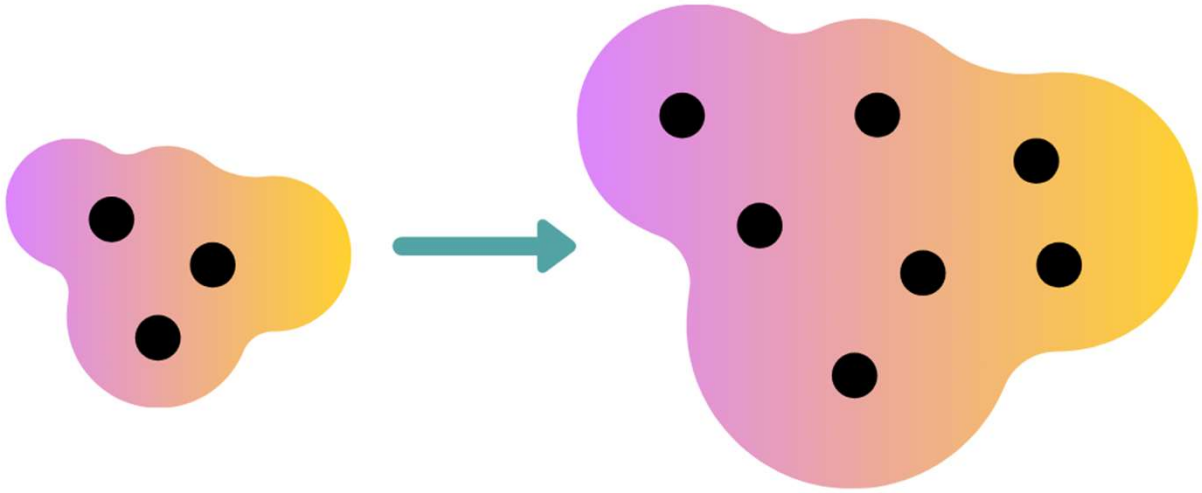
基于深度学习的生成模型
Deep Generative Models



Generating new data using past data

什么是（深度）生成模型

生成：从一个分布里采样

Learn to Build

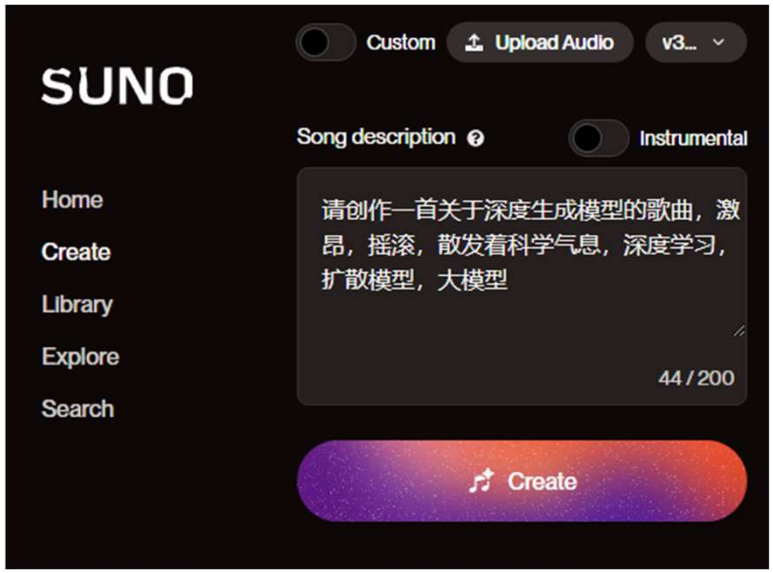
深度生成模型

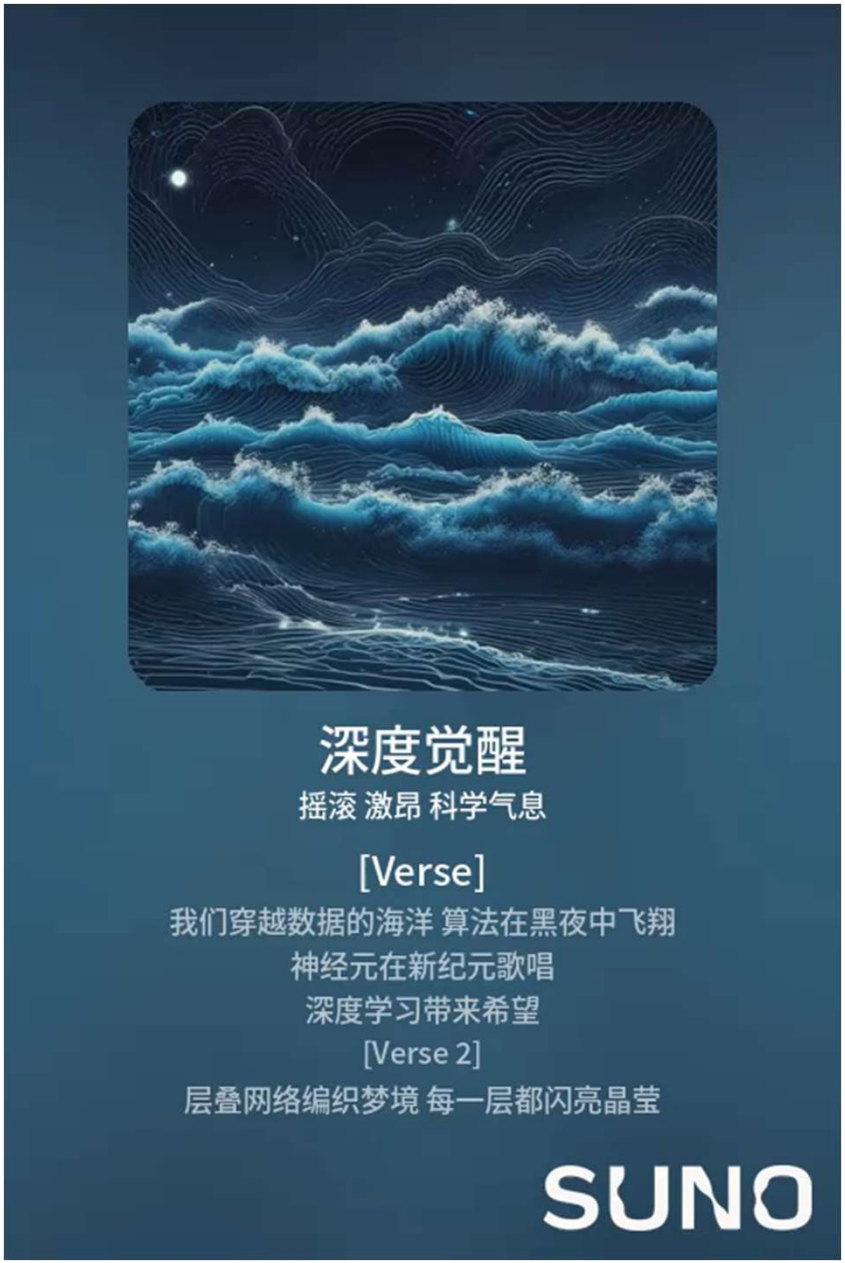
从一个分布里采样随机变量 Z

合成

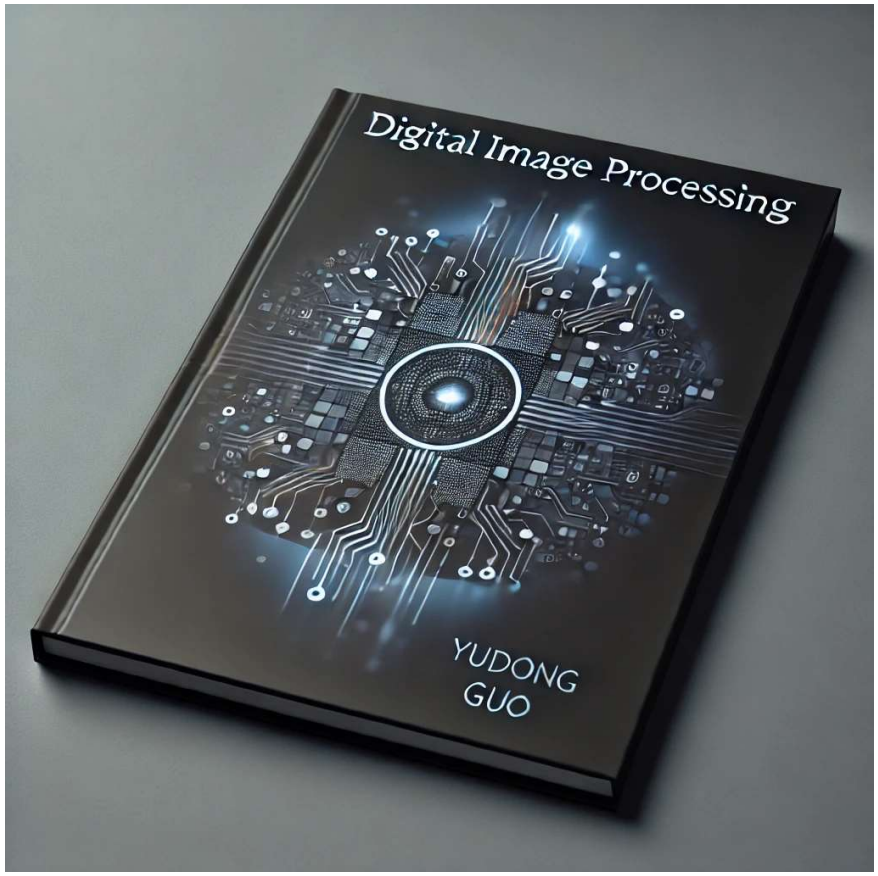
1D数据生成

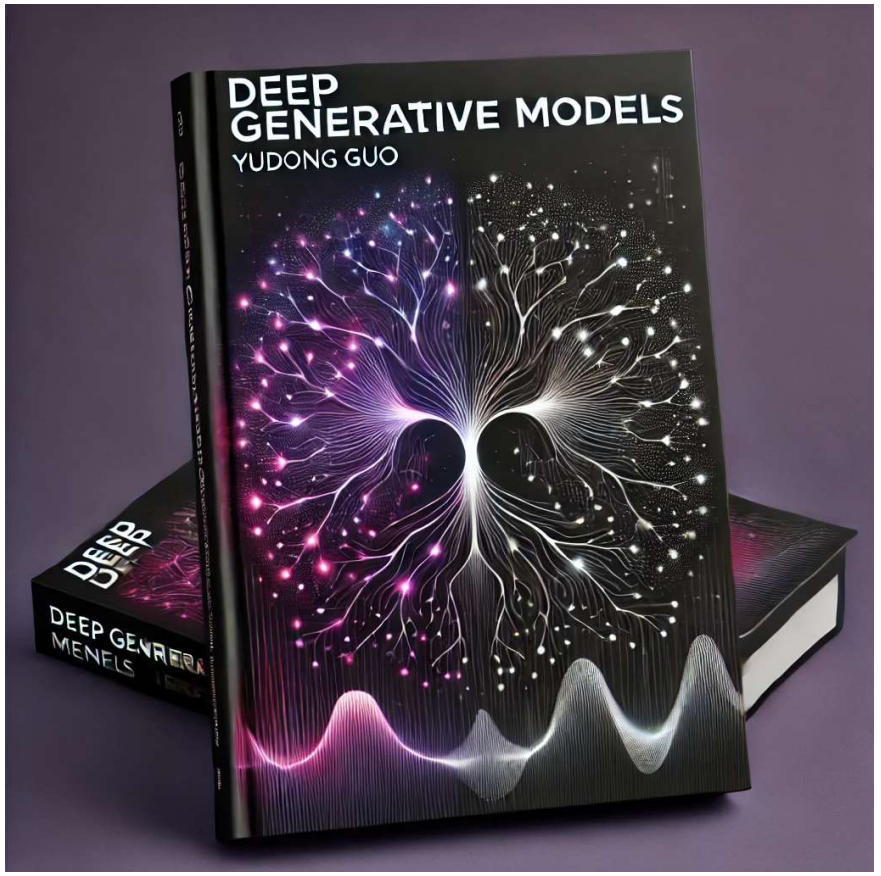
1D数据生成


SUNO
Custom
Upload Audio
v3...
Song description
Instrumental
Home
Create
Library
Explore
Search
请创作一首关于深度生成模型的歌曲，激昂，摇滚，散发着科学气息，深度学习，扩散模型，大模型
44 / 200
Create


深度觉醒
摇滚 激昂 科学气息
[Verse]
我们穿越数据的海洋 算法在黑夜中飞翔
神经元在新纪元歌唱
深度学习带来希望
[Verse 2]
层叠网络编织梦境 每一层都闪亮晶莹
SUNO

2D数据生成


Digital Image Processing
YUDONG
GUO


DEEP
GENERATIVE MODELS
YUDONG GUO

Video / 3D 数据生成

sora

如何构建一个深度生成模型?

之前学的深度学习 — Supervised Learning

Supervised Learning

Data: (x, y)
x is data, y is label

Goal: Learn a function to map x -> y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

Supervised Learning Tasks

Supervised Learning

Data: (x, y)
x is data, y is label

Goal: Learn a function to map x -> y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

Classification

Supervised Learning Tasks

Supervised Learning

Data: (x, y)
x is data, y is label

Goal: Learn a function to map x -> y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

A cat sitting on a suitcase on the floor

Image captioning

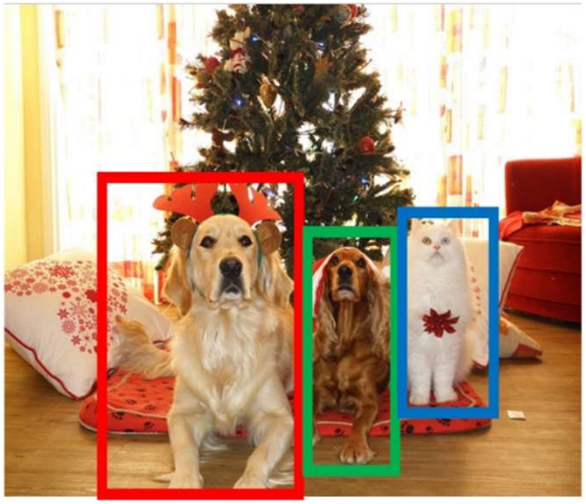
Supervised Learning Tasks

Supervised Learning

Data: (x, y)
x is data, y is label

Goal: Learn a function to map x -> y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



DOG, DOG, CAT

Object Detection

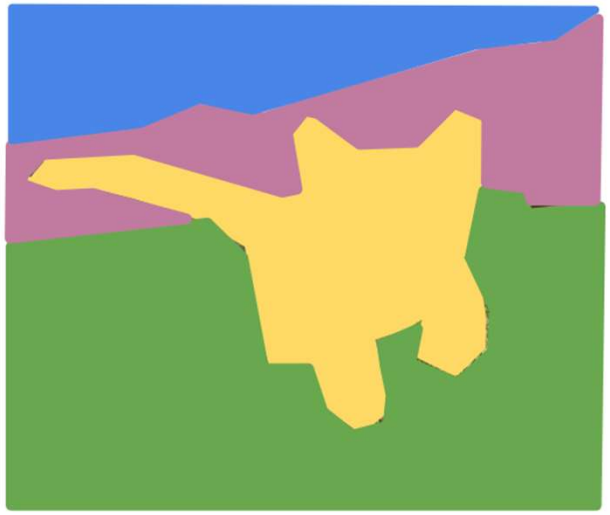
Supervised Learning Tasks

Supervised Learning

Data: (x, y)
x is data, y is label

Goal: Learn a function to map x -> y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



GRASS, CAT, TREE, SKY

Semantic Segmentation

生成 — A type of Unsupervised Learning

Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden structure of the data

Examples: Clustering, dimensionality reduction, density estimation, etc.

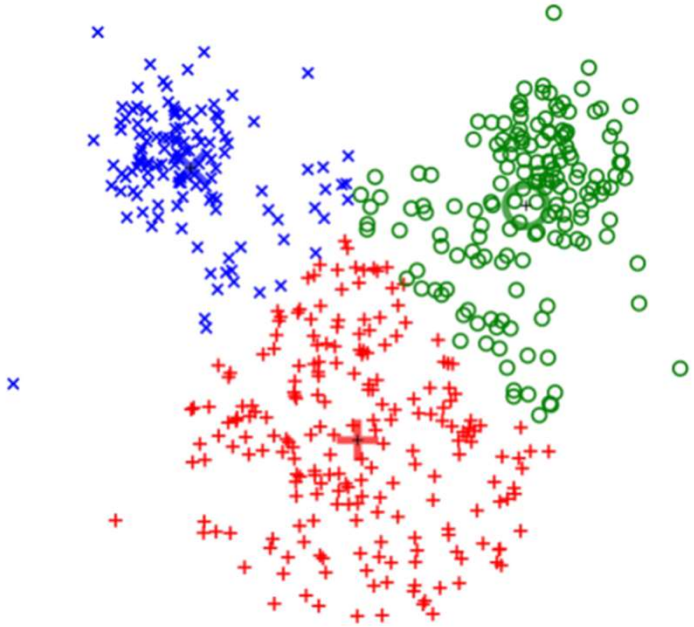
Unsupervised Learning

Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden structure of the data

Examples: Clustering, dimensionality reduction, density estimation, etc.



K-means clustering

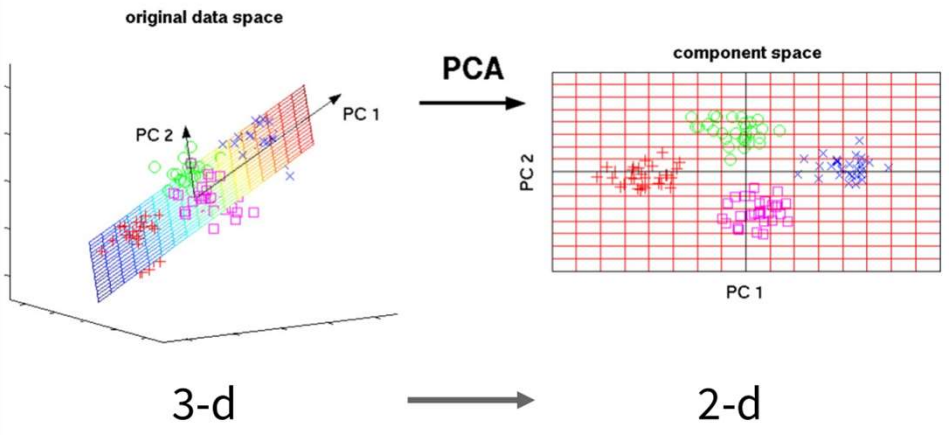
Unsupervised Learning

Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden structure of the data

Examples: Clustering, dimensionality reduction, density estimation, etc.



Principal Component Analysis
(Dimensionality reduction)

Unsupervised Learning

Unsupervised Learning

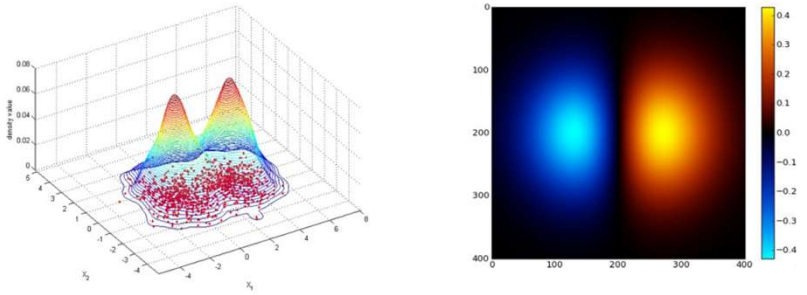
Data: x
Just data, no labels!

Goal: Learn some underlying hidden structure of the data

Examples: Clustering, dimensionality reduction, density estimation, etc.

Figure copyright Ian Goodfellow, 2016. Reproduced with permission.

1-d density estimation



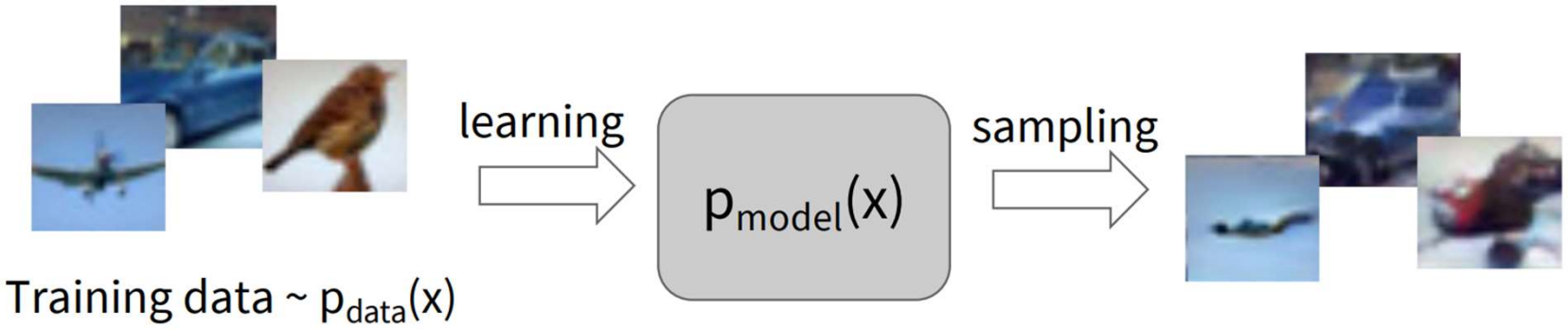
2-d density estimation

Modeling p(x)

2-d density images left and right are CC0 public domain

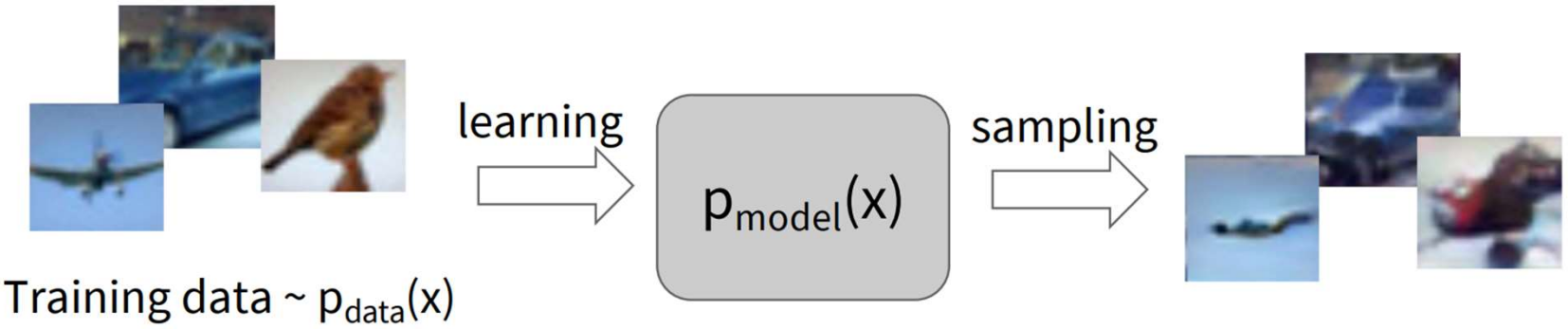
Generative Modeling

Given training data, generate new samples from same distribution



Generative Modeling

Given training data, generate new samples from same distribution

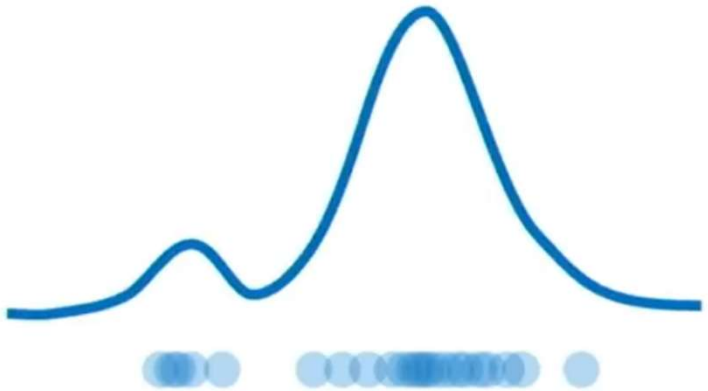


Objectives:

1. Learn $p_{model}(x)$ that approximates $p_{data}(x)$
2. Sampling new x from $p_{model}(x)$

Key in Generative Modeling

Density Estimation



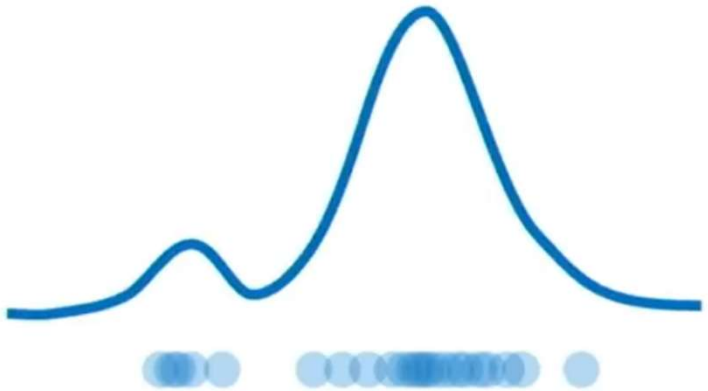
Sample Generation

Input samples

Training data $\sim P_{data}(x)$

Key in Generative Modeling

Density Estimation



Sample Generation

Input samples

Training data ~ $P_{data}(x)$

Formulate as density estimation problems:

- Explicit density estimation: explicitly define and solve for $p_{model}(x)$
- Implicit density estimation: learn model that can sample from $p_{model}(x)$ without explicitly defining it.

如何从大量复杂数据中估计概率密度函数?

Taxonomy of Generative Models

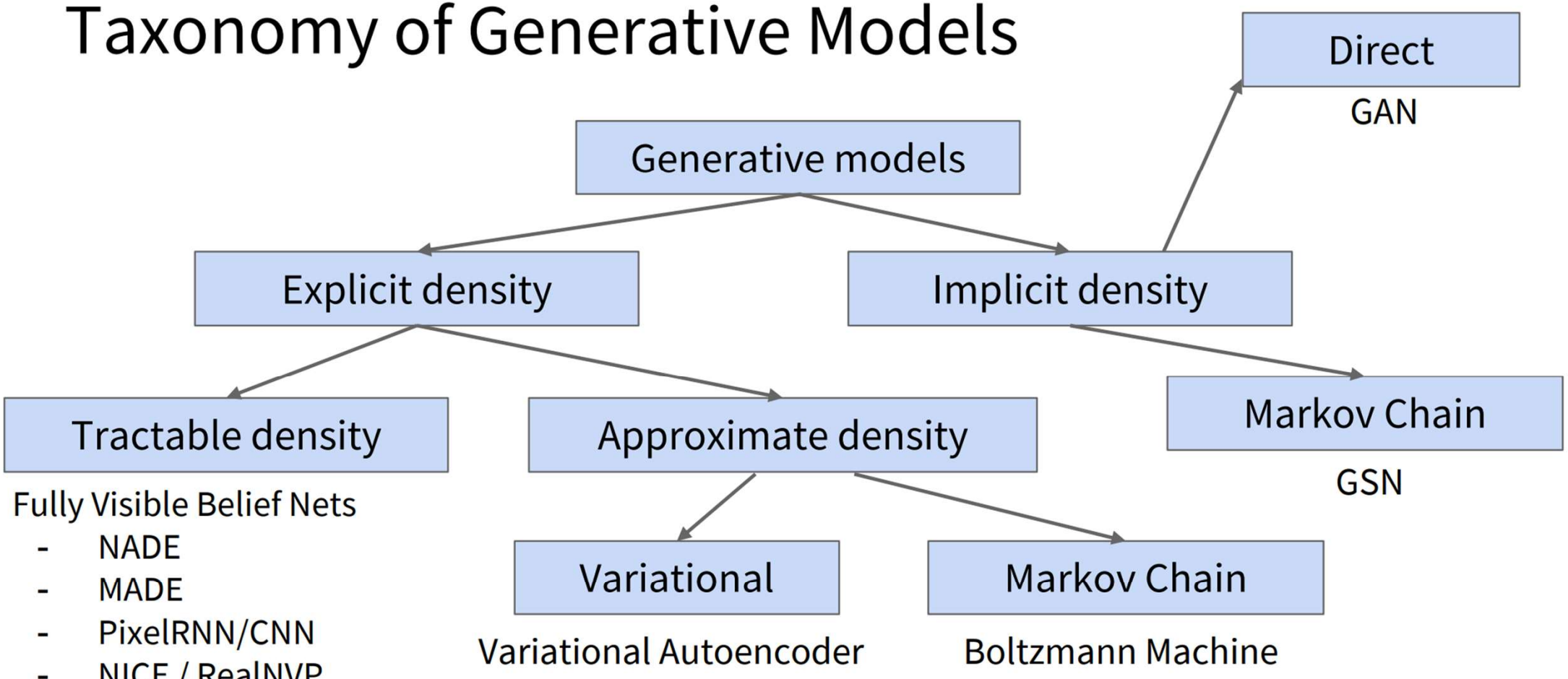


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Taxonomy of Generative Models

介绍三种不同类型的
常用的深度生成模型

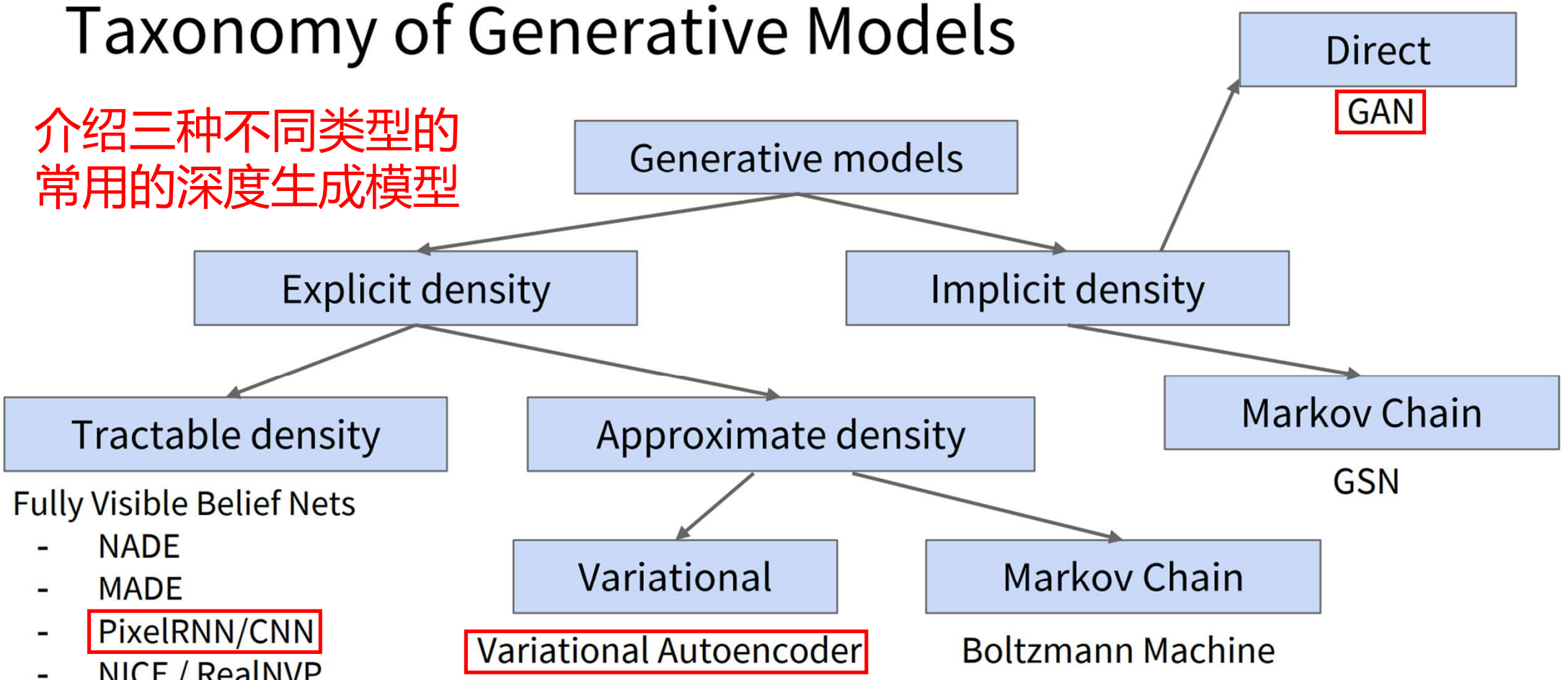


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

PixelRNN and PixelCNN

A very brief overview

Key Idea

Fully visible belief network (FVBN)

Explicit density model

$$p(x) = p(x_1, x_2, \ldots, x_n)$$

Likelihood of image x

Joint likelihood of each pixel in the image

Key Idea

Fully visible belief network (FVBN)

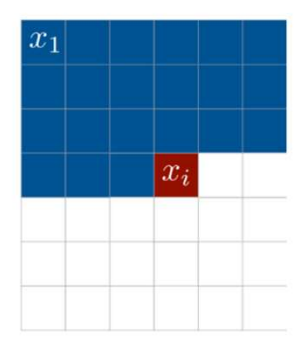
Explicit density model

Use chain rule to decompose likelihood of an image x into product of 1-d distributions:

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$

Likelihood of image x

Probability of i'th pixel value given all previous pixels



Then maximize likelihood of training data

Key Idea

Fully visible belief network (FVBN)

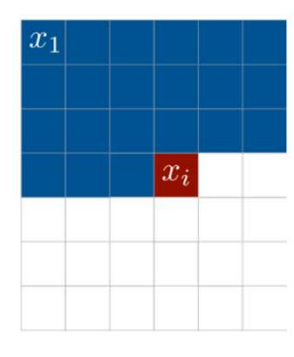
Explicit density model

Use chain rule to decompose likelihood of an image x into product of 1-d distributions:

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$

Likelihood of image x

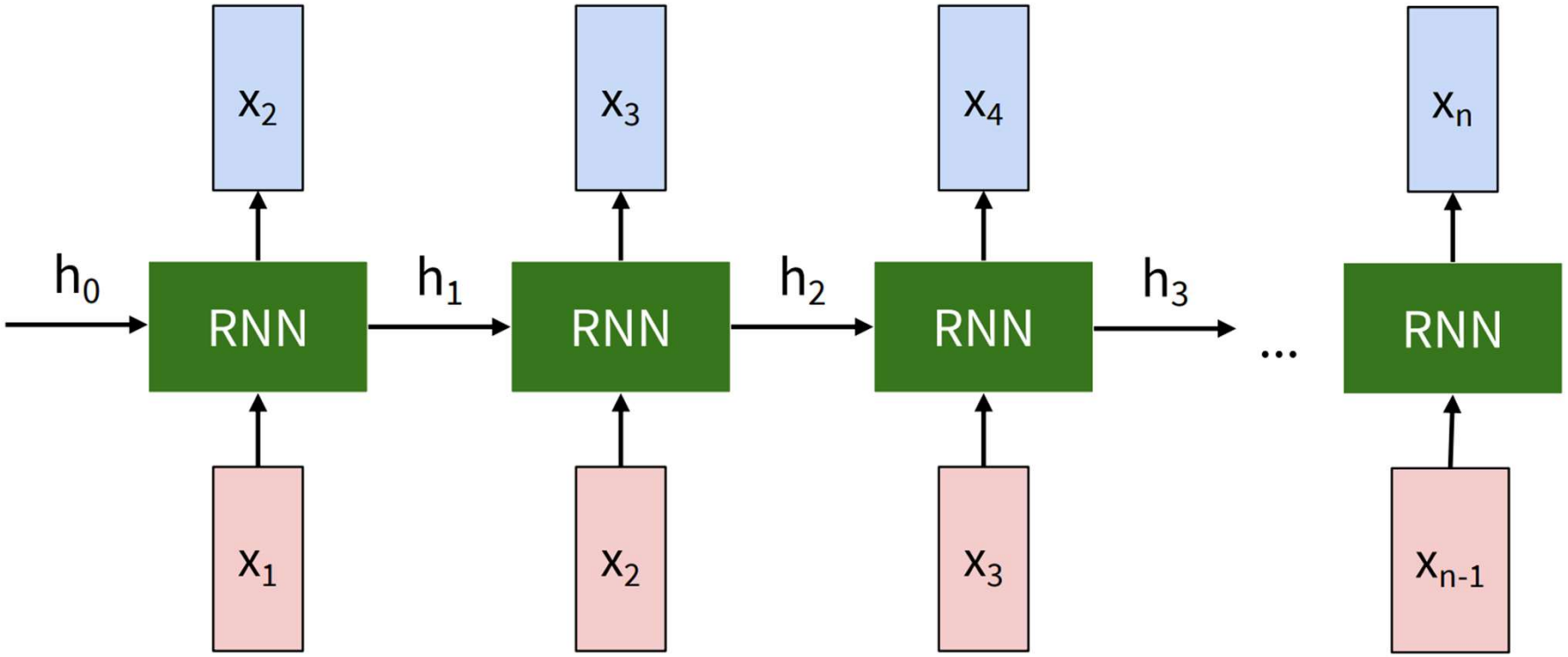
Probability of i'th pixel value given all previous pixels



Then maximize likelihood of training data

用神经网络来与预测逐像素的（复杂）条件概率

Recurrent Neural Network



$$p(x_i|x_1,\ldots,x_{i-1})$$

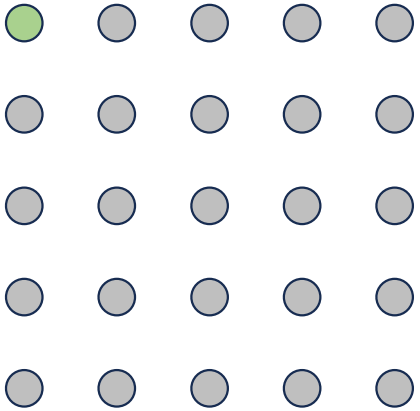
时序 / 循环
神经网络

将历史信息
H_{i-1} (hidden)
和当前状态
x_i
作为输入预测
下一状态
x_{i+1}
并更新状态
H_i

PixelRNN [van der Oord et al. 2016]

Generate image pixels starting from corner

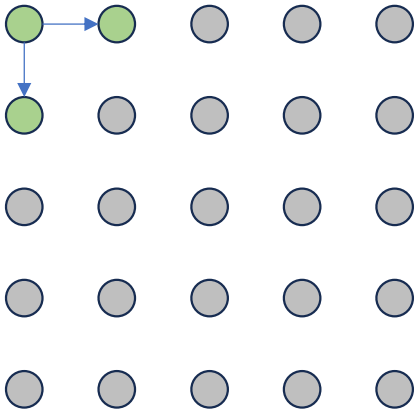
Dependency on previous pixels modeled using an RNN (LSTM)



PixelRNN [van der Oord et al. 2016]

Generate image pixels starting from corner

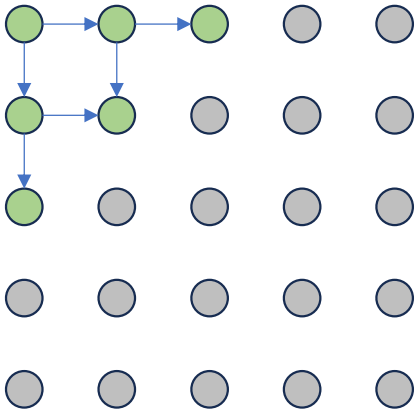
Dependency on previous pixels modeled using an RNN (LSTM)



PixelRNN [van der Oord et al. 2016]

Generate image pixels starting from corner

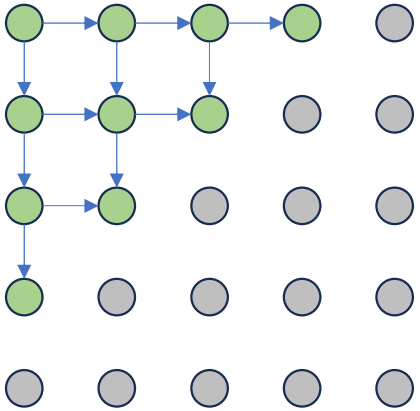
Dependency on previous pixels modeled using an RNN (LSTM)



PixelRNN [van der Oord et al. 2016]

Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)

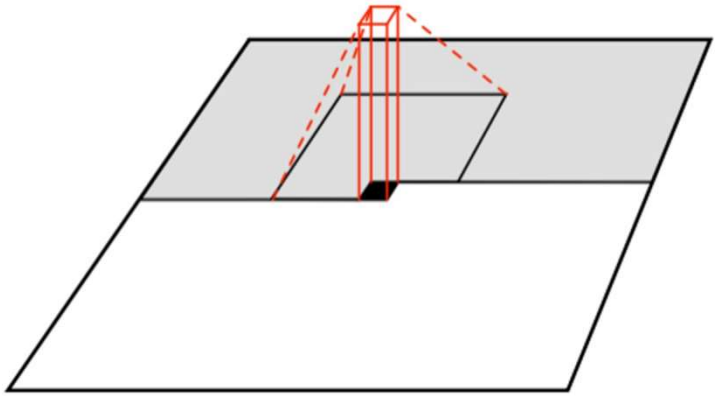


Drawback: sequential generation is slow in both training and inference!

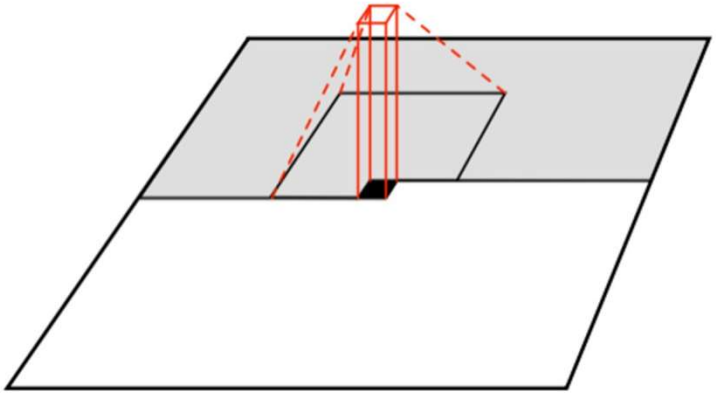
PixelCNN [van der Oord et al. 2016]

Still generate image pixels starting from corner

Dependency on previous pixels now modeled using a CNN over context region
(masked convolution)

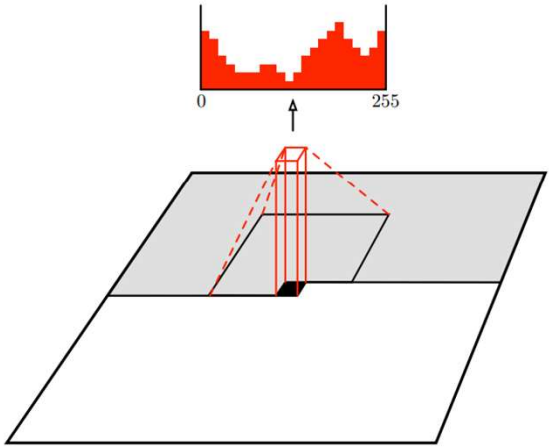


PixelCNN [van der Oord et al. 2016]



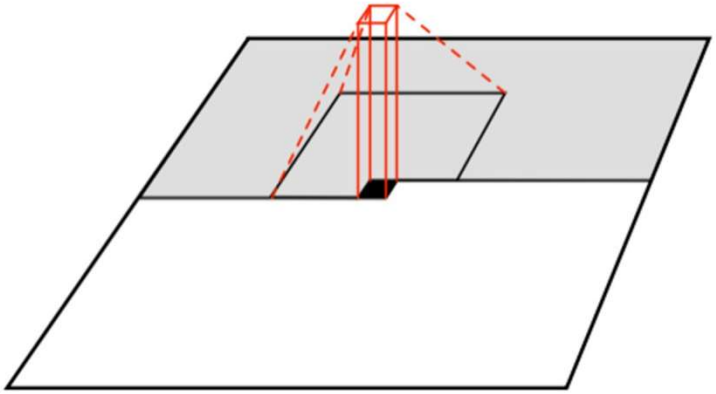
Still generate image pixels starting from corner

Dependency on previous pixels now modeled using a CNN over context region (masked convolution)



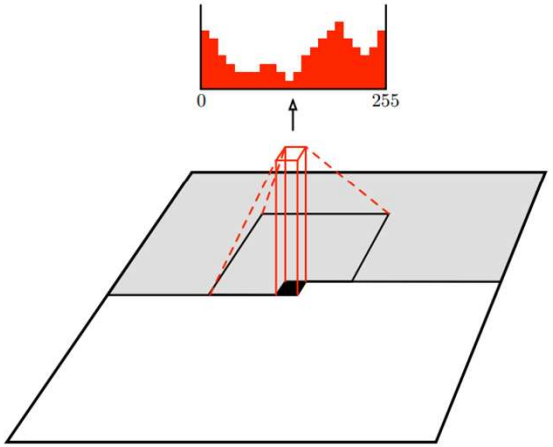
1	1	1	1	1
1	1	1	1	1
1	1	0	0	0
0	0	0	0	0
0	0	0	0	0

PixelCNN [van der Oord et al. 2016]



Still generate image pixels starting from corner

Dependency on previous pixels now modeled using a CNN over context region (masked convolution)

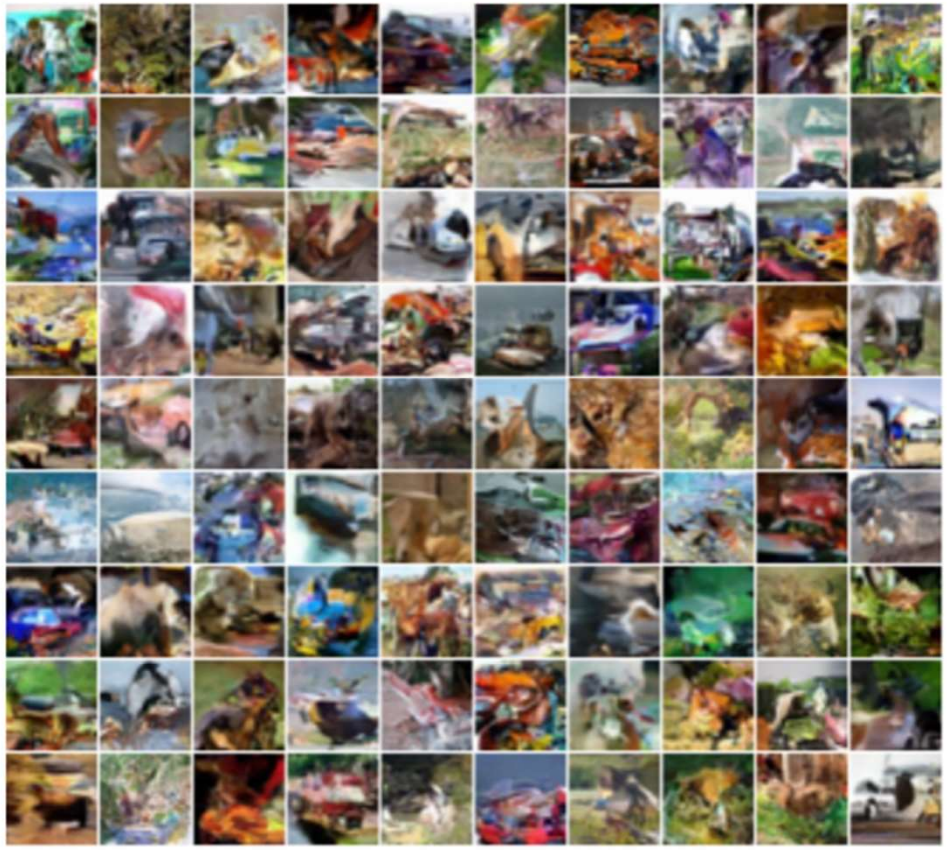


1	1	1	1	1
1	1	1	1	1
1	1	0	0	0
0	0	0	0	0
0	0	0	0	0

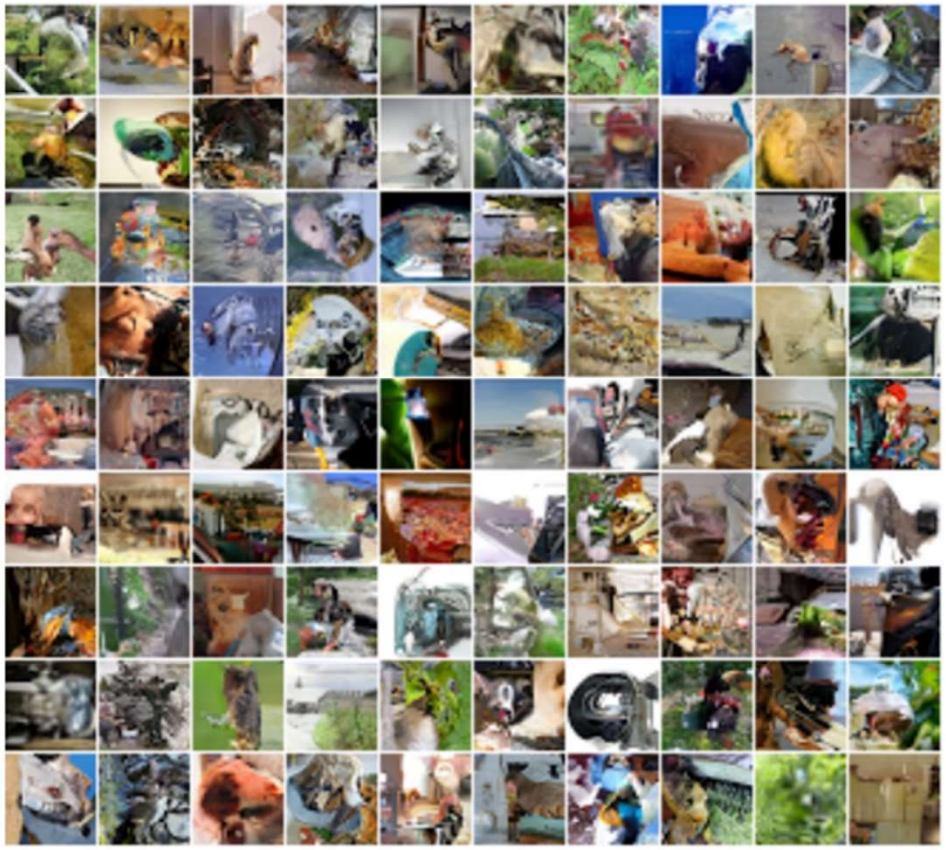
Training is faster than PixelRNN
(can parallelize convolutions since context region values known from training images)

Generation is still slow:
For a 32x32 image, we need to do forward passes of the network 1024 times for a single image

Generation Samples

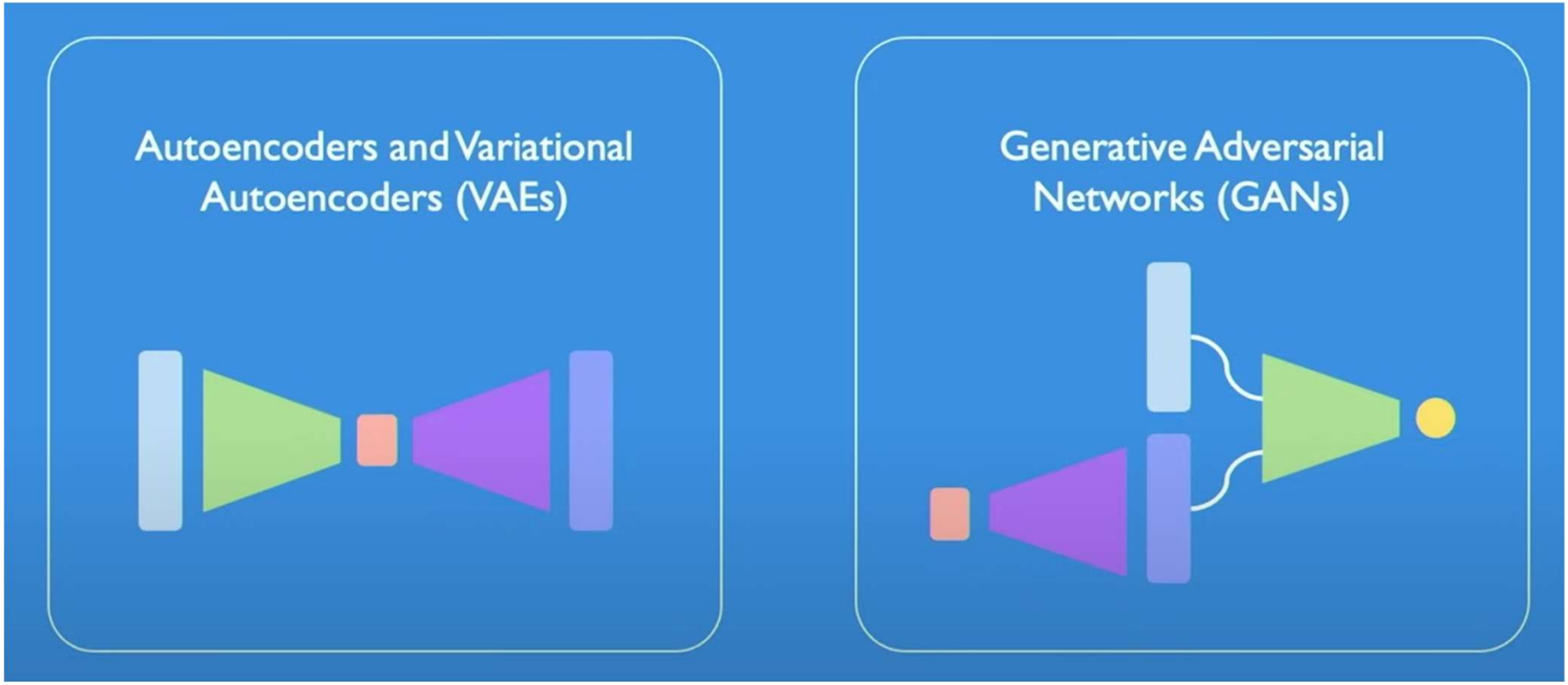


32x32 CIFAR-10

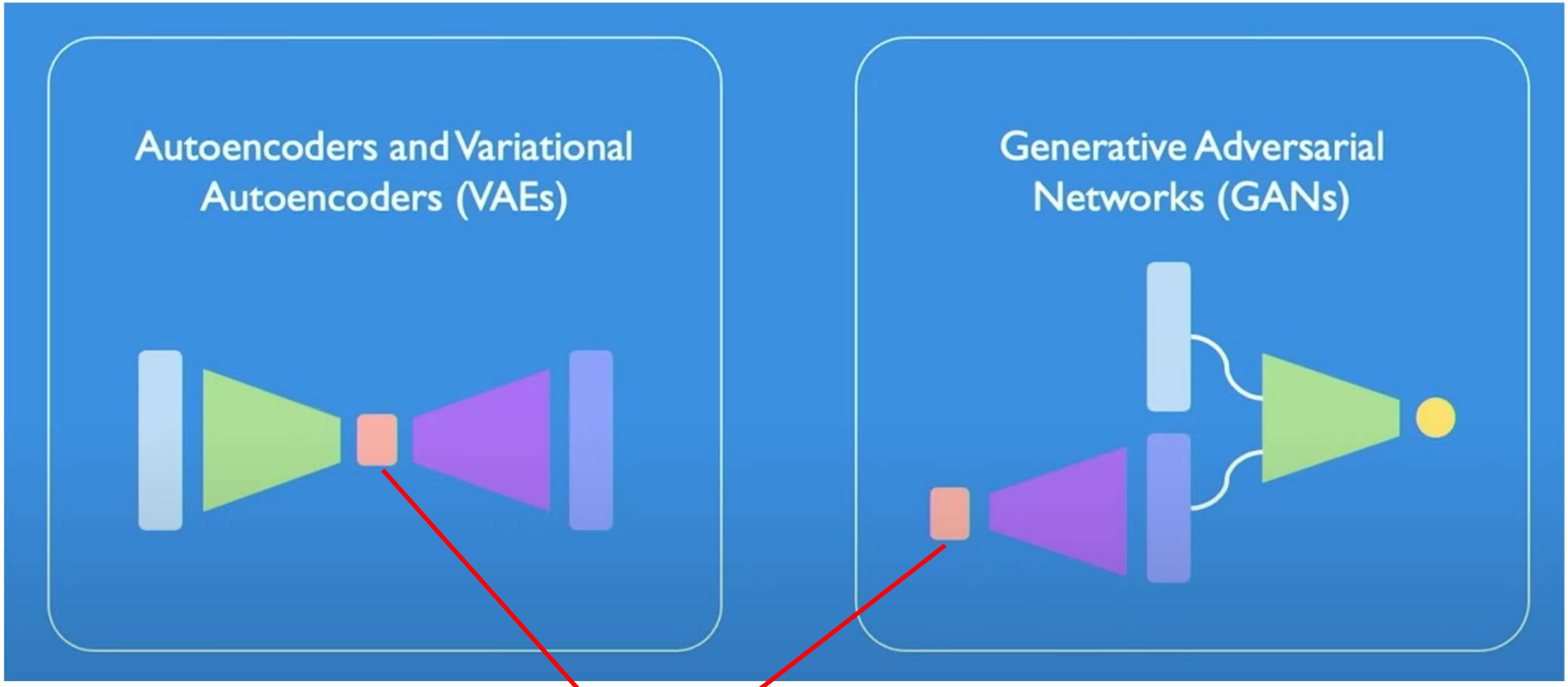


32x32 ImageNet

更常用的深度生成模型


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

更常用的深度生成模型


Autoencoders and Variational Autoencoders (VAEs)
Generative Adversarial Networks (GANs)
Latent Variable (隐变量)

Why Latent Variable

PixelRNN/CNNs define tractable density function, optimize likelihood of training data:

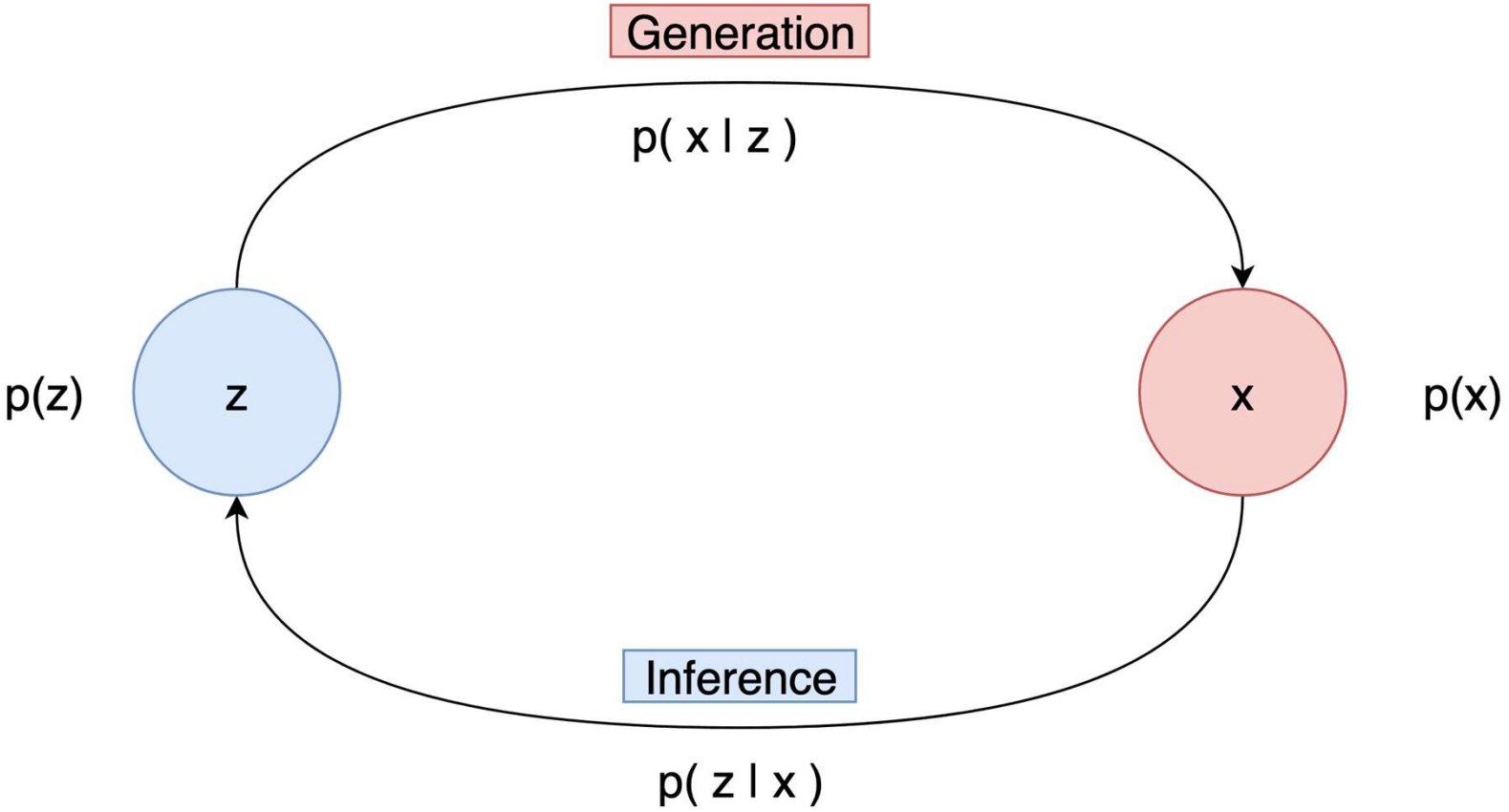
$$p_\theta(x) = \prod_{i=1}^{n} p_\theta(x_i | x_1, ..., x_{i-1})$$

Variational Autoencoders (VAEs) define intractable density function with latent z:

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

No dependencies among pixels, can generate all pixels at the same time!

Latent Variable

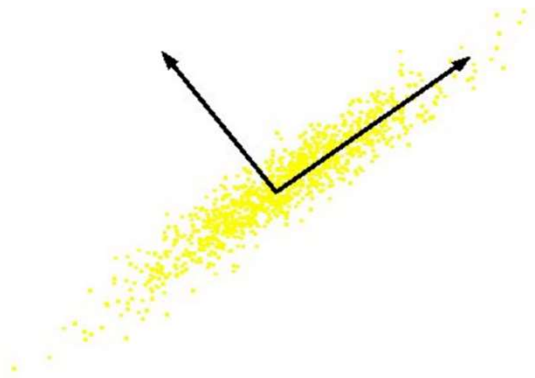


X: 图像
高维复杂数据

Z: 隐变量
低维简单分布

最常用的降维 — PCA

What is PCA: Unsupervised technique for extracting variance structure from high dimensional datasets.



PCA用于将高维数据投影到低维子空间，同时最大程度地保留数据的方差信息。它通过寻找数据的主轴（主成分），使得在这些轴上的投影方差最大，且主成分之间相互正交。

从数据逼近角度理解

问题定义

给定一个高维数据矩阵 $X \in \mathbb{R}^{n \times p}$，我们希望将其投影到一个低维空间，得到投影得分矩阵 $T \in \mathbb{R}^{n \times k}$，其中 $k < p$。我们的目标是：

- **找到一个投影矩阵 $W \in \mathbb{R}^{p \times k}$**，使得投影后的数据 $T = XW$ 能够最佳地近似原始数据 X，即**最小化原始数据与重构数据之间的差异。**

这个问题可以形式化为：

$$\min_{W} \quad \|X - TW^T\|_F^2 = \min_{W} \quad \|X - XWW^T\|_F^2$$

其中：

- $\|\cdot\|_F$ 表示 Frobenius 范数（矩阵元素的平方和的平方根）。
- $T = XW$ 是投影到低维空间的表示。
- $X_{\text{reconstructed}} = TW^T = XWW^T$ 是从低维表示重构的高维数据。

PCA求解

4. 转化为最大化问题

重构误差可以展开为：

$$\begin{aligned}\|\bar{X} - \bar{X}WW^T\|_F^2 &= \mathrm{tr}\left[(\bar{X} - \bar{X}WW^T)^T(\bar{X} - \bar{X}WW^T)\right] \\ &= \mathrm{tr}(\bar{X}^T\bar{X}) - \mathrm{tr}(W^T\bar{X}^T\bar{X}W)\end{aligned}$$

由于 $\mathrm{tr}(\bar{X}^T\bar{X})$ 为常数，最小化重构误差等价于最大化：

$$\max_W \quad \mathrm{tr}(W^T\bar{X}^T\bar{X}W)$$

令协方差矩阵 $S = \bar{X}^T\bar{X}$，目标函数变为：

$$\max_W \quad \mathrm{tr}(W^TSW)$$

约束条件仍为 $W^TW = I_k$。

PCA求解

6. 拉格朗日乘子法

构建拉格朗日函数：

$$L(W, \Lambda) = \text{tr}(W^T S W) - \text{tr}[\Lambda(W^T W - I_k)]$$

对 W 求导：

$$2SW - 2W\Lambda = 0 \quad \Rightarrow \quad SW = W\Lambda$$

这表明 W 的列向量是协方差矩阵 S 的特征向量，Λ 是对应的特征值对角矩阵。

7. 选择前 k 个特征向量

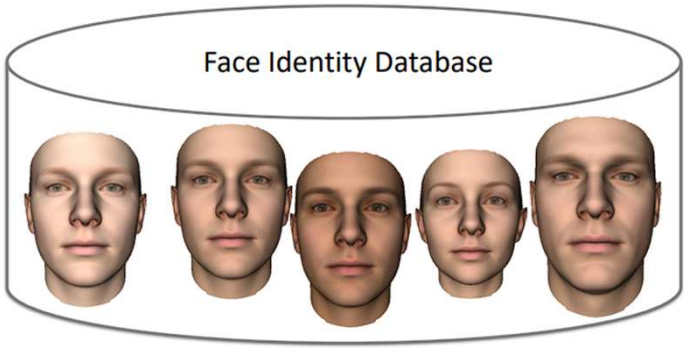
将 S 的特征值按照从大到小排序：

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \geq 0$$

对应的特征向量为 $v_1, v_2, \ldots, v_p$。选择前 k 个最大的特征值对应的特征向量构成矩阵 W：

$$W = [v_1, v_2, \ldots, v_k]$$

人脸三维几何PCA



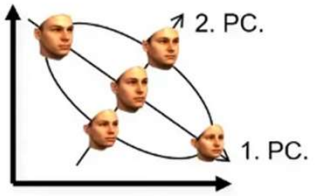
A vector space of 3D shapes and colors of a class of objects

- linear combinations of shapes $\mathbf{S}$ and textures $\mathbf{T}$

$$\mathbf{S} = \sum_i \alpha_i \boldsymbol{S}_i = \alpha_1 \cdot + \alpha_2 \cdot + \alpha_3 \cdot + \alpha_4 \cdot + \ldots$$

$$\boldsymbol{T} = \sum_i \beta_i \boldsymbol{T}_i = \beta_1 \cdot + \beta_2 \cdot + \beta_3 \cdot + \beta_4 \cdot + \ldots$$

- Often: Principal Component Analysis (PCA)



A morphable model for the synthesis of 3D faces. SIGGRAPH 1999.

人脸图片做PCA然后采样生成?

https://github.com/NVlabs/ffhq-dataset

人脸图片做PCA然后采样生成?

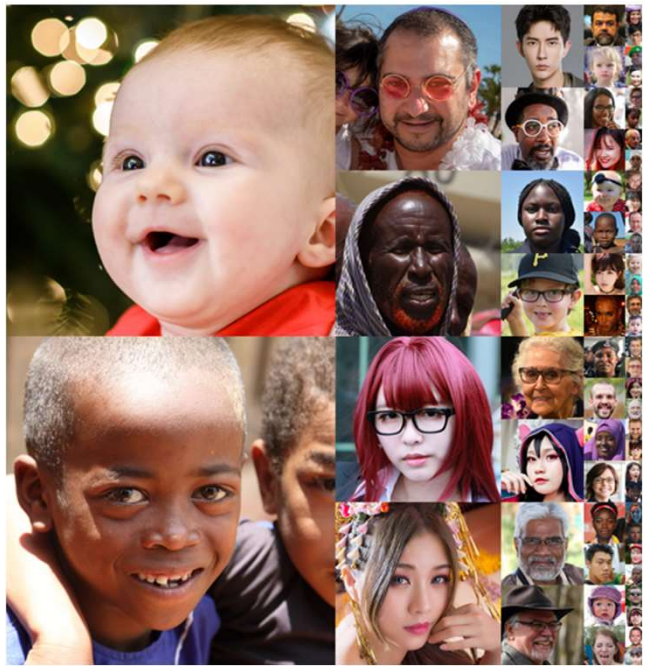
```python
# Read and preprocess images
for i in range(NUM_IMAGES):
    filename = os.path.join(IMAGE_FOLDER, '{:05d}.png'.format(i))
    with Image.open(filename) as img:
        img_array = np.array(img)
        if IMAGE_SIZE is None:
            IMAGE_SIZE = img_array.shape
        else:
            if img_array.shape != IMAGE_SIZE:
                raise ValueError(f"Image size mismatch at {filename}")
        img_flat = img_array.flatten()
        image_list.append(img_flat)

# Perform PCA
pca = PCA(n_components=NUM_COMPONENTS)
pca.fit(data)

# Transform data to PCA space
scores = pca.transform(data)  # Shape: (NUM_IMAGES, NUM_COMPONENTS)

# Generate random samples in PCA space
mean = np.mean(scores, axis=0)
std = np.std(scores, axis=0)
random_scores = np.random.normal(loc=mean, scale=std, size=(NUM_SAMPLES, NUM_COMPONENTS))
```

Problem



PCA

生成 / 最近邻

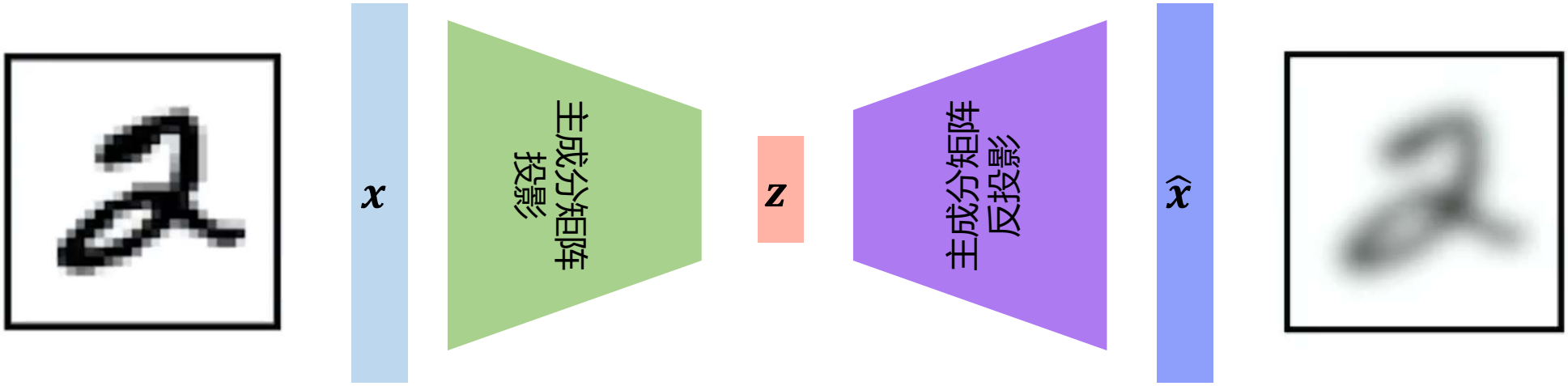
表达能力
不足

利用深度学习的强大能力

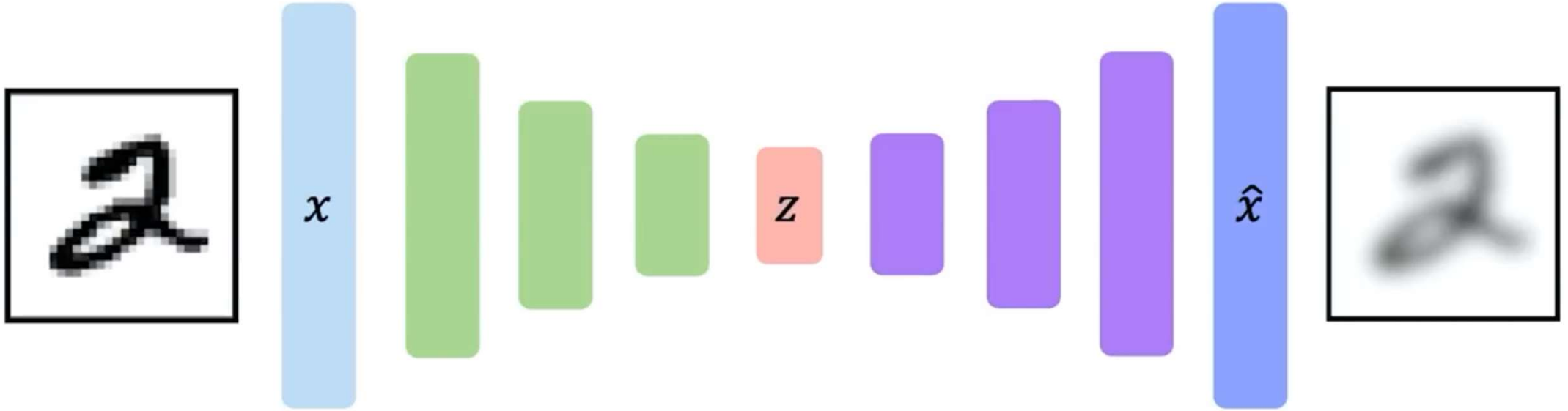
CNN

Replace PCA with
Deep Network

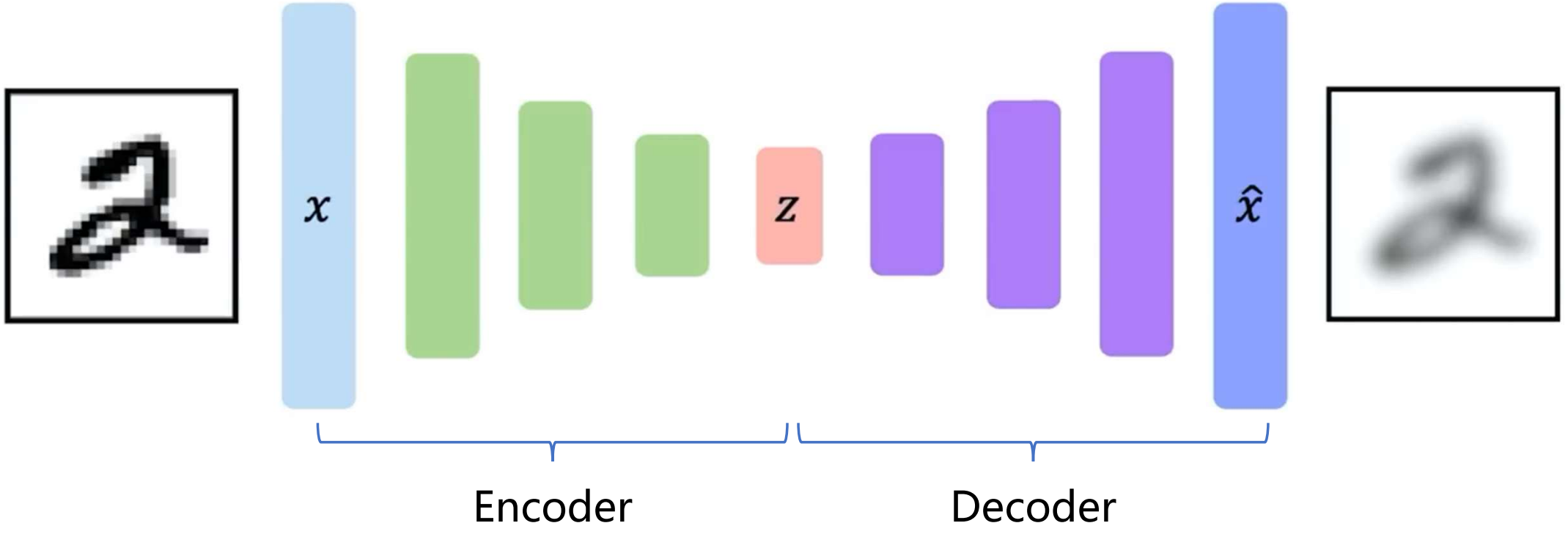
PCA



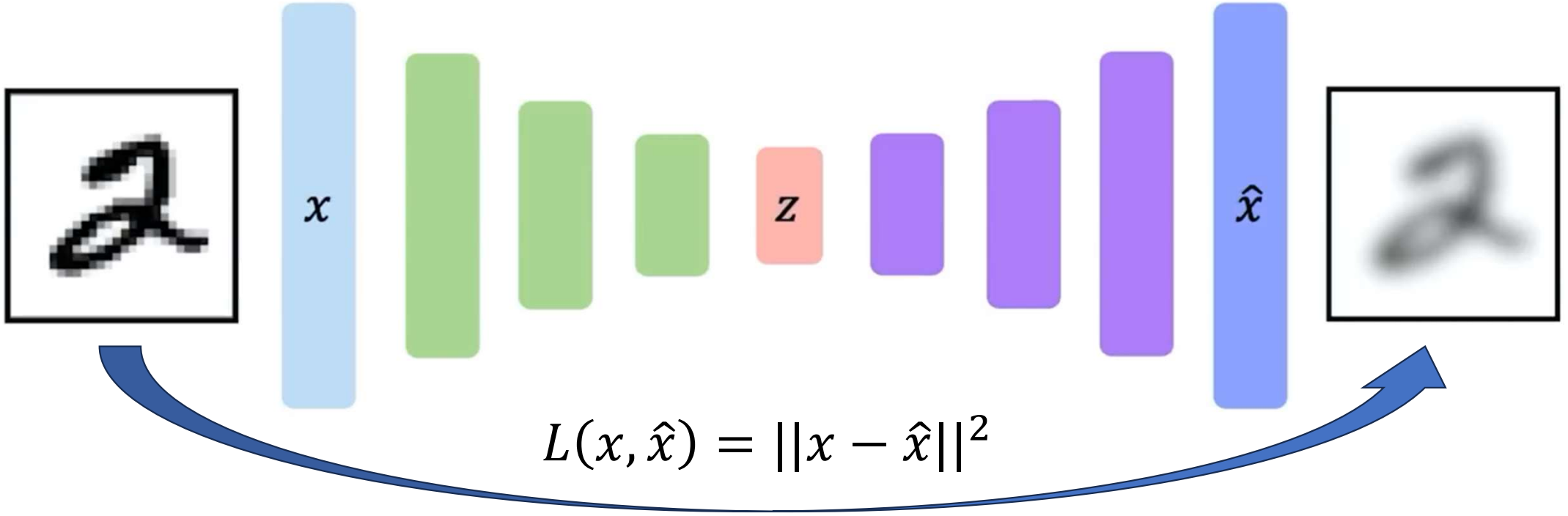
深度学习降维-AutoEncoder



深度学习降维-AutoEncoder



深度学习降维-AutoEncoder

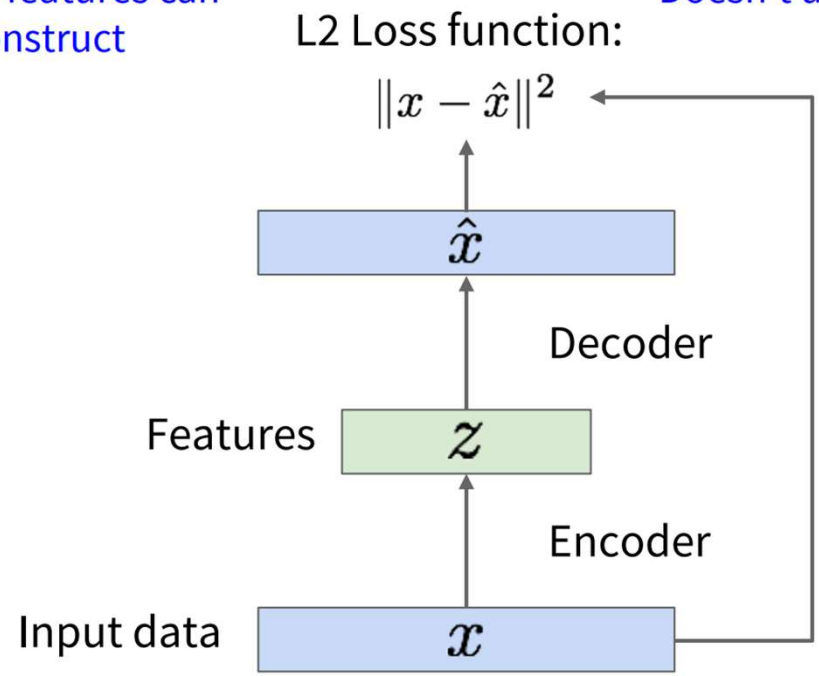


Auto:自动根据原始数据训练
无需人工标注等label

AutoEncoder — Learn Compact Features

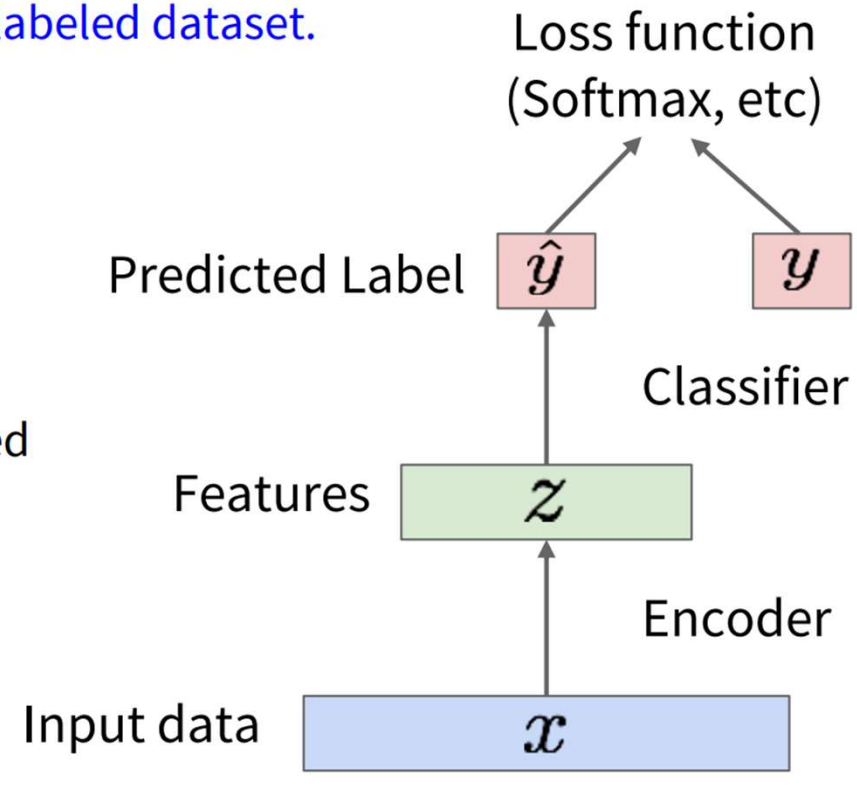
Train such that features can be used to reconstruct original data

Doesn't use labels!



AutoEncoder — Learn Compact Features

Transfer from large, unlabeled dataset to small, labeled dataset.



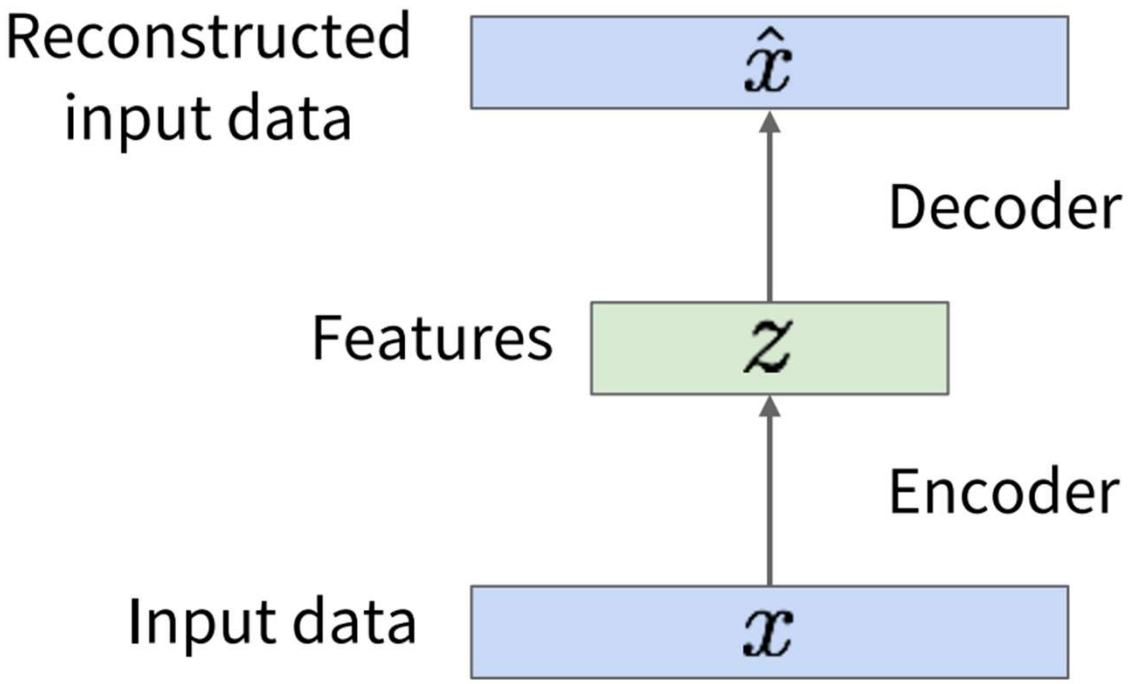
Encoder can be used to initialize a supervised model

Fine-tune encoder jointly with classifier

bird plane
dog deer truck

Train for final task (sometimes with small data)

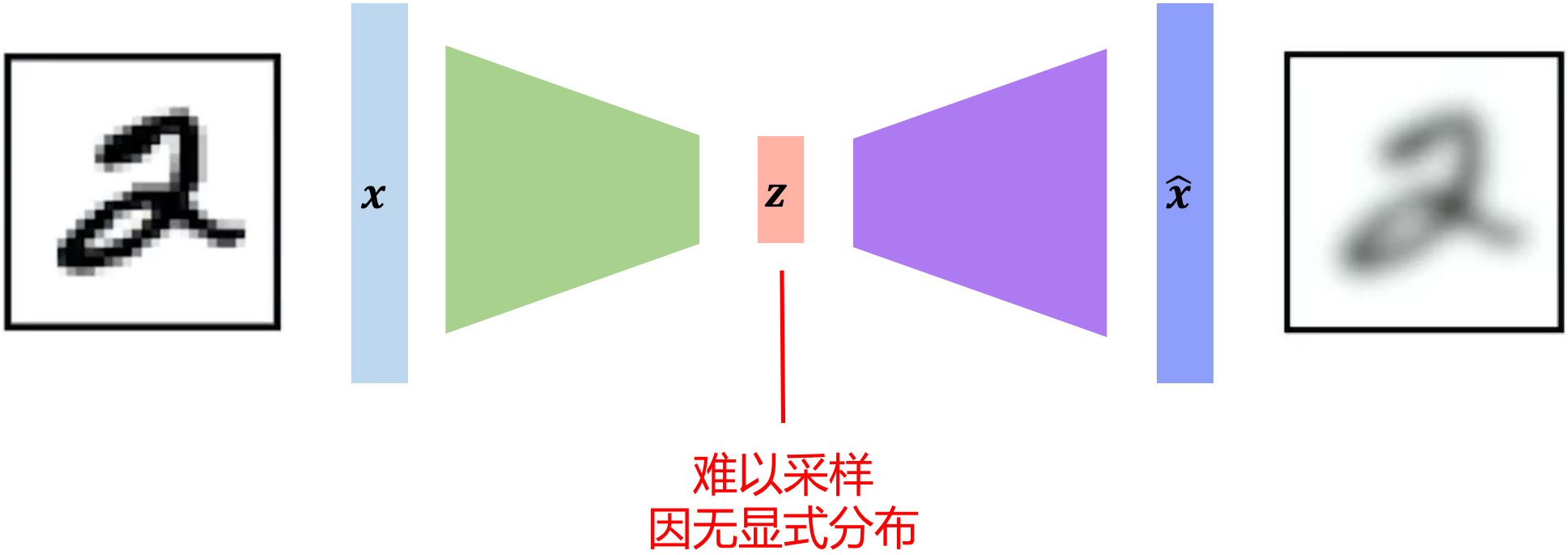
AutoEncoder是否适合做生成?



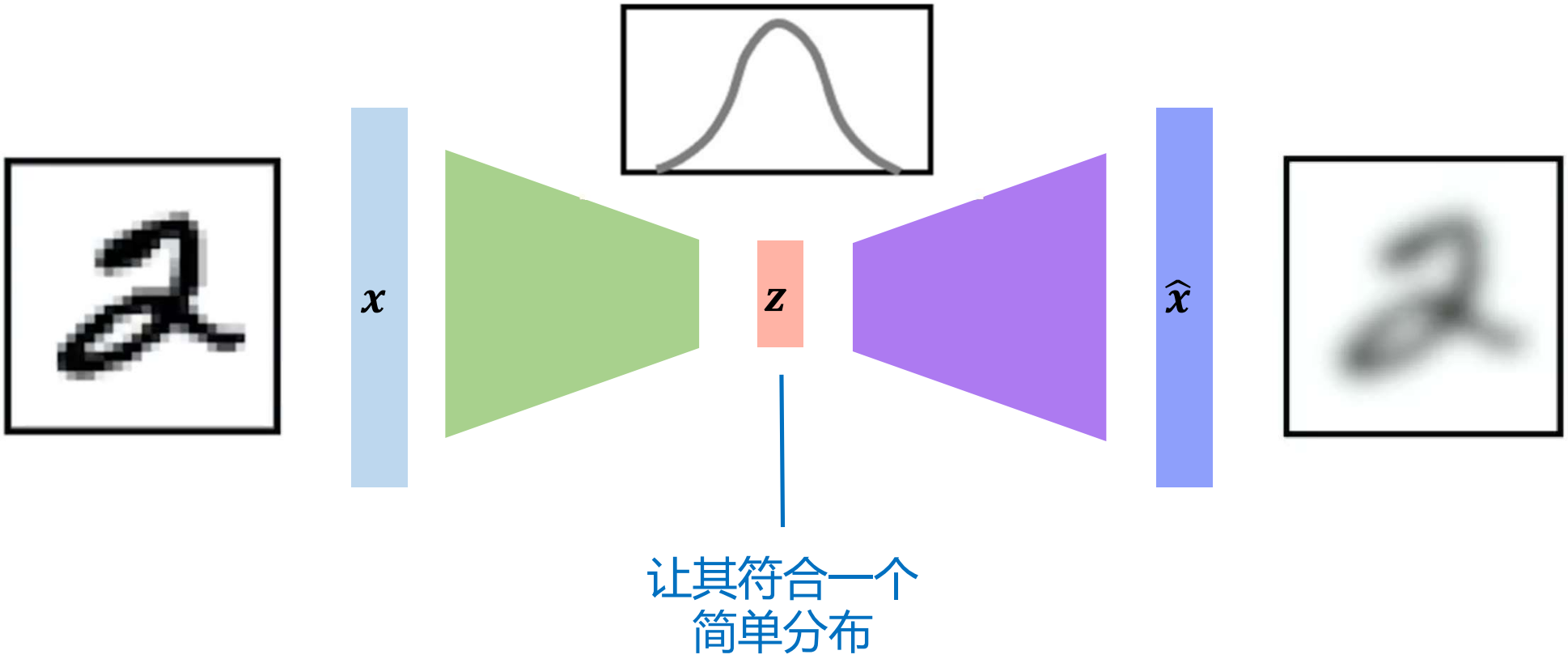
But we can't generate new images from an autoencoder because we don't know the space of z.

How do we make autoencoder a generative model?

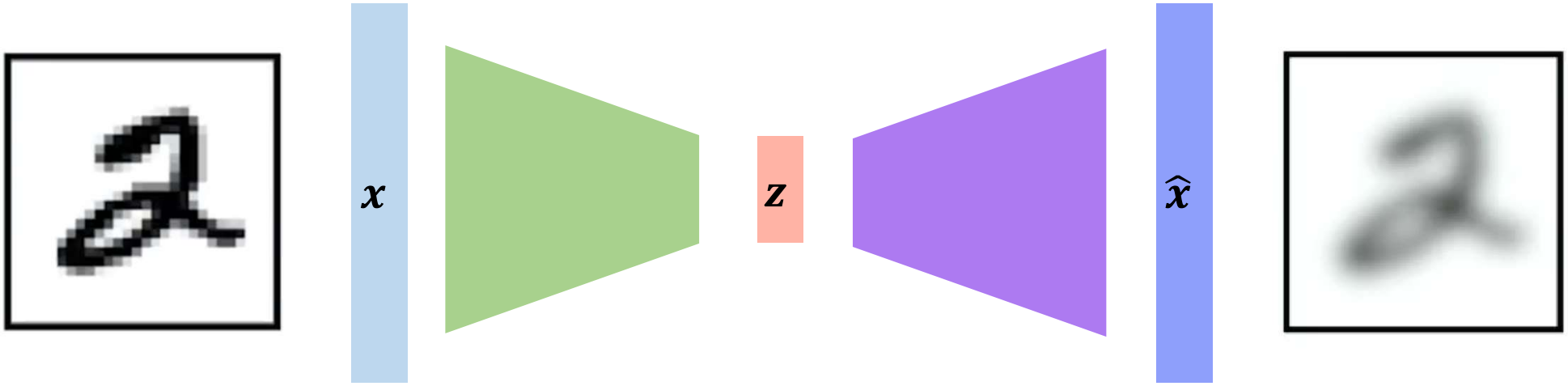
如何让Autoencoder适合生成



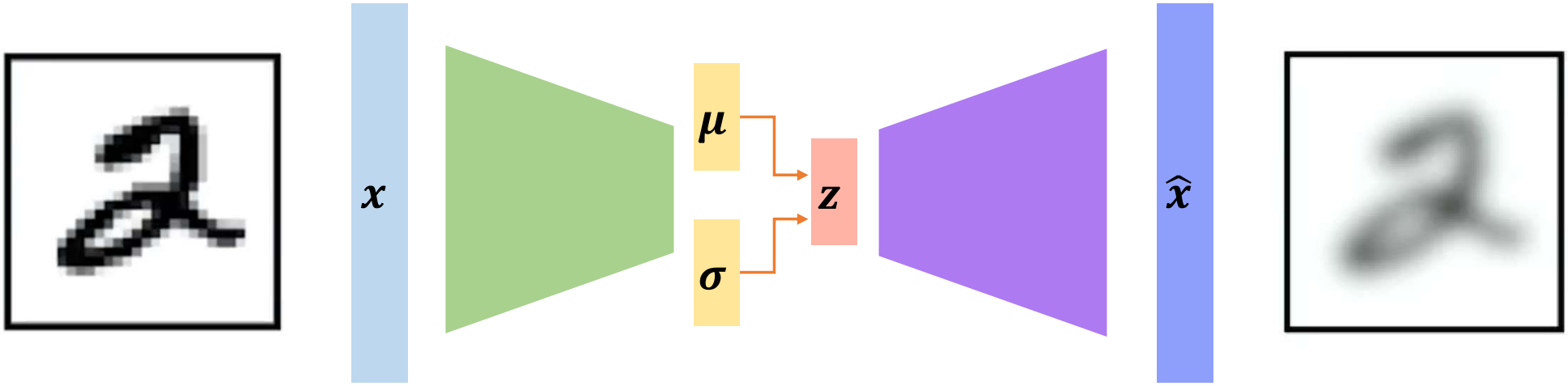
如何让Autoencoder适合生成



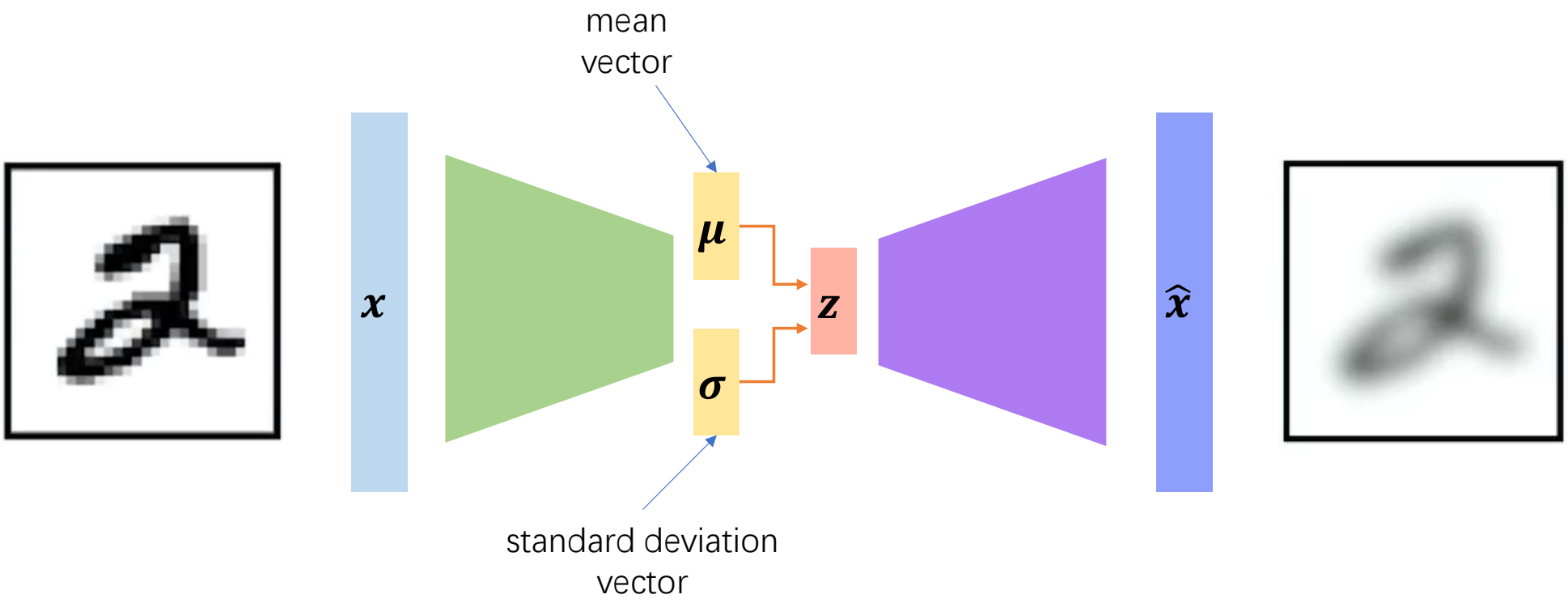
Variational Autoencoder



Variational Autoencoder

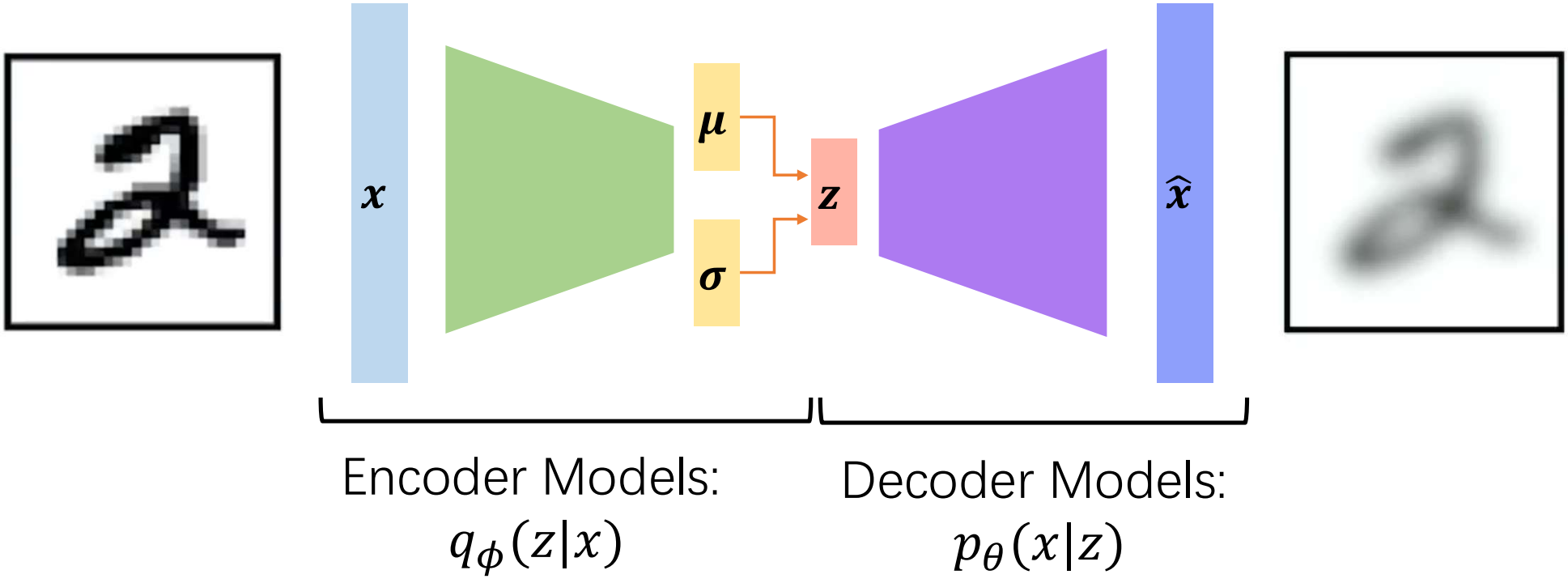


Variational Autoencoder

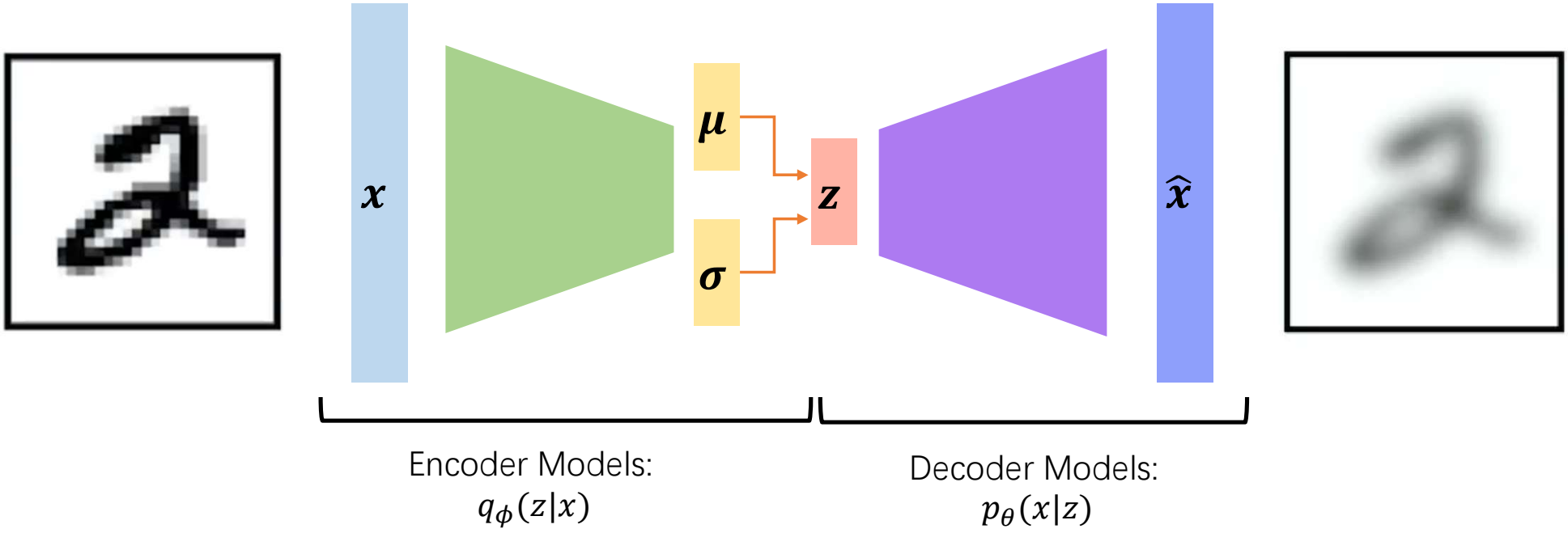


如果latent code符合一个正态分布
那就可以通过采样正态分布来生成Sample

How to Optimize VAE

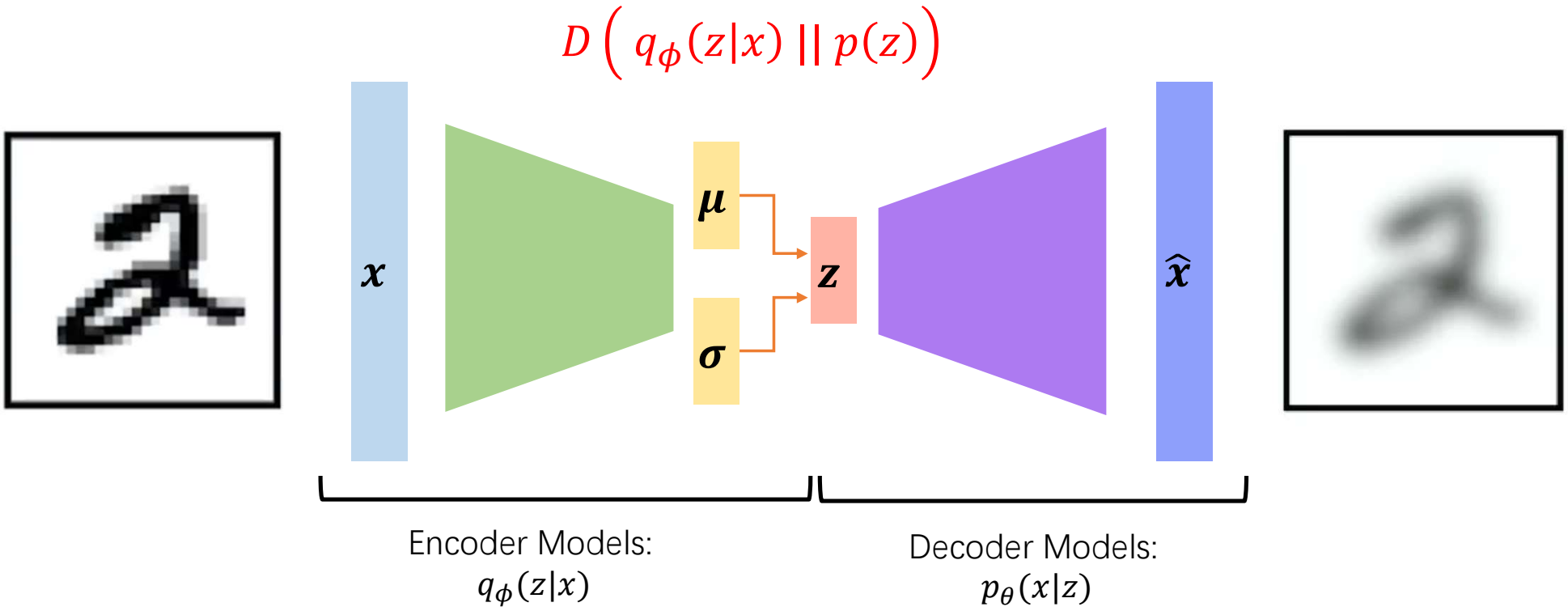


How to Optimize VAE



$$L(\phi, \theta, x) = (reconstruction\ loss) + (regularization\ term)$$

How to Optimize VAE



$$L(\phi, \theta, x) = (reconstruction\ loss) + (regularization\ term)$$

Distance between two distribution?

$$D\left(q_\phi(z|x) \,||\, p(z)\right)$$

Kullback–Leibler divergence

21 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

In mathematical statistics, the **Kullback–Leibler** (**KL**) **divergence** (also called **relative entropy** and **I-divergence**[1]), denoted $D_{\mathrm{KL}}(P \parallel Q)$, is a type of statistical distance: a measure of how one reference probability distribution P is different from a second probability distribution Q.[2][3] Mathematically, it is defined as

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right).$$

KL Divergence

If P=Q, D = 0

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right)$$

$$\mathrm{D}(\mathrm{f}||\mathrm{g}) = \int \mathrm{f}(\mathrm{x}) \ln \frac{\mathrm{f}(\mathrm{x})}{\mathrm{g}(\mathrm{x})} \mathrm{dx}$$

D(P,Q) >= 0

$$\begin{aligned}
&\int \mathrm{f}(\mathrm{x}) \ln \frac{\mathrm{f}(\mathrm{x})}{\mathrm{g}(\mathrm{x})} \mathrm{dx} \\
&= -\int \mathrm{f}(\mathrm{x}) \ln \frac{\mathrm{g}(\mathrm{x})}{\mathrm{f}(\mathrm{x})} \mathrm{dx} \\
&\geq -\int \mathrm{f}(\mathrm{x})\left(\frac{\mathrm{g}(\mathrm{x})}{\mathrm{f}(\mathrm{x})} - 1\right)\mathrm{dx} \\
&= -\int \mathrm{g}(\mathrm{x}) - \mathrm{f}(\mathrm{x})\mathrm{dx} \\
&= 0.
\end{aligned}$$

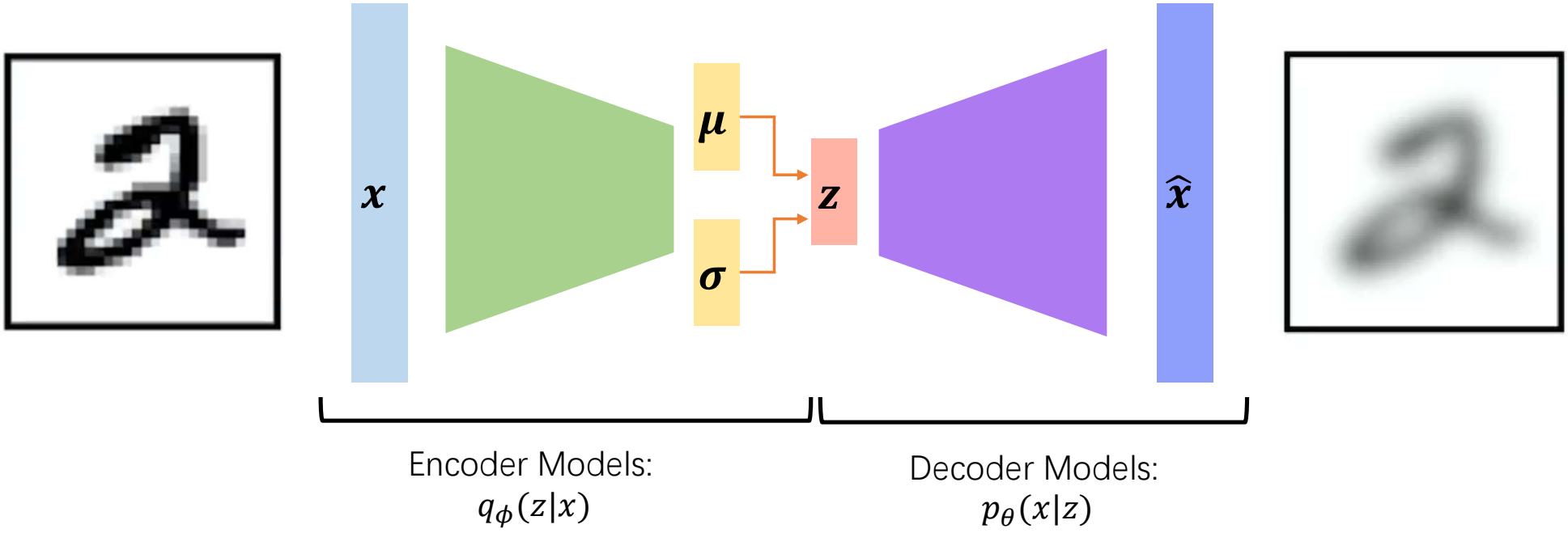
假设 Z 符合标准正态分布

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right)$$

$$D\left(q_\phi(\mathrm{z}|x) \parallel p(z)\right)$$

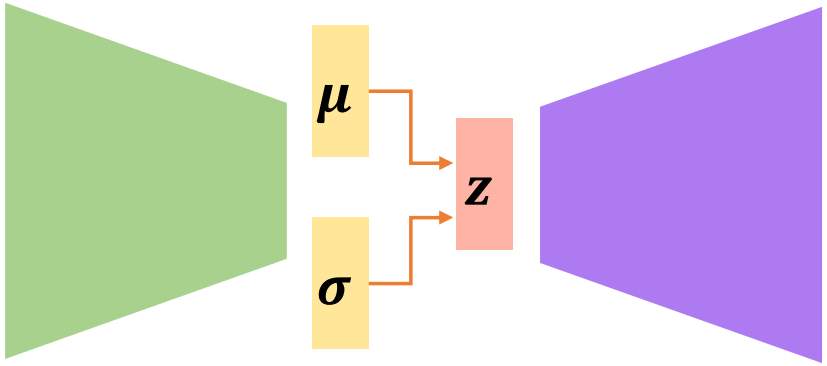
$$= -\frac{1}{2}\sum_{j=0}^{k-1}\left(\sigma_j + \mu_j^2 - 1 - \log\sigma_j\right)$$

How to Optimize VAE



$$L(\phi, \theta, x) = (reconstruction\ loss) + (regularization\ term)$$

How to obtain z during training

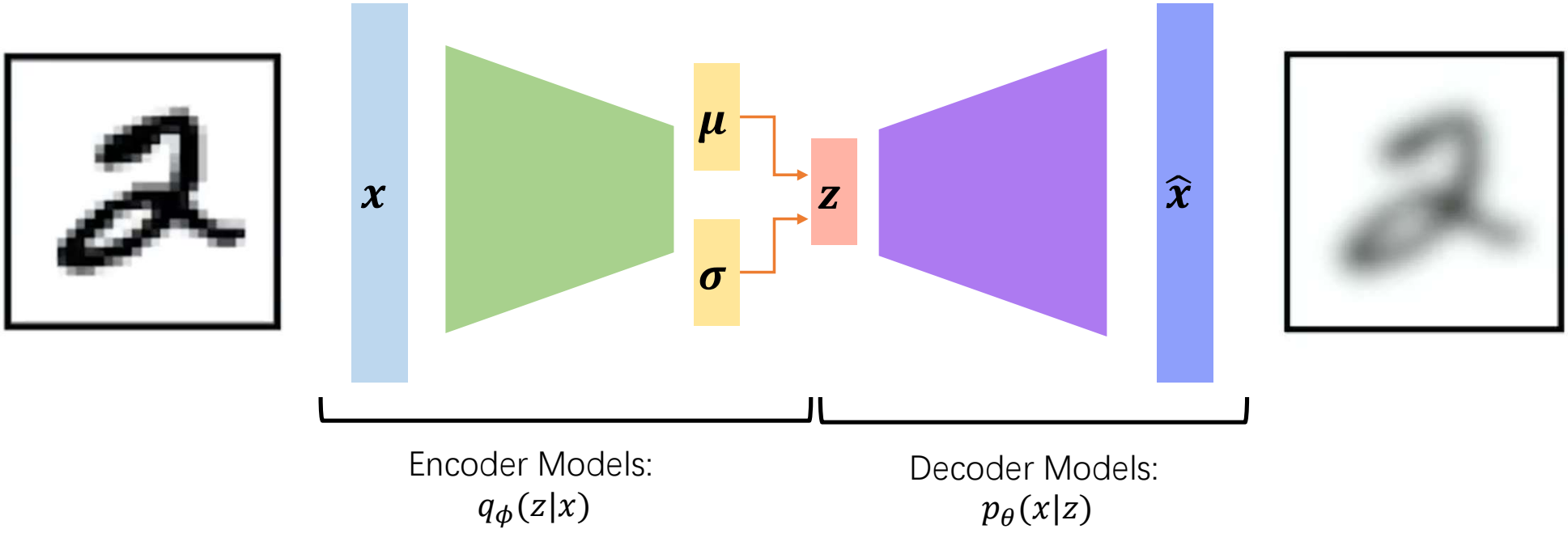


$$z \sim N(\mu, \sigma^2)$$

随机采样n-dim标准正态ε

$$z = \mu + \sigma \cdot \varepsilon$$

Total Differentiable Process



$$L(\phi, \theta, x) = (reconstruction\ loss) + (regularization\ term)$$

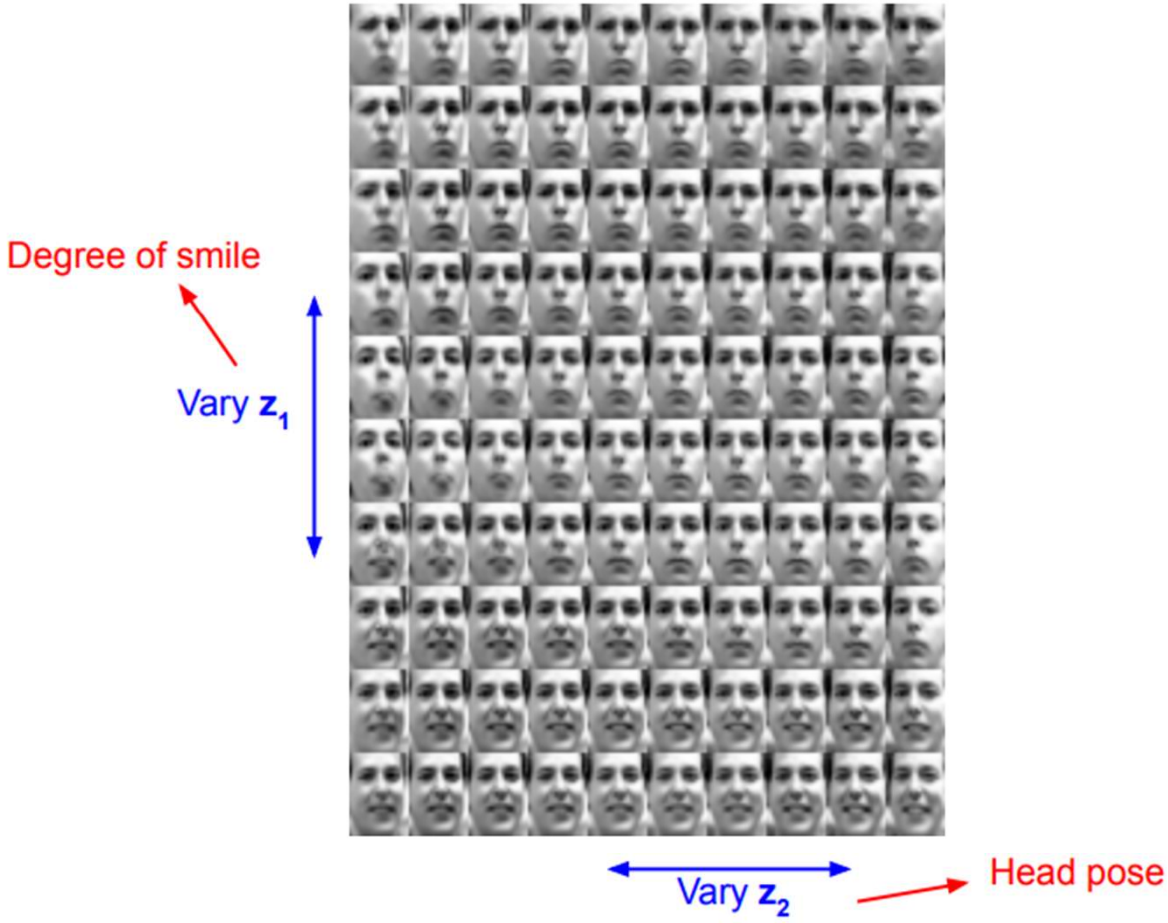
VAE for Image Generation

维度间正交

不同维

编码不同属性

可实现解耦生成



Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

VAE for Better Image Generation

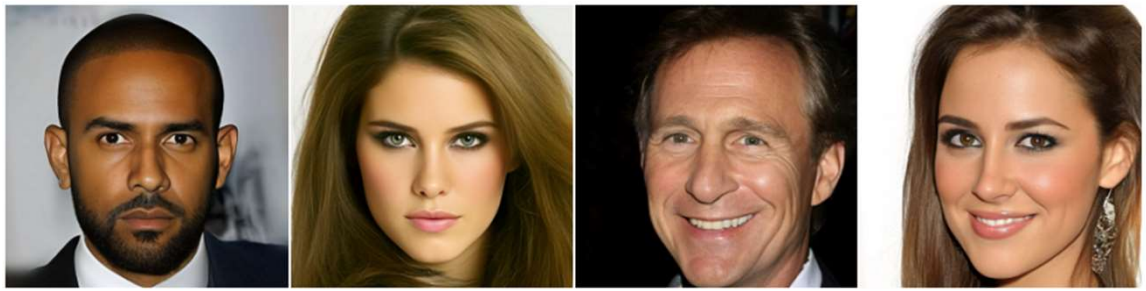
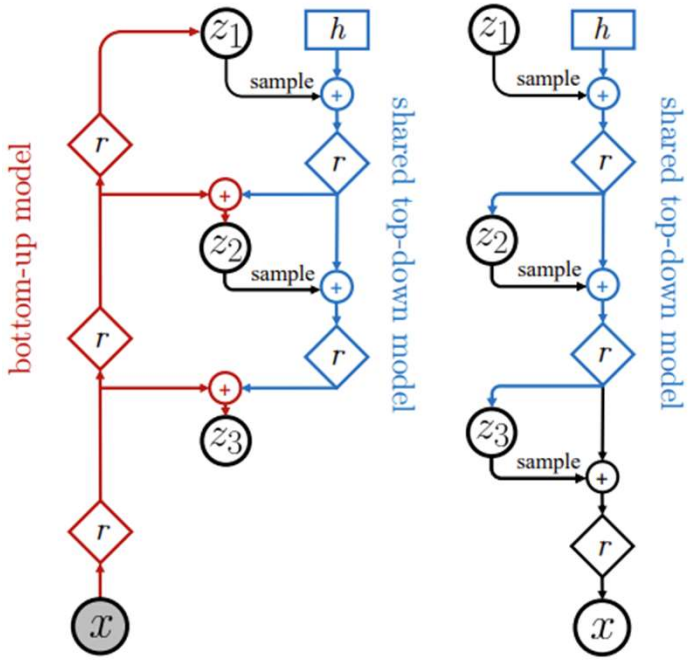


Figure 1: 256×256-pixel samples generated by NVAE, trained on CelebA HQ [28].

(e) FFHQ ($t = 0.5$)

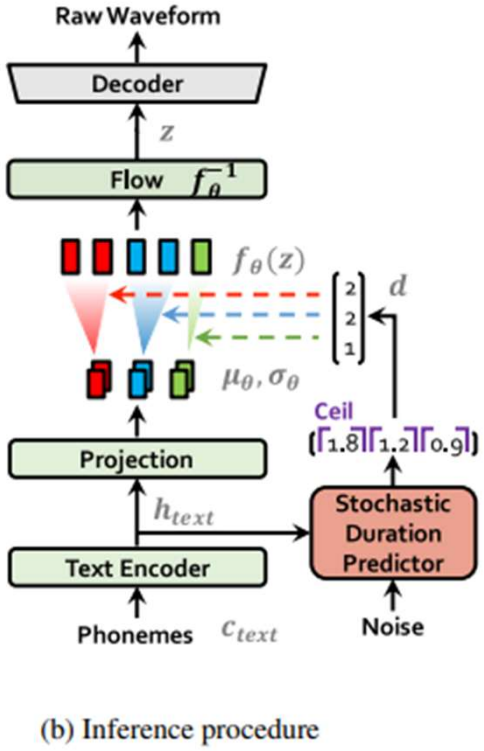


(a) Bidirectional Encoder (b) Generative Model

Figure 2: The neural networks implementing an encoder $q(\boldsymbol{z}|\boldsymbol{x})$ and generative model $p(\boldsymbol{x}, \boldsymbol{z})$ for a 3-group hierarchical VAE. ◇ denotes residual neural networks, ⊕ denotes feature combination (e.g., concatenation), and □ is a trainable parameter.

NVAE: A Deep Hierarchical Variational Autoencoder, NIPS 2020

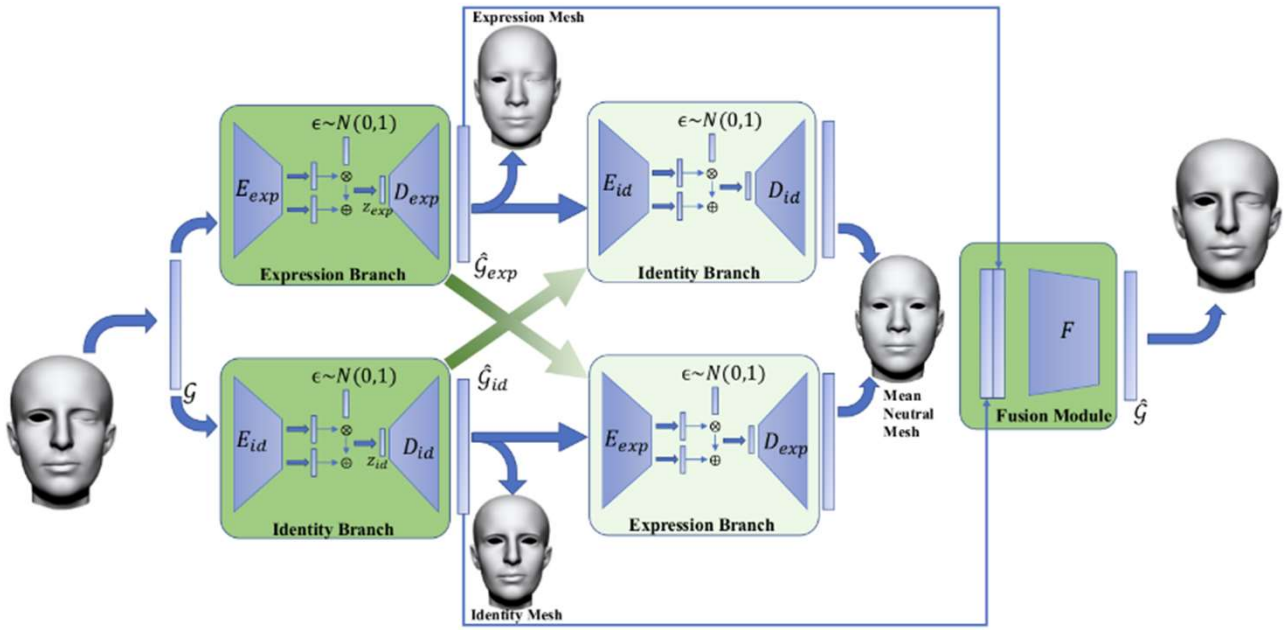
VAE for Audio Generation

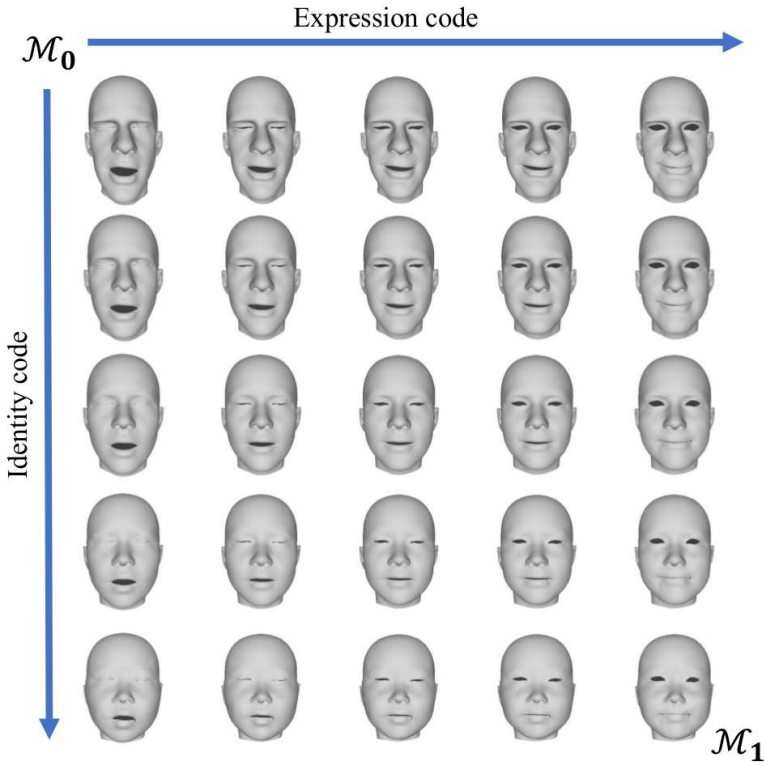


(b) Inference procedure

Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech. PMLR 2021.

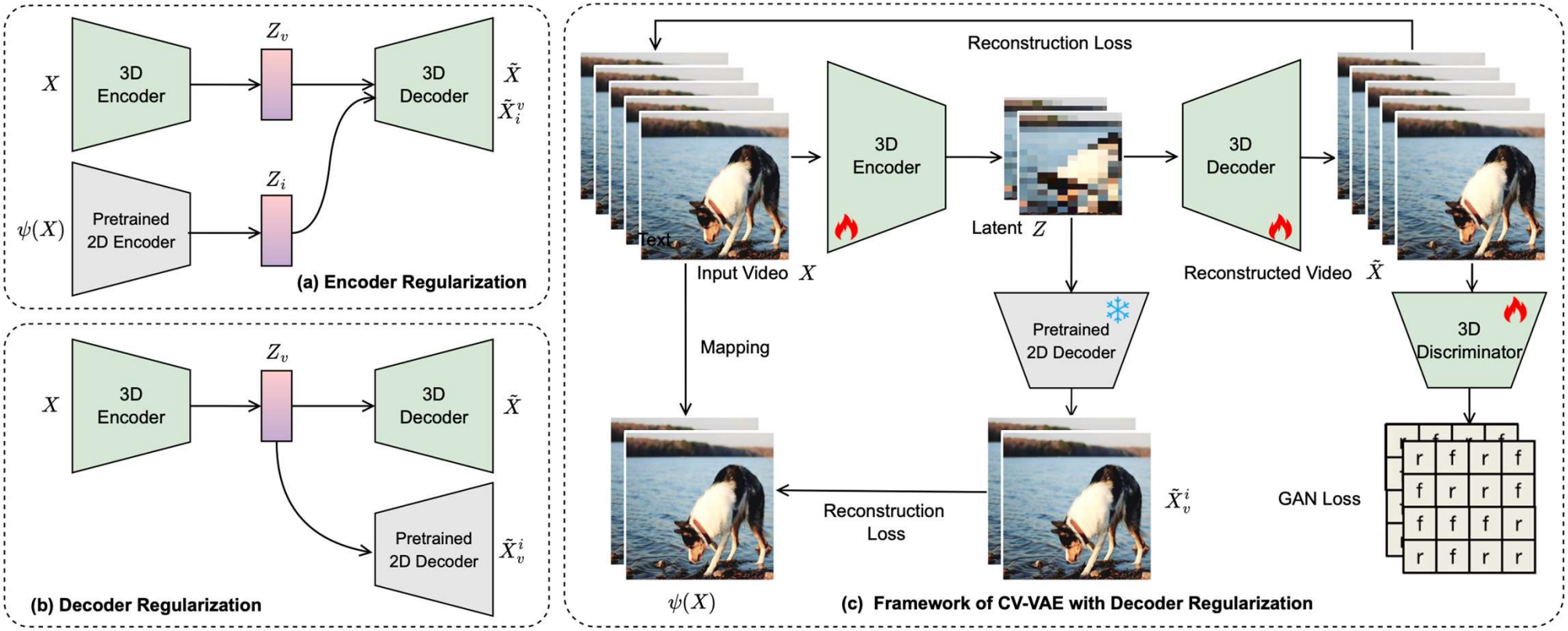
VAE for 3D Mesh





Disentangled Representation Learning for 3D Face Shape. CVPR 2019.

VAE for Video

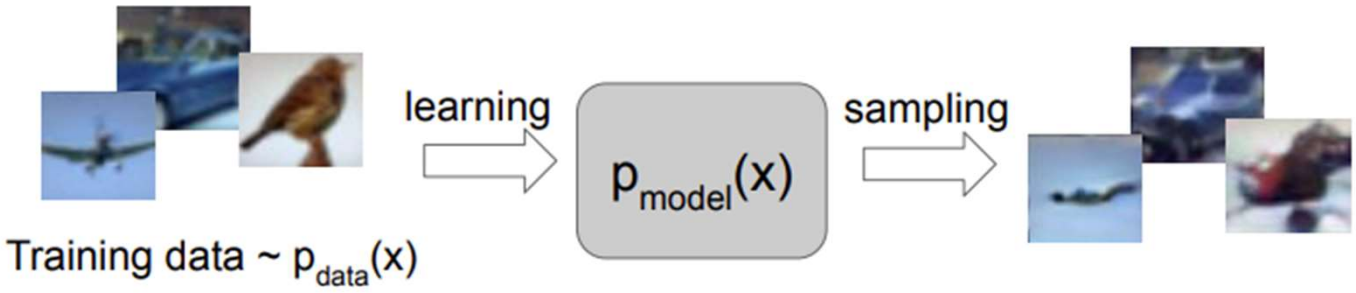


CV-VAE: A Compatible Video VAE for Latent Generative Video Models. 2024.

VAE for Video

CV-VAE: A Compatible Video VAE for Latent Generative Video Models. 2024.

Course Summary



Directly Model the Distribution:

PixelRNN / PixelCNN

Approximate the Distribution with Normal Distrubution :

PCA

Better Approximate the Distribution with Deep Learning:

VAE

中国科学技术大学

University of Science and Technology of China

谢谢观看!