STAT 453: Introduction to Deep Learning and Generative Models

Ben Lengerich

Lecture 17: Generative Adversarial Networks

November 3, 2025

Reading: See course homepage

Project

- Midway report due this Friday
- Rubric

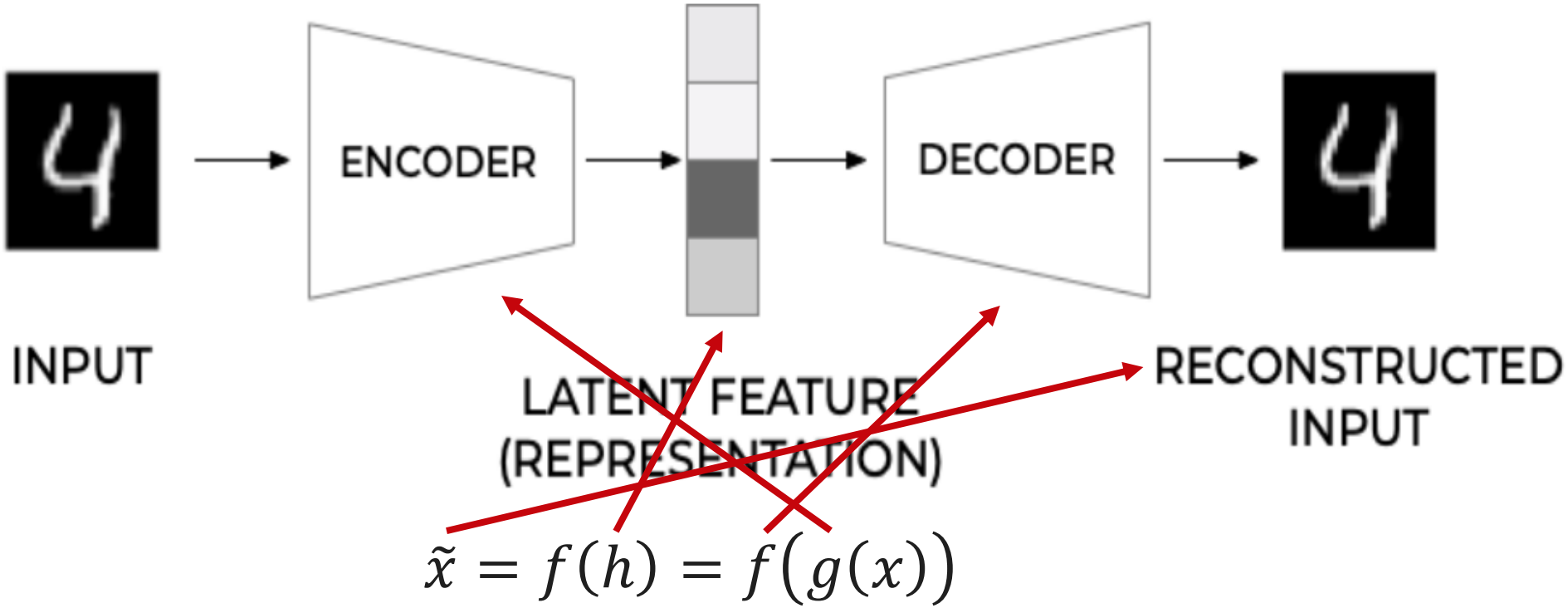
A New Resource

https://badgercompute.wisc.edu/

Interactive computing environment

Last Time: Autoencoders

Autoencoders

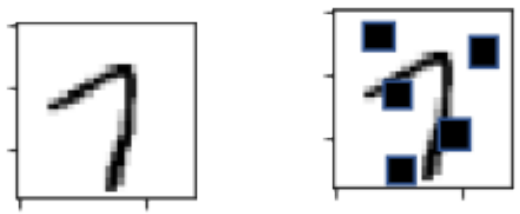


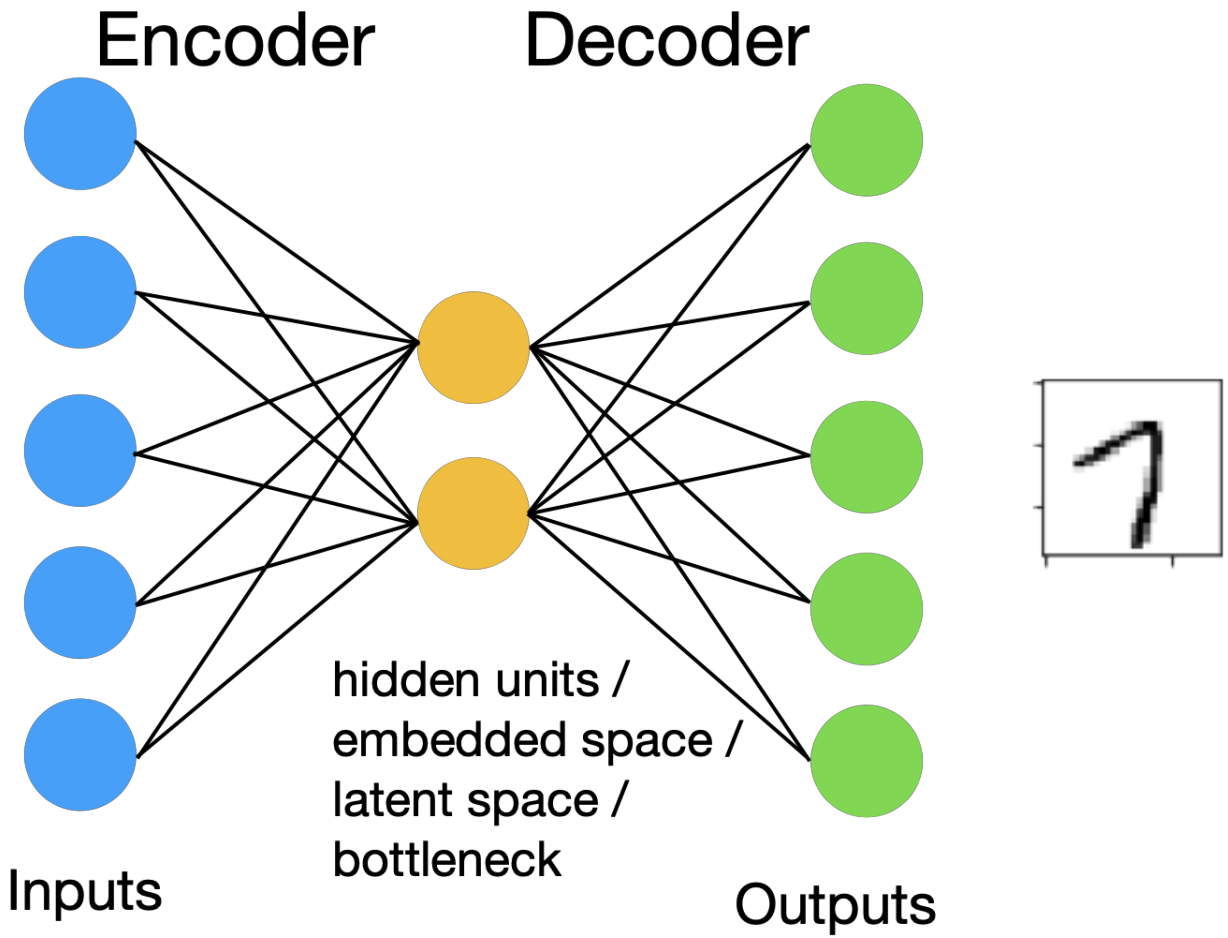
$$\tilde{x} = f(h) = f\big(g(x)\big)$$

[Michelucci 2022]

Denoising Autoencoders

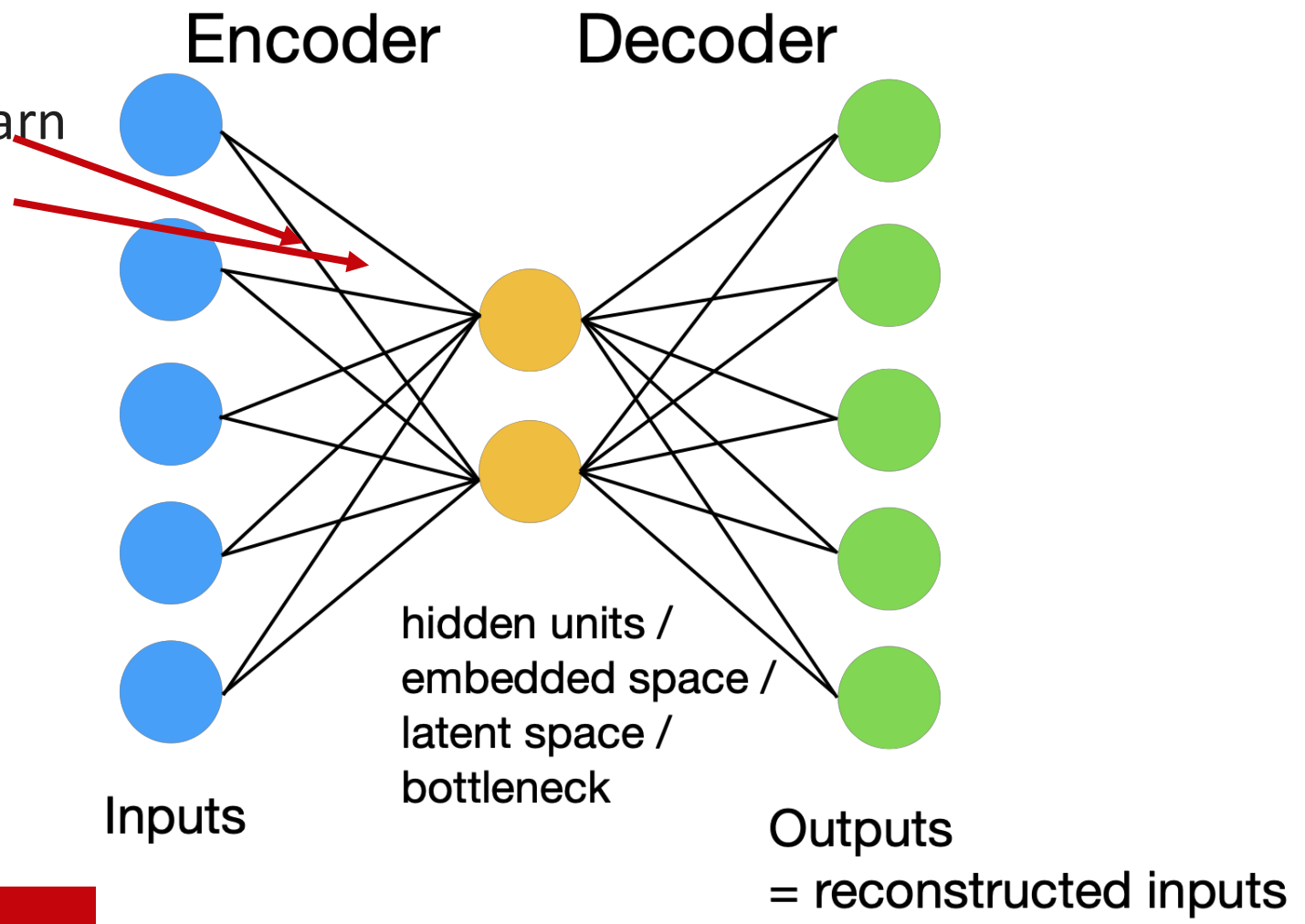
Add dropout after the input, or add noise to the input to learn to denoise inputs





Autoencoders and Dropout

Add dropout layers to force the network to learn redundant features

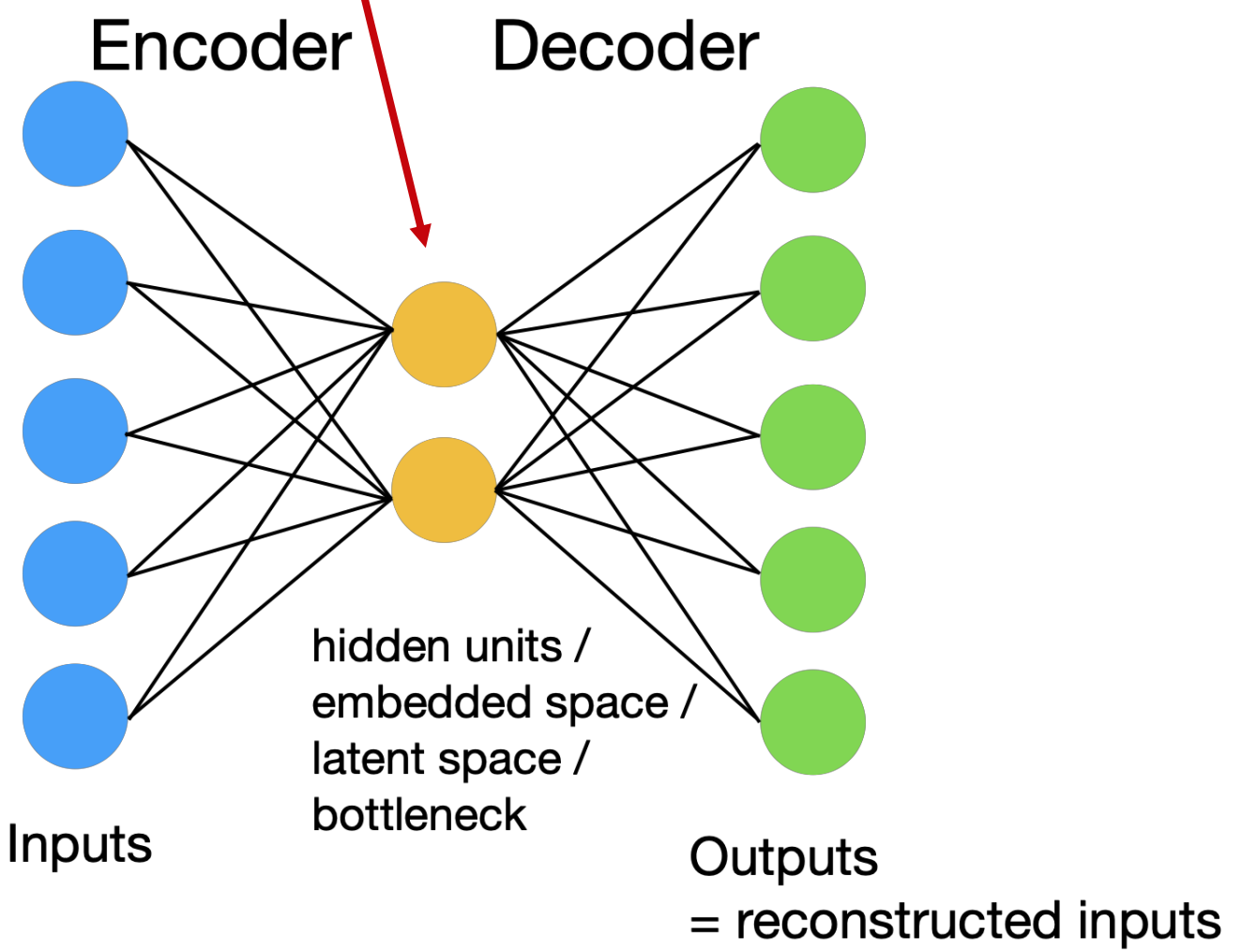


Sparse Autoencoders

$$\sum_i |Enc_i(\mathbf{x})|$$

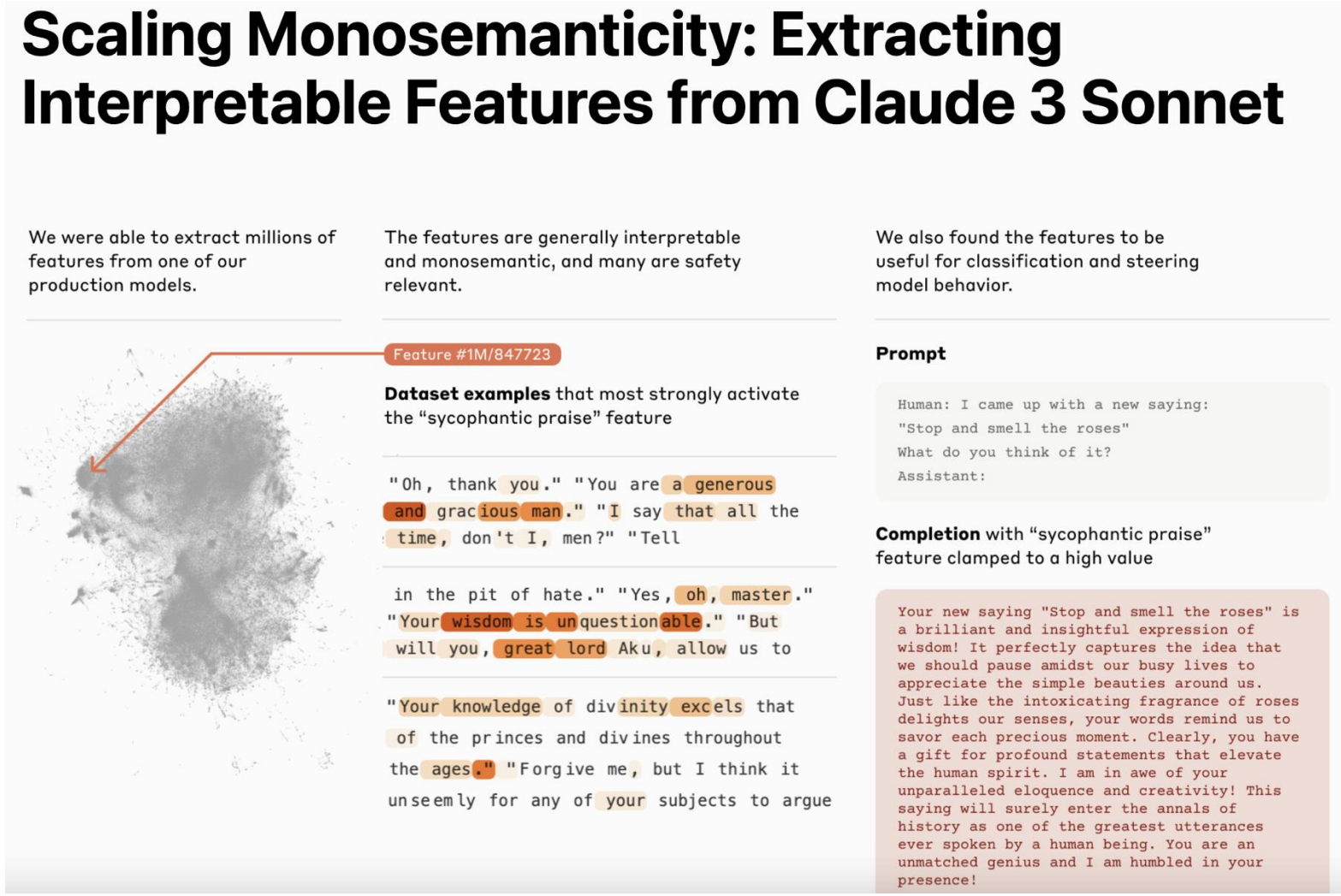
$$\mathcal{L} = ||\mathbf{x} - Dec(Enc(\mathbf{x}))||_2^2 + \sum_i |Enc_i(\mathbf{x})|$$

Add L1 penalty to the loss to learn sparse feature representations



Sparse Autoencoders

Useful for post-hoc interpretability



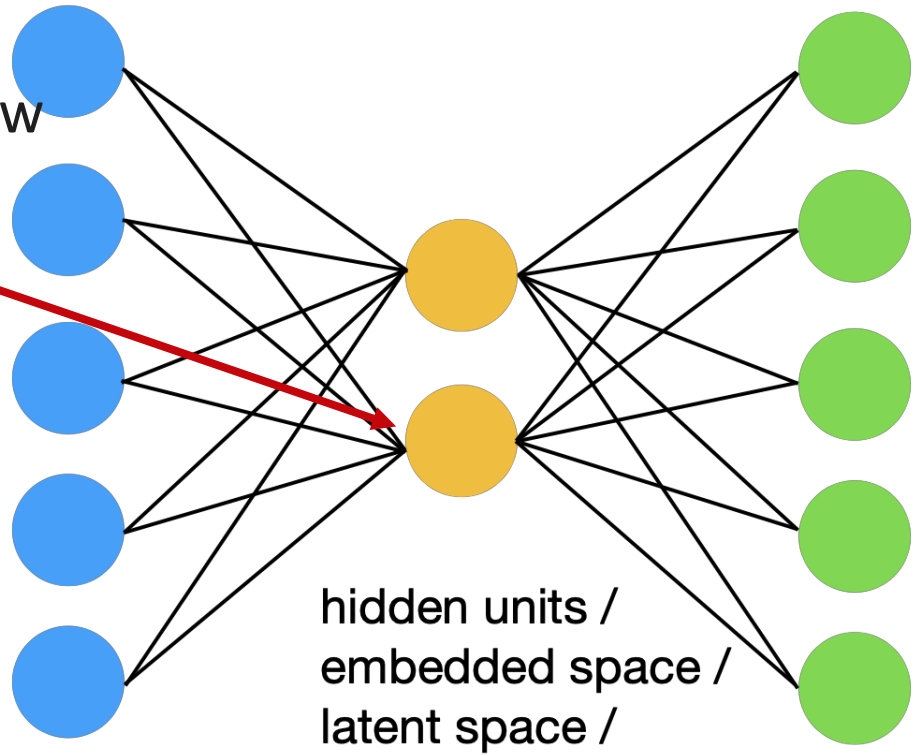
Variational Autoencoders

Kullback-Leibler divergence term
where $p(z) = \mathcal{N}\left(\mu = 0, \sigma^2 = 1\right)$

$$L^{[i]} = -\mathbb{E}_{z \sim q_w(z|x^{[i]})}\left[\log p_w\left(x^{[i]}|z\right)\right] + \mathbf{KL}\left(q_w\left(z|x^{[i]}\right)||p(z)\right)$$

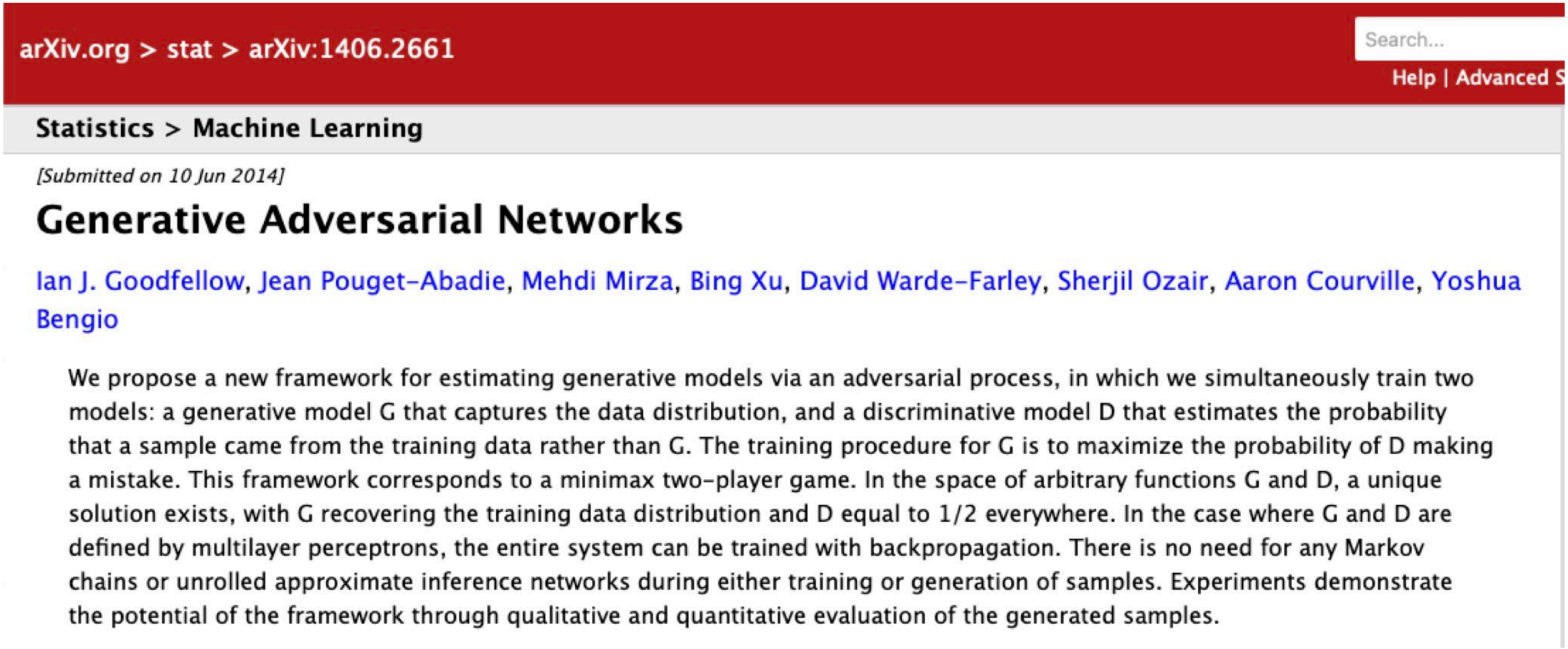
Define latent dim to follow Normal distribution

Enables sampling



Generative Adversarial Networks (GANs)

Generative Adversarial Nets (GANs)



arXiv.org > stat > arXiv:1406.2661

Search...

Help | Advanced S

Statistics > Machine Learning

[Submitted on 10 Jun 2014]

Generative Adversarial Networks

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio

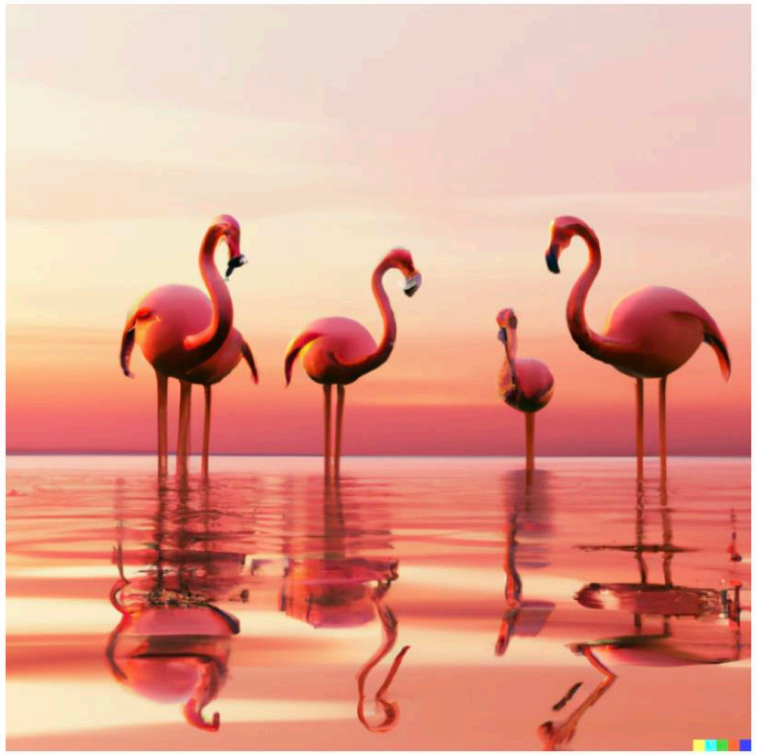
We propose a new framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to 1/2 everywhere. In the case where G and D are defined by multilayer perceptrons, the entire system can be trained with backpropagation. There is no need for any Markov chains or unrolled approximate inference networks during either training or generation of samples. Experiments demonstrate the potential of the framework through qualitative and quantitative evaluation of the generated samples.

https://arxiv.org/abs/1406.2661

Lots of GAN Applications

Text-based image generation via diffusion models

"People walking on a beach during sunrise, a reflection of the sun on the water, realistic"



"Flamingos standing on water, red sunset, pink-red water reflection, photo-realistic, 4k"

Dall-E 2 by OpenAI, April 2022

Lots of GAN Applications

"A photo of a Shiba Inu dog with a backpack riding a bike. It is wearing sunglasses and a beach hat."

"A blue jay standing on a large basket of rainbow macarons"

"A brain riding a rocketship heading towards the moon"

Imagen by Google, April 2022

Lots of GAN Applications

Human Faces generated by **VAEs**

Celebrity faces [Radford 2015]

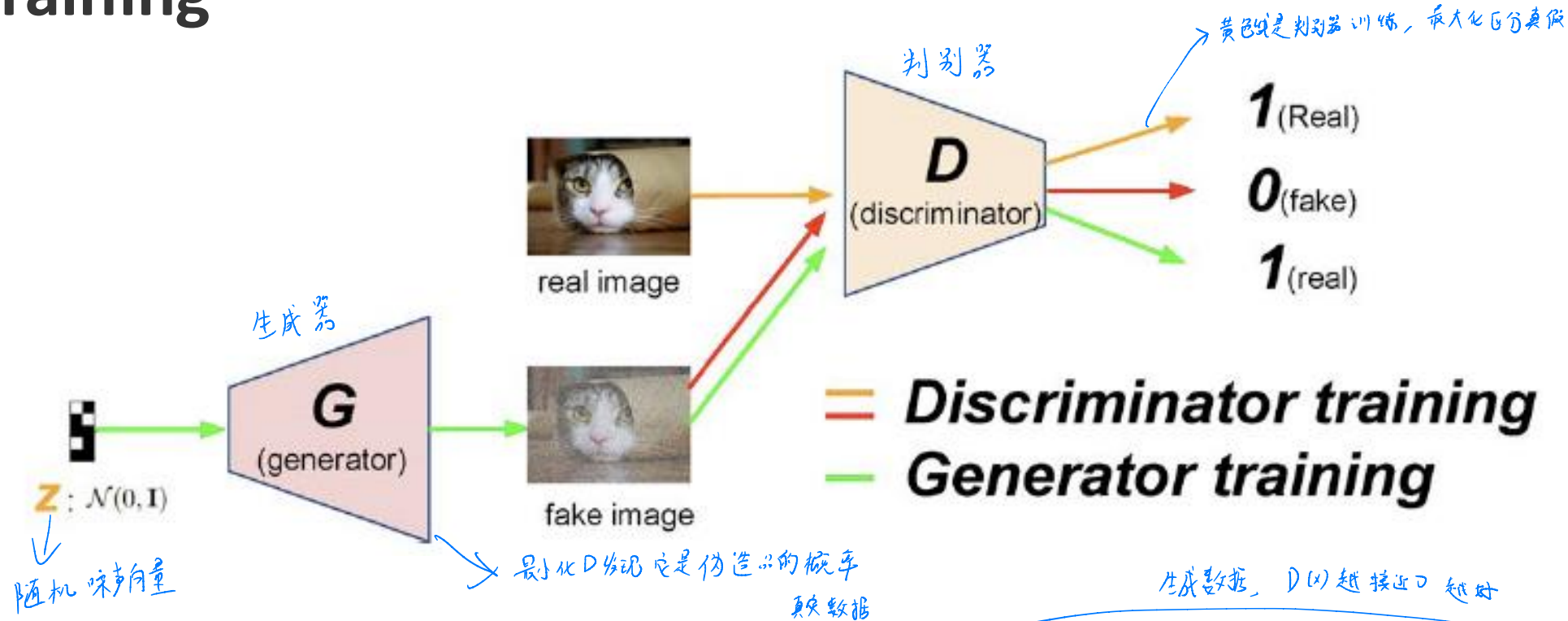
Human Faces generated by **GANs**

https://becominghuman.ai/generative-adversarial-networks-gans-human-creativity-2fc61283f3f6

Generative Adversarial Nets (GANs)

- The original purpose is to generate new data
- Classically for generating new images, but applicable to wide range of domains
- Learns the training set distribution and can generate new images that have never been seen before
- Similar to VAE, and in contrast to e.g., autoregressive models or RNNs (generating one word at a time), GANs generate the whole output all at once

GAN Training



Discriminator: $\max_D \mathcal{L}_D = \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})}\left[\log D(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{x}\sim G(\boldsymbol{z}),\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\log(1-D(\boldsymbol{x}))\right]$

Generator: $\min_G \mathcal{L}_G = \mathbb{E}_{\boldsymbol{x}\sim G(\boldsymbol{z}),\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\log(1-D(\boldsymbol{x}))\right].$

GAN Training – An Adversarial Game

Discriminator: learns to become better at distinguishing real from generated images

Generator: learns to generate better images to fool the discriminator

GAN Training – Putting it together

Discriminator: $\max_D \mathcal{L}_D = \mathbb{E}_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})} [\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x} \sim G(\boldsymbol{z}), \boldsymbol{z} \sim p(\boldsymbol{z})} [\log(1 - D(\boldsymbol{x}))]$

Generator: $\min_G \mathcal{L}_G = \mathbb{E}_{\boldsymbol{x} \sim G(\boldsymbol{z}), \boldsymbol{z} \sim p(\boldsymbol{z})} [\log(1 - D(\boldsymbol{x}))] .$

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

GAN Training – Putting it together

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

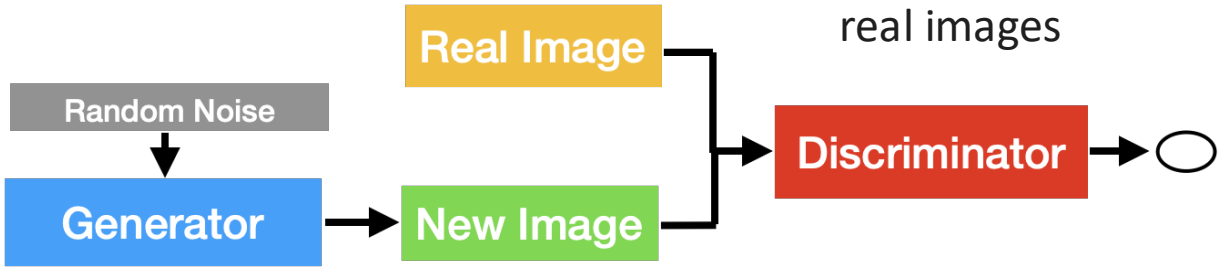
Discriminator gradient for update (gradient ascent):

要最大化 D，所以是梯度上升

$$\nabla_{\mathbf{w}_D} \frac{1}{n} \sum_{i=1}^{n} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right]$$

want large probability on real images

want small probability on generated images



GAN Training – Putting it together

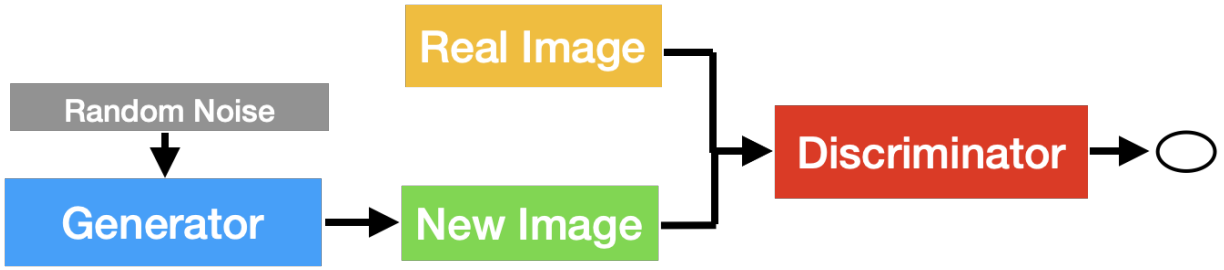
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}\ (\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Generator gradient for update (gradient descent):

生成器是梯度下降

$$\nabla_{\mathbf{w}_G} \frac{1}{n} \sum_{i=1}^{n} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$$

Want discriminator to predict poorly on fake images



Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

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

  end for

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

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Goodfellow, Ian, 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, pp. 2672-2680. 2014.

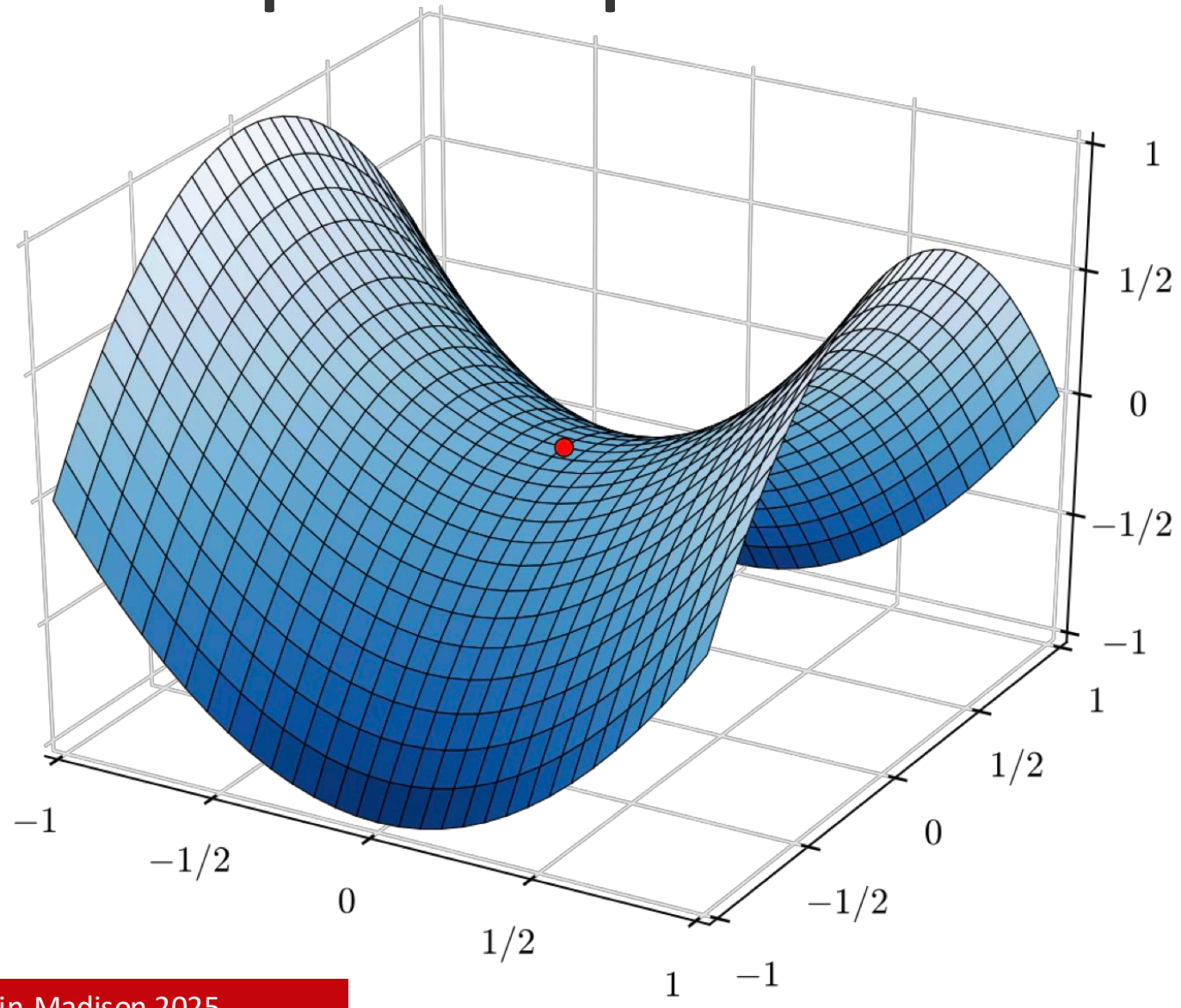
GAN Training – Convergence?

- Converges when Nash-equilibrium (Game Theory concept) is reached in the minmax (zero-sum) game

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

- Nash-equilibrium in Game Theory is reached when the actions of one player won't change depending on the opponent's actions
- Here, this means that the GAN produces realistic images and the discriminator outputs random predictions (probabilities close to 0.5)

GAN Training – Saddle point interpretation



GAN Training Problems

- Oscillation between generator and discriminator loss
- Mode collapse (generator produces examples of a particular kind only)
- Discriminator is too strong, such that the gradient for the generator vanishes and the generator can't keep up

Instead of gradient descent with

$$\nabla_{\mathbf{w}_G} \frac{1}{n} \sum_{i=1}^{n} \log \left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$$

Do gradient ascent with

$$\nabla_{\mathbf{w}_G} \frac{1}{n} \sum_{i=1}^{n} \log \left(D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$$

"Non-saturating" GAN

GAN Training Problems

- Oscillation between generator and discriminator loss
- Mode collapse (generator produces examples of a particular kind only)
- Discriminator is too strong, such that the gradient for the generator vanishes and the generator can't keep up
- Discriminator is too weak, and the generator produces non-realistic images that fool it too easily (rare problem, though)
- Sensitive to learning rate and other hyper parameters

GAN Training Problems

- Lots of Tips & Tricks:

https://github.com/soumith/ganhacks

Deep Convolutional GAN

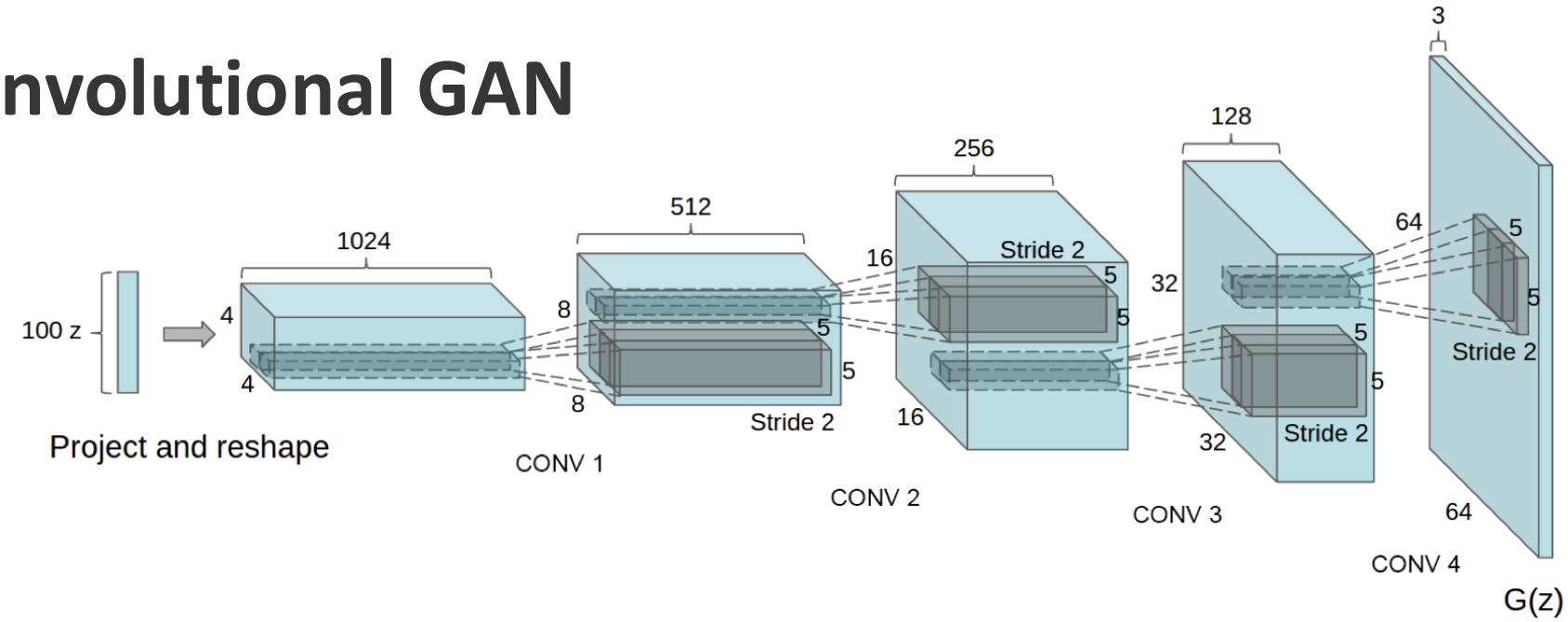


Figure 1: DCGAN generator used for LSUN scene modeling. A 100 dimensional uniform distribution Z is projected to a small spatial extent convolutional representation with many feature maps. A series of four fractionally-strided convolutions (in some recent papers, these are wrongly called deconvolutions) then convert this high level representation into a 64×64 pixel image. Notably, no fully connected or pooling layers are used.

Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

Bonus

GANs and VAEs: A Unified View

Bonus

GANs

- GAN: Implicit distribution over $x \sim p_\theta(x \mid y)$

$$p_\theta(\boldsymbol{x}|y) = \begin{cases} p_{g\theta}(\boldsymbol{x}) & y = 0 \\ p_{data}(\boldsymbol{x}) & y = 1. \end{cases}$$

(distribution of generated images)

(distribution of real images)

- $x \sim p_{g_\theta}(x) \leftrightarrow x = G_\theta(z), z \sim p(z \mid y = 0)$
- $x \sim p_{data}(x)$

Bonus

GANs: Rewrite in Variational-EM format

- The familiar "Variational-EM" format:

$$\max_{\boldsymbol{\phi}} \mathcal{L}_\phi = \mathbb{E}_{\boldsymbol{x}=G_\theta(\boldsymbol{z}),\boldsymbol{z}\sim p(\boldsymbol{z}|y=0)} \left[\log(1 - D_\phi(\boldsymbol{x}))\right] + \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})} \left[\log D_\phi(\boldsymbol{x})\right]$$

$$\max_{\boldsymbol{\theta}} \mathcal{L}_\theta = \mathbb{E}_{\boldsymbol{x}=G_\theta(\boldsymbol{z}),\boldsymbol{z}\sim p(\boldsymbol{z}|y=0)} \left[\log D_\phi(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})} \left[\log(1 - D_\phi(\boldsymbol{x}))\right]$$

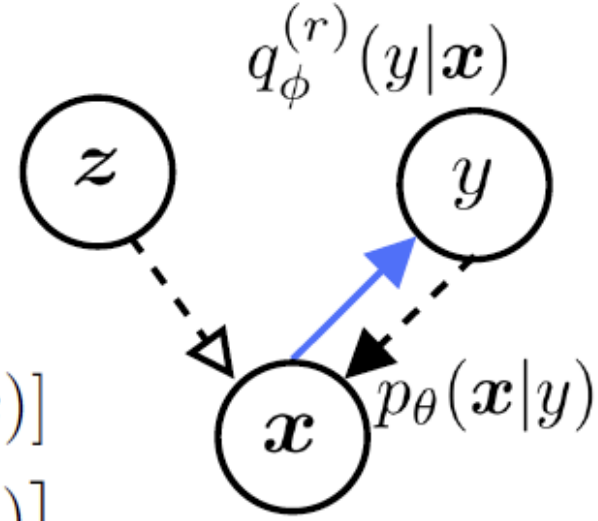
$$= \mathbb{E}_{\boldsymbol{x}=G_\theta(\boldsymbol{z}),\boldsymbol{z}\sim p(\boldsymbol{z}|y=0)} \left[\log D_\phi(\boldsymbol{x})\right]$$

- Implicit distribution over $x \sim p_\theta(x \mid y)$:

$$x = G_\theta(z), z \sim p(z \mid y = 0)$$

- Discriminator distribution $\mathrm{q}_\phi(\mathrm{y} \mid x)$:

$$q_\phi^r(y \mid x) = q_\phi(1 - y \mid x)$$



$$\max_{\boldsymbol{\phi}} \mathcal{L}_\phi = \mathbb{E}_{p_\theta(\boldsymbol{x}|y)p(y)} \left[\log q_\phi(y|\boldsymbol{x})\right]$$

$$\max_{\boldsymbol{\theta}} \mathcal{L}_\theta = \mathbb{E}_{p_\theta(\boldsymbol{x}|y)p(y)} \left[\log q_\phi^r(y|\boldsymbol{x})\right]$$

Bonus

Variational EM vs. GANs

- Variational EM
  - Objectives

$$\max_{\phi} \mathcal{L}_{\phi,\theta} = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] + KL\left(q_\phi(z|x)||p(z)\right)$$

$$\max_{\theta} \mathcal{L}_{\phi,\theta} = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] + KL\left(q_\phi(z|x)||p(z)\right)$$

  - Single objective for both θ and ϕ
  - Extra prior regularization by $p(z)$
- The reconstruction term:
  - Maximize the conditional log-likelihood of x with the generative distribution $p_\theta(x \mid z)$ conditioning on the latent code z inferred by $q_\phi(z \mid x)$
- $p_\theta(x \mid z)$ is the generative model
- $q_\phi(z \mid x)$ is the inference model

- GAN
  - Objectives

$$\max_{\phi} \mathcal{L}_{\phi} = \mathbb{E}_{p_\theta(\boldsymbol{x}|y)p(y)}\left[\log q_\phi(y|\boldsymbol{x})\right]$$

$$\max_{\theta} \mathcal{L}_{\theta} = \mathbb{E}_{p_\theta(\boldsymbol{x}|y)p(y)}\left[\log q_\phi^r(y|\boldsymbol{x})\right]$$

  - Two objectives
- Maximize the conditional log-likelihood of y with the distribution $q_\phi(y \mid x)$ conditioning on data/generation x from $p_\theta(x \mid y)$
- $q_\phi(y \mid x)$ is the generative model
- $p_\theta(x \mid y)$ is the inference model

Bonus

GANs vs VAEs: A Symmetry

Hu et al. "Unifying Deep Generative Models"

	GANs (InfoGAN)	VAEs
Generative distribution	$p_\theta(\boldsymbol{x}\|y) = \begin{cases} p_{g_\theta}(\boldsymbol{x}) & y = 0 \\ p_{data}(\boldsymbol{x}) & y = 1. \end{cases}$	$p_\theta(\boldsymbol{x}\|\boldsymbol{z}, y) = \begin{cases} p_\theta(\boldsymbol{x}\|\boldsymbol{z}) & y = 0 \\ p_{data}(\boldsymbol{x}) & y = 1. \end{cases}$
Discriminator distribution	$q_\phi(y\|\boldsymbol{x})$	$q_*(y\|\boldsymbol{x})$, perfect, degenerated
$\boldsymbol{z}$-inference model	$q_\eta(\boldsymbol{z}\|\boldsymbol{x}, y)$ of InfoGAN	$q_\eta(\boldsymbol{z}\|\boldsymbol{x}, y)$
KLD to minimize	$\min_\theta \mathrm{KL}\left(p_\theta(\boldsymbol{x}\|y) \,\|\| \, q^r(\boldsymbol{x}\|\boldsymbol{z}, y)\right)$ $\sim \min_\theta \mathrm{KL}(P_\theta \,\|\| \, Q)$	$\min_\theta \mathrm{KL}\left(q_\eta(\boldsymbol{z}\|\boldsymbol{x}, y) q^r_*(y\|\boldsymbol{x}) \,\|\| \, p_\theta(\boldsymbol{z}, y\|\boldsymbol{x})\right)$ $\sim \min_\theta \mathrm{KL}(Q \,\|\| \, P_\theta)$

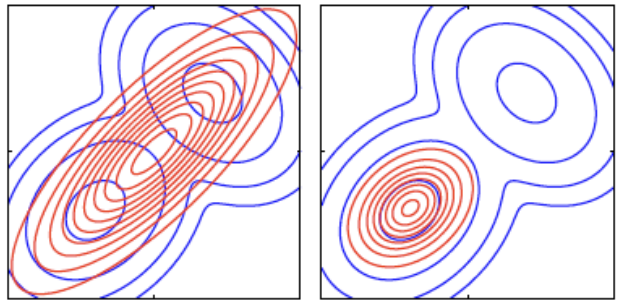
Bonus

GANs vs VAEs: A Symmetry

Hu et al. "Unifying Deep Generative Models"

	GANs (InfoGAN)	VAEs
KLD to minimize	$\min_\theta \mathrm{KL}\left(p_\theta(\boldsymbol{x}\|y) \,\|\| \, q^r(\boldsymbol{x}\|\boldsymbol{z}, y)\right)$ $\sim \min_\theta \mathrm{KL}(P_\theta \,\|\| \, Q)$	$\min_\theta \mathrm{KL}(q_\eta(\boldsymbol{z}\|\boldsymbol{x}, y) q^r_*(y\|\boldsymbol{x}) \,\|\| \, p_\theta(\boldsymbol{z}, y\|\boldsymbol{x}))$ $\sim \min_\theta \mathrm{KL}(Q \,\|\| \, P_\theta)$

- Asymmetry of KLDs inspires combination of GANs and VAEs
  - GANs: $\min_\theta \mathrm{KL}(P_\theta \| Q)$ tends to missing mode
  - VAEs: $\min_\theta \mathrm{KL}(Q \| P_\theta)$ tends to cover regions with small values of p_{data}



Mode covering Mode missing

Bonus

GANs and Mode Collapse

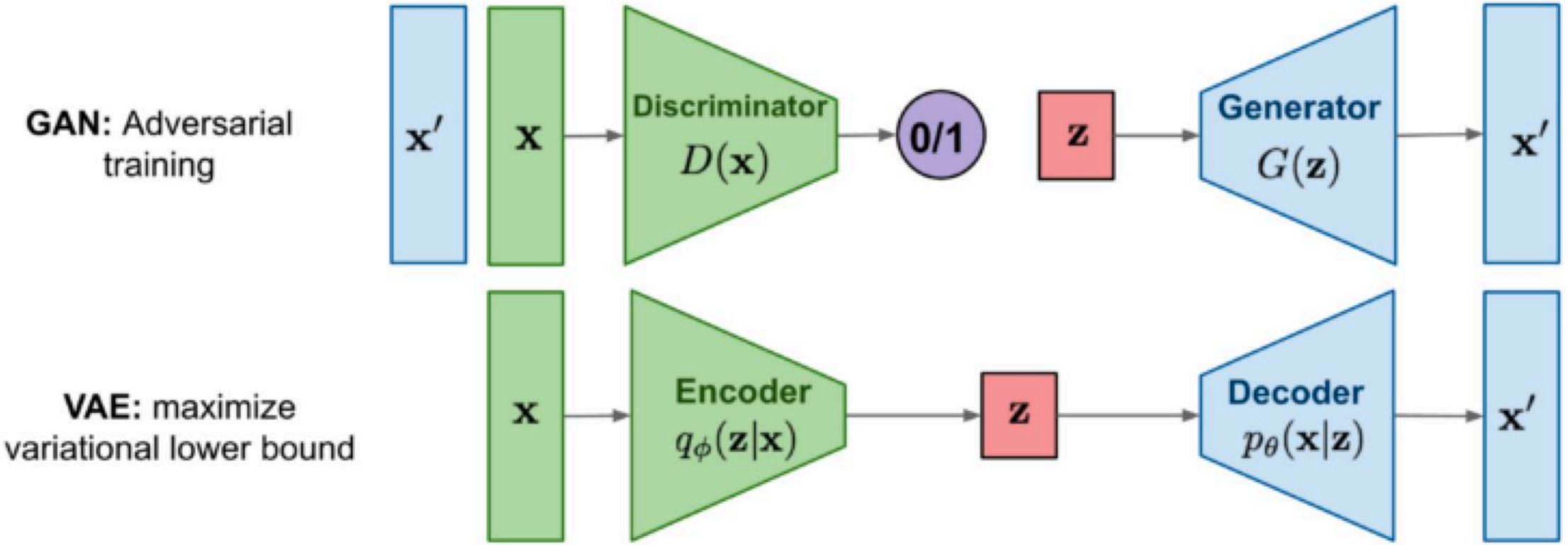
Human Faces generated by **VAEs**

Celebrity faces [Radford 2015]

Human Faces generated by **GANs**

https://becominghuman.ai/generative-adversarial-networks-gans-human-creativity-2fc61283f3f6

What to remember



Source: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Questions?