CS492D: Diffusion Models and Their Applications

Introduction to Generative Models: GAN and VAE

LECTURE 2
MINHYUK SUNG

Fall 2024
KAIST

https://pixabay.com/photos/pictures-photos-photo-collection-382009/

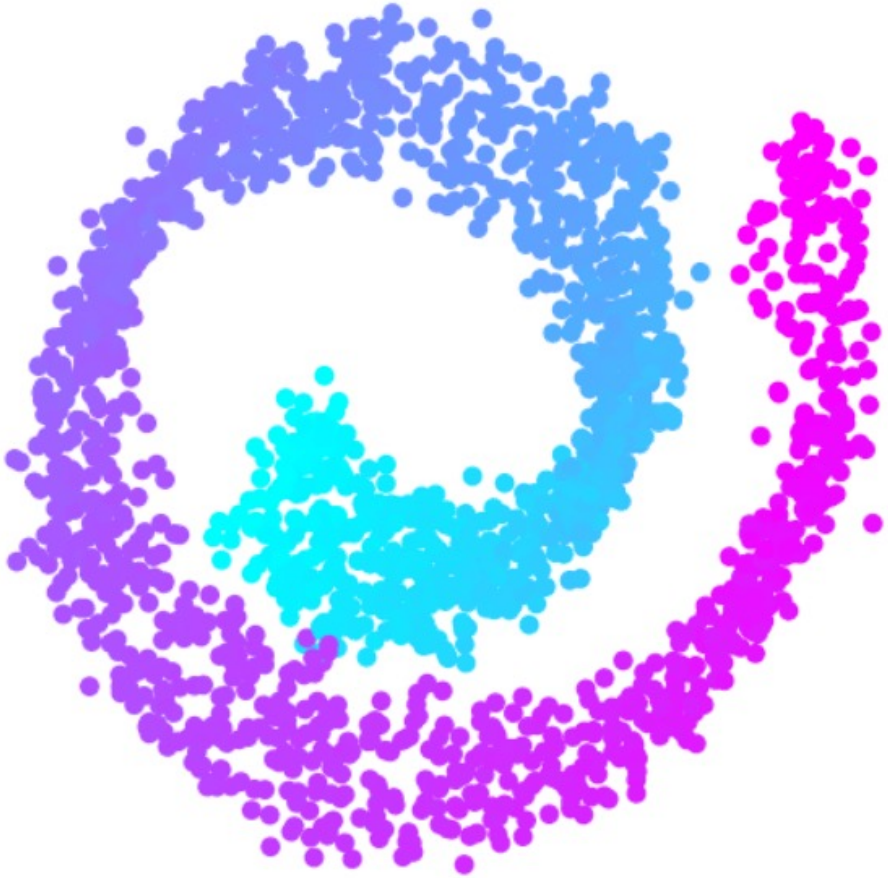
Let's consider a collection of real photos.

https://pixabay.com/photos/pictures-photos-photo-collection-382009/

How can we generate a new realistic photo?

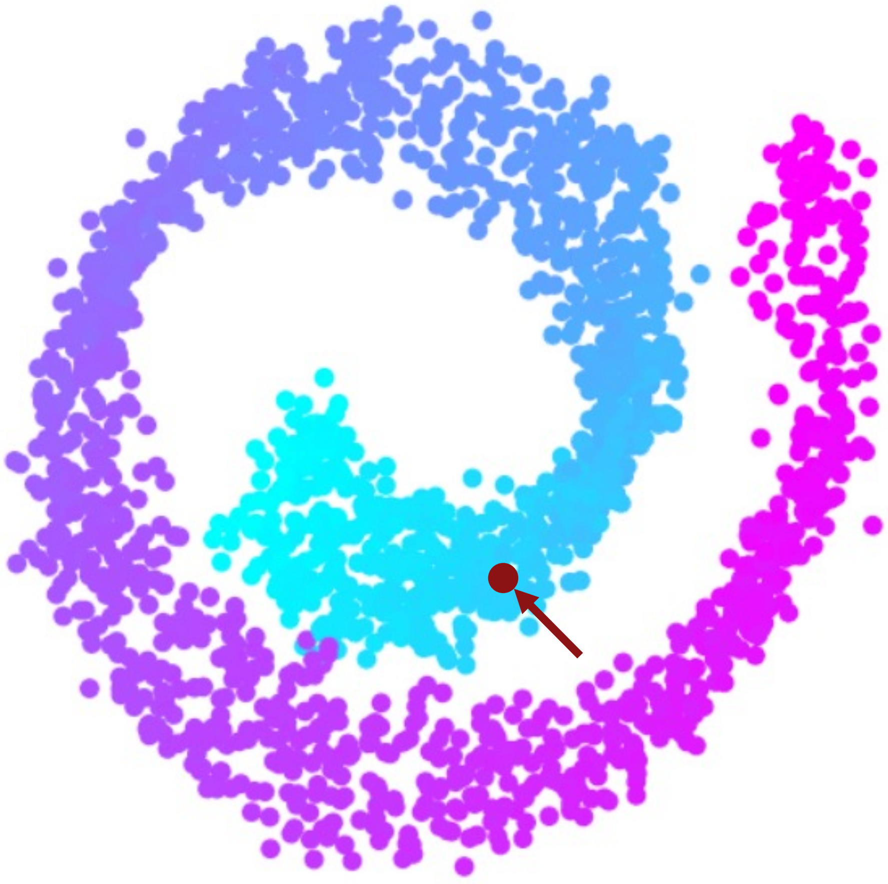
https://www.geeksforgeeks.org/swiss-roll-reduction-with-lle-in-scikit-learn/

Let's consider a simpler case: a collection of 2D points.



https://www.geeksforgeeks.org/swiss-roll-reduction-with-lle-in-scikit-learn/

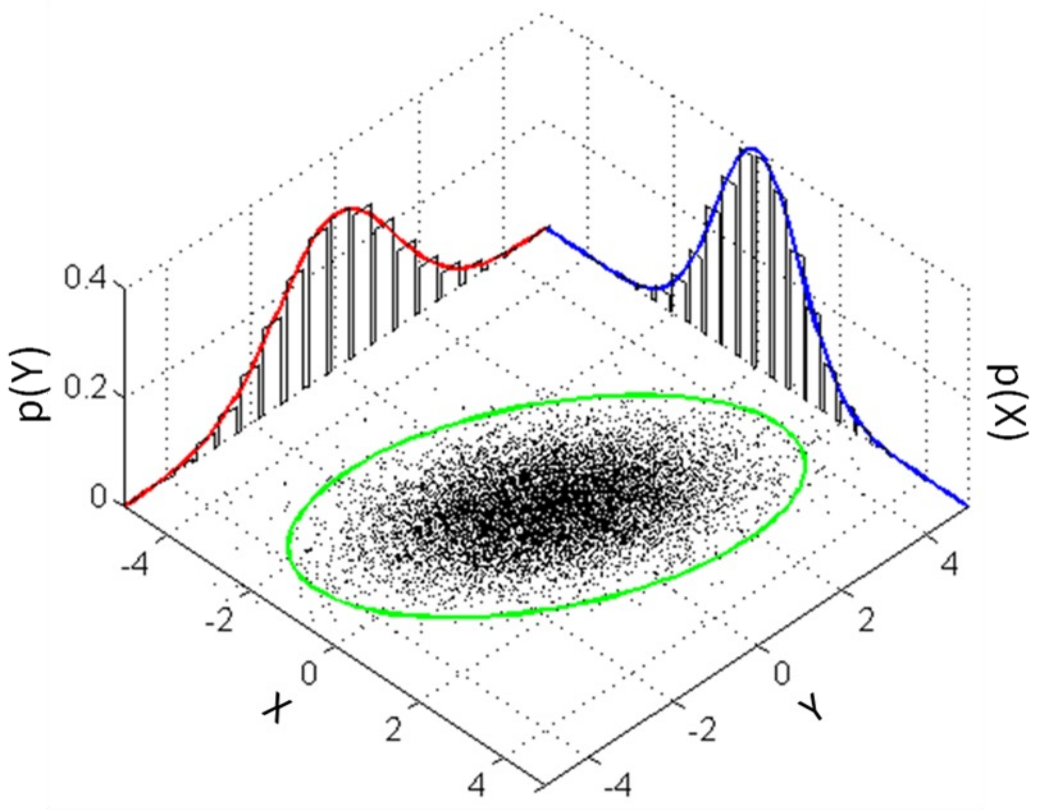
How can we sample a new plausible 2D point?



Statistical Perspective

From a statistical perspective, we will view this as

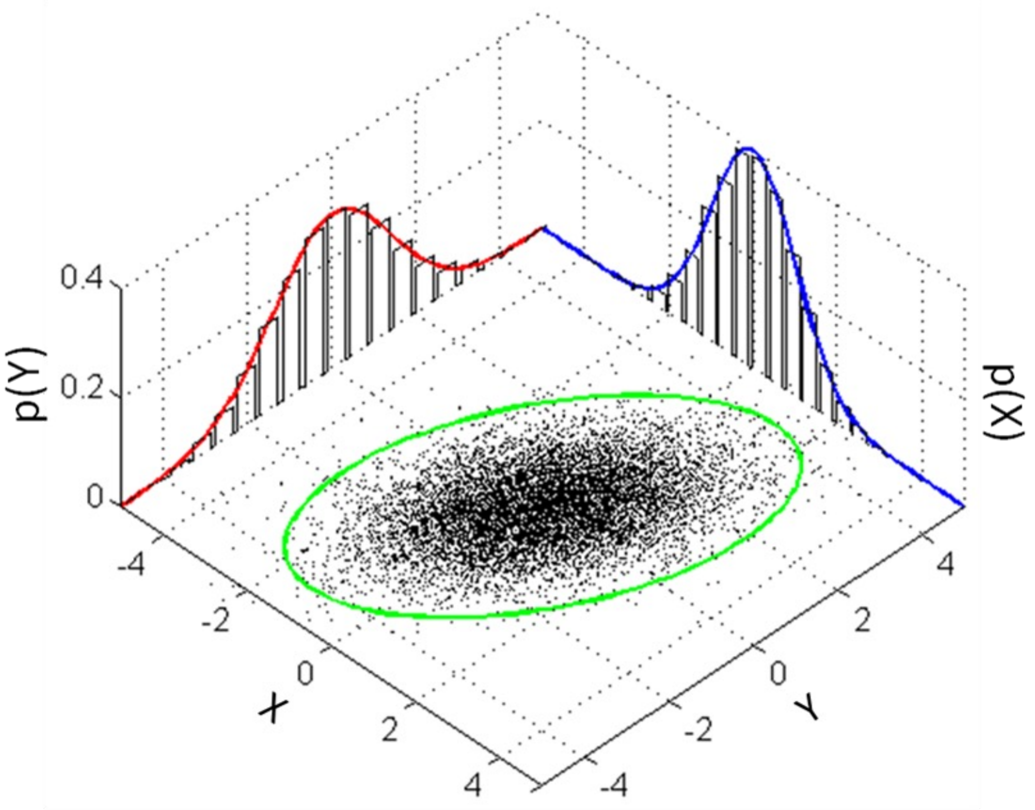
- there being a probability distribution of the data, and
- the given points are samples from the probability distribution.



Statistical Perspective

If the probability density function (PDF) of the distribution is known, we can sample from it directly.

How?

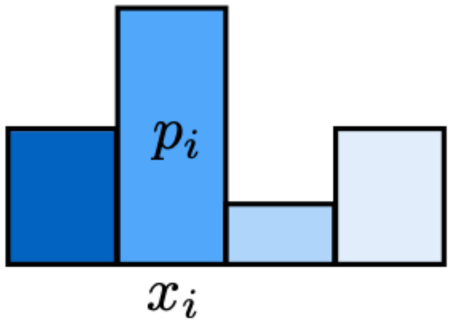


Discrete Probability Distributions

- n discrete values x_i with probability p_i.
- Requirements of a PDF:

$$p_i \geq 0$$

$$\sum_{i=1}^{n} p_i = 1$$



Q. How can we sample based on this PDF?

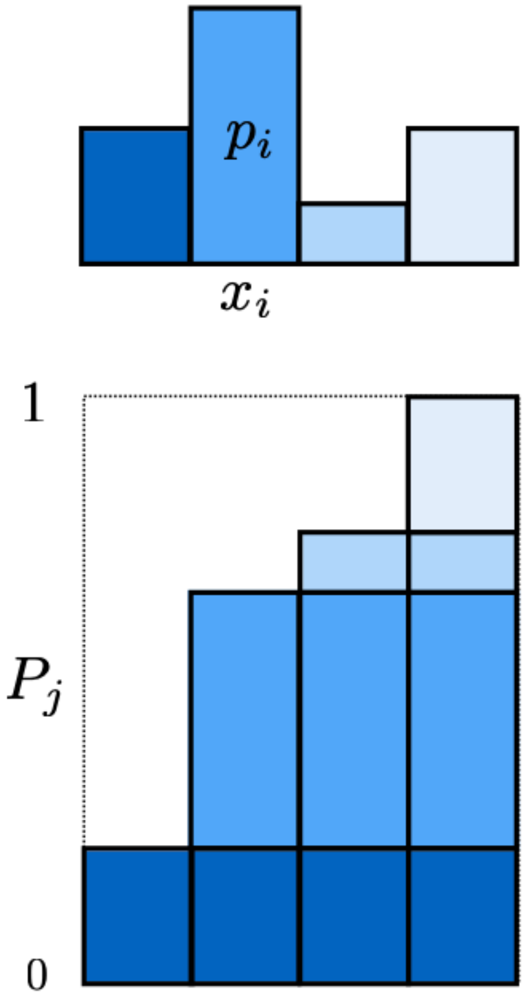
Cumulative Distribution Function (CDF)

Cumulative distribution function:

$$P_i = \sum_{j=1}^{i} p_j$$

where

$$0 \leq P_i \leq 1$$
$$P_n = 1$$



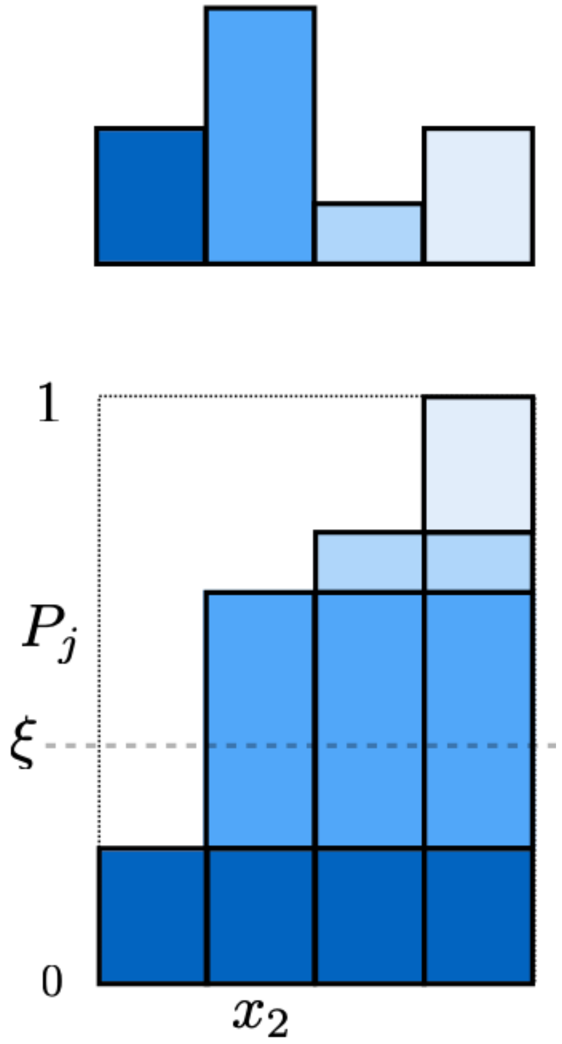
Inverse Transform Sampling

To randomly select an event,

- Draw $u \sim \mathcal{U}(0,1)$.

- Take x_i if

$$P_{i-1} \leq u \leq P_i$$



Continuous Probability Distributions

Given a PDF $p(x)$,

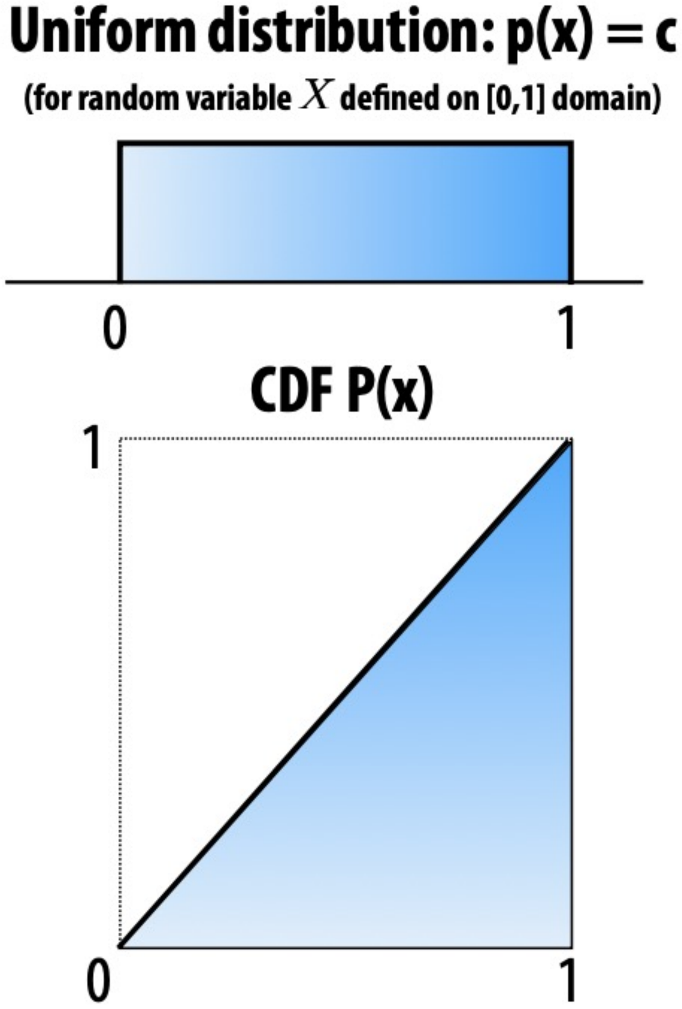
- **CDF** $F_X(x)$

$$= \Pr(X \leq x) = \int_0^x p(t)\mathrm{d}t$$

- $\Pr(a < X \leq b)$

$$= \int_a^b p(t)\mathrm{d}t = F_X(b) - F_X(a)$$

- $F_X(1) = 1$

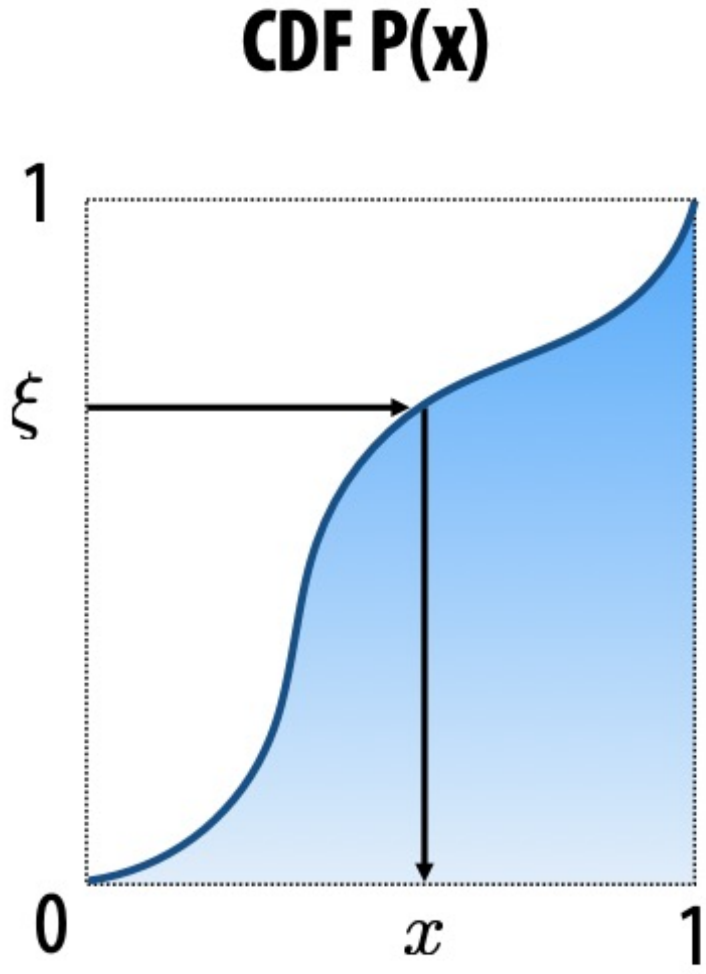


Inverse Transform Sampling

To randomly sample based on the given PDF,

- Compute **CDF** $F_X(x)$
- Draw $u \sim \mathcal{U}(0,1)$.
- Take $x = F_X^{-1}(u)$.

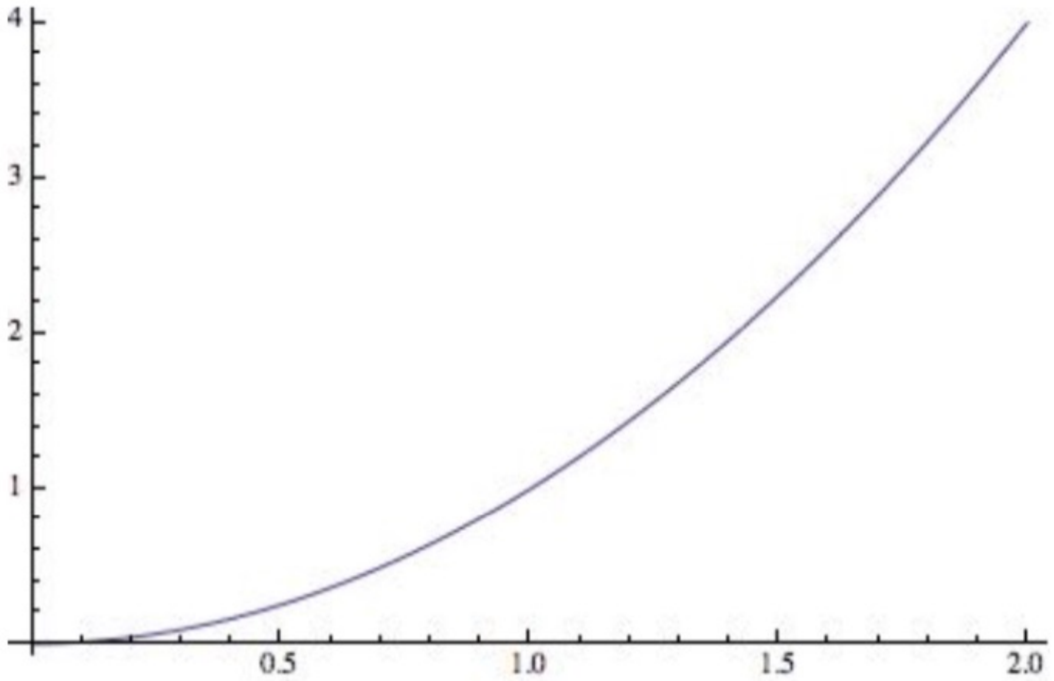
Need to know the inverse function $F_X^{-1}(x)$.



Inverse Transform Sampling – Example

PDF

$$p(x) = \frac{3}{8}x^2 \qquad x \in [0, 2]$$



Q. What is the inverse of CDF?

Inverse Transform Sampling – Example

PDF

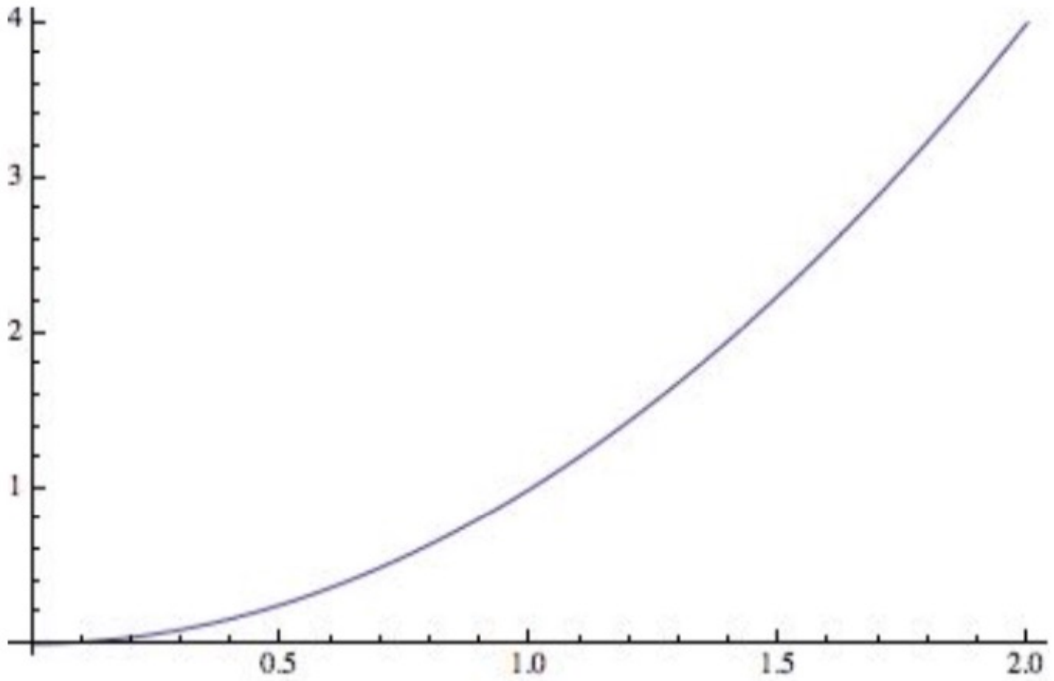
$$p(x) = \frac{3}{8}x^2 \qquad x \in [0, 2]$$

CDF

$$F_X(x) = \int_0^x p(x)\mathrm{d}x = \frac{1}{8}x^3$$

Inverse of CDF

$$F_X^{-1}(x) = 2\sqrt[3]{x}$$



Inverse Transform Sampling – Example

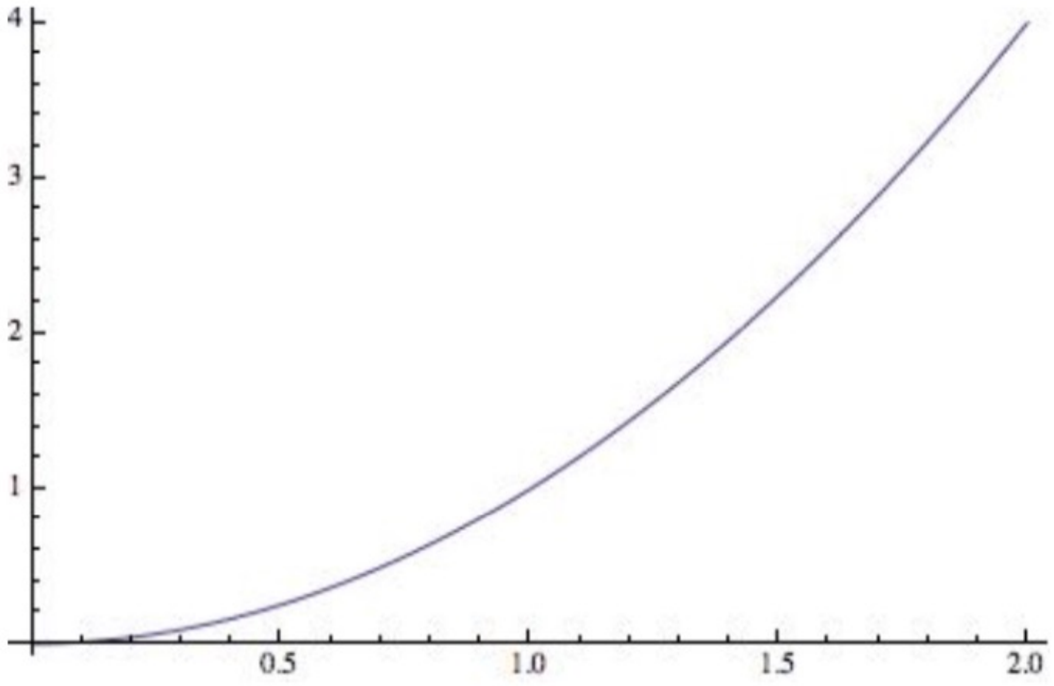
Sampling from

$$p(x) = \frac{3}{8}x^2$$

1. Draw a sample

   $u \sim \mathcal{U}(0, 1).$

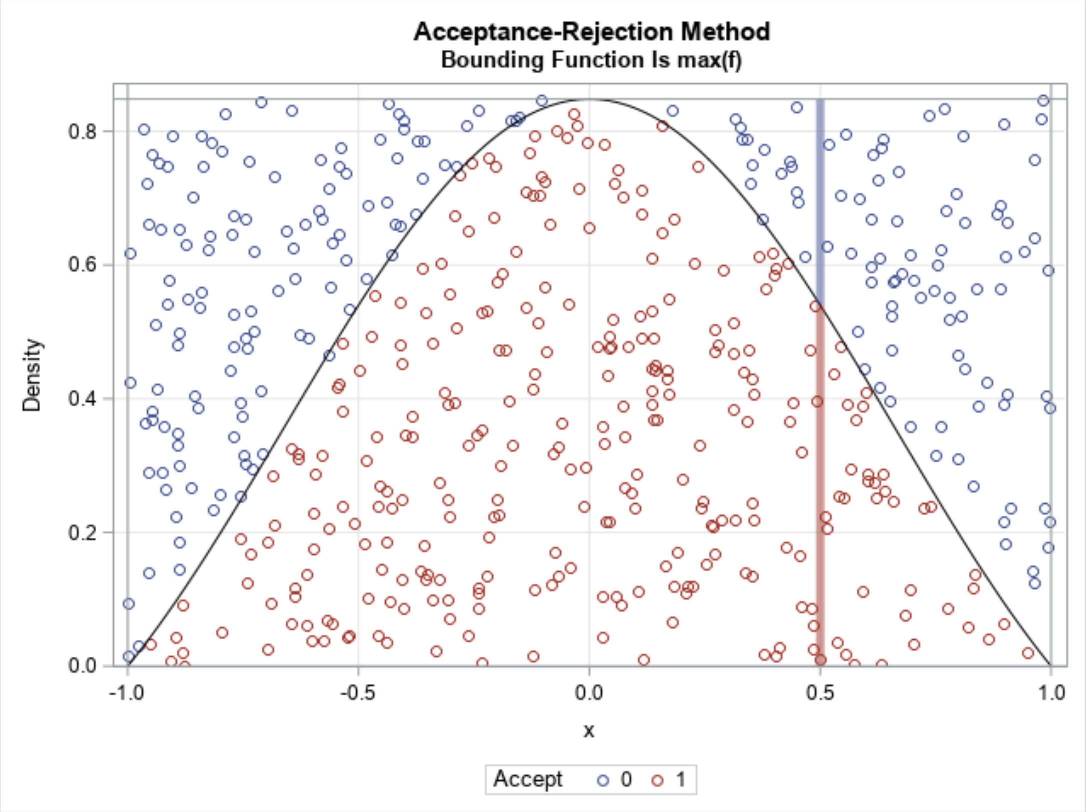
2. Take $2\sqrt[3]{u}$.



Rejection Sampling

If the inverse of the CDF cannot be computed:

1. Let $q(x)$ be an upper bound distribution: $\forall x\ q(x) \geq p(x)$.
2. Draw $x \sim q(x)$.
3. Draw a $h \sim \mathcal{U}(0, q(x))$.
4. Accept the sample x if $h \leq p(x)$; otherwise reject it.



Reparameterization Trick

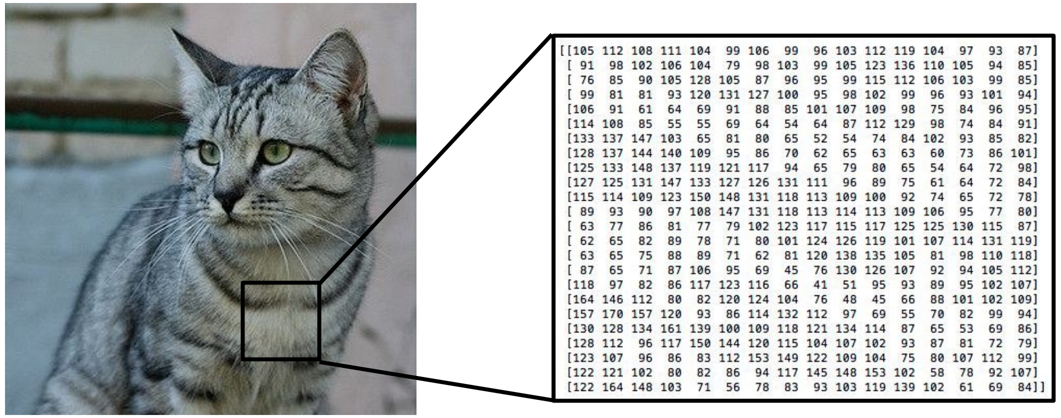
- A sample from a normal distribution $z \sim N(\mu, \Sigma)$ can be rewritten as follows:

$$z = \mu + \Sigma^{\frac{1}{2}}\epsilon \ \text{ where } \epsilon \sim N(0, \mathrm{I}).$$

- We just need a standard normal sampler to sample from an arbitrary normal distribution.

Statistical Perspective for Real Images

- Let's consider RGB images with a resolution of 256×256.
- An image can be represented by a 256×256×3 vector.
- This means that an image is a point in a 256×256×3-dimensional space.

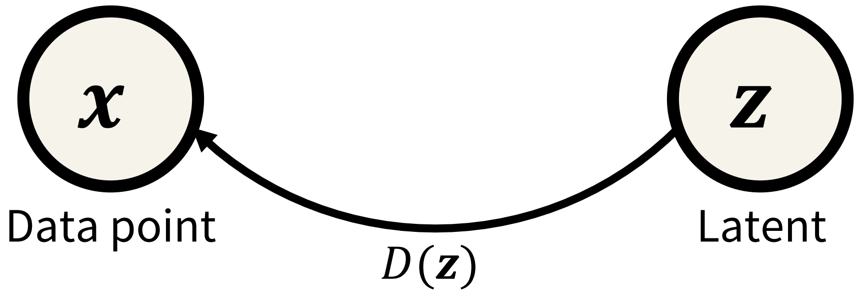


Statistical Perspective for Real Images

- Let us consider real images $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$ as samples from a data distribution $p(\boldsymbol{x})$ that measures how likely it is for an image to be a real photo.
- Can we derive the PDF of the data distribution…?

The Basic Idea

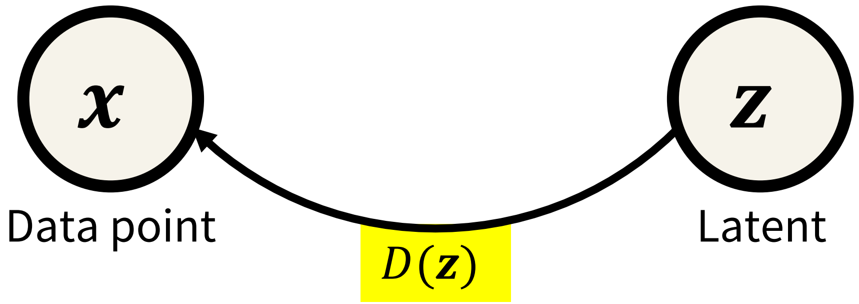
- Map a *simple* distribution $p(\boldsymbol{z})$ (e.g., a standard normal distribution $\mathcal{N}(\boldsymbol{x}; \mathbf{0}, \mathbf{I})$) to the data distribution $p(\boldsymbol{x})$.
    - $\boldsymbol{z}$: Latent variable
    - $p(\boldsymbol{z})$: Latent distribution
- Sample from $p(\boldsymbol{z})$ and map it to a data point.



The Basic Idea

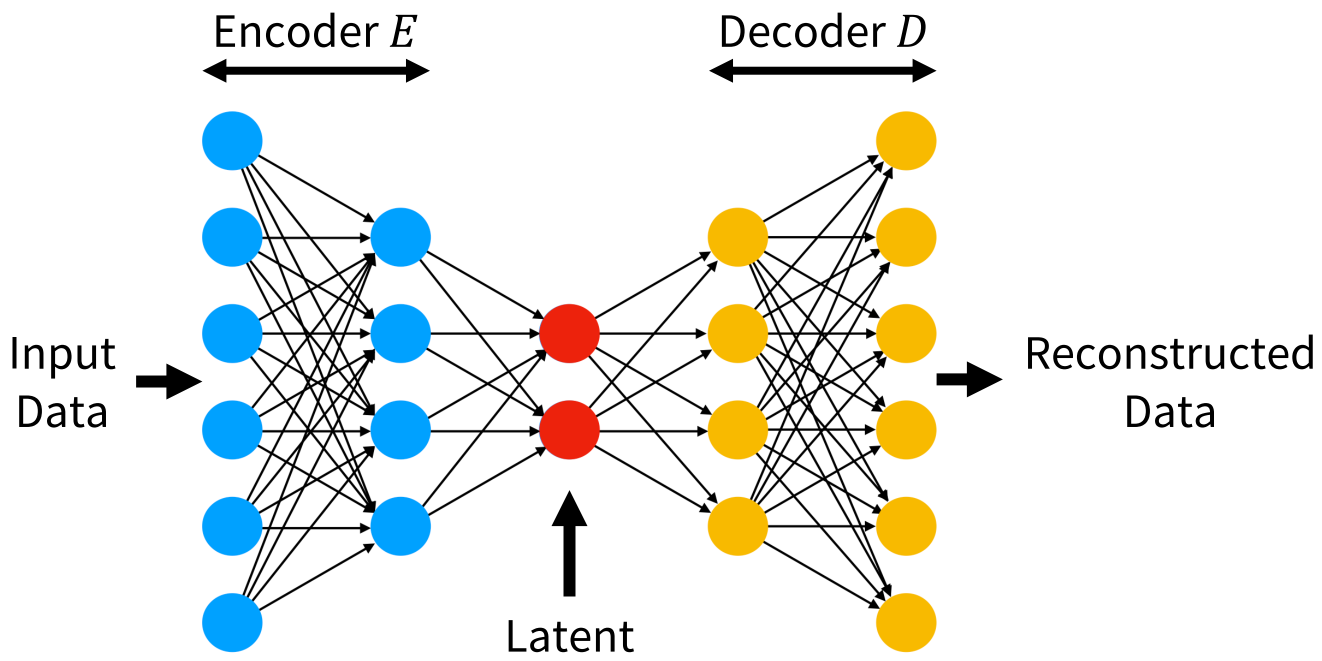
How to map a latent distribution $p(\boldsymbol{z})$ to the data distribution $p(\boldsymbol{x})$?

Let's use a neural network!



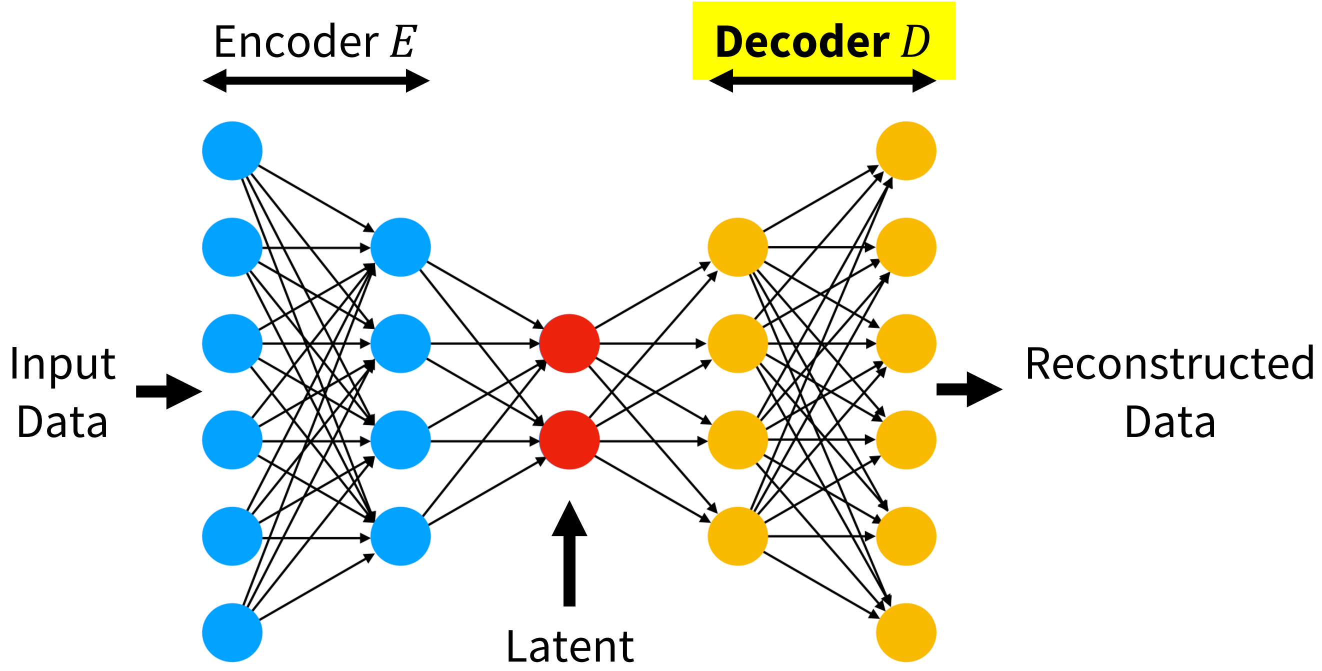
Autoencoder

An autoencoder is a neural network designed to reconstruct the input data while encoding it into a lower-dimensional latent vector.



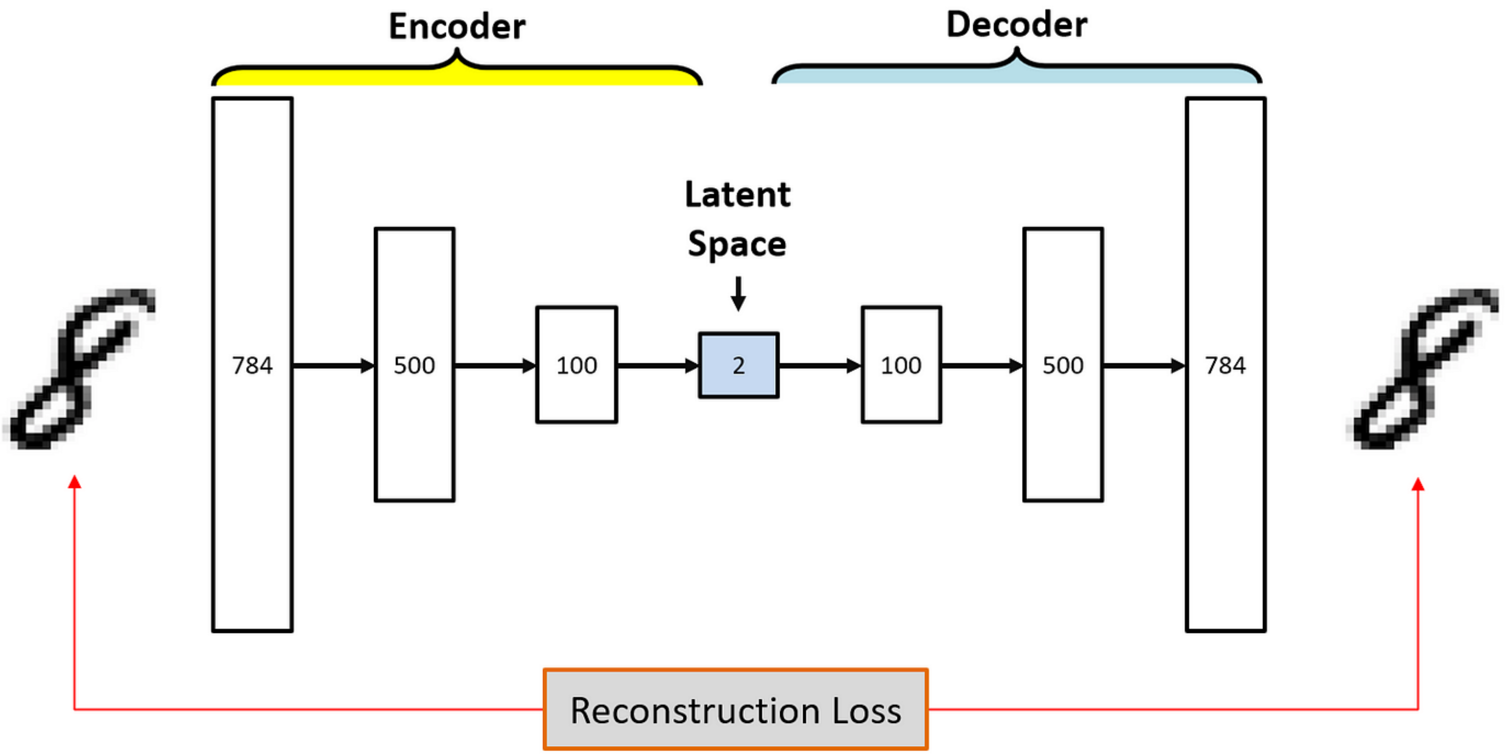
Autoencoder

- What we need is the **decoder** (latent → input data).
- But how do we guarantee that a latent is mapped to a data point of the data distribution?



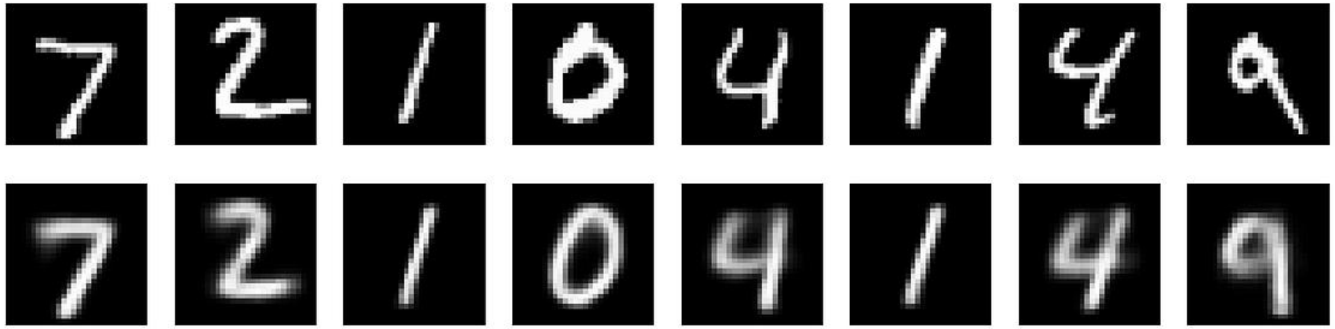
[EXTRA] Autoencoder Example

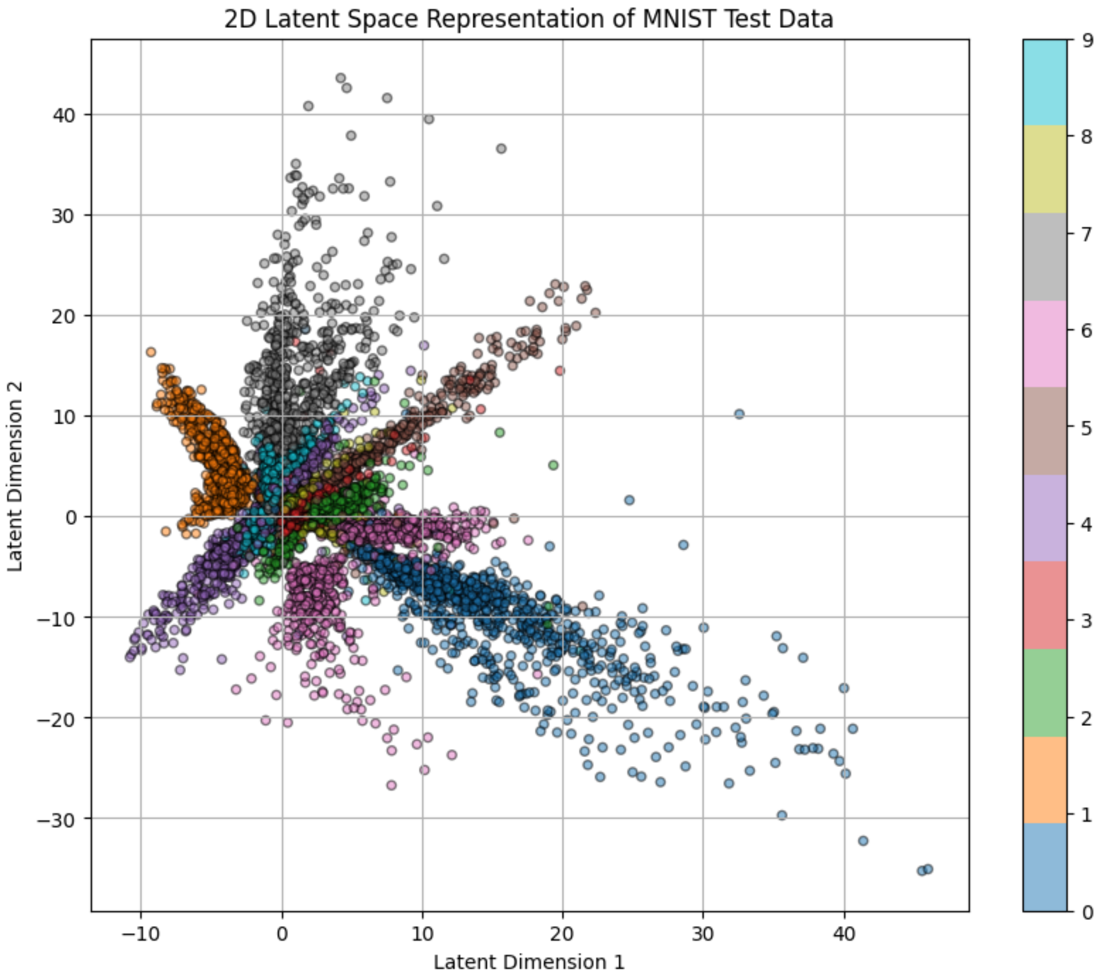
Ex) Latent Dimension = 2



[EXTRA] Autoencoder Example

Ex) Latent Dimension = 2





Two Methods

1. Generative Adversarial Network (GAN)
2. Variational Autoencoder (VAE)

Generative Adversarial Network (GAN)

Goodfellow et al., Generative Adversarial Networks, NeurIPS 2014.

Off-Topic: Adversarial Examples

Neural networks work well, but how vulnerable is a neural network?

Images that are *correctly* classified

Szegedy et al., Intriguing properties of neural networks, arXiv 2013.

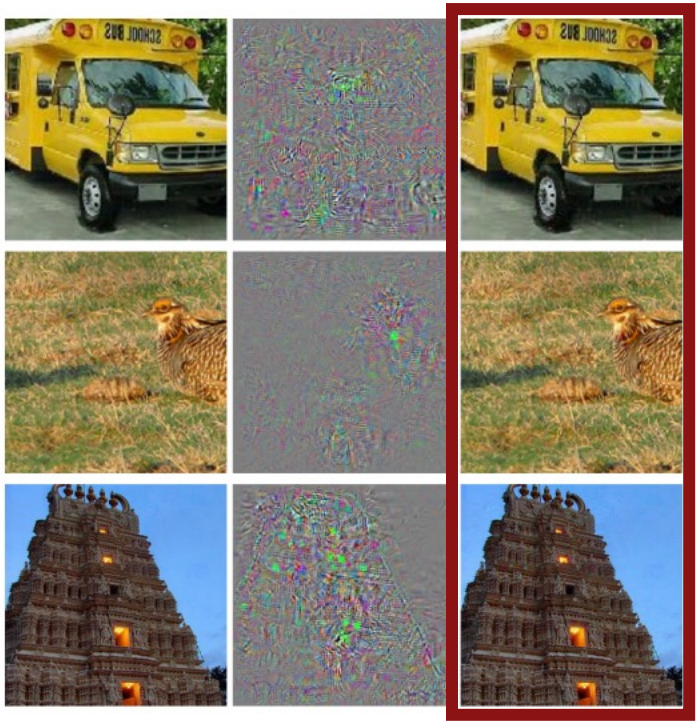
Off-Topic: Adversarial Examples

Neural networks work well, but how vulnerable is a neural network?

Difference between the two multiplied by $10\times$

Off-Topic: Adversarial Examples

Neural networks work well, but how vulnerable is a neural network?



Images that are *incorrectly* classified as ostrich

Wikipedia

Off-Topic: Adversarial Examples

- Let's train a network to predict an image that causes the classifier to fail. Adversarial attack!
- Can we also finetune the classifier to prevent it from failing due to adversarial attacks?
- What happens if we make them compete against each other?

Generative Adversarial Network (GAN)

Think about

- a real/fake image classifier → **Discriminator**
- an adversarial attack network → **Generator (Decoder)**
  that tries to make a real-like generated image.

Generative Adversarial Network (GAN)

- **Generator** (or decoder) $\boldsymbol{G}$ takes a latent sampled from a unit Gaussian as input and generates a synthetic image.

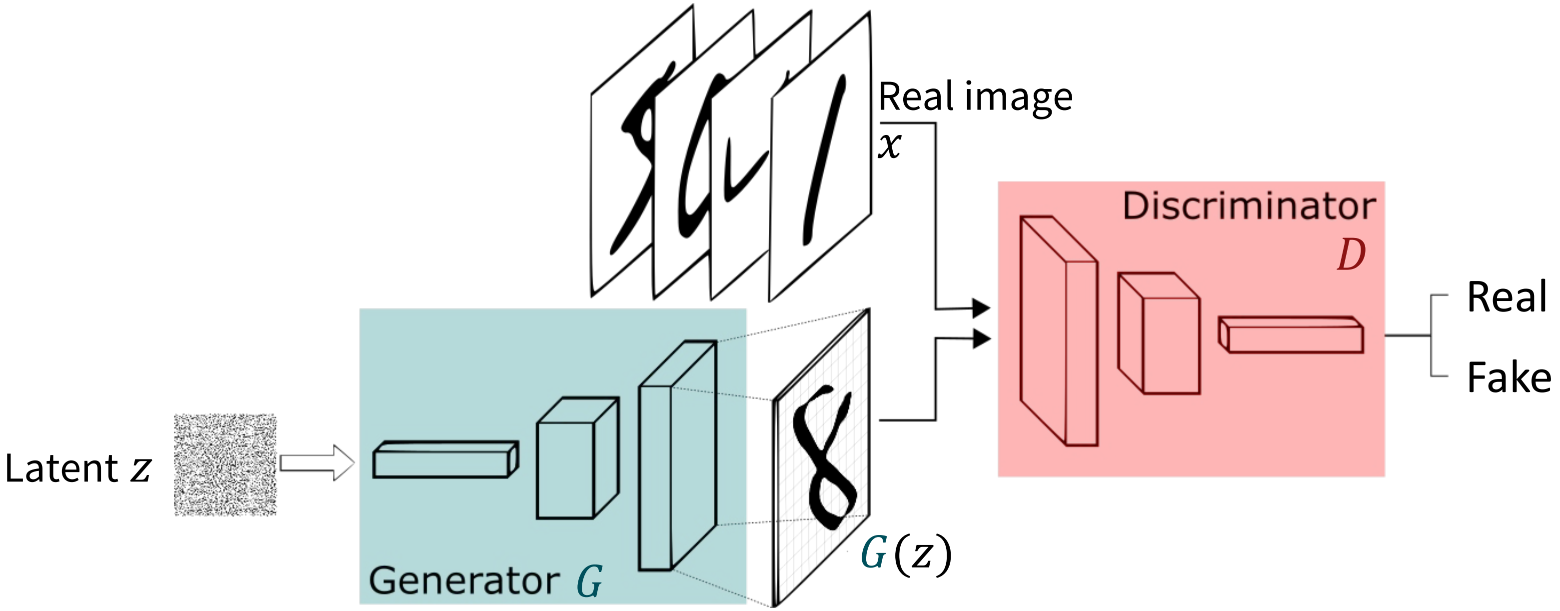
- **Discriminator** $\boldsymbol{D}$ takes an image as input and classifies it as either real or fake (generated).

Make them compete against each other!

Generative Adversarial Network (GAN)

Loss function:

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$



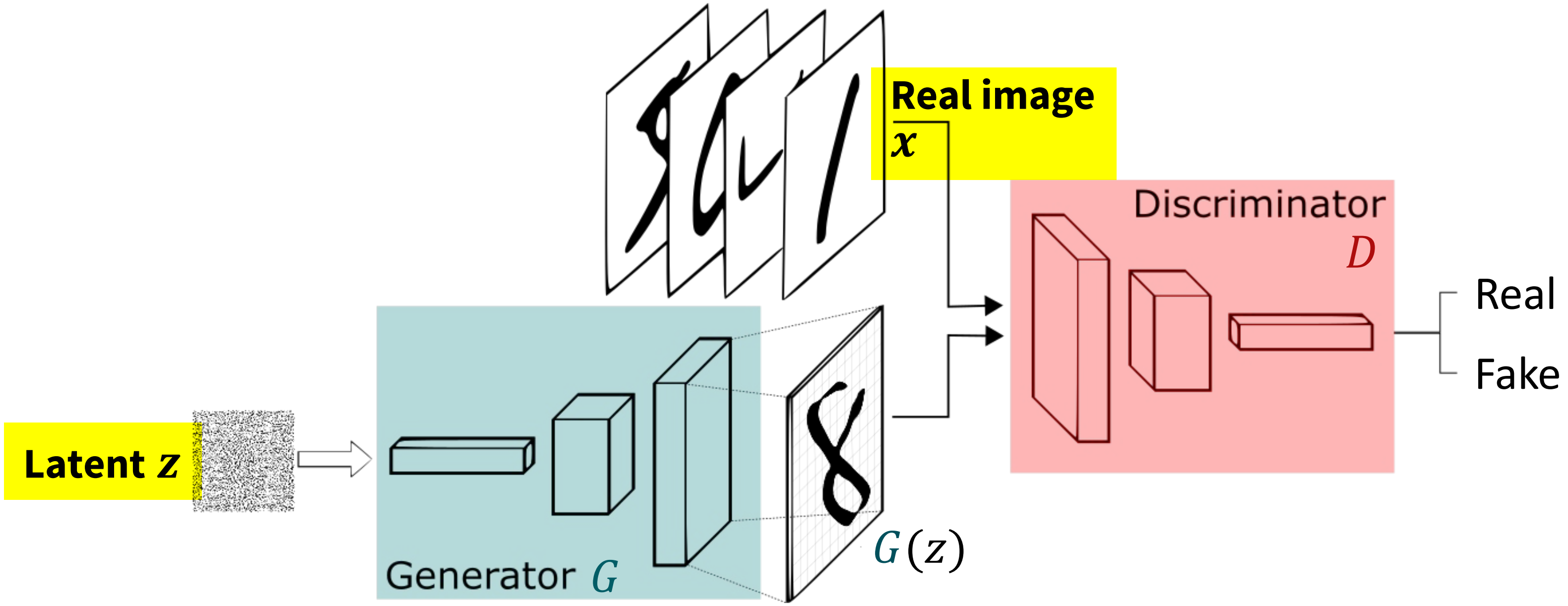
Generative Adversarial Network (GAN)

Loss function:

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Real image sampled from the data distribution

Latent sampled from the latent distribution

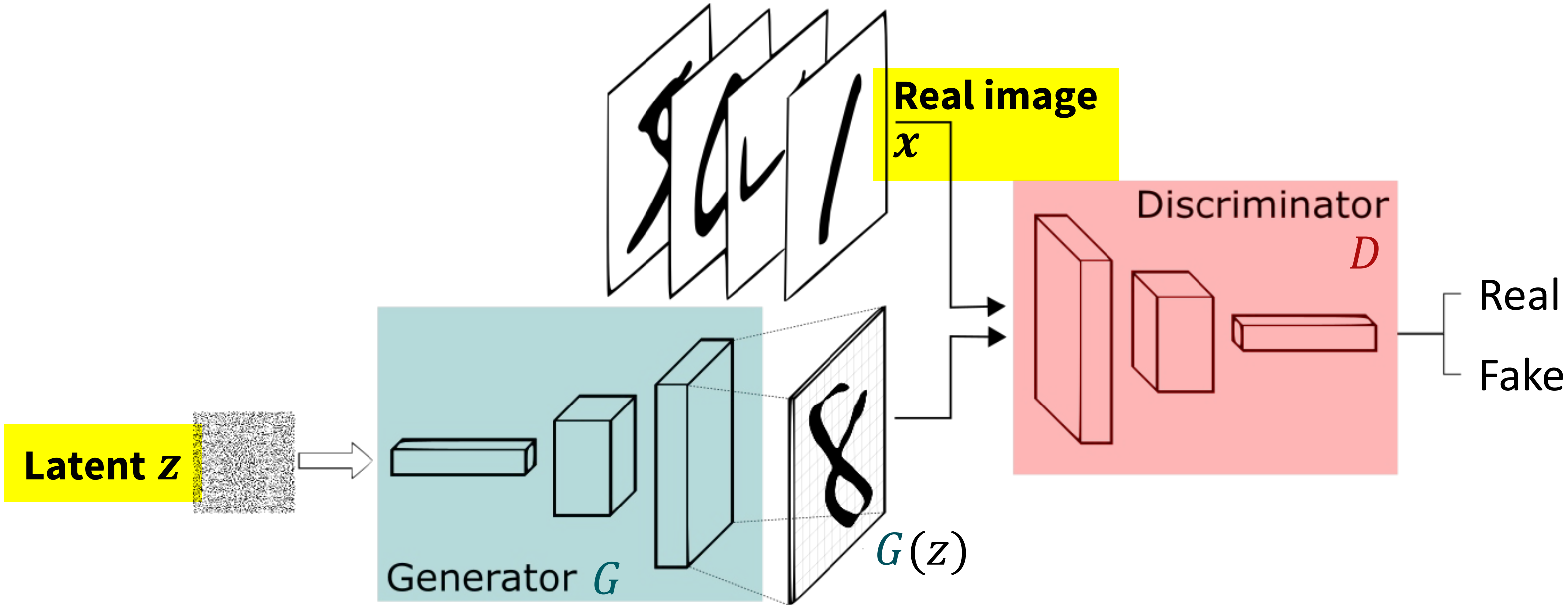


Generative Adversarial Network (GAN)

Loss function:

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}[\log(1 - D(\underline{G(\boldsymbol{z})}))]$$

The *fake* image synthesized from the latent $\boldsymbol{z}$.



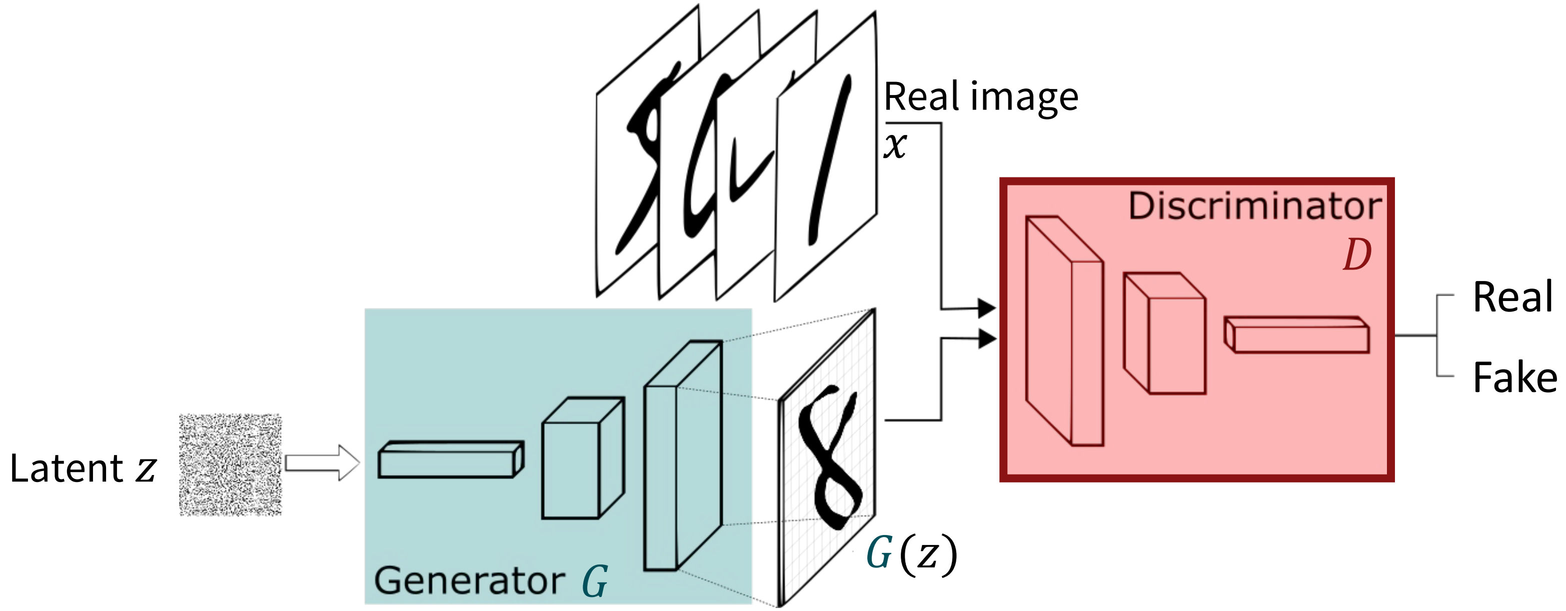
Generative Adversarial Network (GAN)

Loss function:

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x})}[\log \underline{D(\boldsymbol{x})}] + \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}[\log(1 - \underline{D(G(\boldsymbol{z}))})]$$

The probability of the *real* image x being a real image.

The probability of the *fake* image $G(\boldsymbol{z})$ being a real image.



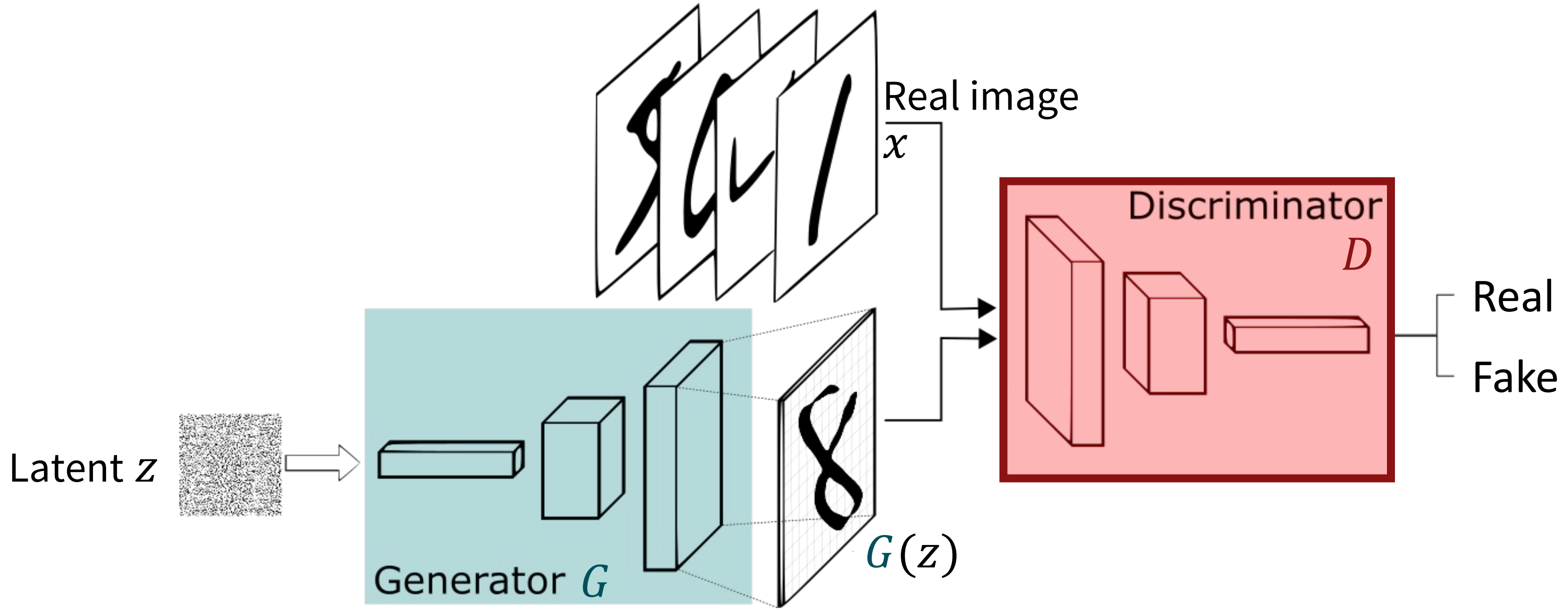
Generative Adversarial Network (GAN)

Loss function:

$$\min_{G} \underline{\max_{D}} V(D,G) = \underbrace{\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[\log D(\boldsymbol{x})]}_{} + \underbrace{\mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]}_{}$$

Discriminator ***increases*** the probability for the real image x...

and ***decreases*** the probability for the synthetic image $G(z)$.

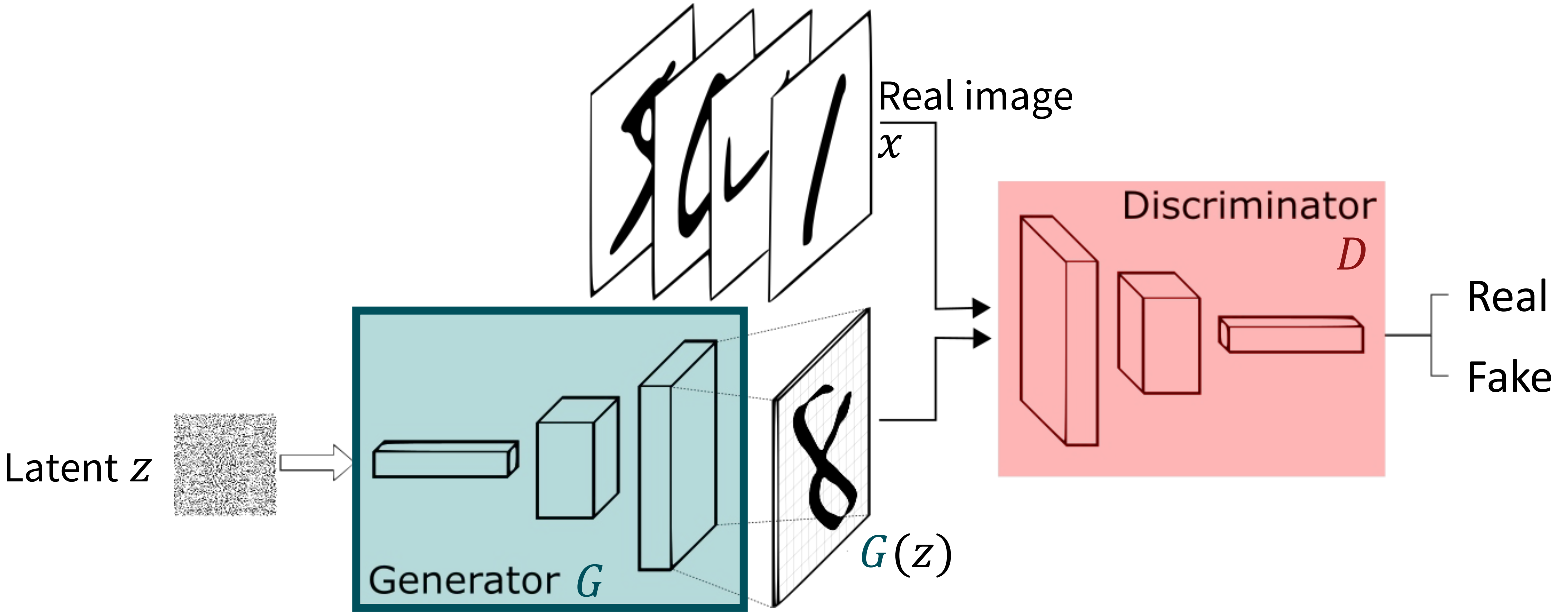


Generative Adversarial Network (GAN)

Loss function:

$$\underline{\min_{G}} \max_{D} V(D, G) = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \underline{\mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]}$$

Generator *increases* the probability for the synthetic image $G(z)$.



Generative Adversarial Network (GAN)

Loss function:

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

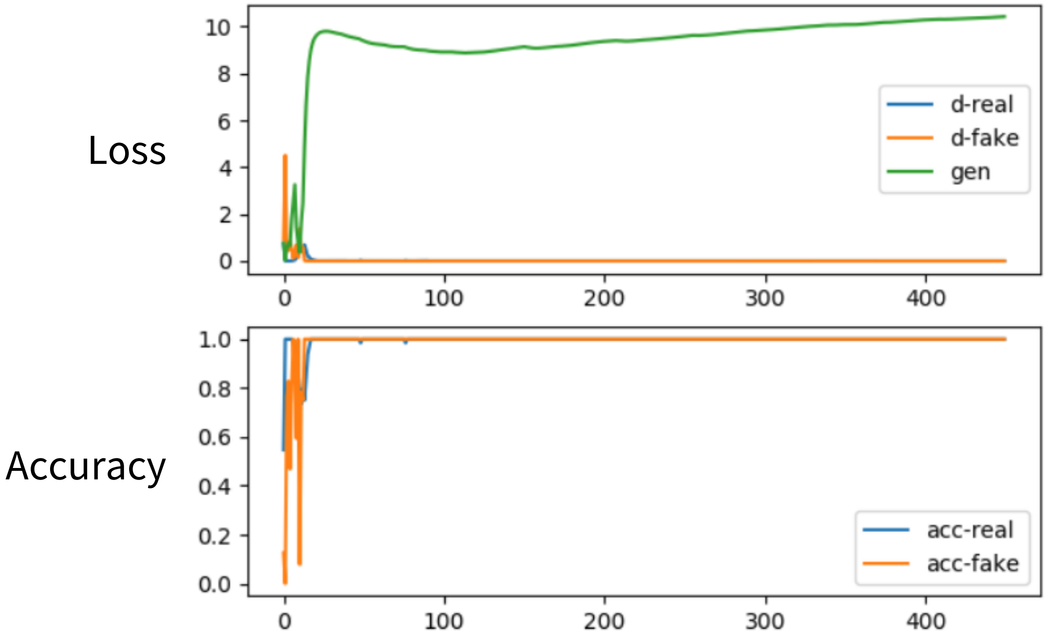
A min-max (minimax) optimization problem, which is known as very difficult to solve!

Challenges in GAN Training

1. Non-convergence & Instability
2. Mode collapse

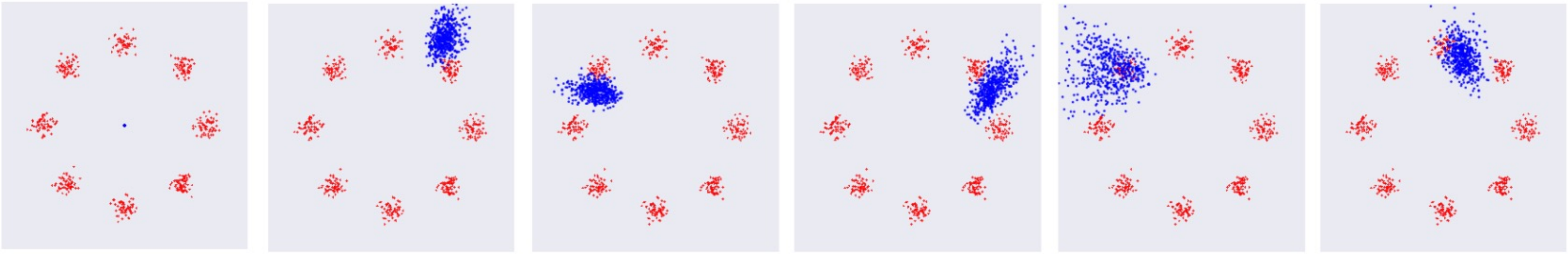
Non-Convergence & Instability

The discriminator becomes too strong and can easily distinguish between real and fake samples, leading to near-zero gradients for the generator and halting its learning.



Mode Collapse

The generator tends to collapses to a few modes of the data distribution instead of capturing the full diversity of the true data distribution.



Advances in GAN

Index	Software name	Language	Backend	Link
1	CGAN	Python	PyTorch	https://github.com/Lornatang/CGAN-PyTorch
2	DCGAN	Python	PyTorch	https://github.com/Natsu6767/DCGAN-PyTorch
3	AAEs	Python	TensorFlow	https://github.com/conan7882/adversarial-autoencoders
4	InfoGAN	Python	TensorFlow	https://github.com/openai/InfoGAN
5	SAD-GAN	—	—	—
6	LSGAN	Python	PyTorch	https://github.com/xudonmao/LSGAN
7	SRGAN	Python	TensorFlow	https://github.com/tensorlayer/SRGAN
8	WGAN	Python	PyTorch	https://github.com/Zeleni9/pytorch-wgan
9	CycleGAN	Python	TensorFlow	https://github.com/junyanz/CycleGAN
10	ProGAN	Python	PyTorch	https://github.com/tkarras/progressive_growing_of_gans
11	MidiNet	Python	TensorFlow	https://github.com/RichardYang40148/MidiNet
12	SN-GAN	Python	PyTorch	https://github.com/hanyoseob/pytorch-SNGAN
13	RGAN	Python	TensorFlow	https://github.com/ratschlab/RGAN
14	StarGAN	Python	PyTorch	https://github.com/yunjey/stargan
15	BigGAN	Python	PyTorch	https://github.com/ajbrock/BigGAN-PyTorch
16	MI-GAN	Python	TensorFlow	https://github.com/hazratali/MI-GAN
17	AttGAN	Python	TensorFlow	https://github.com/LynnHo/AttGAN-Tensorflow
18	PATE-GAN	Python	TensorFlow	https://github.com/vanderschaarlab/mlforhealthlabpub/tree/main/alg/pategan
19	DM-GAN	Python	PyTorch	https://github.com/MinfengZhu/DM-GAN
20	SinGAN	Python	PyTorch	https://github.com/tamarott/SinGAN
21	POLY-GAN	Python	PyTorch	https://github.com/nile649/POLY-GAN
22	MIEGAN	—	—	—
23	VQGAN	Python	PyTorch	https://github.com/dome272/VQGAN-pytorch
24	DALL-E	Python	PyTorch	https://github.com/lucidrains/DALLE-pytorch
25	CEGAN	—	—	—
26	Seismogen	Python	PyTorch	https://github.com/Miffka/seismogen
27	MetroGAN	Python	PyTorch	https://github.com/zwy-Giser/MetroGAN
28	M3GAN	Python	PyTorch	https://github.com/SLZWVICTOR/M3GAN
29	CNTS	Python	PyTorch	https://github.com/BomBooooo/CNTS/tree/main
30	RidgeGAN	Python	PyTorch	https://github.com/rahisha-thottolil/ridgegan

StyleGAN2

Variational Autoencoder (VAE)

Kingma and Welling , Auto-Encoding Variational Bayes, ICLR 2014.

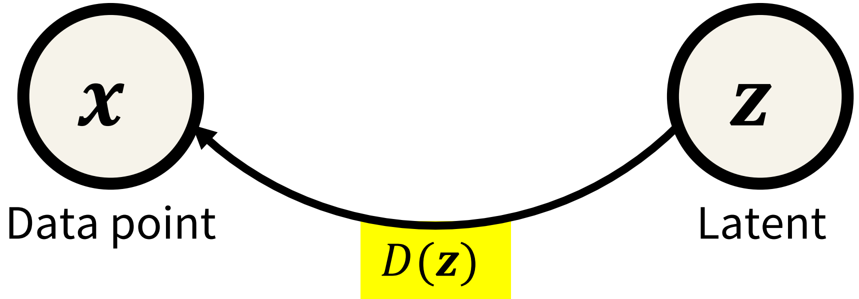
Variational Autoencoder (VAE)

- How to make a generative model *without* solving a minimax problem?
- Let's represent the mapping from the latent distribution $p(\boldsymbol{z})$ to the data distribution $p(\boldsymbol{x})$ as a conditional distribution $p(\boldsymbol{x}|\boldsymbol{z})$.

Variational Autoencoder (VAE)

Let's consider the decoder (generator) as predicting the mean of the conditional distribution $p(\boldsymbol{x}|\boldsymbol{z})$:

$$p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; D(\boldsymbol{z}), \underbrace{\sigma^2 \mathbf{I}}_{\text{Fixed variance}})$$



Basics

- Maginal distribution
- Expected value
- Bayes' rule
- Kullback–Leibler (KL) Divergence
- Jensen's inequality

Marginal Distribution

The marginal distribution of a subset of a set of random variables is the probability distribution of the variables contained in the subset.

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z}$$

Take the integral over $\boldsymbol{z}$ to marginalize $\boldsymbol{x}$.

Marginal Distribution

Q. Given the joint probability table below, what is $p(\boldsymbol{x} = \mathbf{1})$?

		$\boldsymbol{y}$		
		1	2	3
$\boldsymbol{x}$	1	0.32	0.03	0.01
	2	0.06	0.24	0.02
	3	0.02	0.03	0.27

Expected Value

The expected value is the arithmetic mean of the possible values a random variable can take, weighted by the probability of those outcomes.

$$\mathbb{E}_{p(\boldsymbol{x})}[\boldsymbol{x}] = \int \boldsymbol{x} \cdot p(\boldsymbol{x}) d\boldsymbol{x}$$

Marginal Distribution

Q. What is $\mathbb{E}_{p(\boldsymbol{x})}[\boldsymbol{x}]$?

		$\boldsymbol{y}$		
		1	2	3
$\boldsymbol{x}$	1	0.32	0.03	0.01
	2	0.06	0.24	0.02
	3	0.02	0.03	0.27

Bayes' Rule

Bayes' rule is a mathematical formula used to determine the conditional probability of events.

$$\underbrace{\boxed{p(\boldsymbol{z}|\boldsymbol{x})}}_{\text{Posterior}} = \frac{\overbrace{\boxed{p(\boldsymbol{x}|\boldsymbol{z})}}^{\text{Likelihood}}\ \underbrace{\boxed{p(\boldsymbol{z})}}_{\text{Prior}}}{\underbrace{\boxed{p(\boldsymbol{x})}}_{\text{Marginal}}}$$

$$p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x}) = p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}) = p(\boldsymbol{x},\boldsymbol{z})$$

Marginal Distribution

Q. What is $p(\boldsymbol{y} = \mathbf{2} | \boldsymbol{x} = \mathbf{1})$?

		$\boldsymbol{y}$		
		1	2	3
$\boldsymbol{x}$	1	0.32	0.03	0.01
	2	0.06	0.24	0.02
	3	0.02	0.03	0.27

Kullback–Leibler (KL) Divergence

Kullback–Leibler (KL) divergence is a measure of how one probability distribution p is different from a reference probability distribution q:

$$D_{\mathrm{KL}}(p \parallel q) = \int p(\boldsymbol{x}) \log\left(\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right) d\boldsymbol{x} = \mathbb{E}_{p(\boldsymbol{x})}\left[\log\left(\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right)\right]$$

Q. What is $D_{KL}(p \parallel p)$?

Kullback–Leibler (KL) Divergence

Q. Homework

When $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \sigma^2\mathbf{I})$ and $q(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \mathbf{0}, \mathbf{I})$,

What is $D_{\mathrm{KL}}(p \parallel q)$?

[EXTRA] KL Divergence with information perspective

Suppose X takes 4 values, each with probability 1/4.

- ABCCBDDBACCDBAA..

• How many bits required?

->A=00, B=01, C=10, D=11

But with different probabilities:

• A: 1/2 , B: 1/4 , C, D: 1/8

Is it still best to assign 2 bits/sym?

-> NO! A: 0 , B: 10 , C: 110 , D: 111

[EXTRA] KL Divergence with information perspective

Amount of info for sym A:

$$\log_2 \frac{1}{P(X = A)} = \log_2 2 = 1$$

Average amount of info in X (for case 1 and 2):

$$\sum_{x=A,B,C,D} p(x) \log_2 \frac{1}{p(x)} = -\sum_{x=A,B,C,D} p(x) \log_2 p(x) = 2$$

$$\sum_{x=A,B,C,D} p(x) \log_2 p(x) = 0.5 * 1 + 0.25 * 2 + 0.125 * 3 * 2 = 1.75$$

[EXTRA] KL Divergence with information perspective

Entropy of distribution p(x):

$$H(p) = -\mathbb{E}_{p(X)}[\log p(X)] = -\sum_x p(x) \log p(x)$$

Cross-entropy between distributions p and q:

$$H(p, q) = -\mathbb{E}_{p(X)}[\log q(X)] = -\sum_x p(x) \log q(x)$$

And.. H(p,q) – H(p) is called Kullback-Leibler (KL) divergence

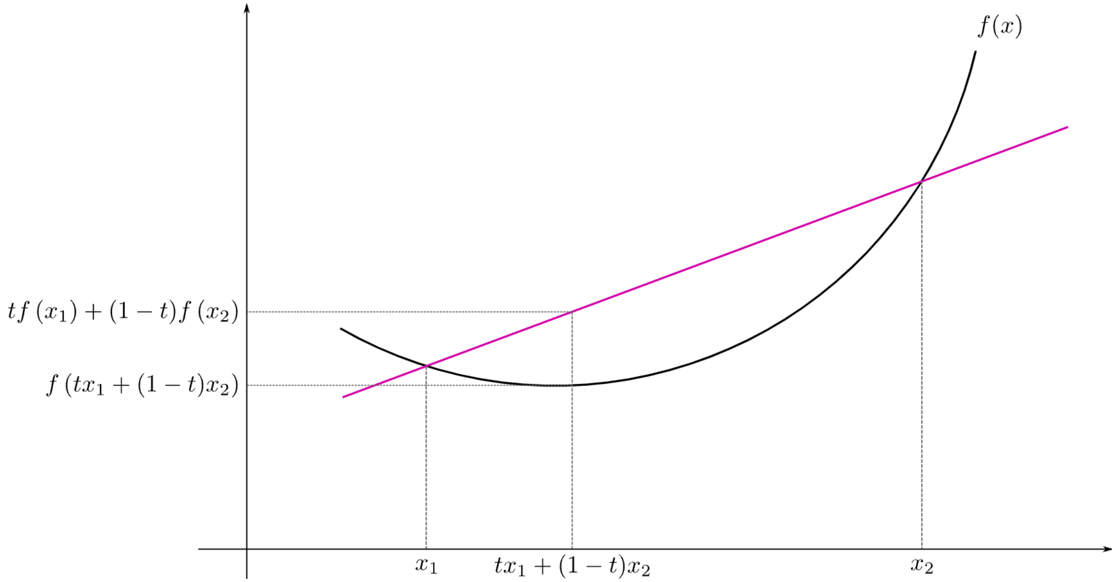
$$D_{KL}(P \| Q) = \mathbb{E}_{X \sim P}\left[\log \frac{P(X)}{Q(X)}\right] = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

Jensen's Inequality

f is a convex function if

$$f(t\boldsymbol{x}_1 + (1-t)\boldsymbol{x}_2) \leq tf(\boldsymbol{x}_1) + (1-t)f(\boldsymbol{x}_2)$$

for all $\boldsymbol{x}_1, \boldsymbol{x}_2$, and $t \in [0,1]$.



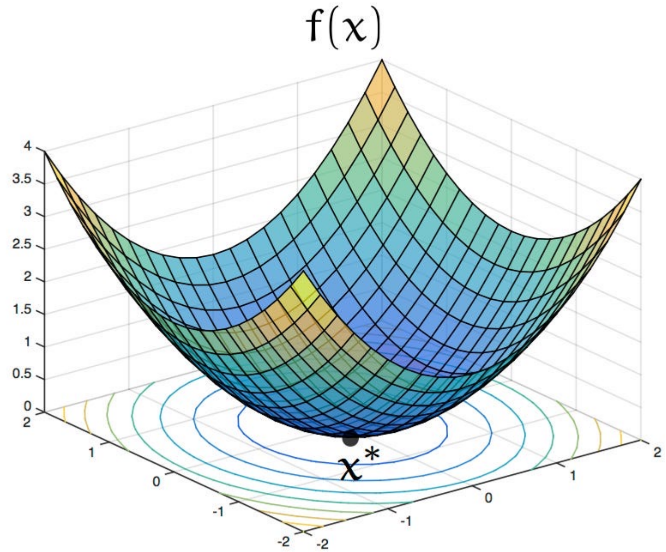
Jensen's Inequality

f is a convex function if

$$f\left(\boxed{\sum_i t_i \boldsymbol{x}_i}\right) \leq \sum_i t_i f(\boldsymbol{x}_1)$$

Affine combination

for all $\boldsymbol{x}_i$ and $t \in [0,1]$.



Jensen's Inequality

if $\boldsymbol{x}$ is a random variable and f is a convex function, then

$$f\left(\mathbb{E}_{p(\boldsymbol{x})}[\boldsymbol{x}]\right) \leq \mathbb{E}_{p(\boldsymbol{x})}[f(\boldsymbol{x})]$$

Since the expected value is an affine combination.

Q. What if f is a concave function?

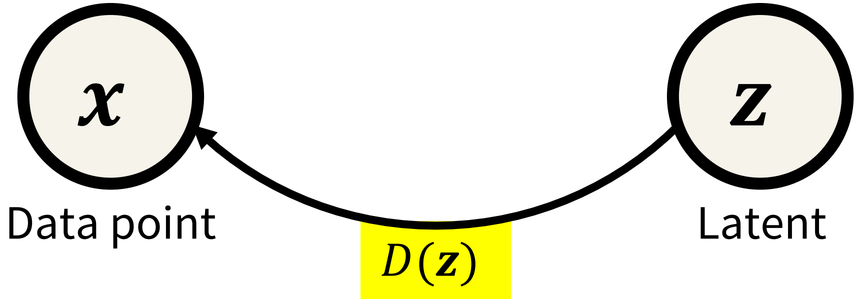
Back to VAE…

- Maginal distribution
- Expected value
- Bayes’ rule
- Kullback–Leibler (KL) Divergence
- Jensen’s inequality

Variational Autoencoder (VAE)

$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \mathbf{0}, \mathbf{I})$$

$$p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; D(\boldsymbol{z}), \sigma^2\mathbf{I})$$



Variational Autoencoder (VAE)

For all given real images $\boldsymbol{x}$, we want to maximize the marginal probability:

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} = \int p(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z}$$

How to compute the integral?

Monte-Carlo method for $\boldsymbol{x}$ and $\boldsymbol{z}$ takes too much time…
→ Intractable.

Variational Autoencoder (VAE)

Or, we can compute

$$p(\boldsymbol{x}) = \frac{p(\boldsymbol{x}, \boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})} = \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{\boxed{p(\boldsymbol{z}|\boldsymbol{x})}}$$

But this conditional distribution
is unknown.

Evidence Lower Bound (ELBO)

- We cannot directly maximize $p(\boldsymbol{x}) = \frac{p(\boldsymbol{x},\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})}$ since $p(\boldsymbol{z}|\boldsymbol{x})$ is unknown.
- Let's think about the lower bound of $\log p(\boldsymbol{x})$:

$$\log p(\boldsymbol{x}) \geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]$$

Evidence Lower Bound (ELBO)

- $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is a variational distribution with parameters ϕ.
- E.g., a Gaussian distribution with mean and variance as parameters.
- Consider $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ as an arbitrary conditional distribution that may not be the same with $p(\boldsymbol{z}|\boldsymbol{x})$.
- We use the proxy distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ since we don't know $p(\boldsymbol{z}|\boldsymbol{x})$.

Evidence Lower Bound (ELBO)

- $\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]$ is the expected value over $\boldsymbol{z}$ sampled from the variational distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$.

- Why $\log p(\boldsymbol{x}) \geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]$?

  Because of the Jensen's inequality.

Evidence Lower Bound (ELBO)

$$\log p(\boldsymbol{x}) = \log \int p(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z}$$

$$= \log \int p(\boldsymbol{x}, \boldsymbol{z}) \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z}$$

$$= \boxed{\log} \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[\frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \right]$$

Concave function

$$\geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \right]$$

Evidence Lower Bound (ELBO)

Q. **Homework**

What is the Jensen gap $\left(\log p(\boldsymbol{x}) - \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]\right)$?

Evidence Lower Bound (ELBO)

Let's decompose ELBO:

$$\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right] = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]$$

$$= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})] - \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p(\boldsymbol{z})}\right]$$

$$= \boxed{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})]} - \boxed{D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{z})\right)}$$

Reconstruction term
to be *maximized*.

Prior matching term
to be *minimized*.

[EXTRA] Understanding VAE's ELBO in paper

Let us consider some dataset $\mathbf{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ consisting of N i.i.d. samples

The marginal likelihood is composed of a sum over the marginal likelihoods of individual datapoints $\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(N)}) = \sum_{i=1}^{N} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$, which can each be rewritten as:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) = D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)})) + \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) \tag{1}$$

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) \geq \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[-\log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) + \log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})\right]$$

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = -D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z})\right]$$