CS 480/680
Introduction to Machine Learning

Lecture 19
Generative Adversarial Networks and Diffusion Models
Deep Generative Models Part II

Kathryn Simone

21 November 2024

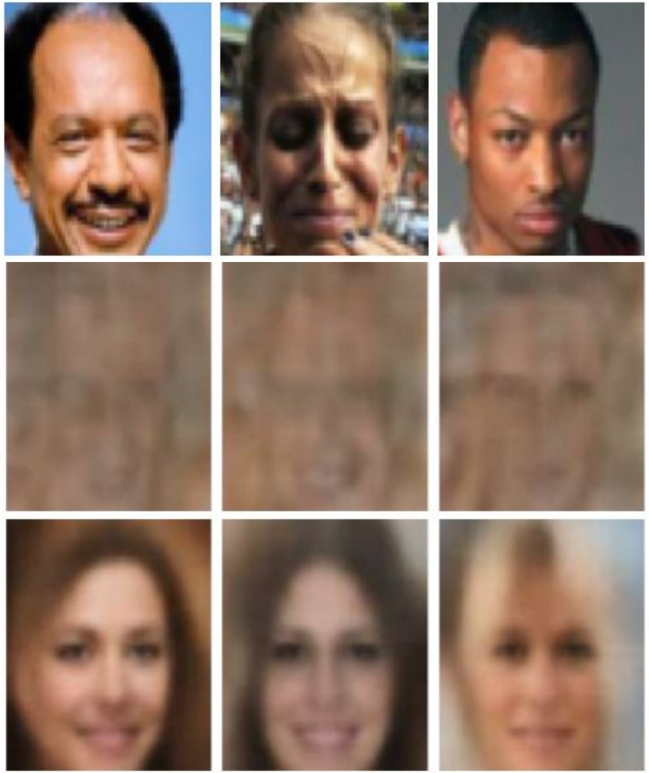
UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

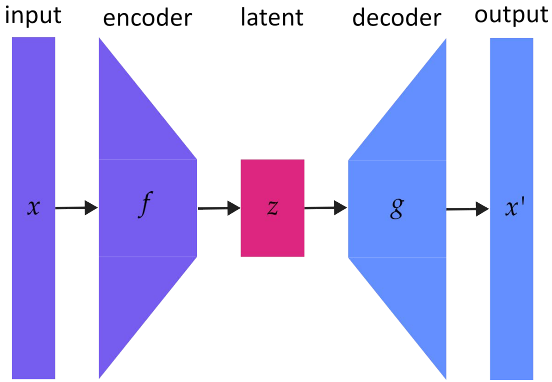
VAEs generate realistic, but low-quality, samples

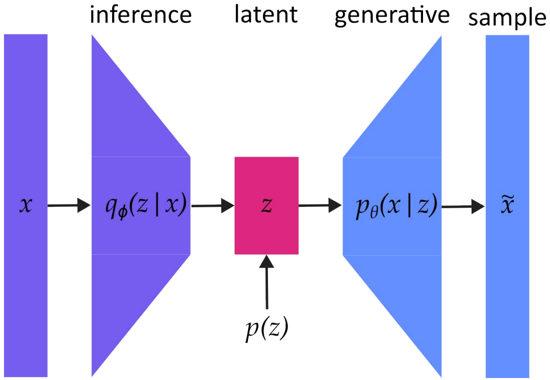
Training

AE Samples

VAE Samples







Contemporary DGMs produce high-quality images

DALL·E 3

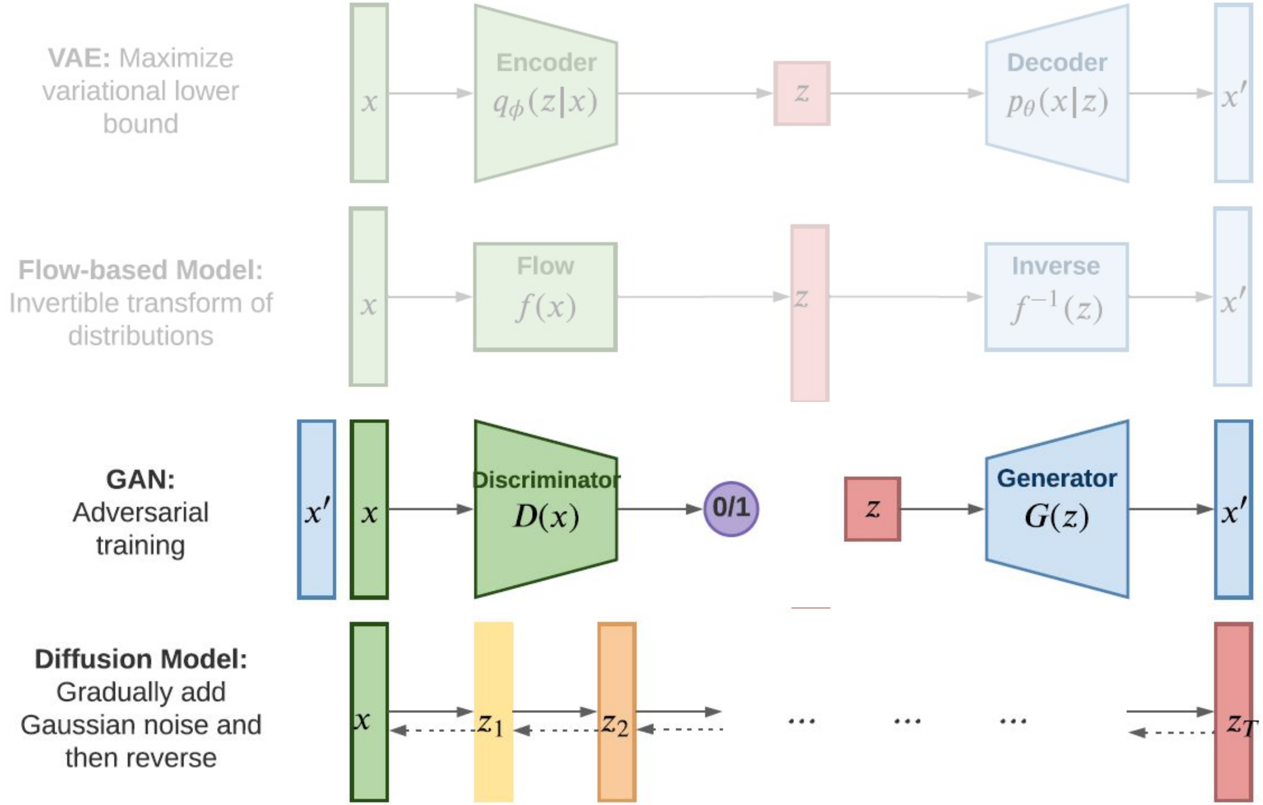
A paper craft art depicting a girl giving her cat a gentle hug. Both sit amidst potted plants, with the cat purring contentedly while the girl smiles. The scene is adorned with handcrafted paper flowers and leaves.

DALL·E 3

Tiny potato kings wearing majestic crowns, sitting on thrones, overseeing their vast potato kingdom filled with potato subjects and potato castles.

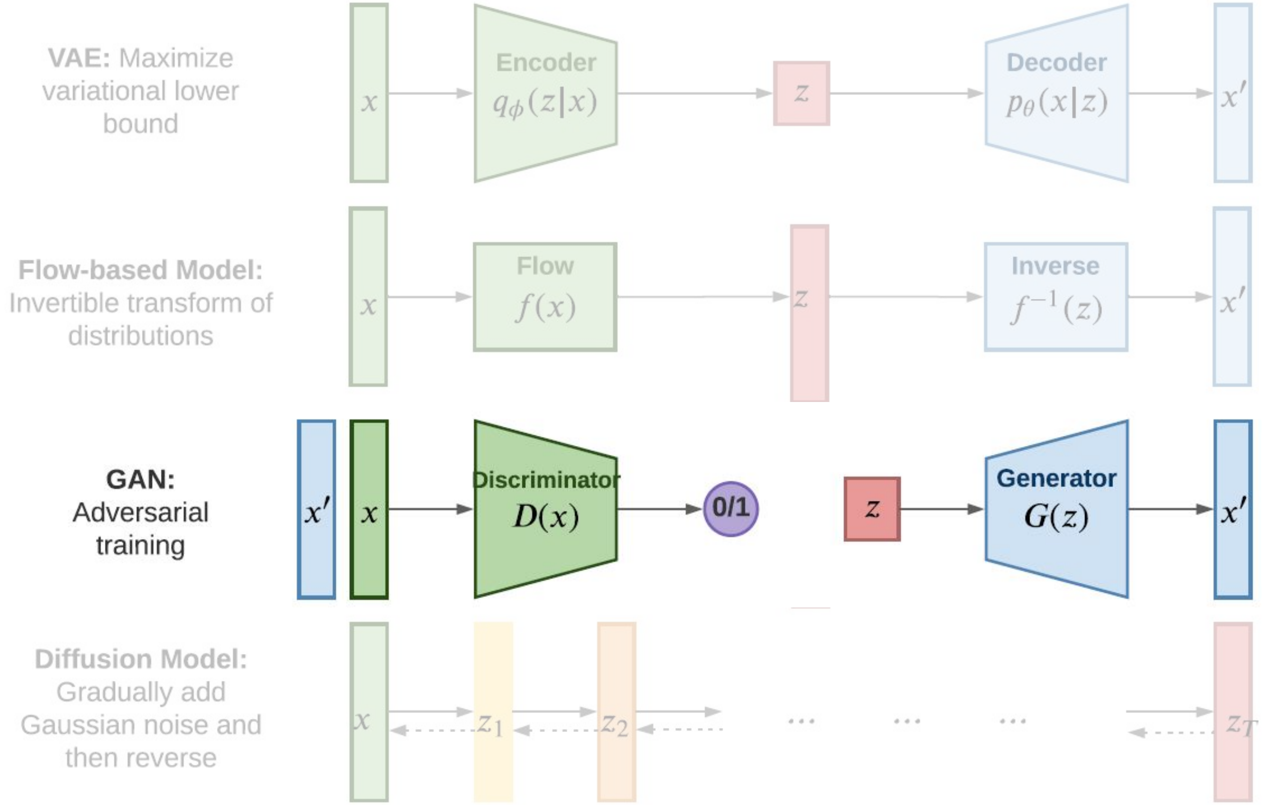
https://openai.com/index/dall-e-3/

Deep Generative Models: Part II



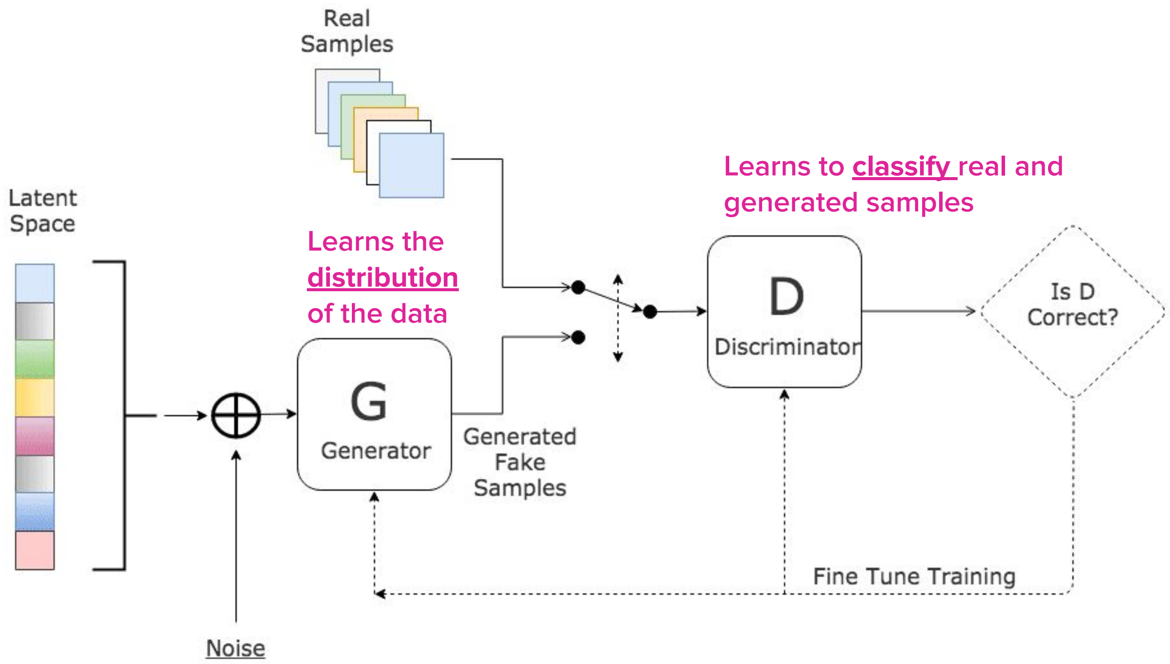
Probabilistic Machine Learning: Section 20.2

Deep Generative Models: Part II



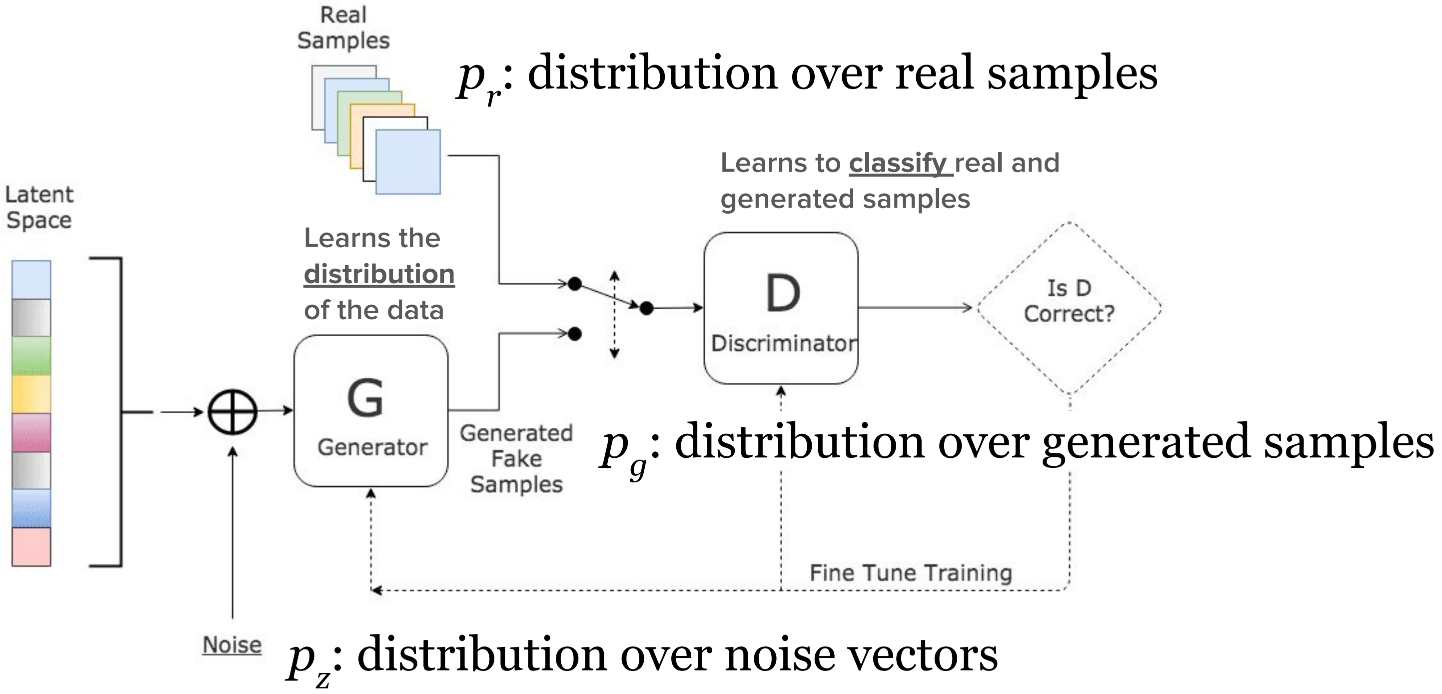
Probabilistic Machine Learning: Section 20.2

A Generative Adversarial Network (GAN) exploits an arms-race between a generator and a discriminator



https://www.kdnuggets.com/2017/01/generative-adversarial-networks-hot-topic-machine-learning.html

A Generative Adversarial Network (GAN) exploits an arms-race between a generator and a discriminator

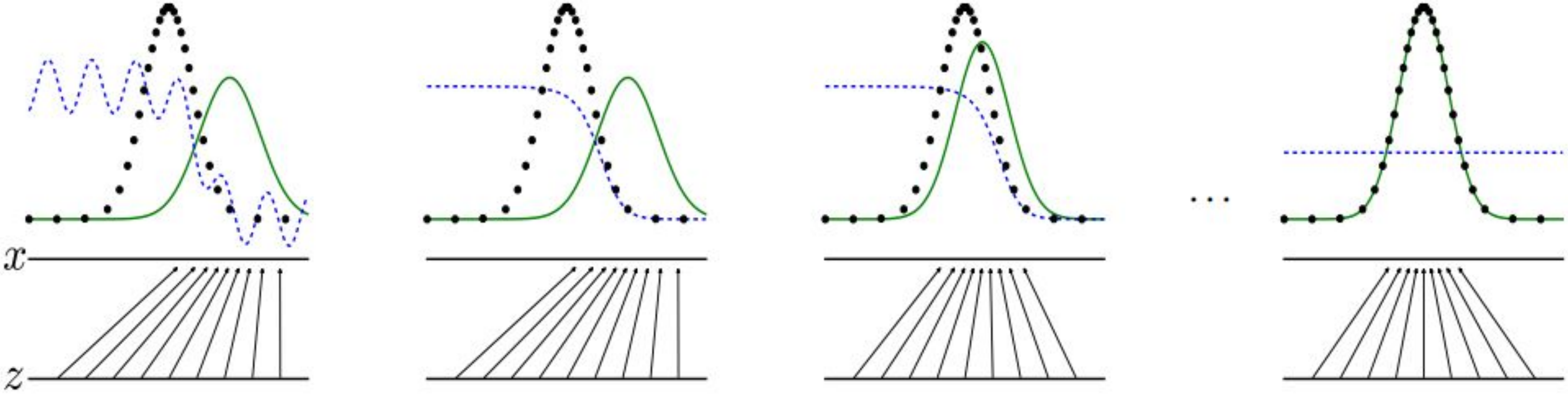


https://www.kdnuggets.com/2017/01/generative-adversarial-networks-hot-topic-machine-learning.html

p_r: distribution over real samples

p_g: distribution over generated samples

Discriminative distribution



Goodfellow et al. 2014

GAN learning proceeds as a “minimax” game

Discriminator goals:

Real samples: $x \sim p_r(x)$

$$\text{Maximize } \mathbb{E}_{x\sim p_r(x)}[\log D(x)]$$

Generated samples: $G(z), z \sim p_z(z)$

$$\text{Maximize } \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$

Generator goals:

$$\text{Maximize } \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$

p_r: Distribution over real samples

p_g: Distribution over generated samples

p_z: Distribution over noise vectors

$D(x)$: Probability emitted by discriminator that x is real

$G(z)$: Sample emitted by generator for noise vector z

$$\begin{aligned}\min_G \max_D L(D,G) &= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \\ &= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log(1 - D(x))]\end{aligned}$$

Convergence is guaranteed under convexity assumption

$$
\begin{aligned}
L(G, D) &= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log(1 - D(x))] \\
&= \int_{\mathcal{X}} p_r(x)\log(D(x))dx + \int_{\mathcal{X}} p_g(x)\log(1 - D(x))dx \\
&= \int_x \left(p_r(x)\log(D(x)) + p_g(x)\log(1 - D(x))\right) dx
\end{aligned}
$$

Letting $\tilde{x} = D(x)$, and assuming the integral can be ignored under the assumption of sufficiently large samples

$$f(\tilde{x}) = p_r(x)\log\tilde{x} + p_g(x)\log(1-\tilde{x})$$

$$
\begin{aligned}
\frac{df(\tilde{x})}{d\tilde{x}} &= p_r(x)\cdot\frac{1}{\ln 10}\cdot\frac{1}{\tilde{x}} - p_g(x)\cdot\frac{1}{\ln 10}\cdot\frac{1}{1-\tilde{x}} \\
&= \frac{1}{\ln 10}\left(\frac{p_r(x)}{\tilde{x}} - \frac{p_g(x)}{1-\tilde{x}}\right) \\
&= \frac{1}{\ln 10}\cdot\frac{p_r(x) - (p_r(x) + p_g(x))\,\tilde{x}}{\tilde{x}(1-\tilde{x})}
\end{aligned}
$$

$$
\begin{aligned}
&\frac{df(\tilde{x})}{d\tilde{x}} = 0 \\
\Longrightarrow\ &\tilde{x}^* = \frac{p_r(x)}{p_r(x) + p_g(x)}
\end{aligned}
$$

The optimal discriminator $D*$ will emit

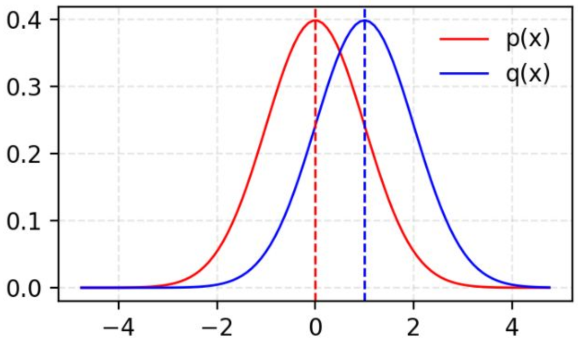
$$D^*(x) = \frac{p_r(x)}{p_r(x) + p_g(x)} \in [0, 1].$$

As the optimal generator will achieve $p_g = p_r$,

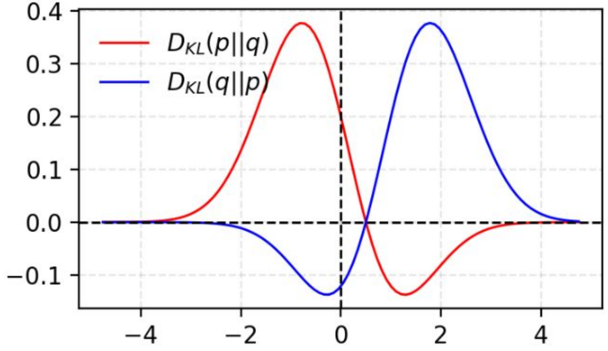
$$
\begin{aligned}
D^*(x) &= \frac{p_r(x)}{p_r(x) + p_r(x)} \\
&= \frac{1}{2}.
\end{aligned}
$$

$$\Longrightarrow\ L(G, D*) = -2\log 2$$

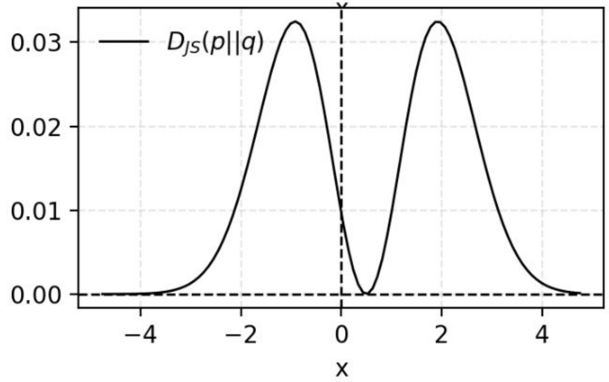
Jensen-Shannon Divergence is a symmetric measure of similarity between probability distributions



$$D_{KL}(p\|q) = \int_x p(x) \log \frac{p(x)}{q(x)} dx$$



$$D_{JS}(p\|q) = \frac{1}{2} D_{KL}(p\|\frac{p+q}{2}) + \frac{1}{2} D_{KL}(q\|\frac{p+q}{2})$$



https://lilianweng.github.io/posts/2017-08-20-gan/

The optimal discriminator computes the Jensen-Shannon divergence between the real and generated distributions

$$
\begin{aligned}
D_{\mathrm{JS}}(p_r \| p_g) &= \frac{1}{2} D_{\mathrm{KL}}\left(p_r \middle\| \frac{p_r + p_g}{2}\right) + \frac{1}{2} D_{\mathrm{KL}}\left(p_g \middle\| \frac{p_r + p_g}{2}\right) \\
&= \frac{1}{2}\left(\log 2 + \int_x p_r(x) \log \frac{p_r(x)}{p_r(x) + p_g(x)}\, dx\right) \\
&\quad + \frac{1}{2}\left(\log 2 + \int_x p_g(x) \log \frac{p_g(x)}{p_r(x) + p_g(x)}\, dx\right) \\
&= \frac{1}{2}\left(\log 4 + L(G, D^*)\right)
\end{aligned}
$$

where:

$$
L(G, D^*) = 2D_{\mathrm{JS}}(p_r \| p_g) - 2\log 2
$$

https://lilianweng.github.io/posts/2017-08-20-gan/

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 $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$ 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 $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

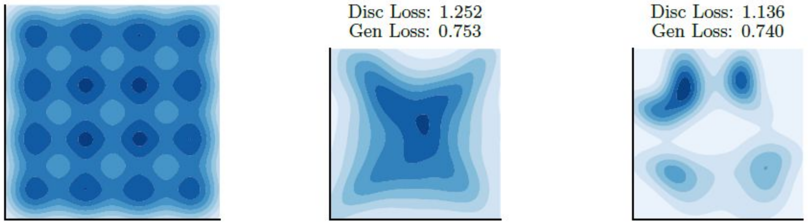
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

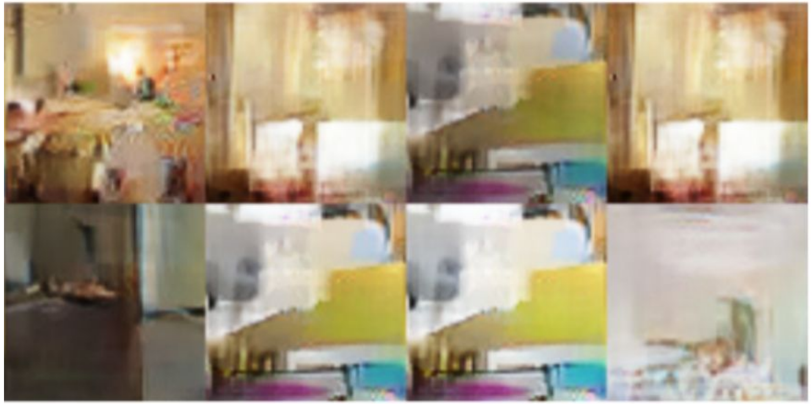
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Goodfellow et al. 2014

Mode collapse/hopping characterizes a loss of diversity



(a) Data *(b) 2000 iterations* *(c) 4000 iterations*



Suspected causes

- Generator learns too fast relative to generator

Improvements

- Update generator less frequently
- Increase batch size
- Increase neural capacity of the discriminator
- More complex optimization methods (e.g. Adam)
- Regularization

Top: PML2, Section 26.3
Bottom: Arjovsky, Chintala, and Bottou, 2017

More issues with training GANs

- Vanishing gradients
  - Discriminator learns too fast
  - The generator cannot learn from a perfect discriminator
- Non-convex objective using neural networks
  - Equilibria of the minimax game are not local minima, but saddle points
  - Nonconvergence will may result in underfitting

Wasserstein GAN is a significant variant

  - Uses an efficient computation of Wasserstein ("Earth-Mover's") distance as the training objective

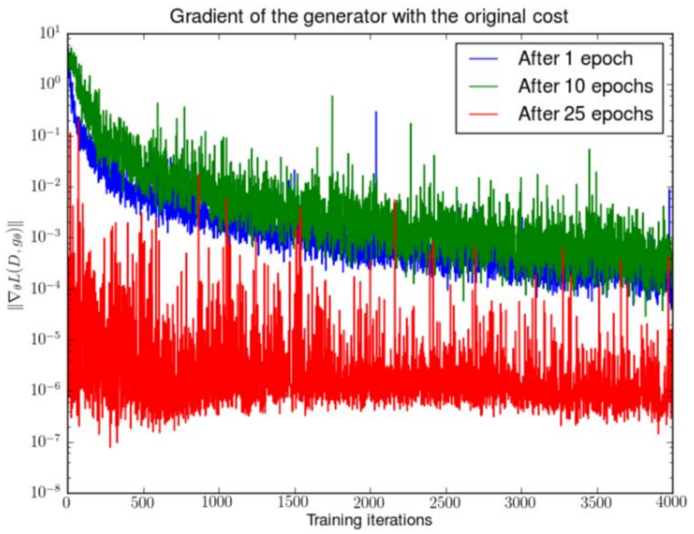
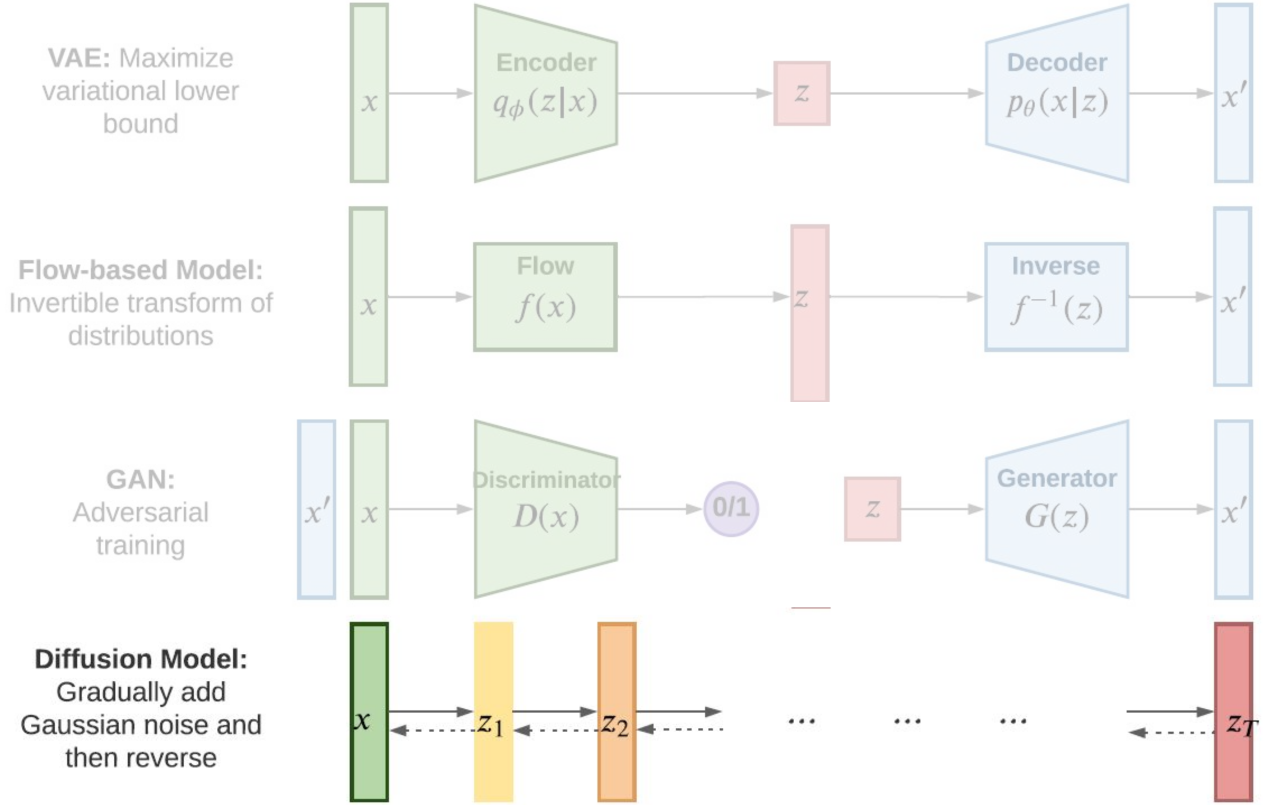


Figure: Arjovsky and Bottou, 2017


WASSERSTEIN
VAHSERSCHTEIN
imgflip.com

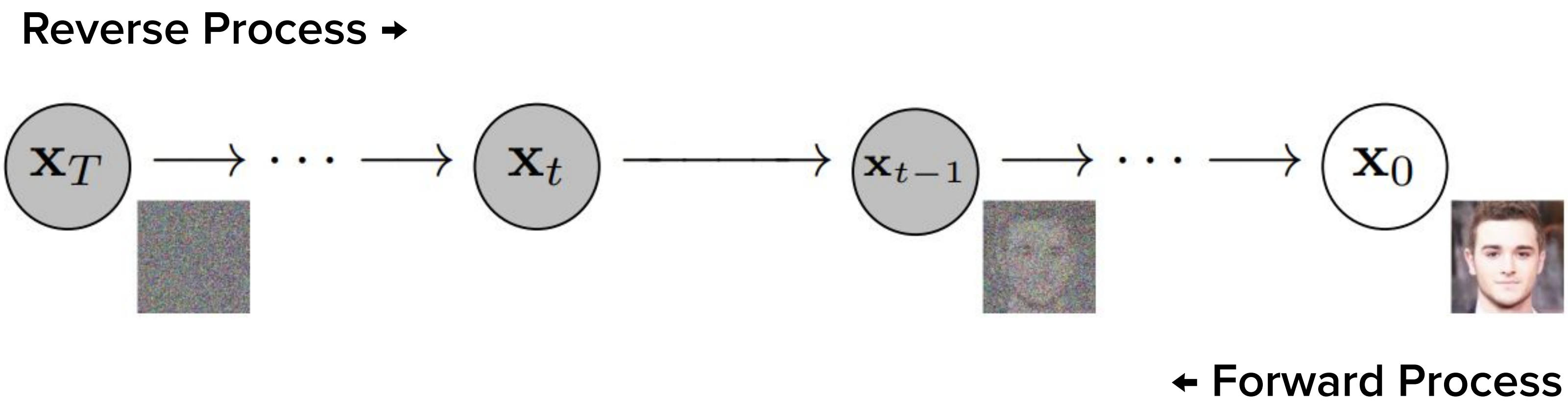
Deep Generative Models: Part II



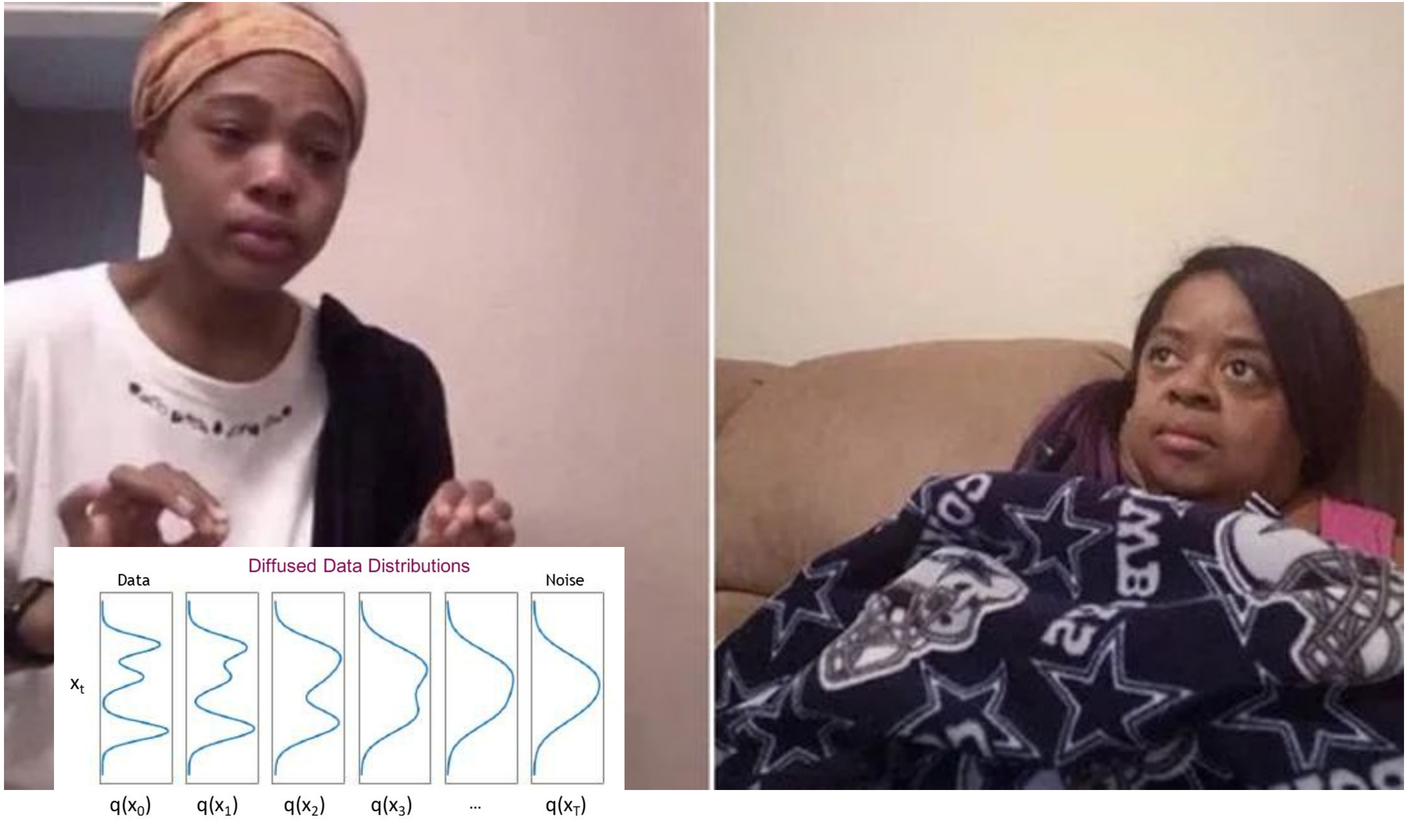
Probabilistic Machine Learning: Section 20.2

Diffusion models are latent variables models that generate through an iterative denoising process

Diffusion models learn to reverse the process of iteratively added noise

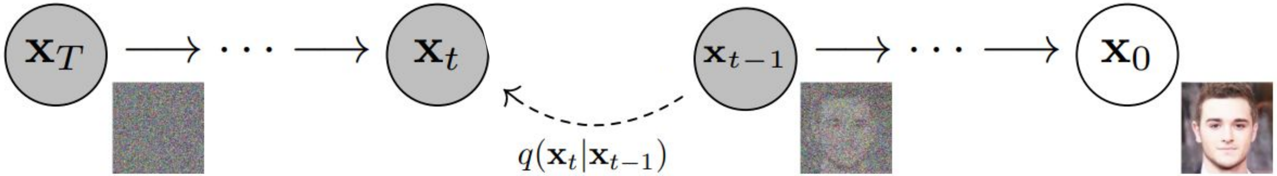


Ho, Jain, and Abbeel, 2020



Diffused Data Distributions
Data
Noise
x_t
$q(x_0)$
$q(x_1)$
$q(x_2)$
$q(x_3)$
...
$q(x_T)$

The forwards process adds Gaussian noise



The forward process adds Gaussian noise:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t \mid \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}\right)$$

$\beta_t \in (0,1)$: noise schedule parameter

The joint distribution over all the latent states:

Using the chain rule of probability

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

Closed-form expressions for the marginals:

$$q(x_t \mid x_0) = \mathcal{N}\left(x_t \mid \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$$

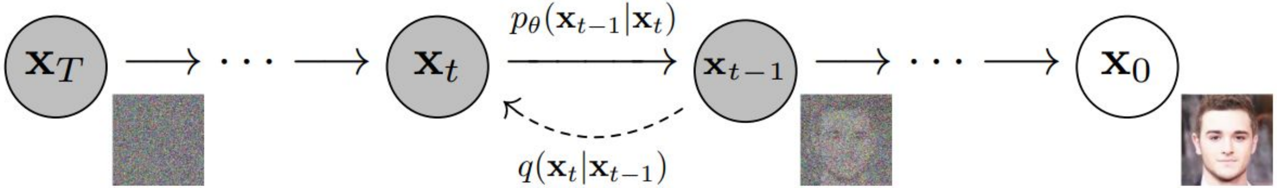
$q(x_t \mid x_0)$ "diffusion kernel"

$$\alpha_t = 1-\beta_t,\ \bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$$

The noise schedule is such that $\bar{\alpha}_T \approx 0$, so that:

$$q(x_T \mid x_0) \approx \mathcal{N}(0, \mathbf{I})$$

Parametrization of the reverse process



Reverse process: $q(x_{t-1} \mid x_t) = ?$

Given x_0, the reverse of one forwards step is:

$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}\left(x_{t-1} \mid \tilde{\mu}(x_t, x_0), \tilde{\beta}_t \mathbf{I}\right)$$

$$\tilde{\mu}(x_t, x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t$$

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

The reverse process parameterized by θ is:

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}\left(x_{t-1} \mid \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\right)$$

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$$

Often, we select $\Sigma_\theta(x_t, t) = \sigma_t^2 \mathbf{I}$
With $\sigma_t^2 = \beta_t$ or $\sigma_t^2 = \tilde{\beta}_t$

Given $p(x_T) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

Simplified training objective of the reverse process

$$L_{t-1}(x_0) = D_{\mathrm{KL}}\left(q(x_{t-1} \mid x_t, x_0) \| p_\theta(x_{t-1} \mid x_t)\right)$$

True reverse process: $q(x_{t-1} \mid x_t)$
Estimated reverse process: $p_\theta(x_{t-1} \mid x_t)$

Since:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$$

$$\epsilon \sim \mathcal{N}(0, \mathbf{I})$$

True reverse mean:

$$\tilde{\mu}_t(x_t, x_0) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}}\epsilon\right)$$

Estimated reverse mean:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right)$$

Sampling from and Training a DDPM

Algorithm 1 Training

repeat
  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
  $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  Take gradient descent step on
    $\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$
until converged

Algorithm 2 Sampling

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
end for
return $\mathbf{x}_0$

Ho, Jain, and Abbeel, 2020

GANs vs Diffusion models

GANs

- Fast generation
- Discriminator can be discarded after training
- Precarious optimization

Diffusion models

- Can generate very high-quality images
- Sampling is necessarily sequential: slow to generate
- More robust training

Now that we're at the end of the lecture, you should be able to...

- ★ State the **theoretical guarantees** associated with training a GAN (**unique equilibrium point, behavior of the optimal discriminator**), and the caveats that undermine these guarantees being met in practice.
- ★ Differentiate **Jensen-Shannon divergence** and **Kullback-Leibler divergence**.
- ★ Diagnose issues with training a GAN (**mode-collapse, vanishing gradient**) and recommend solutions.
- ★ Discriminate the **forward and reverse processes** of a diffusion model.
- ★ Recommend either a GAN or a diffusion model for particular image-generation applications, taking into account the **tradeoffs between detail, computation time,** and **ease of optimization**.

Housekeeping

Lectures 20-22 are guest lectures

- **Schedule**
  - Tuesday, Nov. 26: Dr. P. Michael Furlong, *Robustness*
  - Thursday, Nov. 28: Saber Malekmohammadi, *Privacy*
  - Tuesday, Dec. 3: Dr. Terrence C. Stewart, *Fairness & Safety*
- **Material will be assessed on the final exam**
- **In-class time will be recorded and uploaded to YouTube**
  - By participating in class, you consent to having these interactions recorded and published

Student Course Perception Survey Now Open

- **CS 480 / CS 680 section 001**
  [Wed, Nov 20 8:30 a.m. to Tue, Dec 3 11:59 p.m.]

- **CS 480 / CS 680 section 002**
  [Wed, Nov 20 8:30 a.m. to Tue, Dec 3 11:59 p.m.]