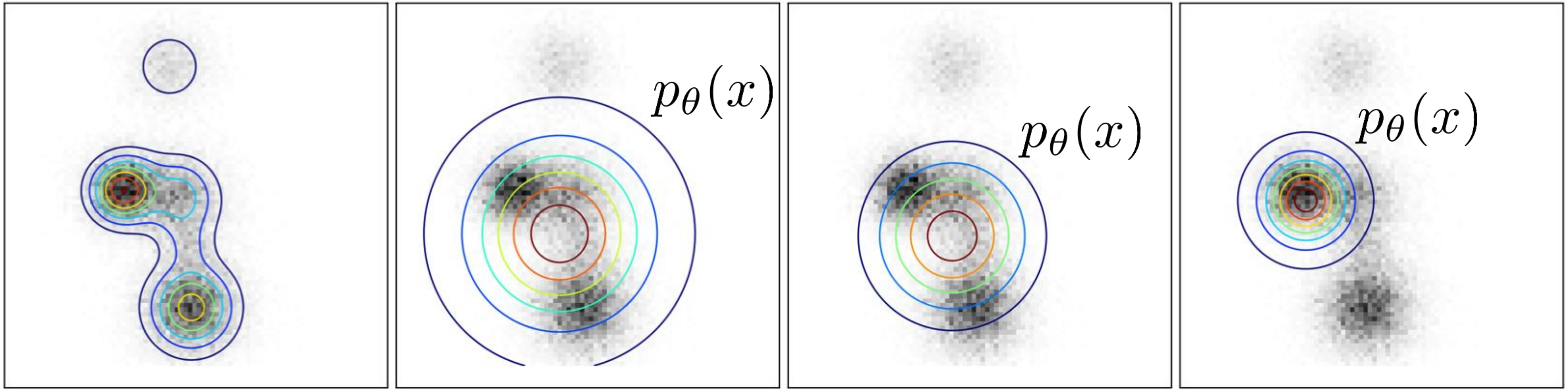
GANs:
Adversarial game

Maximize likelihood of **generator samples** in approximate $p_{\text{data}}(x)$

Image Credit: *How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary?*, Ferenc Huszár

Why Adversarial?

$\approx KL(p_{\text{data}} \parallel p_\theta)$ $\approx JS(p_{\text{data}} \parallel p_\theta)$ $\approx KL(p_\theta \parallel p_{\text{data}})$



$p_{\text{data}}(x)$

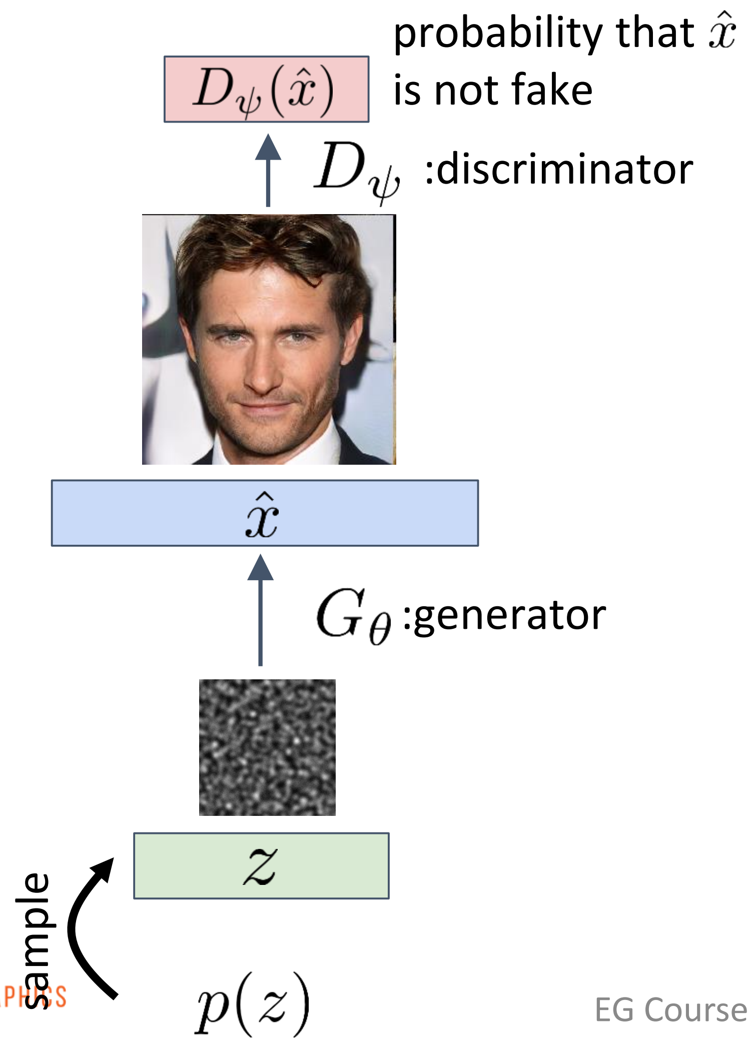
VAEs:
Maximize likelihood of **data samples** in $p_\theta(x)$

GANs:
Adversarial game

Maximize likelihood of **generator samples** in approximate $p_{\text{data}}(x)$

Image Credit: *How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary?*, Ferenc Huszár

GAN Objective



fake/real classification loss (BCE):

$$\begin{aligned} L(\theta, \psi) = & -0.5\ \mathbb{E}_{x \sim p_{\text{data}}} \log D_\psi(x) \\ & -0.5\ \mathbb{E}_{x \sim p_\theta} \log(1 - D_\psi(x)) \end{aligned}$$

Discriminator objective:

$$\min_\psi L(\theta, \psi)$$

Generator objective:

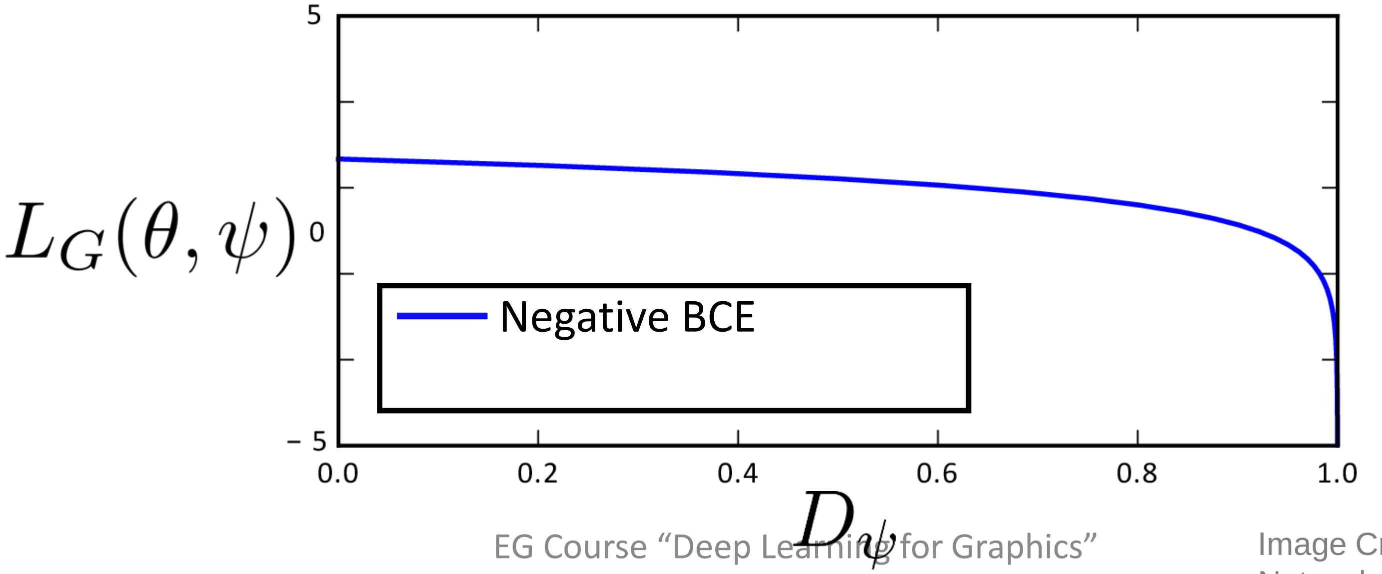
$$\max_\theta L(\theta, \psi)$$

Non-saturating Heuristic

$$\begin{aligned} L(\theta,\psi) = & -0.5\ \mathbb{E}_{x\sim p_{\text{data}}}\ \log D_\psi(x) \\ & -0.5\ \mathbb{E}_{x\sim p_\theta}\ \log(1-D_\psi(x)) \end{aligned}$$

Generator loss is negative binary cross-entropy:

$$L_G(\theta,\psi) = 0.5\ \mathbb{E}_{x\sim p_\theta}\ \log(1-D_\psi(x))$$ poor convergence



EUROGRAPHICS 2018

Image Credit: NIPS 2016 Tutorial: Generative Adversarial Networks, Ian Goodfellow

Non-saturating Heuristic

Generator loss is negative binary cross-entropy:

$$L_G(\theta, \psi) = 0.5\ \mathbb{E}_{x \sim p_\theta}\ \log(1 - D_\psi(x))$$ poor convergence

Flip target class instead of flipping the sign for generator loss:

$$L_G(\theta, \psi) = -0.5\ \mathbb{E}_{x \sim p_\theta}\ \log D_\psi(x)$$ good convergence – like BCE

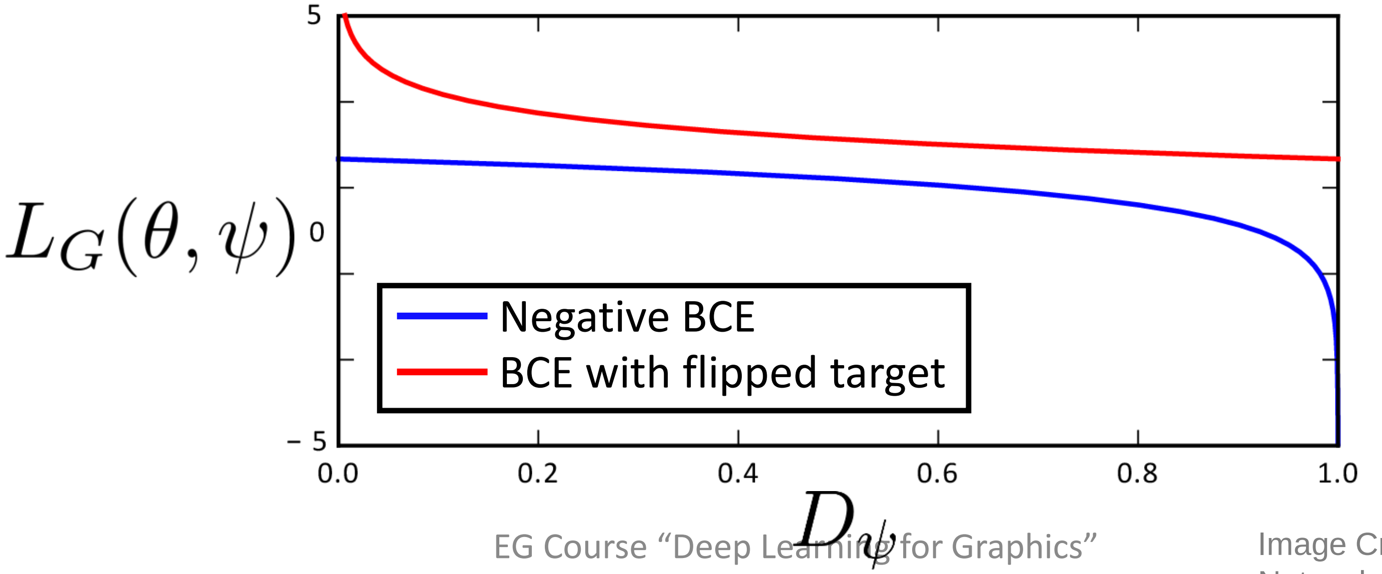


Image Credit: NIPS 2016 Tutorial: Generative Adversarial Networks, Ian Goodfellow

GAN Training

Generator training

Loss:

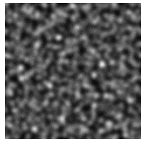
$$L_G(\theta, \psi) = -\log D_\psi(\hat{x})$$

$D_\psi(\hat{x})$

D_ψ :discriminator

$\hat{x}$

G_θ :generator



z

sample

$p(z)$

Discriminator training

Loss:

$$L_D(\theta, \psi) = -0.5 \log(1 - D_\psi(\hat{x})) - 0.5 \log D_\psi(x_i)$$

$D_\psi(\hat{x})$ $D_\psi(x_i)$

D_ψ :discriminator

$\hat{x}$

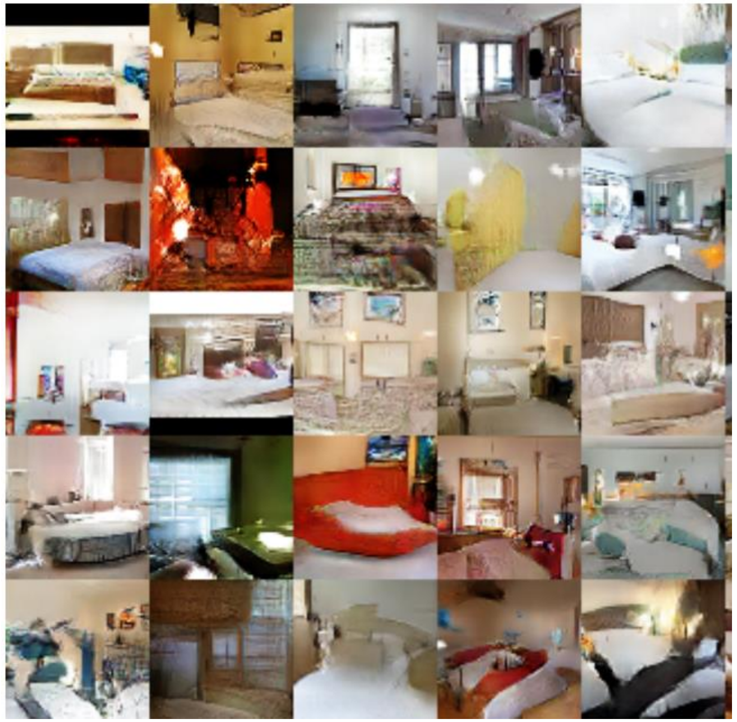
x_i

from dataset

Interleave in each training step

DCGAN

- First paper to successfully use CNNs with GANs
- Due to using novel components (at that time) like batch norm., ReLUs, etc.



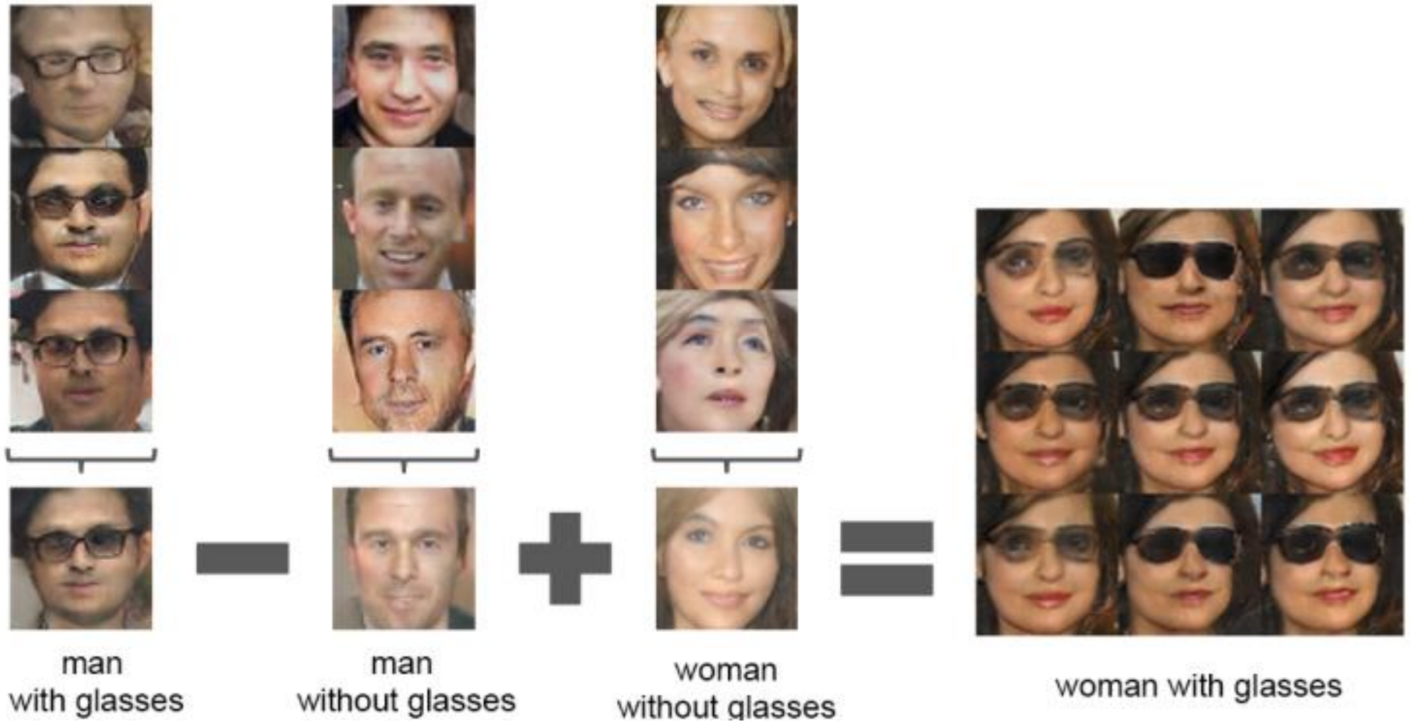
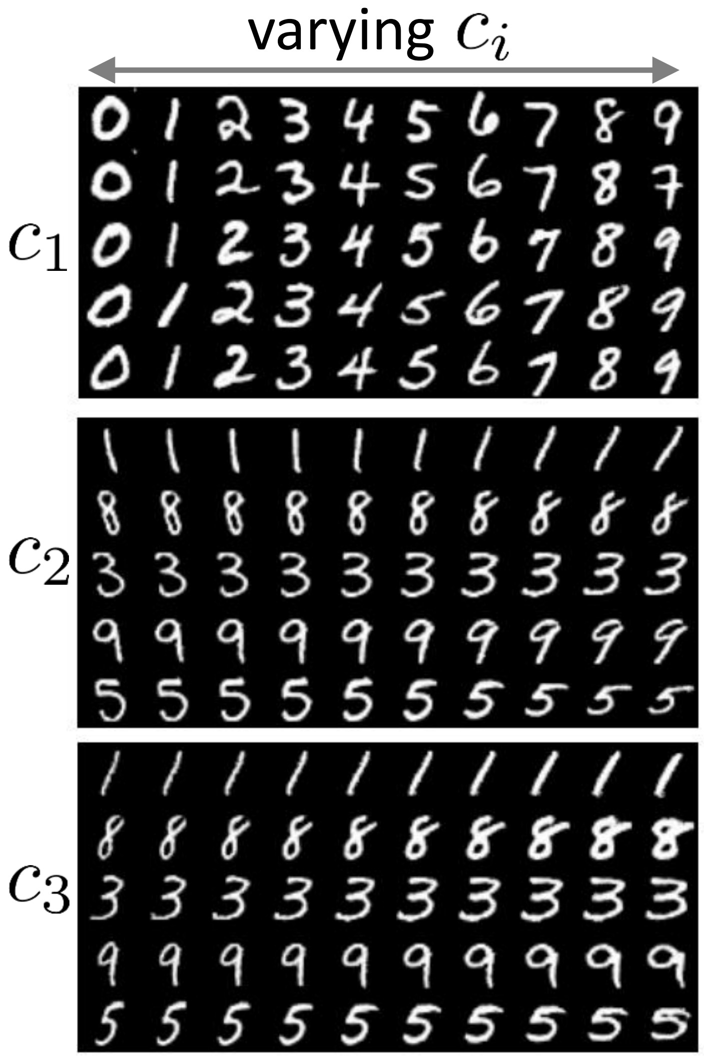


Image Credit: *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*, Radford et al.

InfoGAN



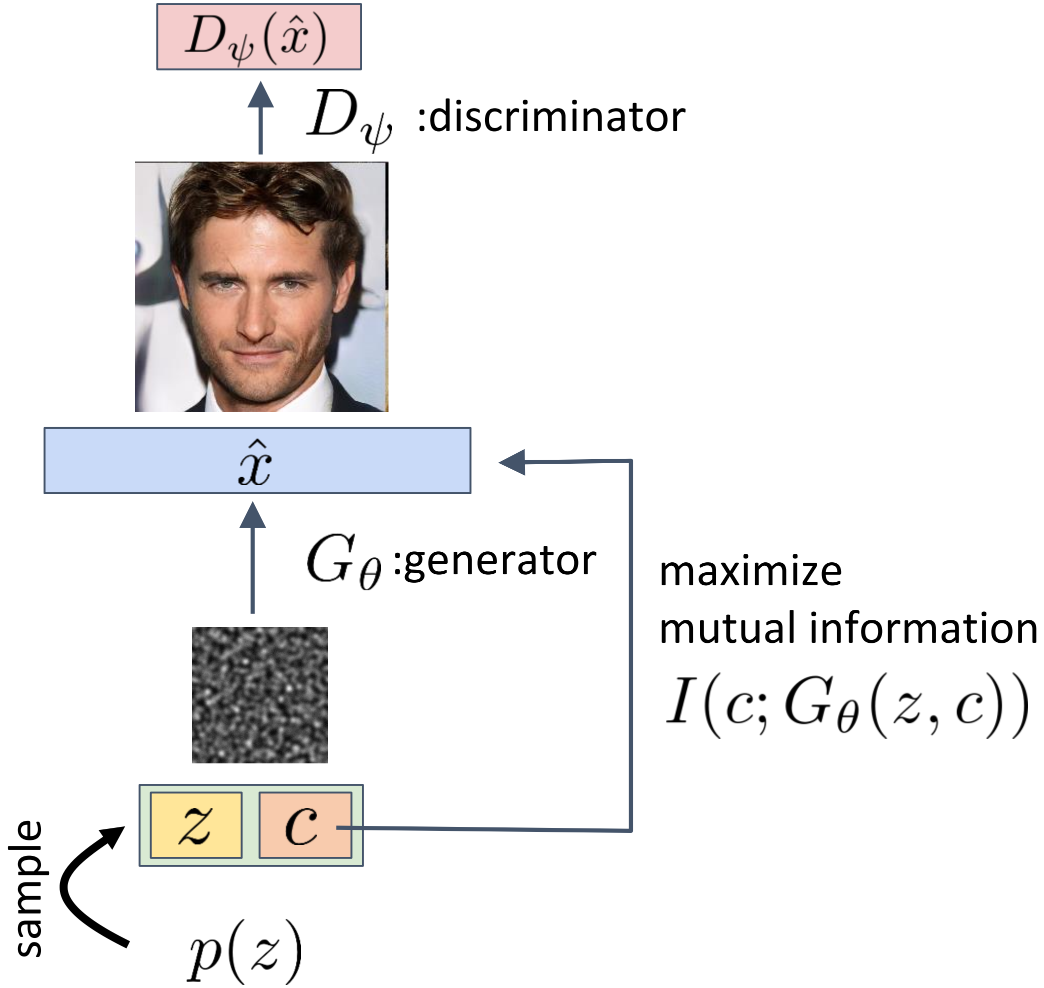


 Image Credit: *InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets*, Chen et al.

Code example

Generative Adversarial Network
(gan.ipynb)

Conditional GANs (CGANs)

- ≈ learn a mapping between images from example pairs
- Approximate sampling from a conditional distribution $p_{\text{data}}(x \mid c)$

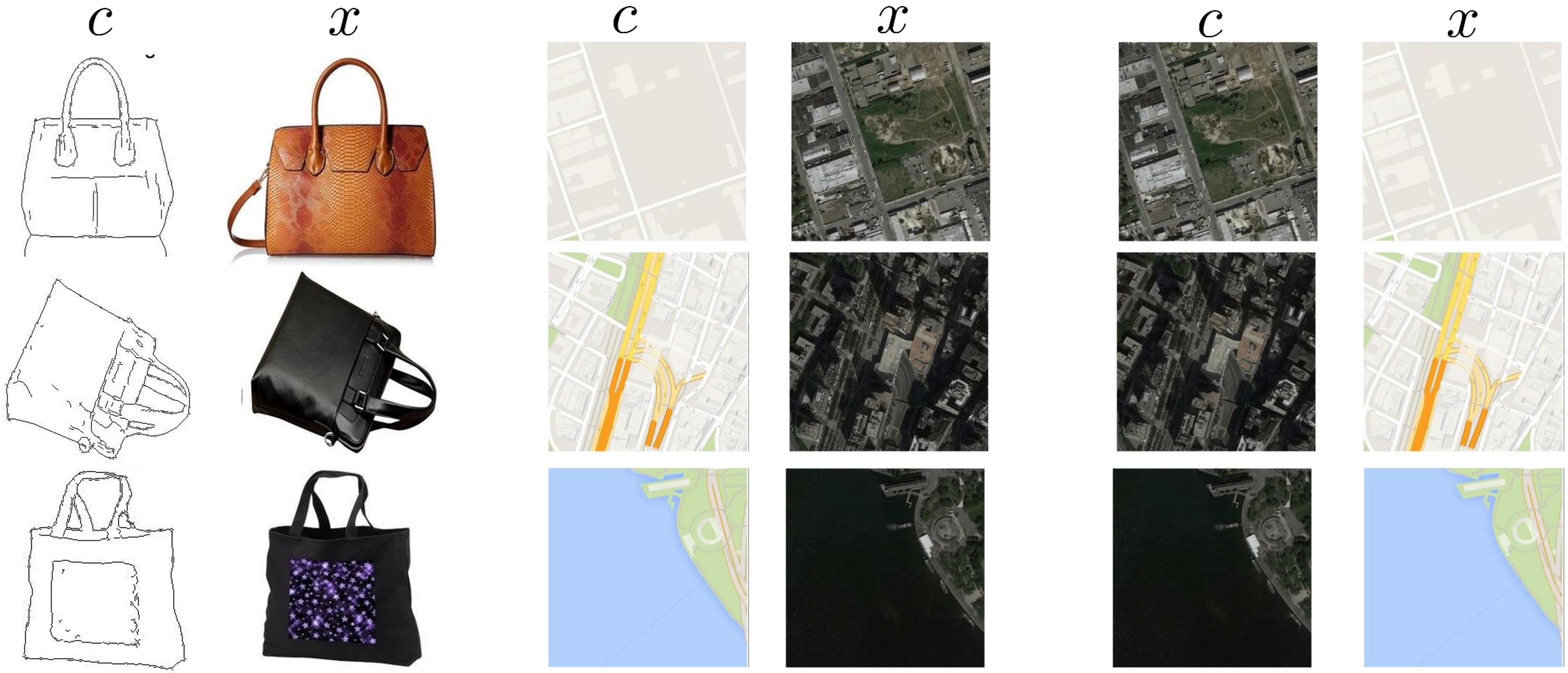


Image Credit: *Image-to-Image Translation with Conditional Adversarial Nets*, Isola et al.

Conditional GANs

Loss:

$$L_G(\theta, \psi) = -\log D_\psi(\hat{x}, c)$$

Loss:

$$L_D(\theta, \psi) = -0.5 \log(1 - D_\psi(\hat{x}, c)) - 0.5 \log D_\psi(x_i, c)$$

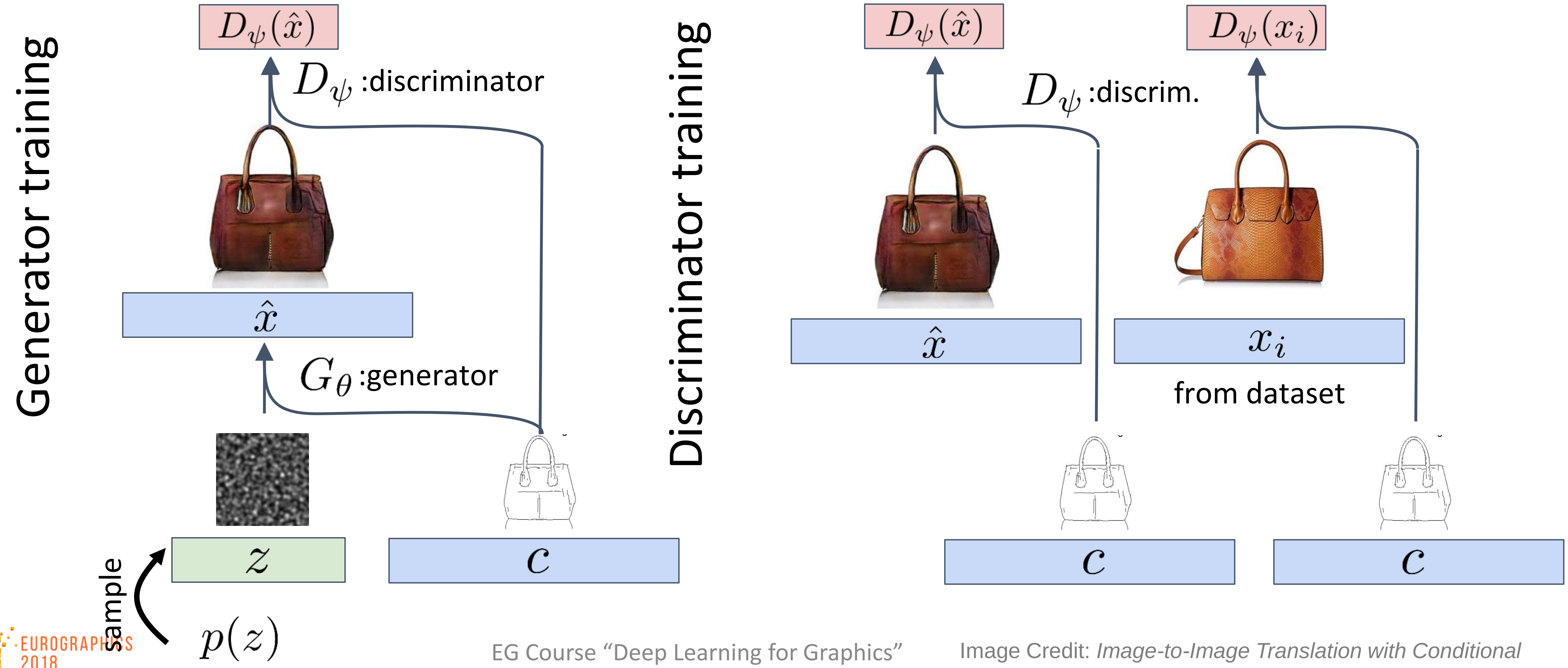


Image Credit: *Image-to-Image Translation with Conditional Adversarial Nets*, Isola et al.

Conditional GANs: Low Variation per Condition

Loss:

$$L_G(\theta, \psi) = -\log D_\psi(\hat{x}, c)$$

Loss:

$$L_D(\theta, \psi) = -0.5 \log(1 - D_\psi(\hat{x}, c)) - 0.5 \log D_\psi(x_i, c)$$

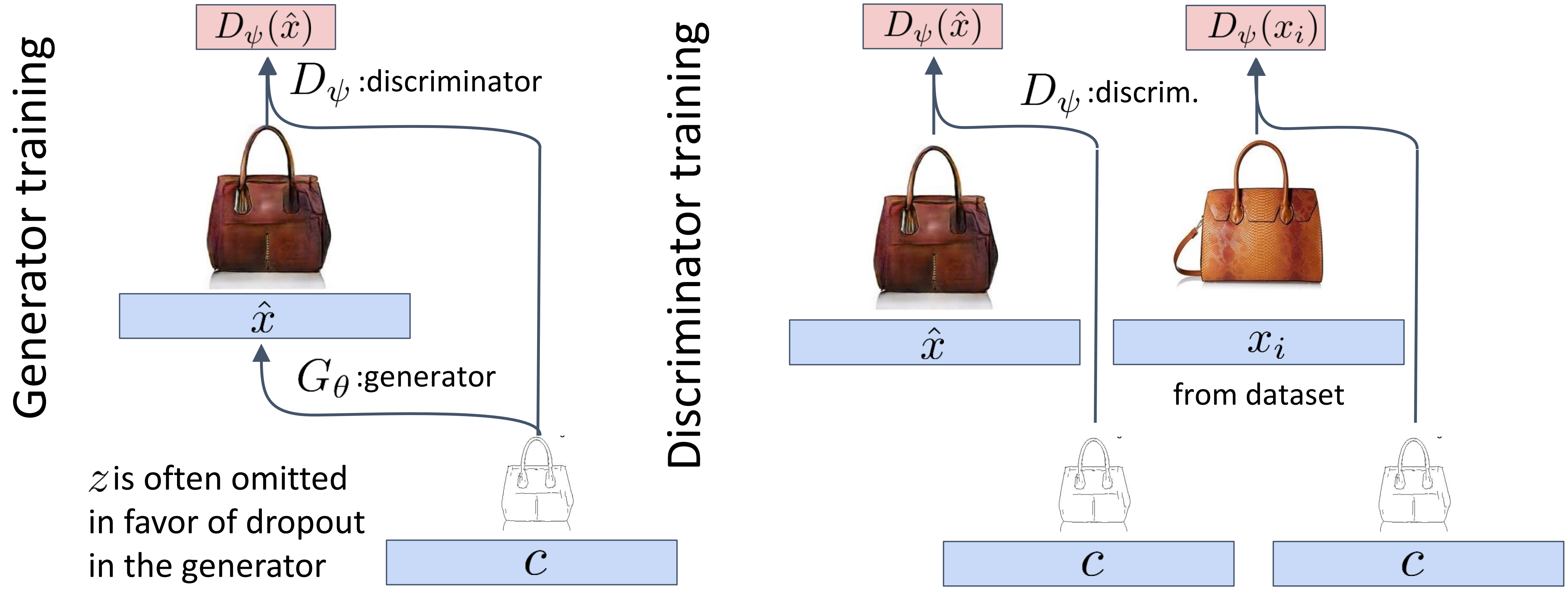


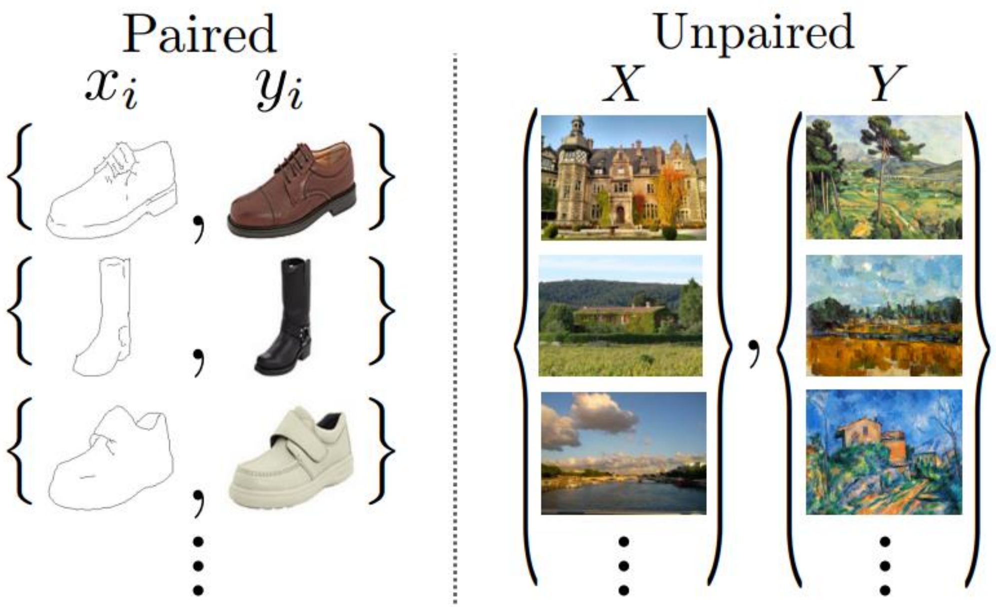
Image Credit: *Image-to-Image Translation with Conditional Adversarial Nets*, Isola et al.

Demos

CGAN
https://affinelayer.com/pixsrv/index.html

CycleGANs

- Less supervision than CGANs: mapping between unpaired datasets
- Two GANs + cycle consistency



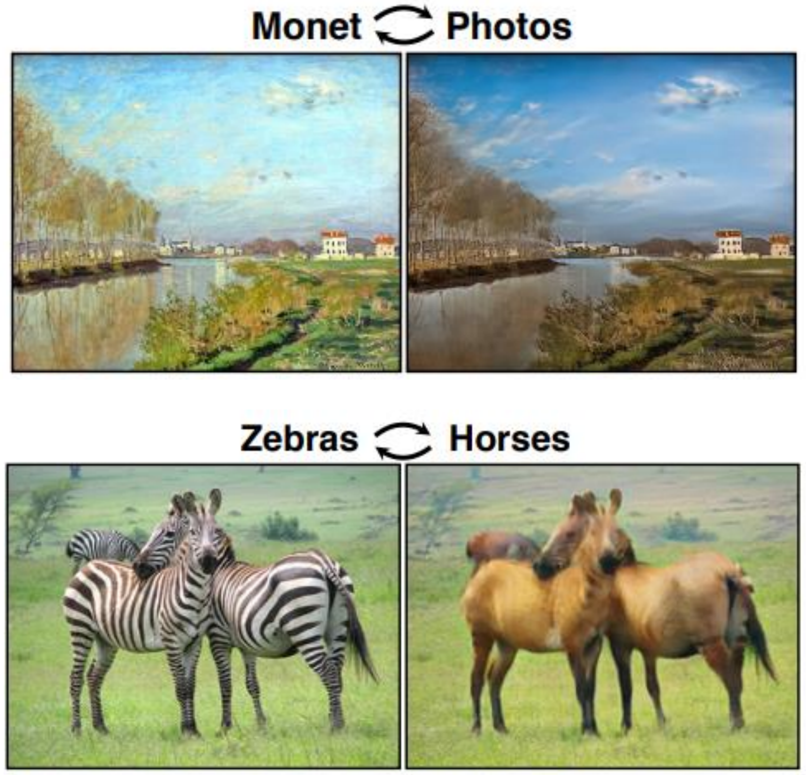


Image Credit: *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*, Zhu et al.

CycleGAN: Two GANs …

- Not conditional, so this alone does not constrain generator input and output to match

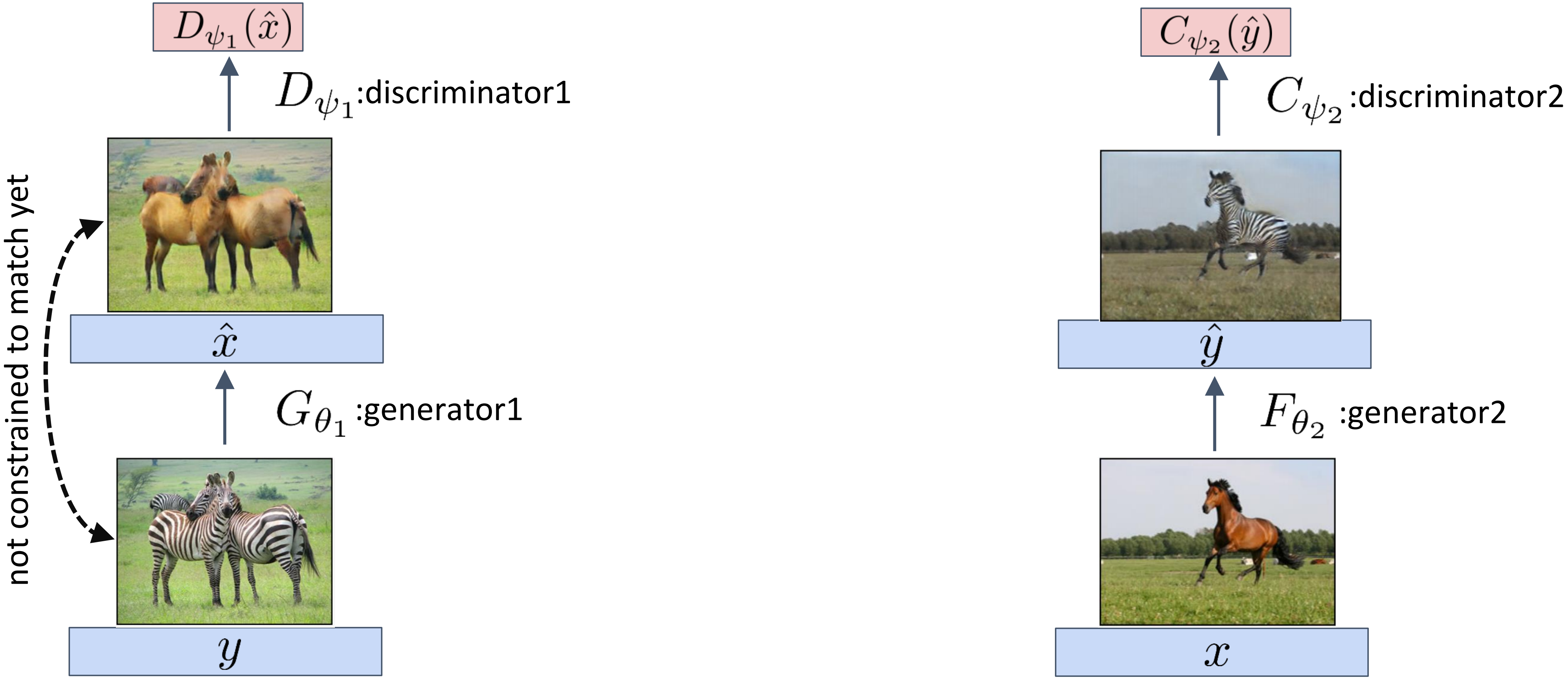
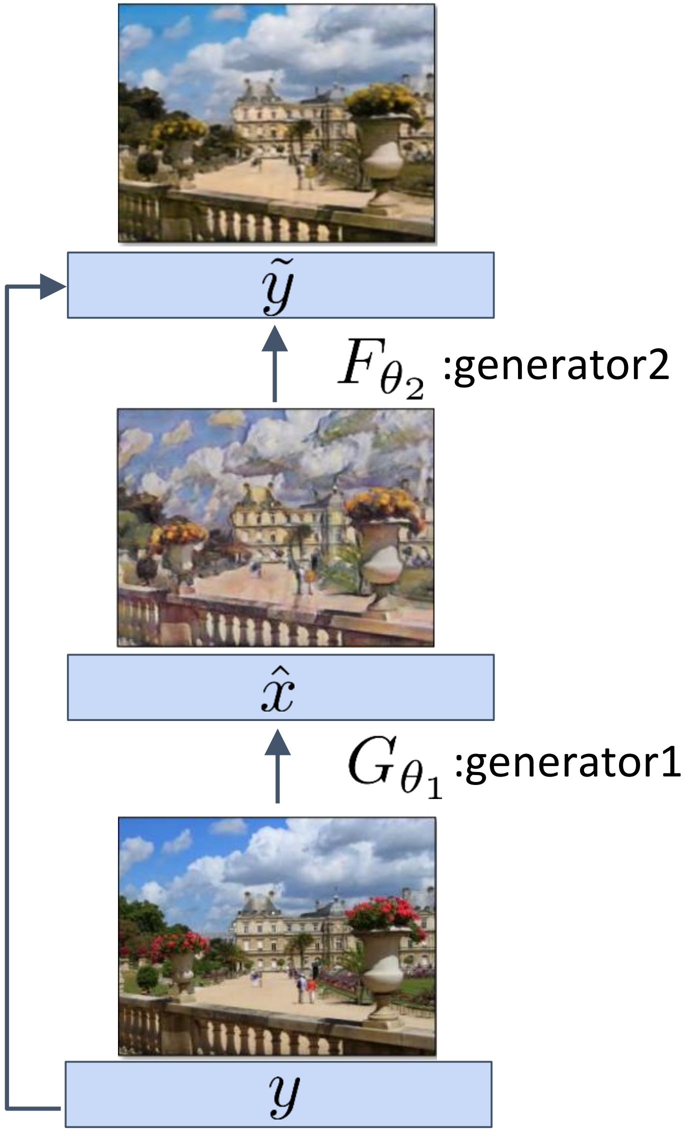


Image Credit: *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*, Zhu et al.

CycleGAN: … and Cycle Consistency

L1 Loss function:
$\|y - \tilde{y}\|_1$



L1 Loss function:
$\|x - \tilde{x}\|_1$

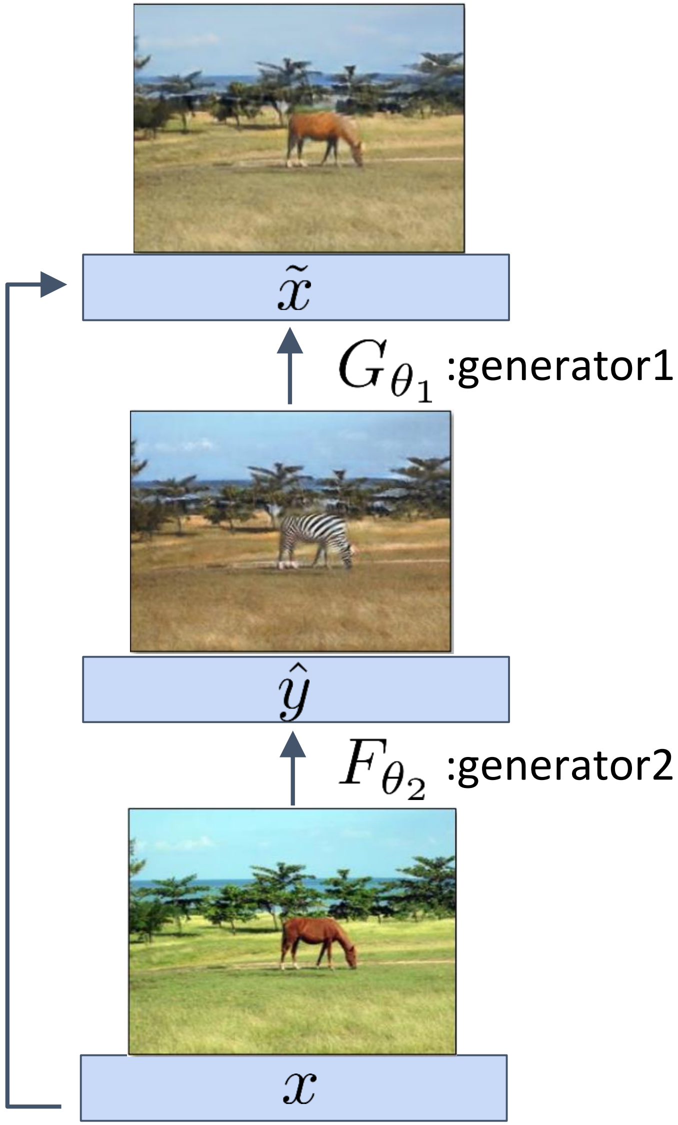


Image Credit: *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*, Zhu et al.

Unstable Training

GAN training can be unstable

Three current research problems (may be related):

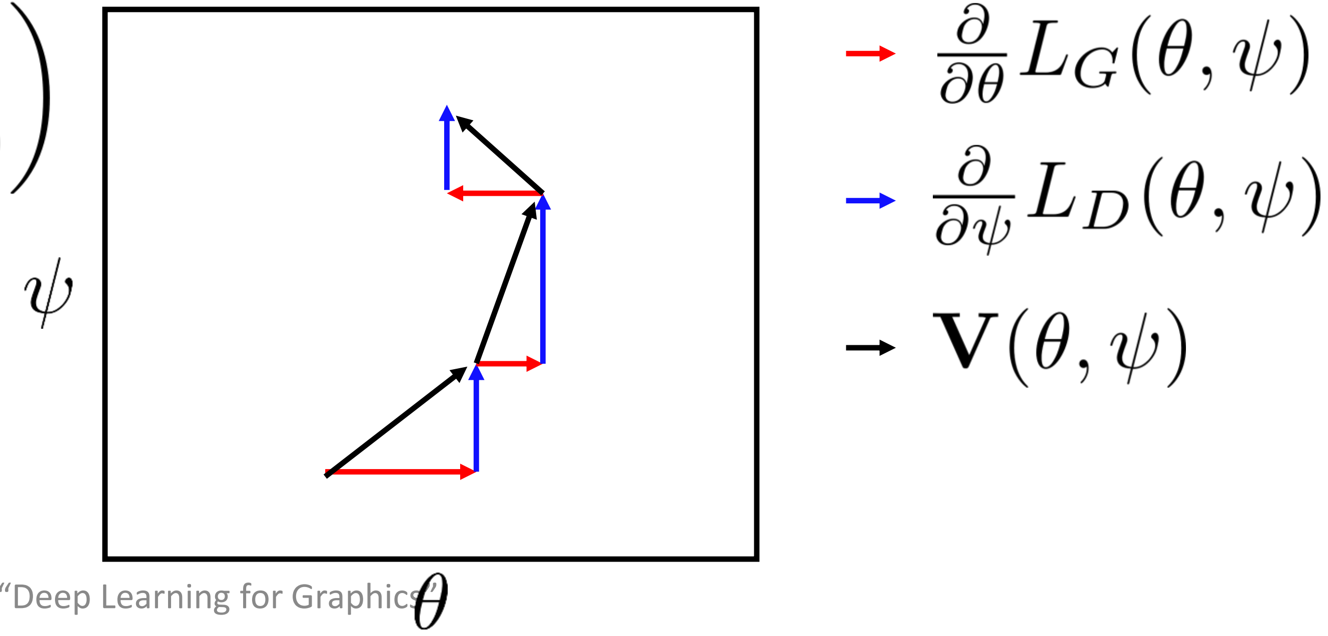
- Reaching a Nash equilibrium (the gradient for both L_G and L_D is 0)
- p_θ and p_{data} initially don't overlap
- Mode Collapse

GAN Training

- Vector-valued loss: $\mathbf{L}(\theta, \psi) = \begin{pmatrix} L_G(\theta, \psi) \\ L_D(\theta, \psi) \end{pmatrix}$

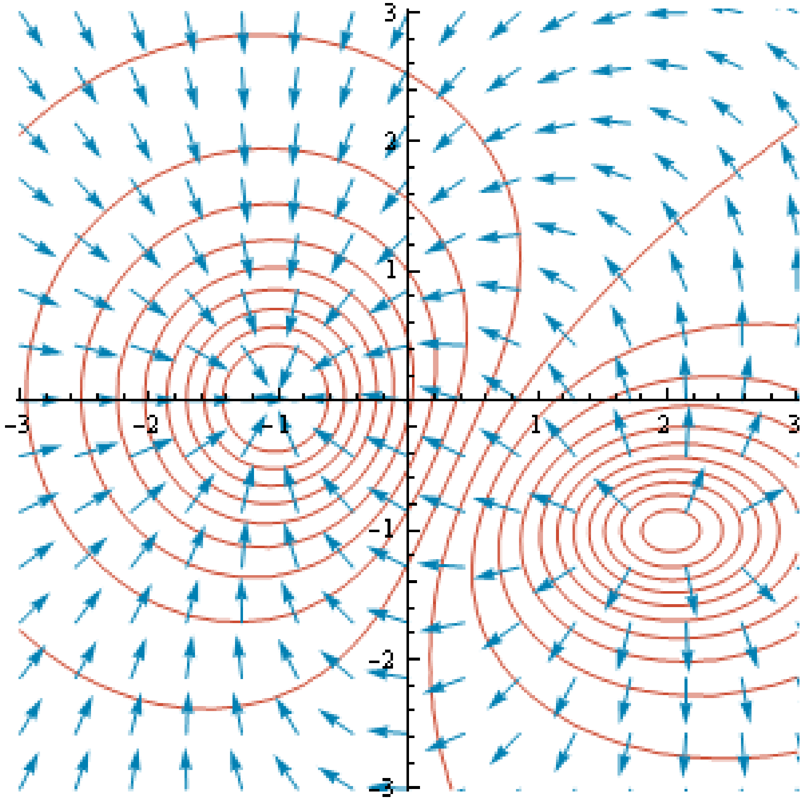
- In each iteration, gradient descent approximately follows this vector over the parameter space (θ, ψ):

$$\mathbf{V}(\theta, \psi) = \begin{pmatrix} \frac{\partial}{\partial \theta} L_G(\theta, \psi) \\ \frac{\partial}{\partial \psi} L_D(\theta, \psi) \end{pmatrix}$$



Reaching Nash Equilibrium

Gradient field example



$\mathbf{V}(\theta, \psi)$ Example

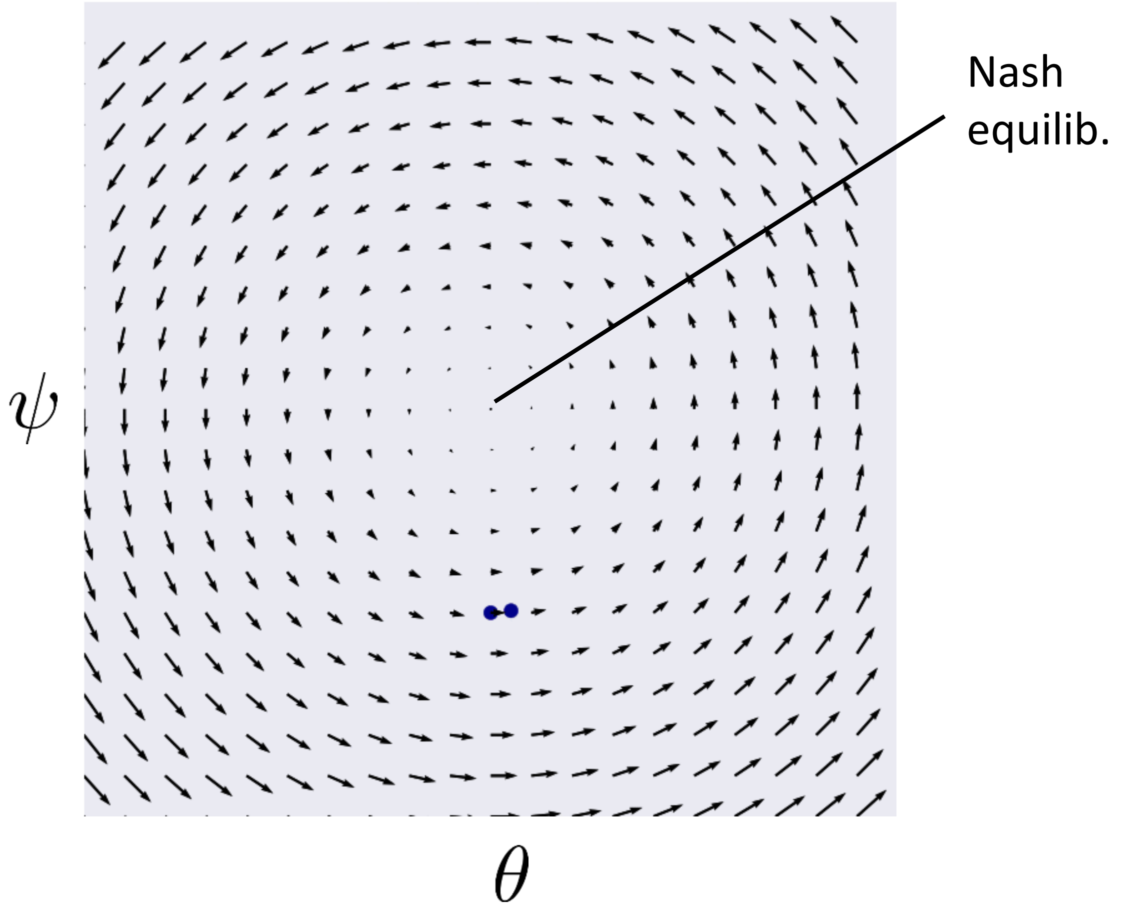
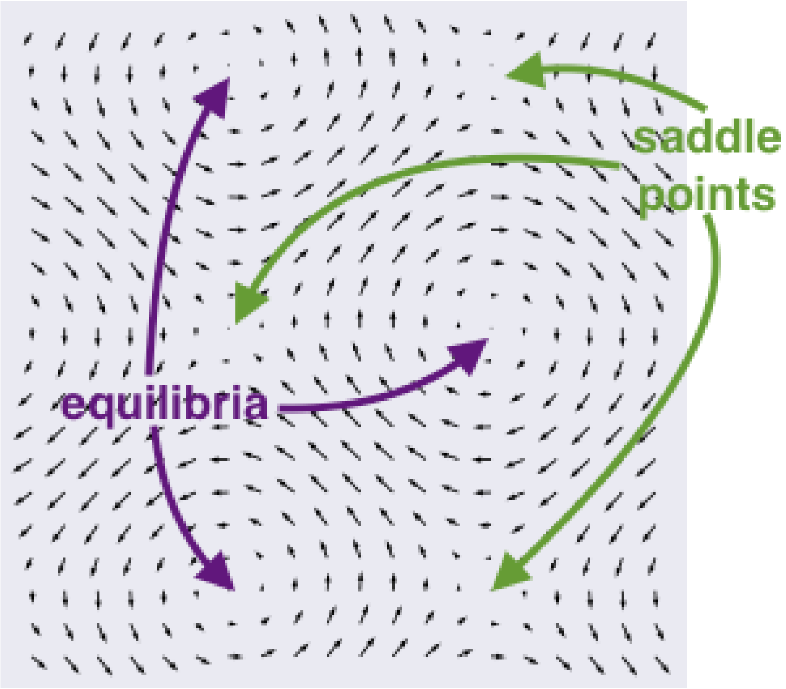


Image Credit: *GANs are Broken in More than One Way: The Numerics of GANs*, Ferenc Huszár

Reaching Nash Equilibrium

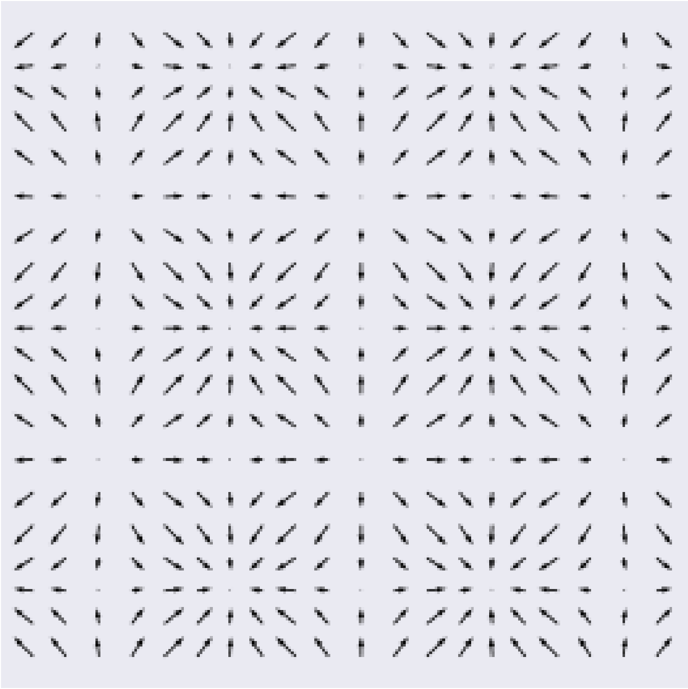
Solution attempt: relaxation with term: $-\nabla L = \frac{\partial}{\partial \theta} \|\mathbf{V}(\theta, \psi)\|_2^2$

Non-conservative field v



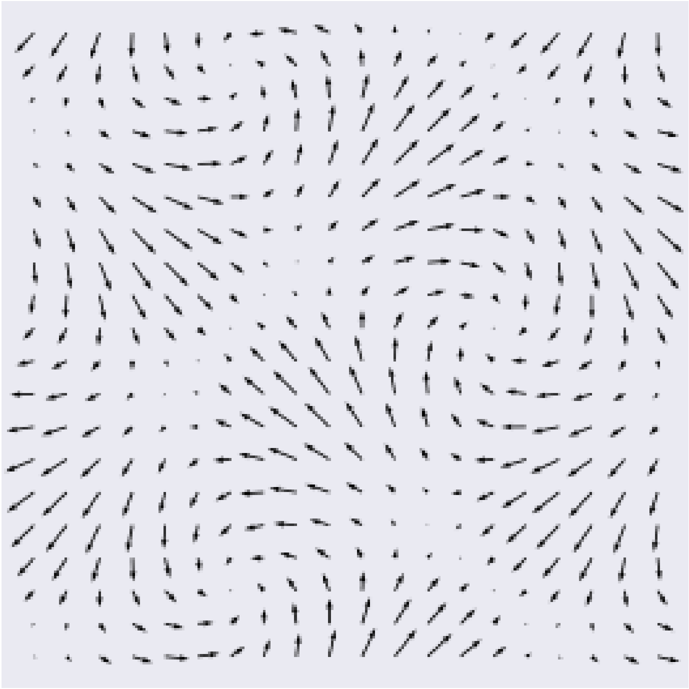
no relaxation has cycles

Conservative field $-\nabla L$



full relaxation introduces bad Nash equilibria

Combined field $v - 0.6\nabla L$

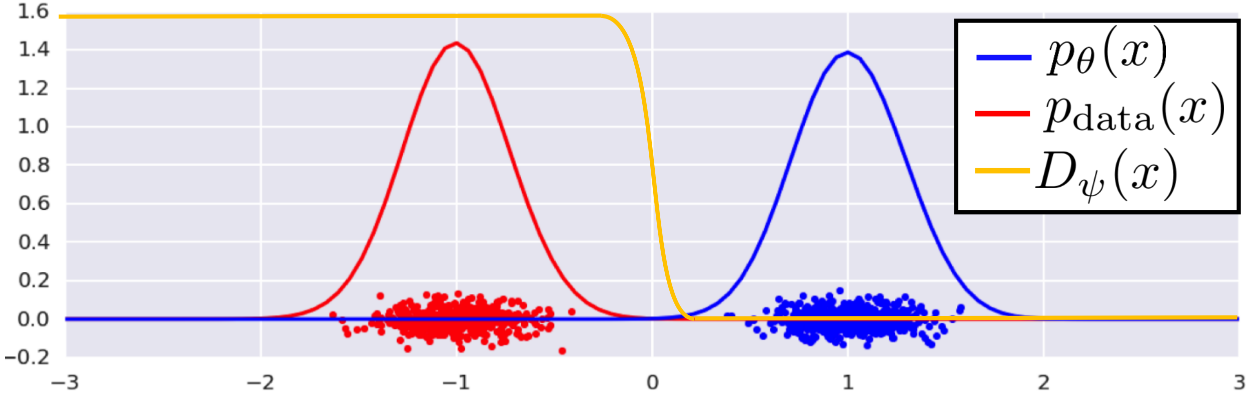


mixture works sometimes

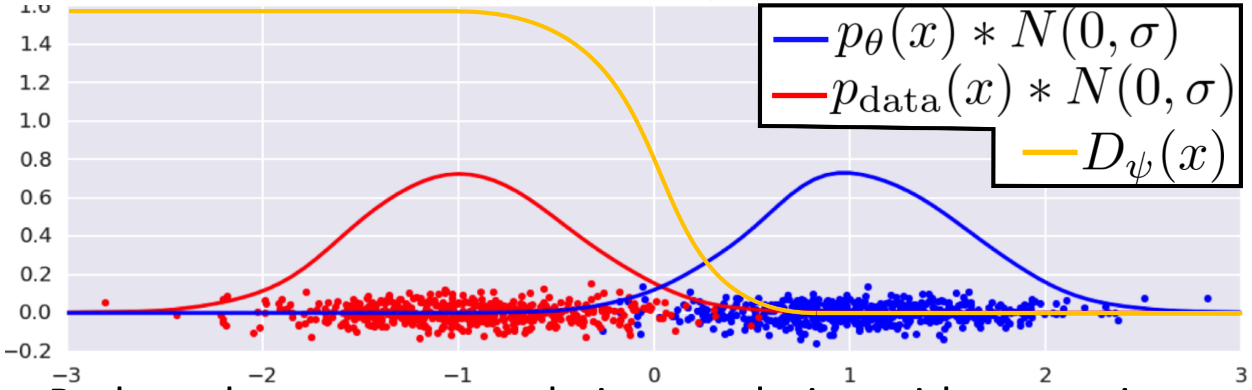
Image Credit: *GANs are Broken in More than One Way: The Numerics of GANs*, Ferenc Huszár

Generator and Data Distribution Don’t Overlap

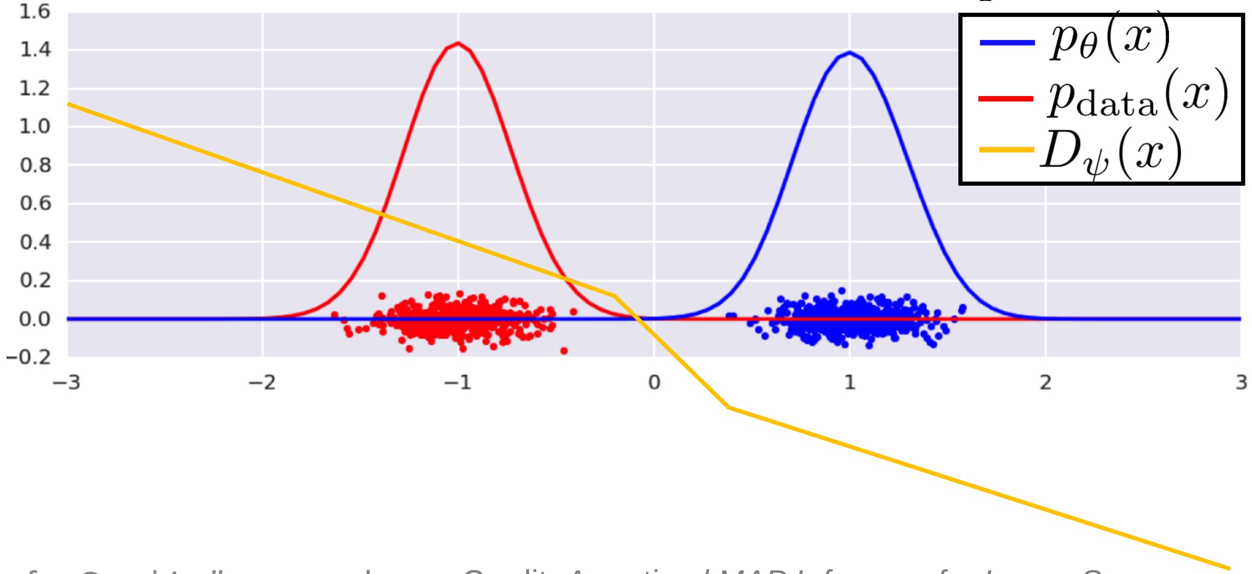
Standard



Instance noise: adding noise to generated and real images



Wasserstein GANs: EMD as distance between p_θ and p_{data}



Roth et al. suggest an analytic convolution with a gaussian:

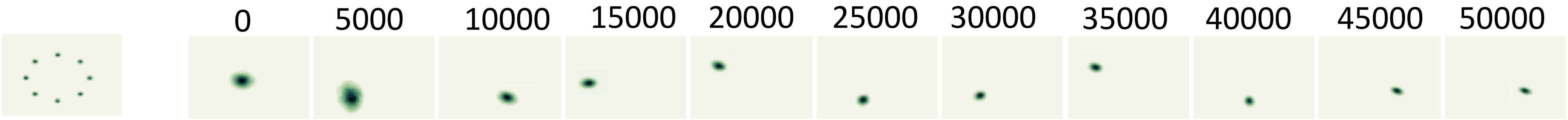
Stabilizing Training of Generative Adversarial Networks through Regularization, Roth et al. 2017

Image Credit: *Amortised MAP Inference for Image Super-resolution*, Sønderby et al.

Mode Collapse

Optimal $D_\psi(x)$: $\dfrac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_\theta(x)}$

p_θ only covers one or a few modes of p_{data}



p_{data} p_θ after n training steps

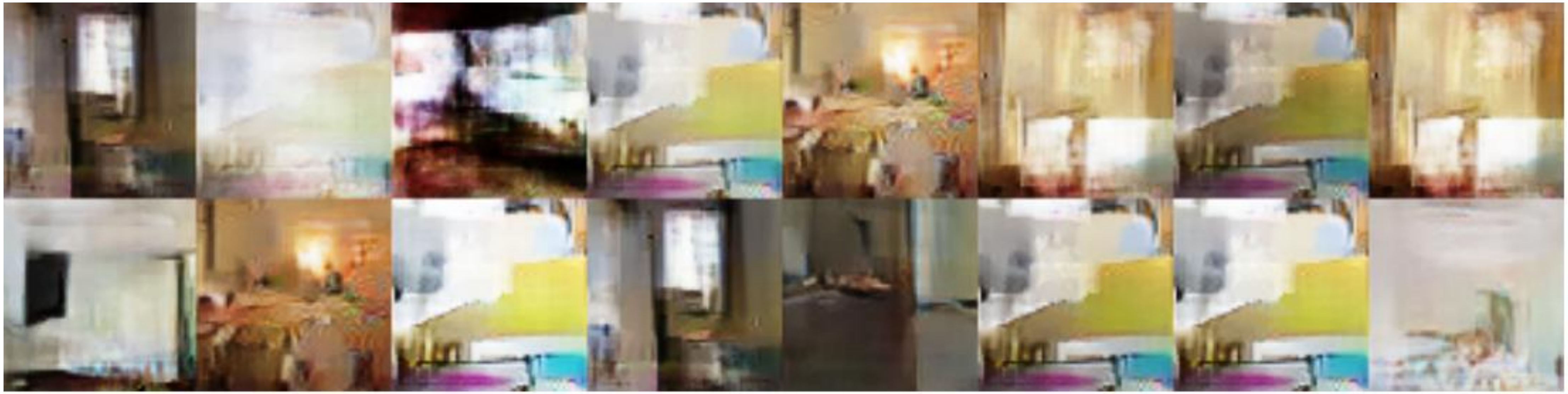


Image Credit: *Wasserstein GAN,* Arjovsky et al.
Unrolled Generative Adversarial Networks, Metz et al.

Mode Collapse

Solution attempts:

- Minibatch comparisons: Discriminator can compare instances in a minibatch (*Improved Techniques for Training GANs*, Salimans et al.)
- Unrolled GANs: Take k steps with the discriminator in each iteration, and backpropagate through all of them to update the generator

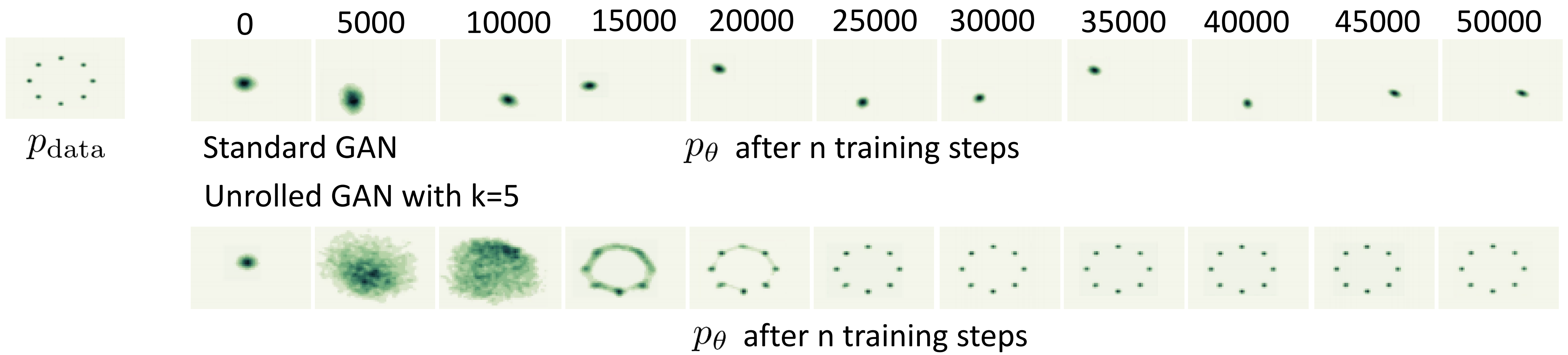


Image Credit: *Wasserstein GAN*, Arjovsky et al.
Unrolled Generative Adversarial Networks, Metz et al.

Progressive GANs

- Resolution is increased progressively during training
- Also other tricks like using minibatch statistics and normalizing feature vectors

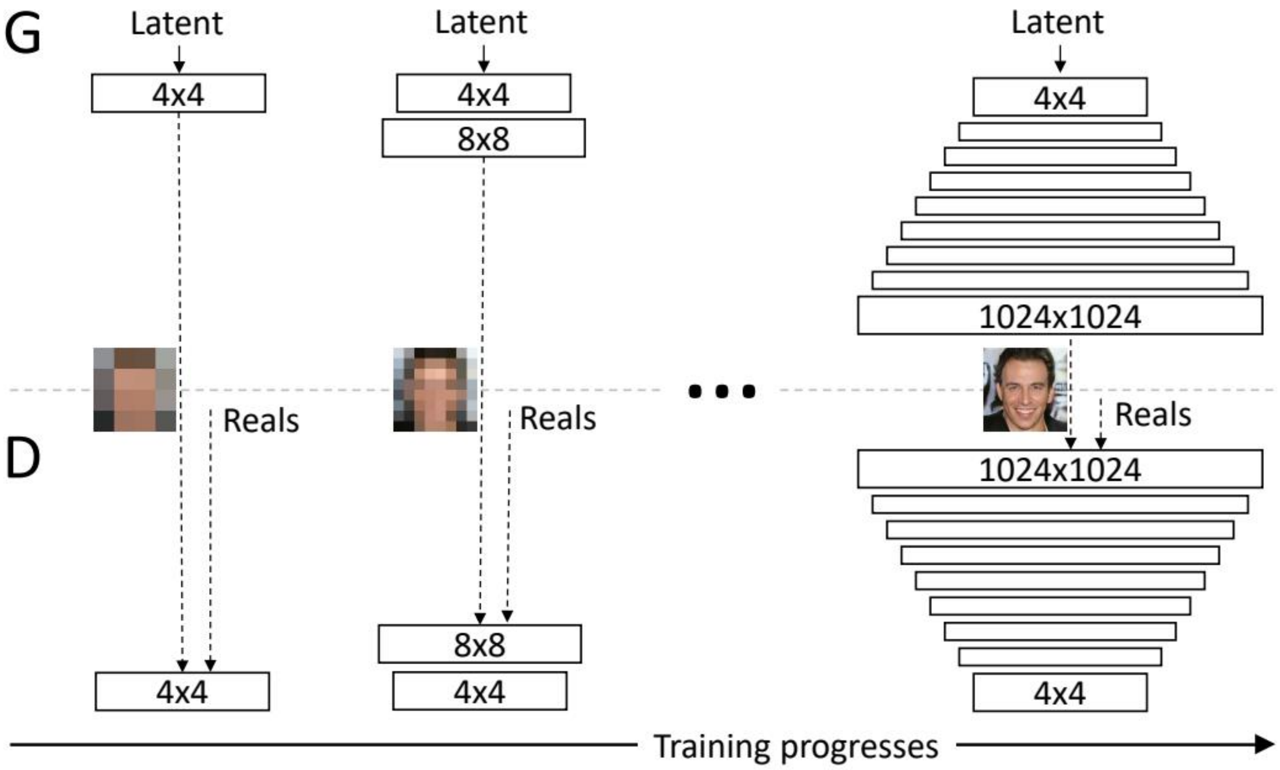


Image Credit: *Progressive Growing of GANs for Improved Quality, Stability, and Variation*, Karras et al.

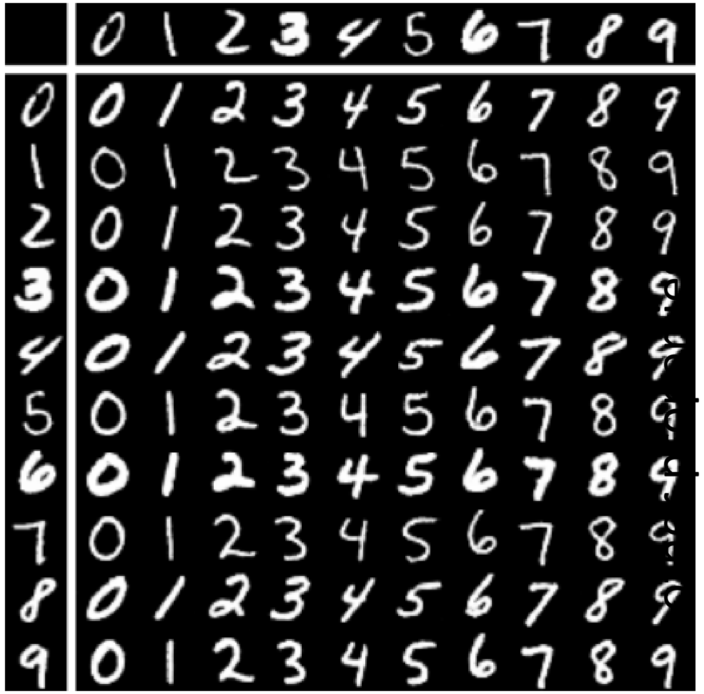
Disentanglement

z — Entangled: different properties may be mixed up over all dimensions

z_a z_b $\cdots$ — Disentangled: different properties are in different dimensions

specified property: number

other properties



specified property: character

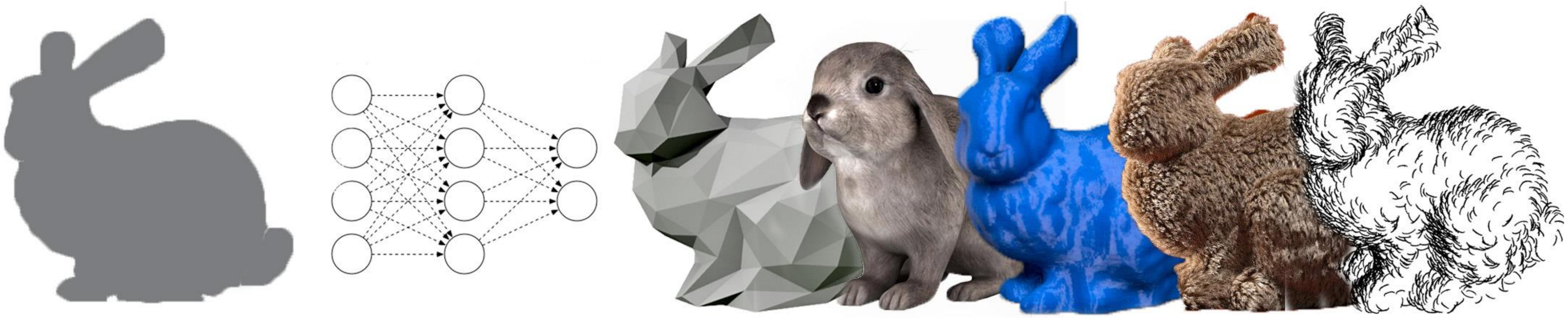
other properties

Image Credit: *Disentangling factors of variation in deep representations using adversarial training*, Mathieu et al.

Summary

- Autoencoders
  - Can infer useful latent representation for a dataset
  - Bad generators
- VAEs
  - Can infer a useful latent representation for a dataset
  - Better generators due to latent space regularization
  - Lower quality reconstructions and generated samples (usually blurry)
- GANs
  - Can not find a latent representation for a given sample (no encoder)
  - Usually better generators than VAEs
  - Currently unstable training (active research)

Thank you!



http://geometry.cs.ucl.ac.uk/dl4g/