Diffusion Models and Scientific Generative AI

—— Lecture 2: Foudations of Generative Models (1)

Professor Siyu Zhu
AI3 institute, Fudan University

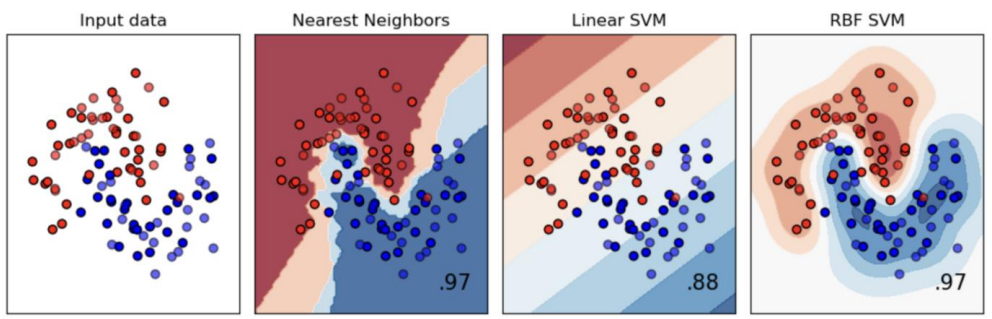
Contents

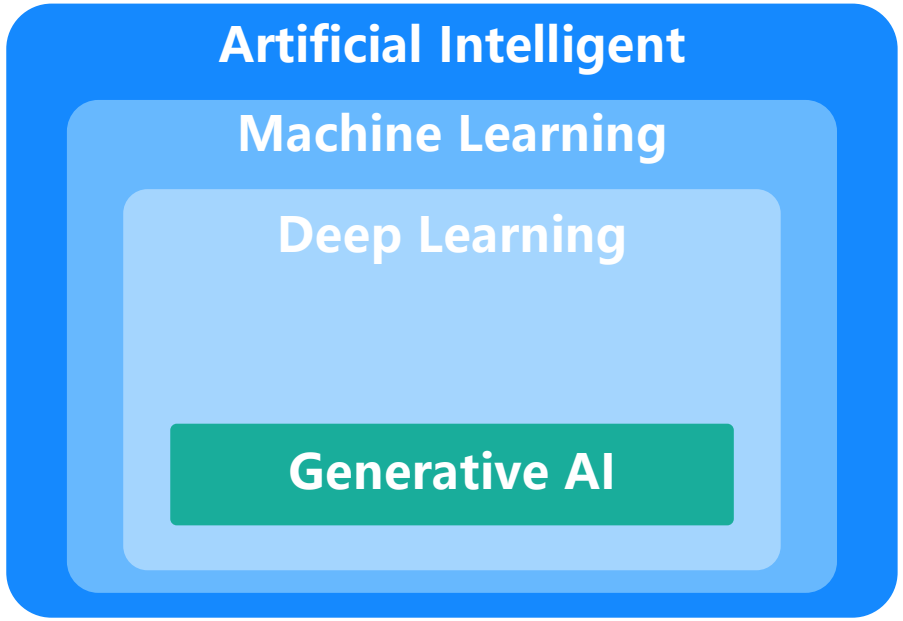
- **Last Class Review**
- Supervised vs Unsupervised Learning
- Generative Model
  - Autoregressive Model (PixelRNN/PixelCNN)
  - Variational Autoencoder (VAEs)
  - Generative Adversarial Networks (GAN)

Course Description

- **Objective**
  - Master the principles and theories of diffusion model-based generative AI.
  - Build programming skills and project development experience.
  - Apply diffusion-based generative AI to real-world problems.
- **Coursework**
  - Attendance 5%
  - Participation 5%
  - Literature Review 45%
  - Course Project 45%
  - **Students are encouraged to develop innovative projects that combine generative models with their academic disciplines.**

Conception of AI



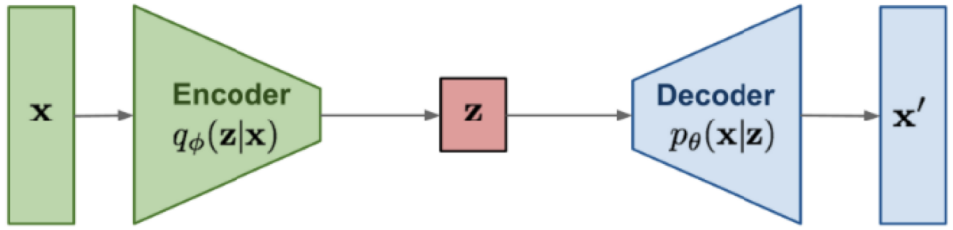


- **Artificial Intelligence (1956)**
  - Refers to computational intelligence designed to replace or surpass human intelligence.
- **Machine Learning (1990)**
  - A subfield of AI focused on analyzing large datasets to identify patterns, enabling systems to automatically learn, improve, and make accurate decisions and predictions.
- **Deep Learning (2012)**
  - A branch of machine learning that utilizes multi-layered neural networks to process data and make decisions.
- **Generative AI Models (2021)**
  - Based on specific prompts, data, or conditions, generative AI can produce new content, such as text, images, audio, video, or other material structures.

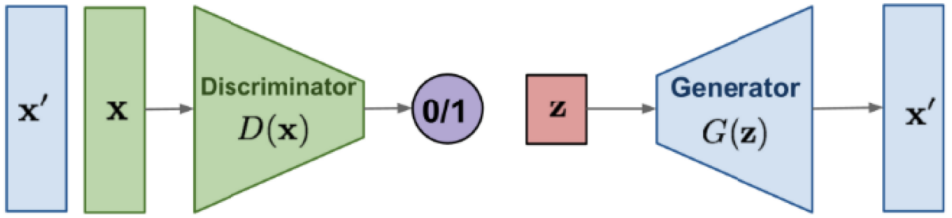
Category of Generative Models

- **Large Vision Models, LVM**

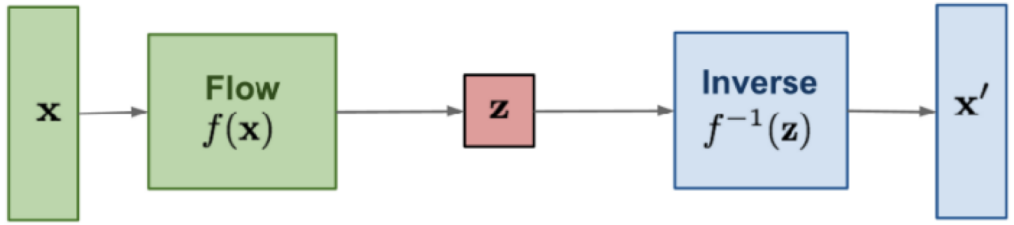
VAE: maximize variational lower bound



GAN: Adversarial training



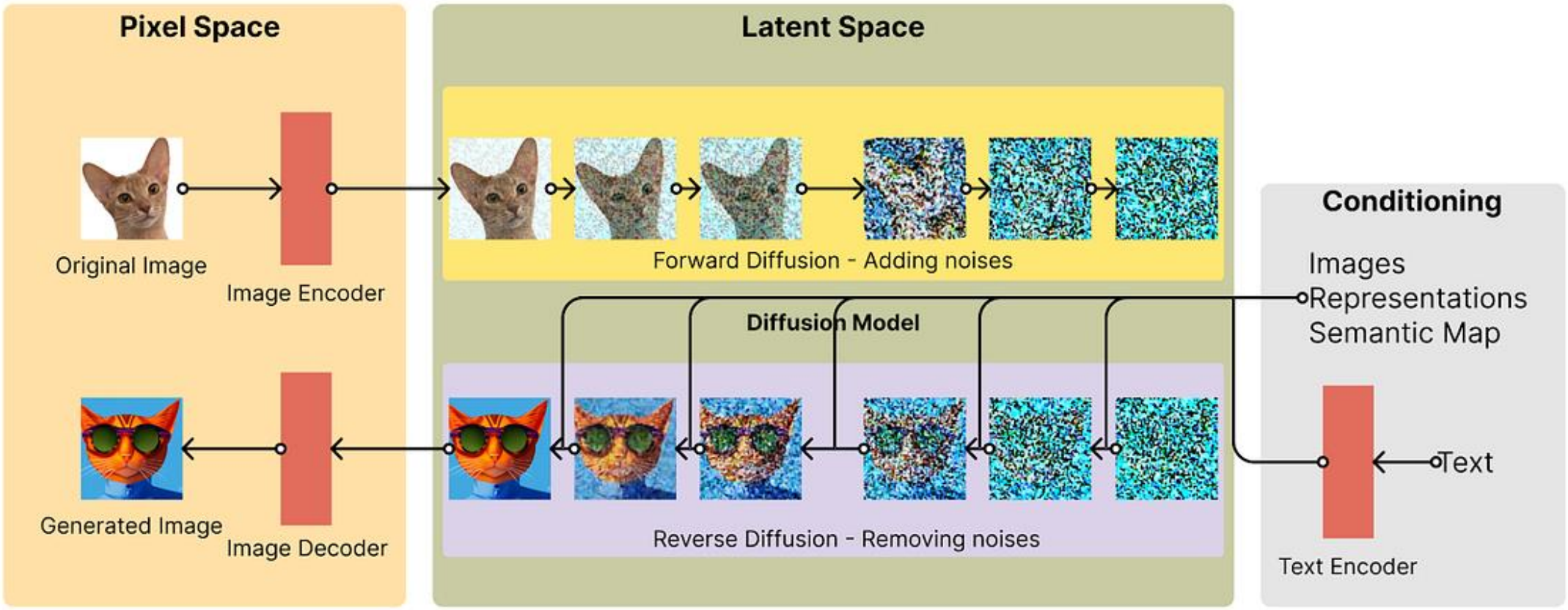
Flow-based models: Invertible transform of distributions



Diffusion models: Gradually add Gaussian noise and then reverse

Large Models: Autoregression v.s. Diffusion

- **Diffusion Model**



Contents

- Last Class Review
- **Supervised vs Unsupervised Learning**
- Generative Model
  - Autoregressive Model (PixelRNN/PixelCNN)
  - Variational Autoencoder (VAE)
  - Generative Adversarial Networks (GAN)

Supervised vs Unsupervised Learning

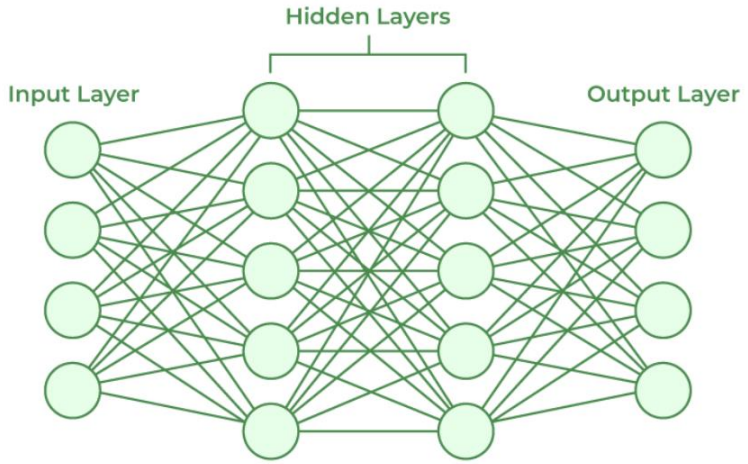
- **Supervised Learning**

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.

x



y

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Supervised vs Unsupervised 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.

Cat

Classification

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Supervised vs Unsupervised 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.

A cat sitting on a suitcase on the floor

Image captioning

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

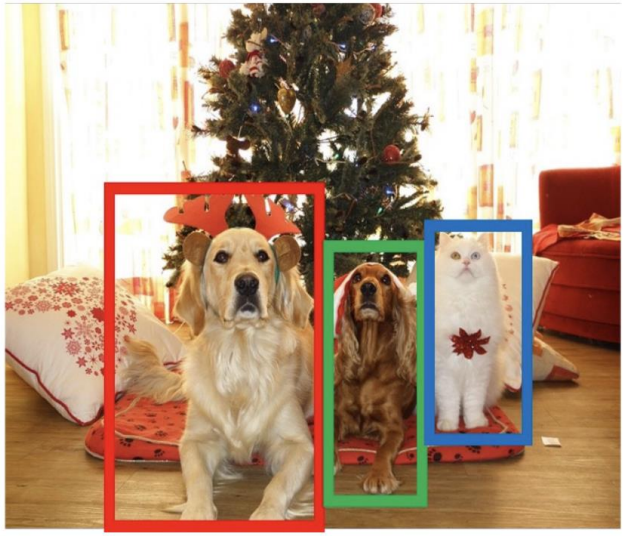
Supervised vs Unsupervised Learning

- **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.



DOG, DOG, CAT

Object Detection

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

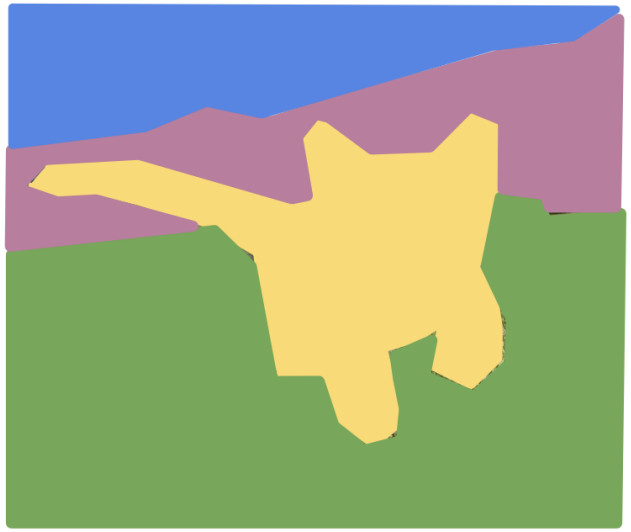
Supervised vs Unsupervised Learning

- **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.



GRASS, CAT, TREE, SKY

Semantic Segmentation

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Supervised vs Unsupervised Learning

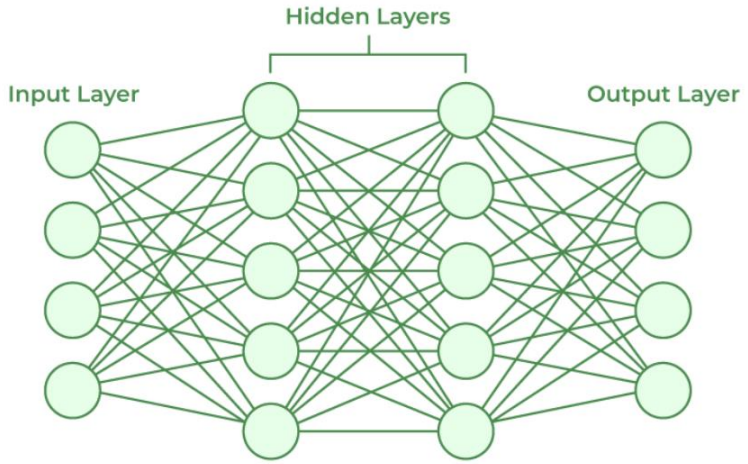
- **Unsupervised Learning**

Data: x
Just data, no labels!

Goal: Learn some underlying hidden structure of the data

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

X



y

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

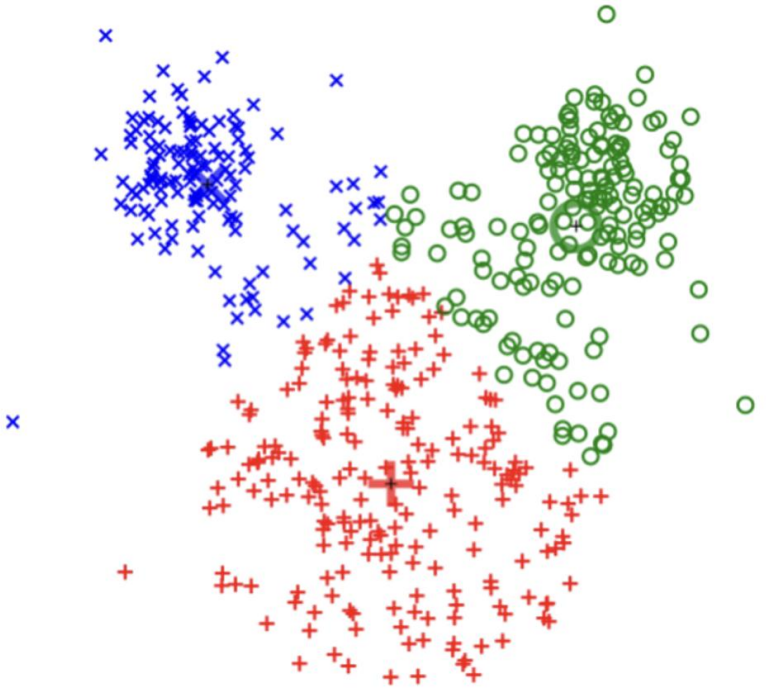
Supervised vs Unsupervised Learning

- **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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

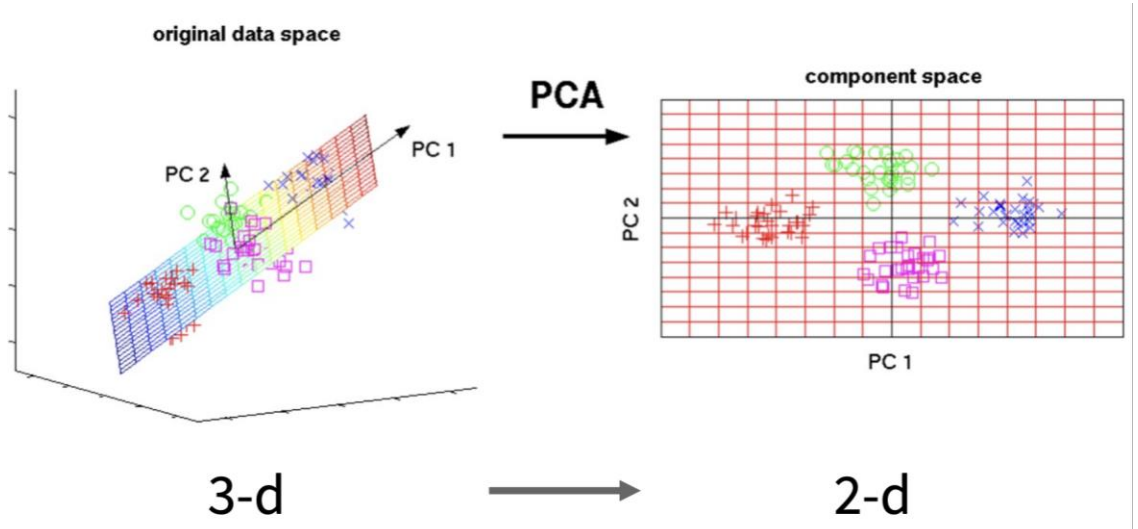
Supervised vs Unsupervised Learning

- **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)

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Supervised vs 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.

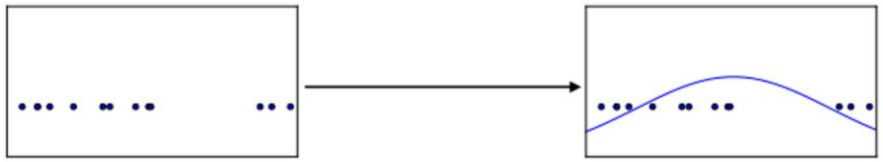
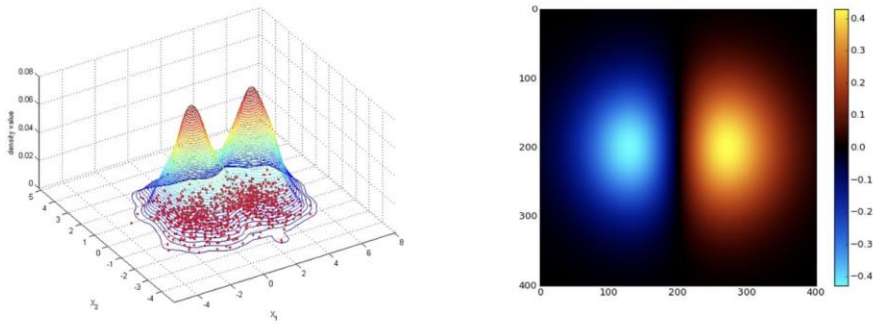


Figure copyright Ian Goodfellow, 2016. Reproduced with permission.

1-d density estimation



2-d density estimation

Modeling p(x)

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Supervised vs Unsupervised 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.

Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden structure of the data

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Supervised vs Unsupervised 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.

Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden structure of the data

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

Self-Supervised Learning

Data: (x, pseudo generated y)
No manual labels!

Goal: Learn to generate good features (reduce the data to useful/generic features)

Example: Classification in downstream applications where we have limited data

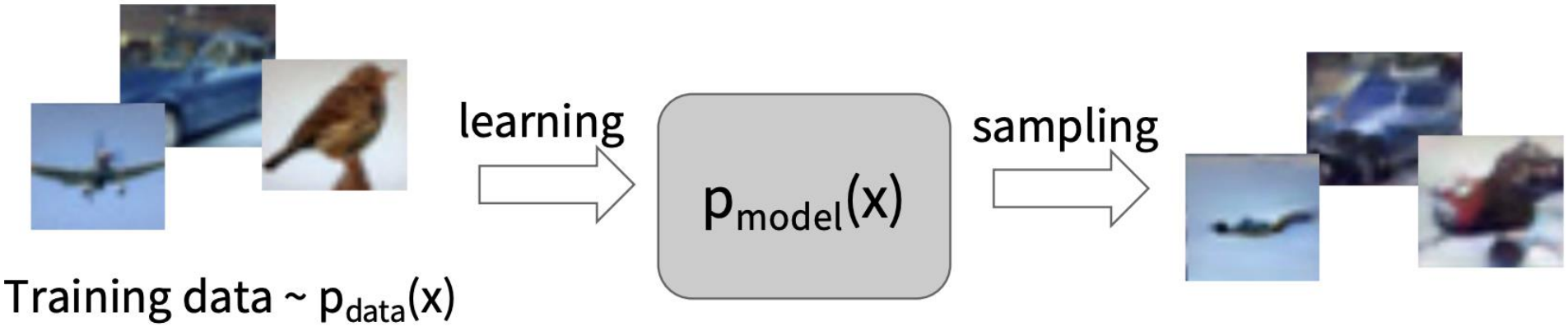
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Contents

- Last Class Review
- Supervised vs Unsupervised Learning
- **Generative Model**
  - Autoregressive Model （PixelRNN/PixelCNN ）
  - Variational Autoencoder (VAE)
  - Generative Adversarial Networks (GAN)

Generative Model

- **Given training data, generate new samples from same distribution.**



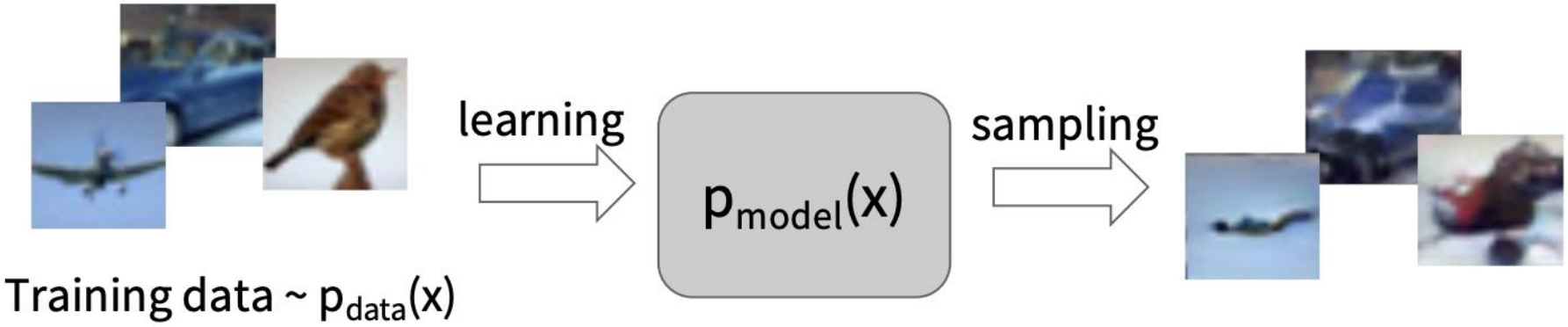
Objectives:

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Generative Model

- **Given training data, generate new samples from same distribution.**



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.

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Generative Model

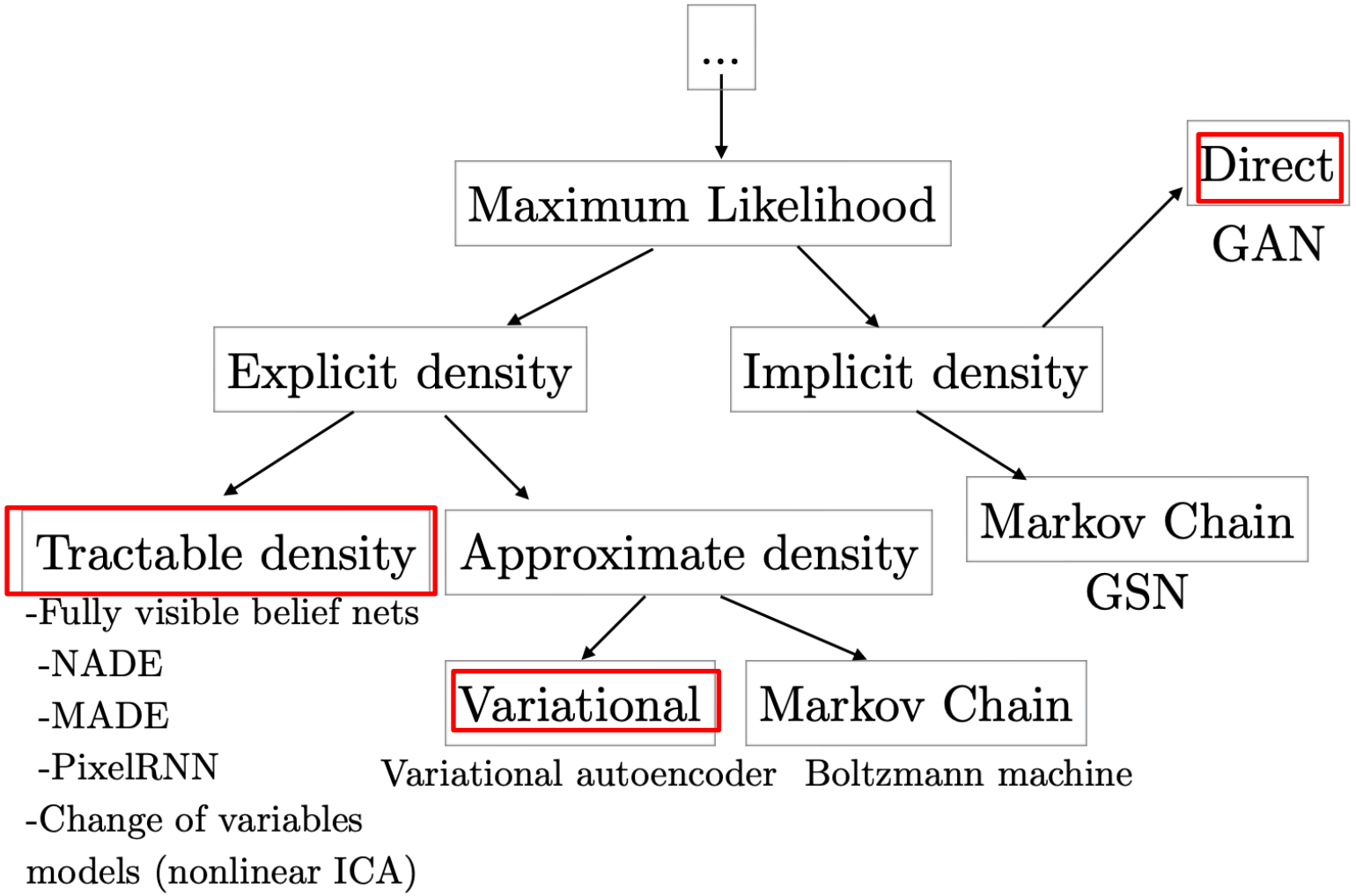
- **Why Generative Models ?**



- Realistic samples for artwork, super-resolution, colorization, etc.
- Learn useful features for downstream tasks such as classification.
- Getting insights from high-dimensional data (physics, medical imaging, etc.)
- Modeling physical world for simulation and planning (robotics and reinforcement learning applications)
- Many more ...

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Taxonomy of Generative Models

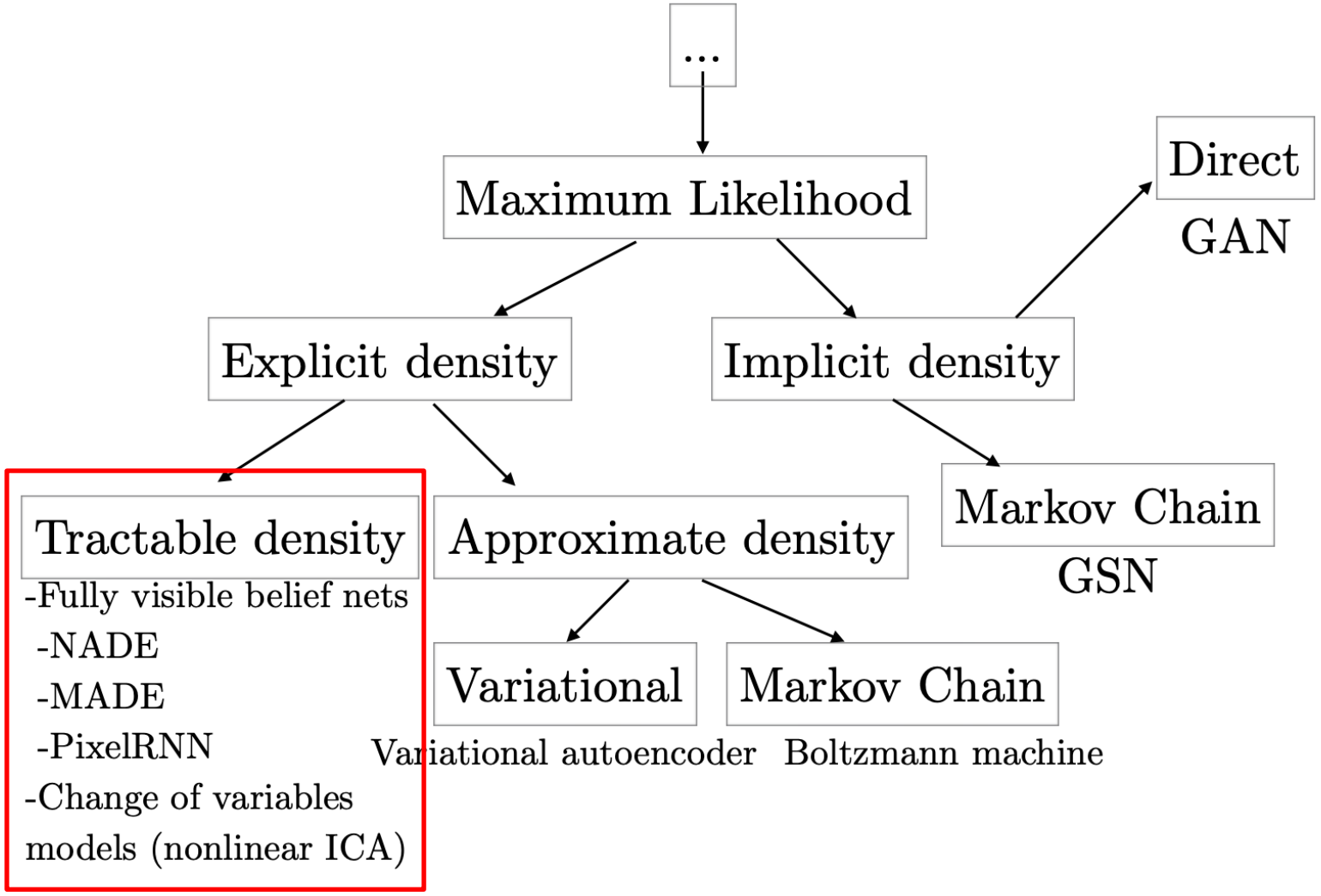


slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Contents

- Last Class Review
- Supervised vs Unsupervised Learning
- **Generative Model**
  - **Autoregressive Model （PixelRNN/PixelCNN ）**
  - Variational Autoencoder (VAE)
  - Generative Adversarial Networks (GAN)

Taxonomy of Generative Models



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Autoregressive Models

- **A type of statistical model used for time series analysis. They predict future values based on past values of the time series itself.**

A simple autoregressive model can be expressed as:

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \epsilon_t$$

where:

- X_t is the value at the current time point.
- c is a constant term.
- $\phi_1, \phi_2, \ldots, \phi_p$ are the model parameters.
- p is the order of the model, indicating how many past values are used.
- ϵ_t is the white noise error term.

Autoregressive Models

- **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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Autoregressive Models

- **Fully visible belief network (FVBN)**

Explicit density model

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

$$p(x) = p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$

Likelihood of image x

Joint likelihood of each pixel in the image

Probability of i'th pixel value given all previous pixels

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

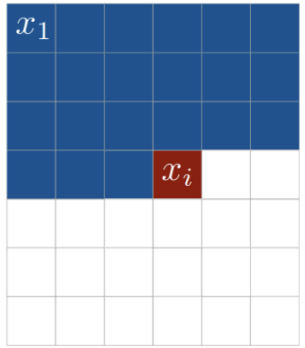
Autoregressive Models

- **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

Complex distribution over pixel values => Express using a neural network!

Then maximize likelihood of training data

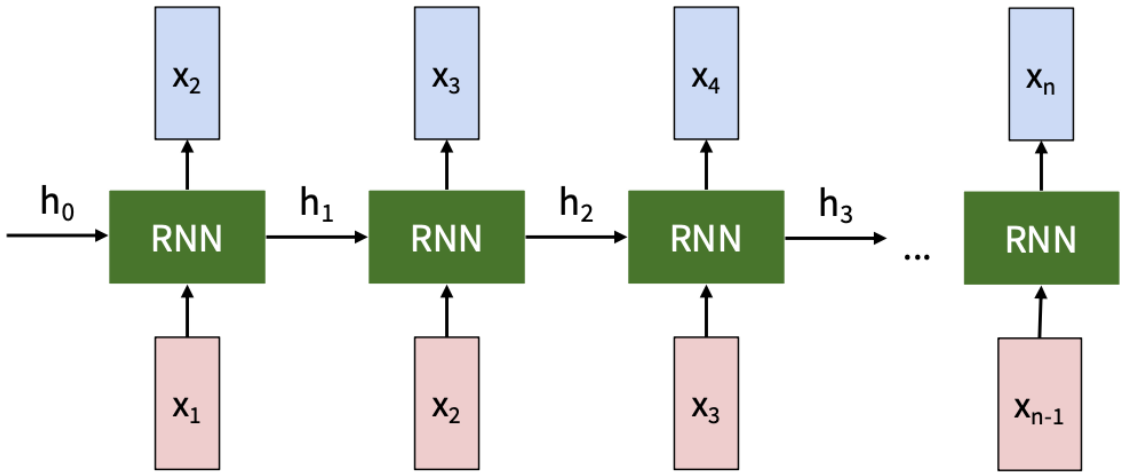
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Autoregressive Models

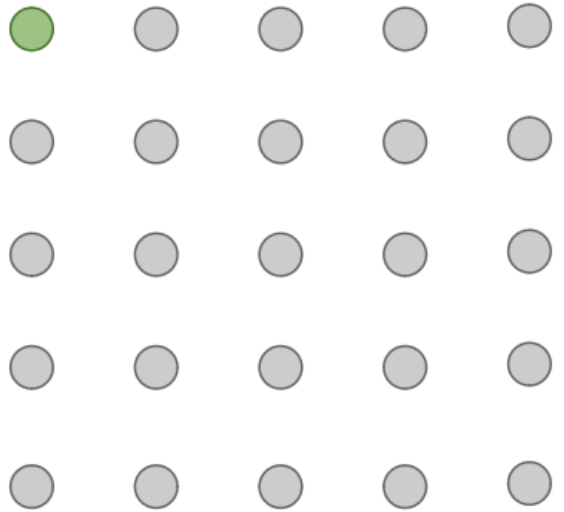
- **PixelRNN [van der Oord et al. 2016]**

Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)



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



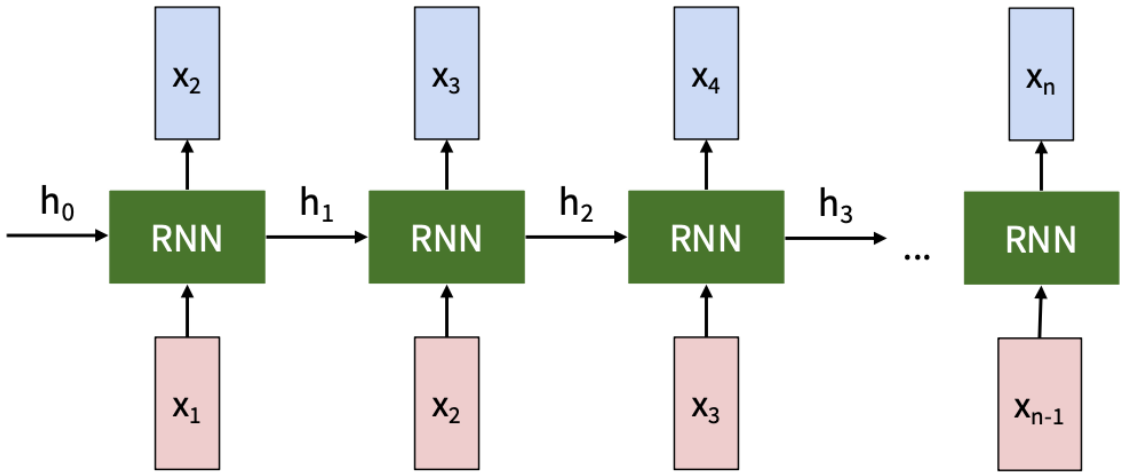
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Autoregressive Models

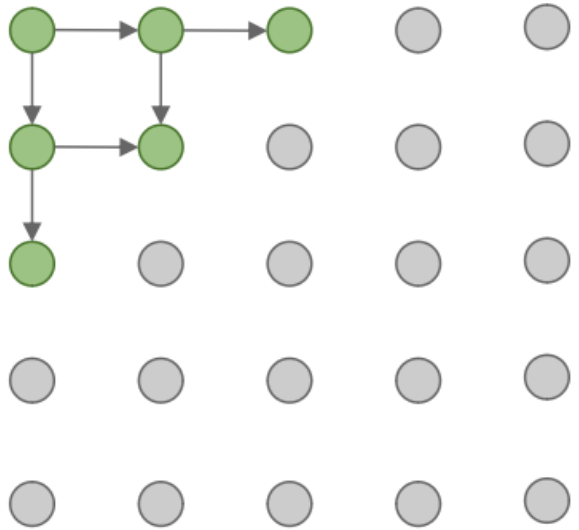
- **PixelRNN [van der Oord et al. 2016]**

Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)



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



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

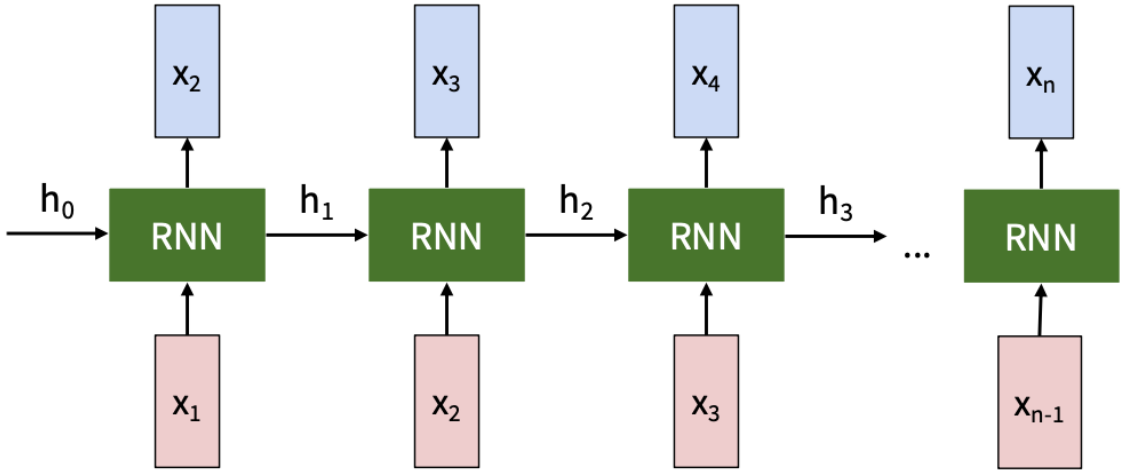
Autoregressive Models

- **PixelRNN [van der Oord et al. 2016]**

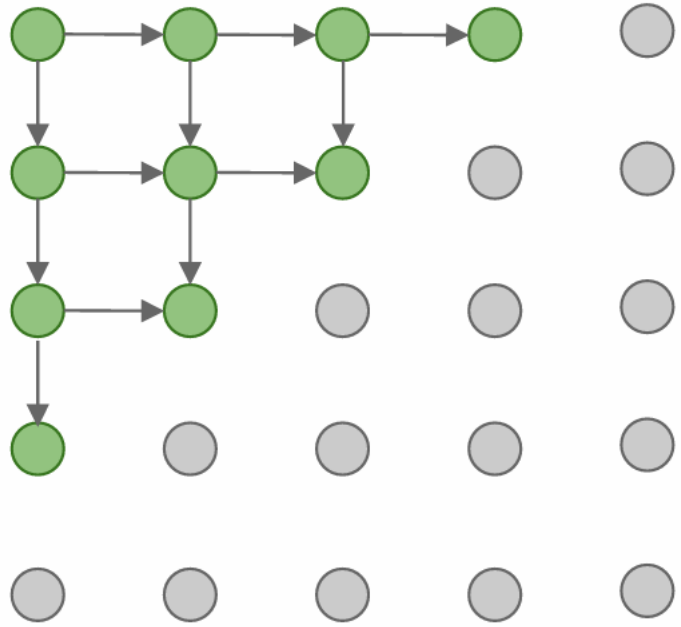
Generate image pixels starting from corner

Dependency on previous pixels modeled using an RNN (LSTM)

Sequential generation is slow in both training and inference!



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



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

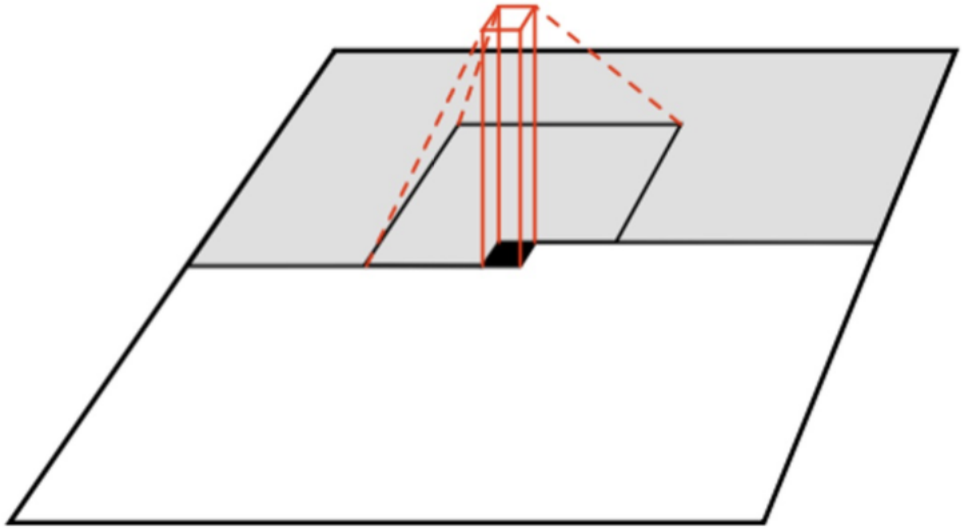
Autoregressive Models

- **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)



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Autoregressive Models

- **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

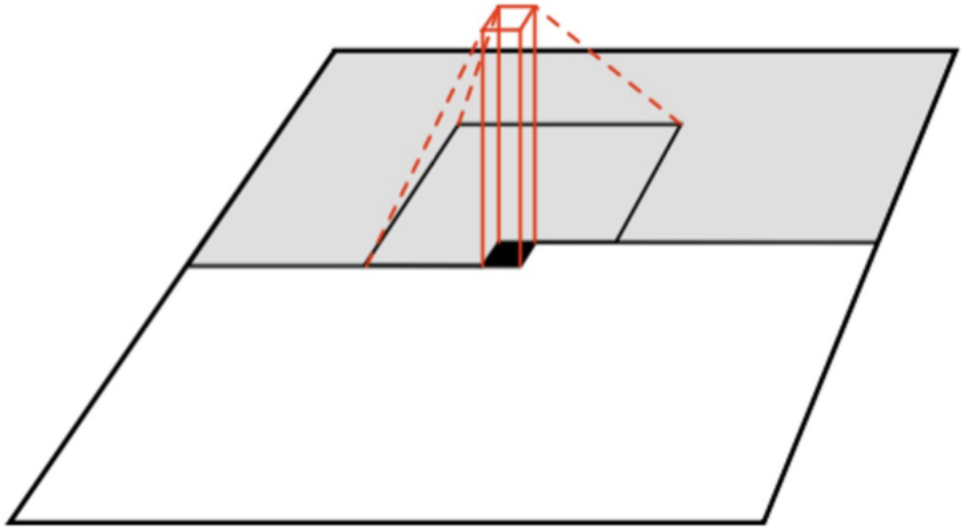
(masked convolution)

Training is faster than PixelRNN

(can parallelize convolutions since context region values known from training images)

Generation is still slow:

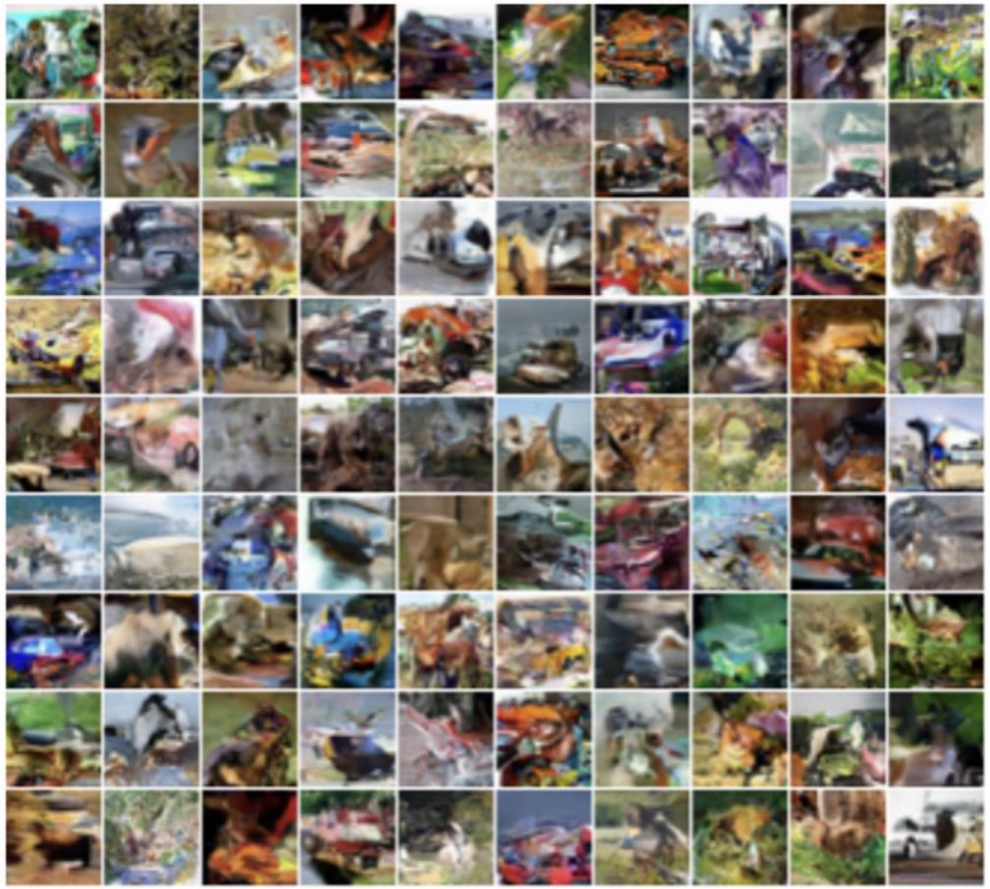
For a 32x32 image, we need to do forward passes of the network 1024 times for a single image



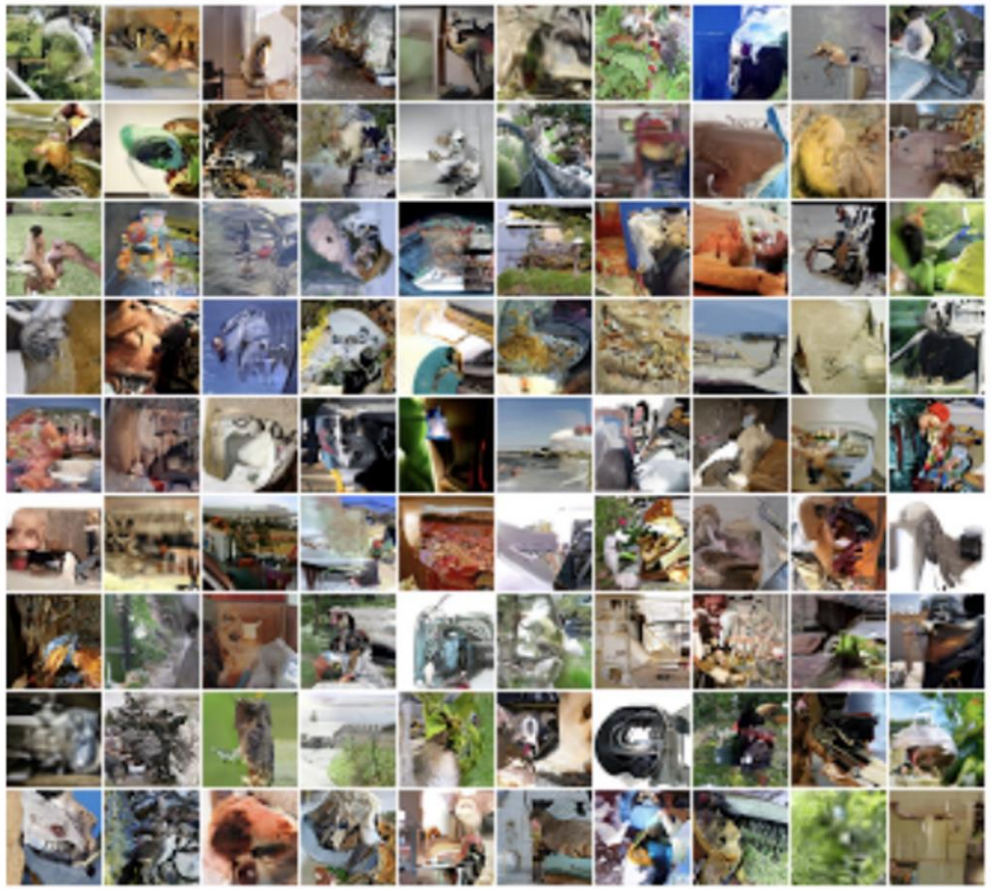
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Autoregressive Models

- **Generation Samples**



32x32 CIFAR-10



32x32 ImageNet

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Autoregressive Models

- **PixelRNN and PixelCNN**

Pros:

- Can explicitly compute likelihood p(x)
- Easy to optimize
- Good samples

Con:

- Sequential generation => slow

Improving PixelCNN performance

- Gated convolutional layers
- Short-cut connections
- Discretized logistic loss
- Multi-scale
- Training tricks
- Etc…

See

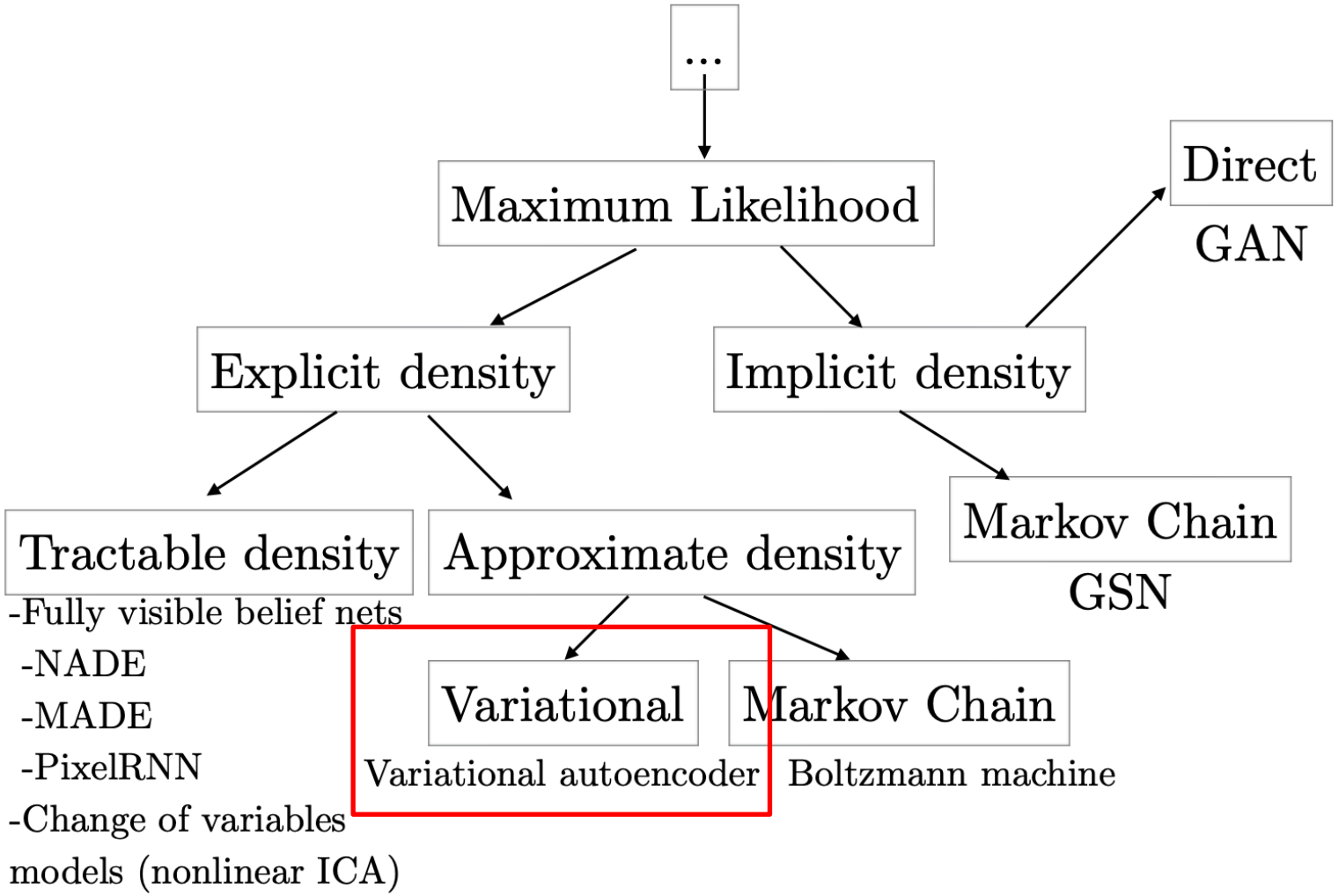
- Van der Oord et al. NIPS 2016
- Salimans et al. 2017 (PixelCNN++)

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Contents

- Last Class Review
- Supervised vs Unsupervised Learning
- **Generative Model**
  - PixelRNN & PixelCNN
  - **Variational Autoencoder (VAE)**
  - Generative Adversarial Networks (GAN)

Taxonomy of Generative Models



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

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

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

- 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:

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

- **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!

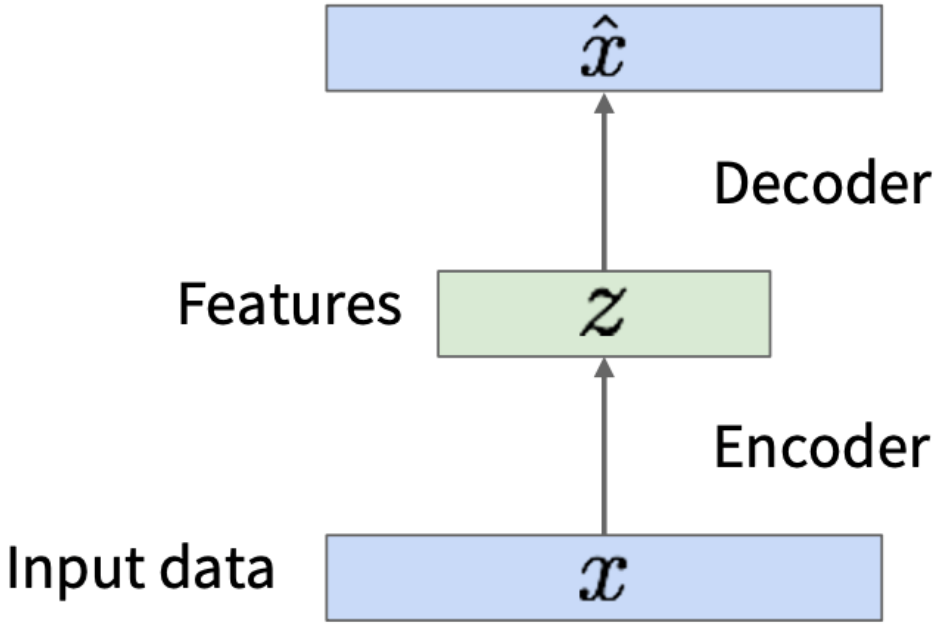
Cannot optimize directly, derive and optimize lower bound on likelihood instead

Why latent z?

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

- **Selfsupervised approach for learning a lower-dimensional feature representation from unlabeled training data.**



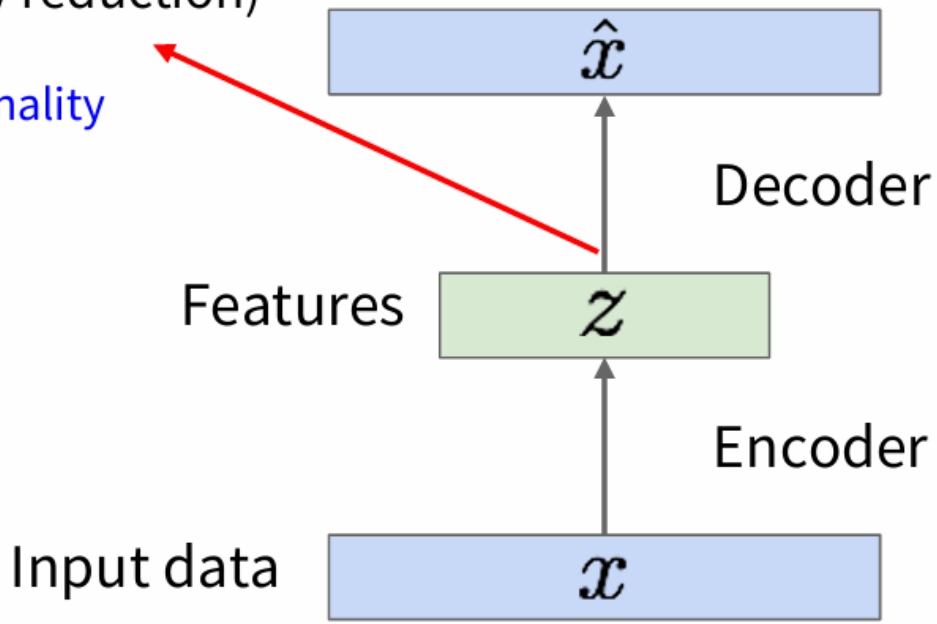
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

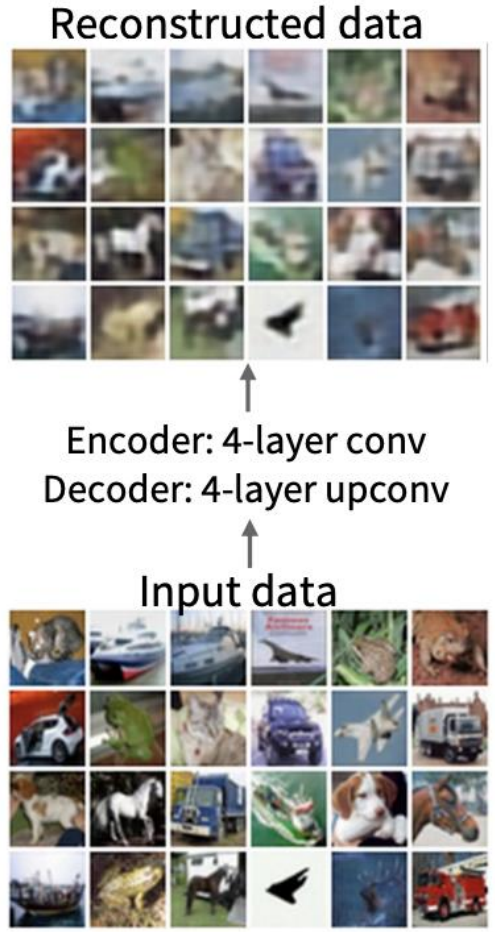
Variational Autoencoder (VAE)

- **Selfsupervised approach for learning a lower-dimensional feature representation from unlabeled training data.**

z usually smaller than x
(dimensionality reduction)

Q: Why dimensionality reduction?

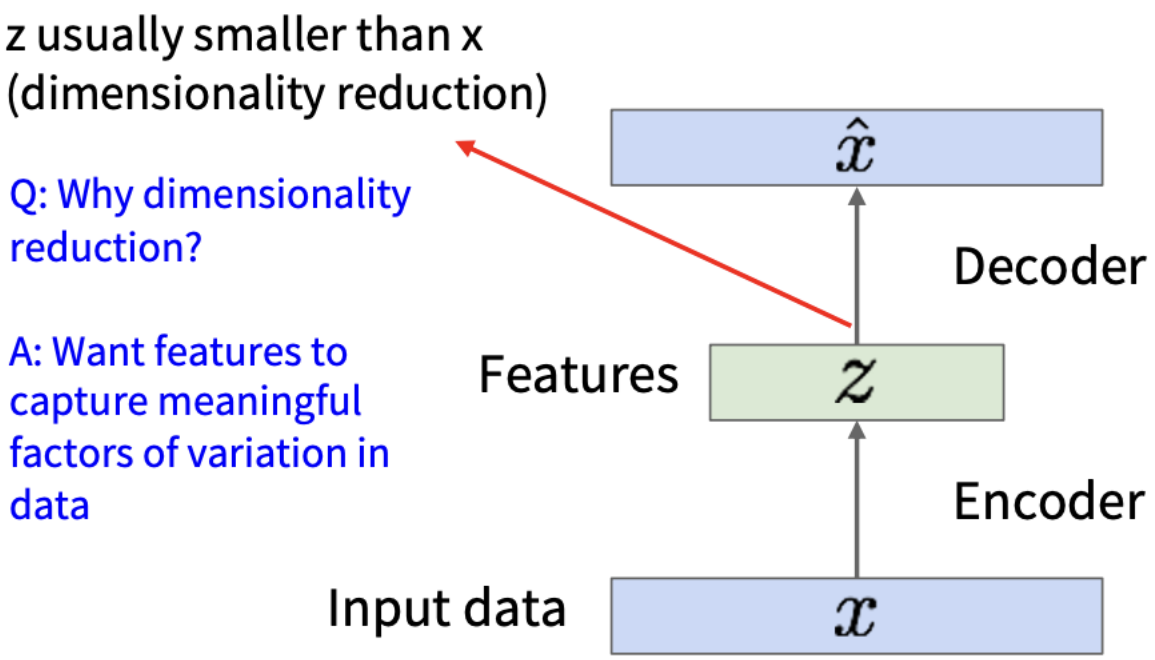




slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

- **Selfsupervised approach for learning a lower-dimensional feature representation from unlabeled training data.**



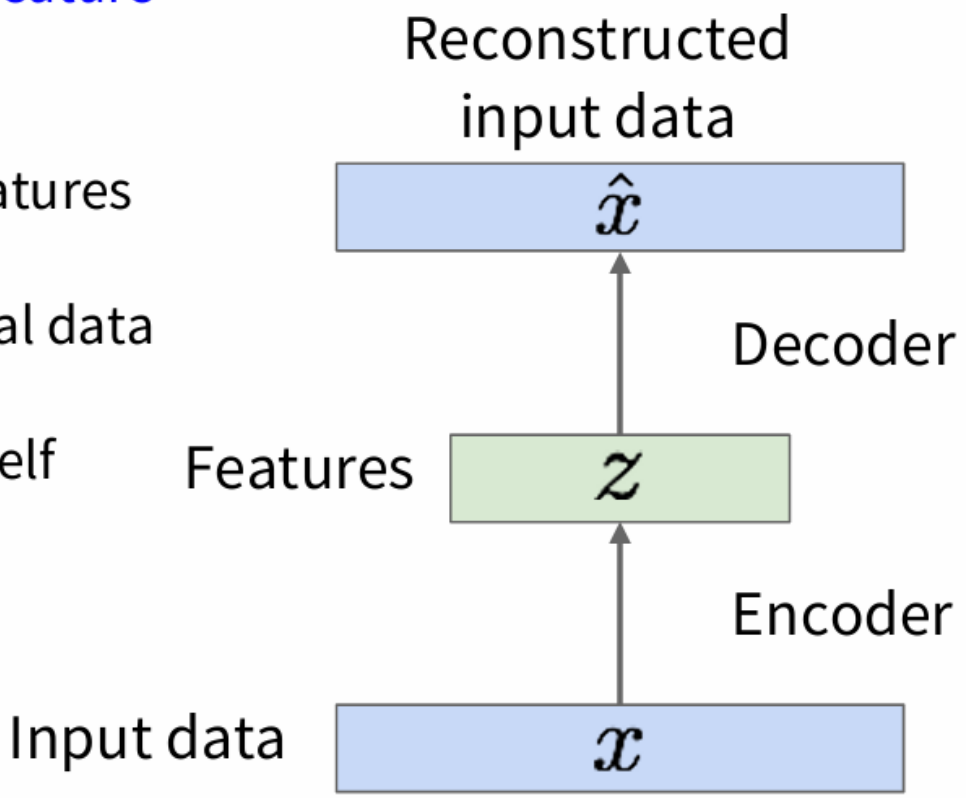
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

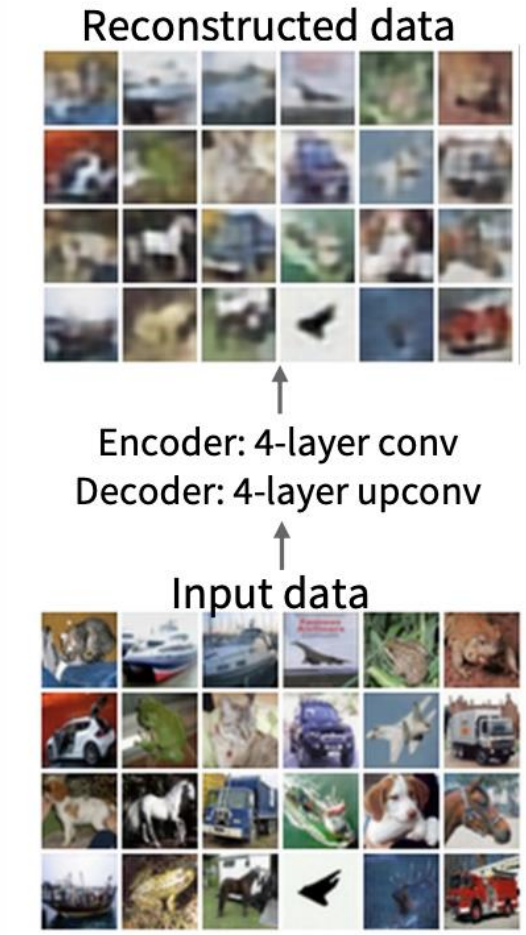
Variational Autoencoder (VAE)

- **Selfsupervised approach for learning a lower-dimensional feature representation from unlabeled training data.**

How to learn this feature representation?

Train such that features can be used to reconstruct original data
"Autoencoding" - encoding input itself





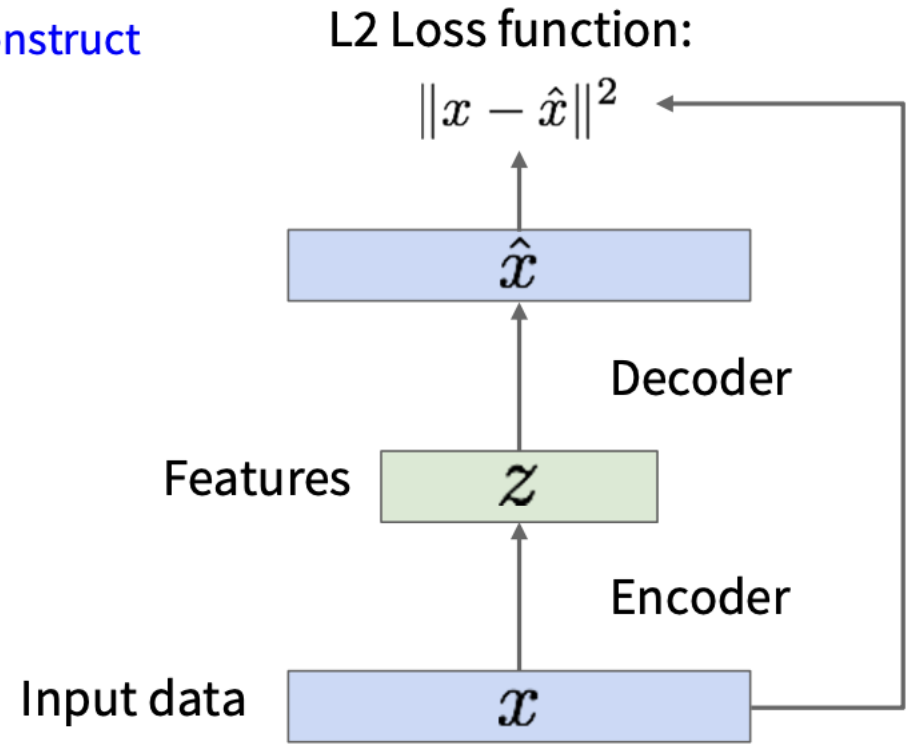
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

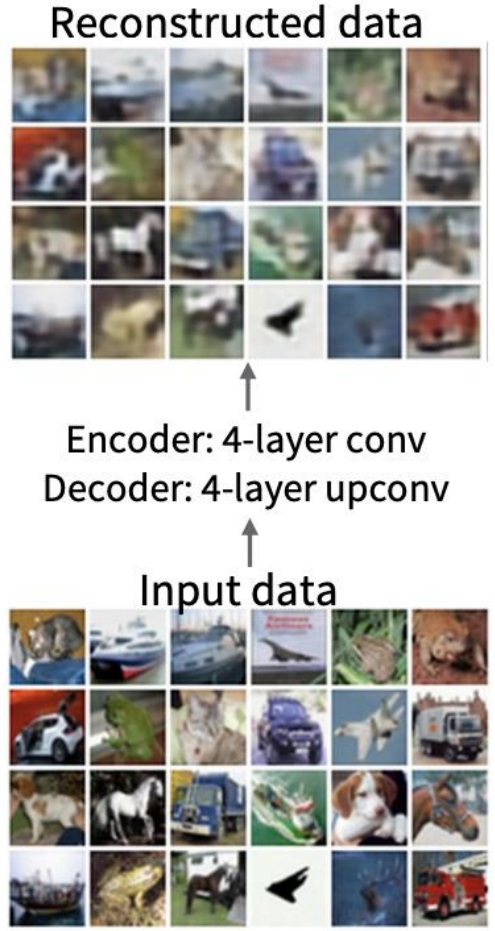
Variational Autoencoder (VAE)

- **Train such that features can be used to reconstruct original data**

Train such that features can be used to reconstruct original data

Doesn't use labels!



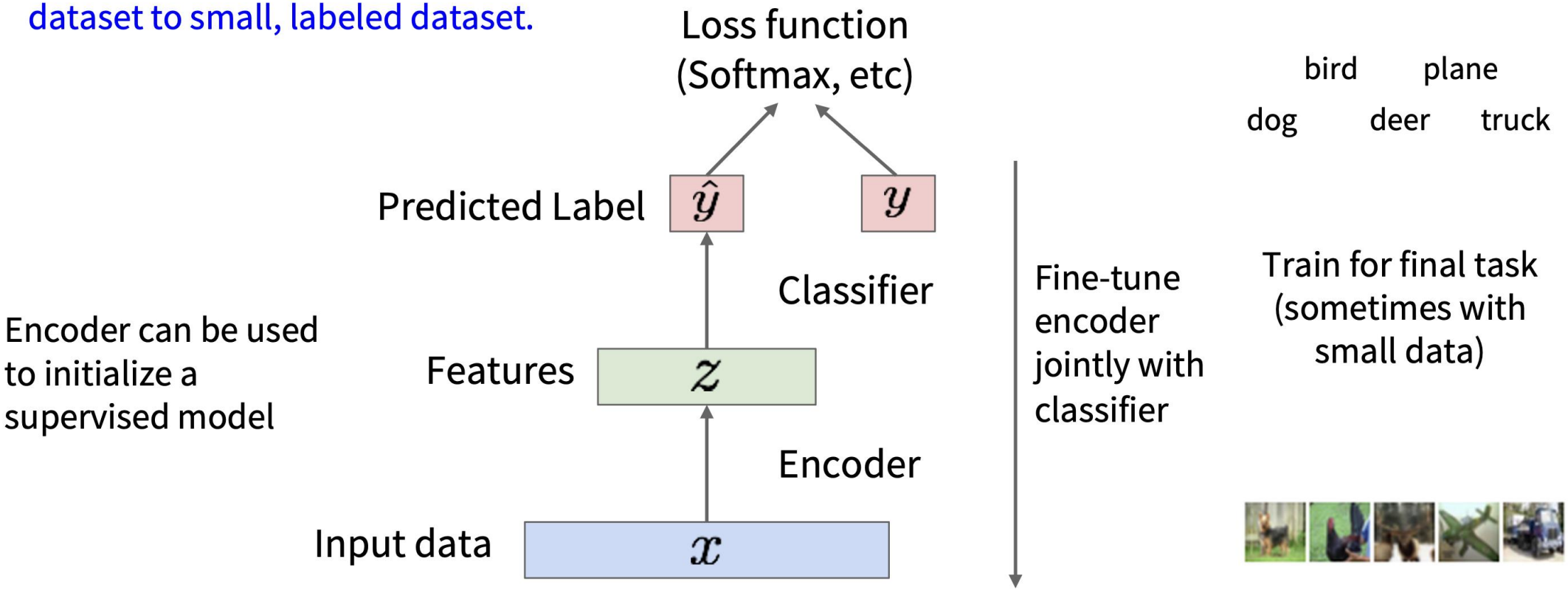


slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

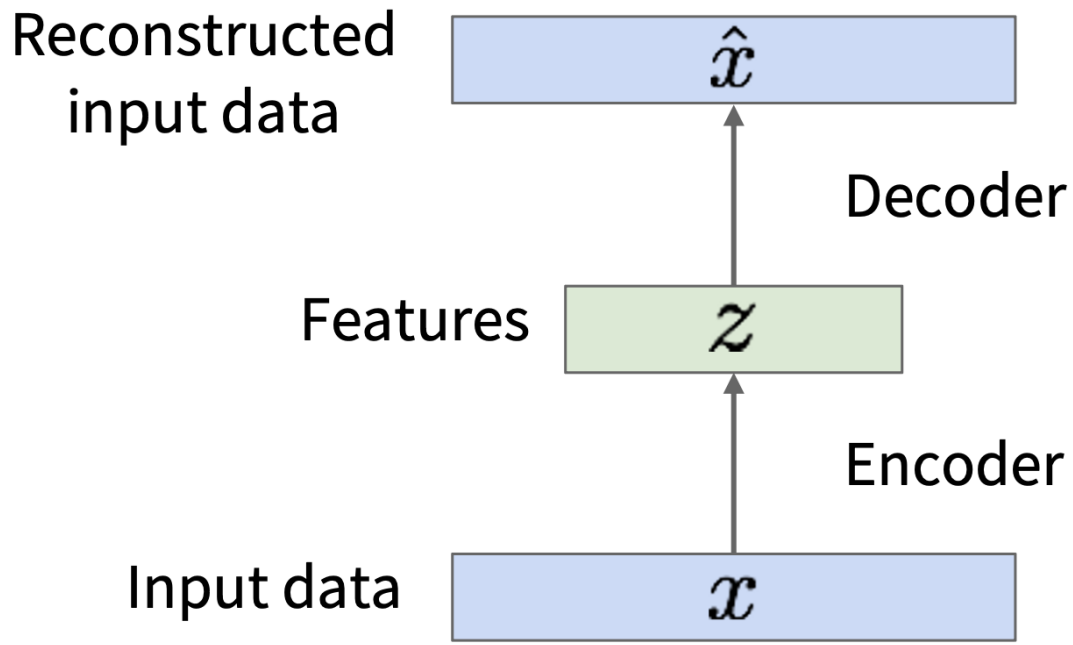
- **Train such that features can be used to align with the prior distribution of features.**

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



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

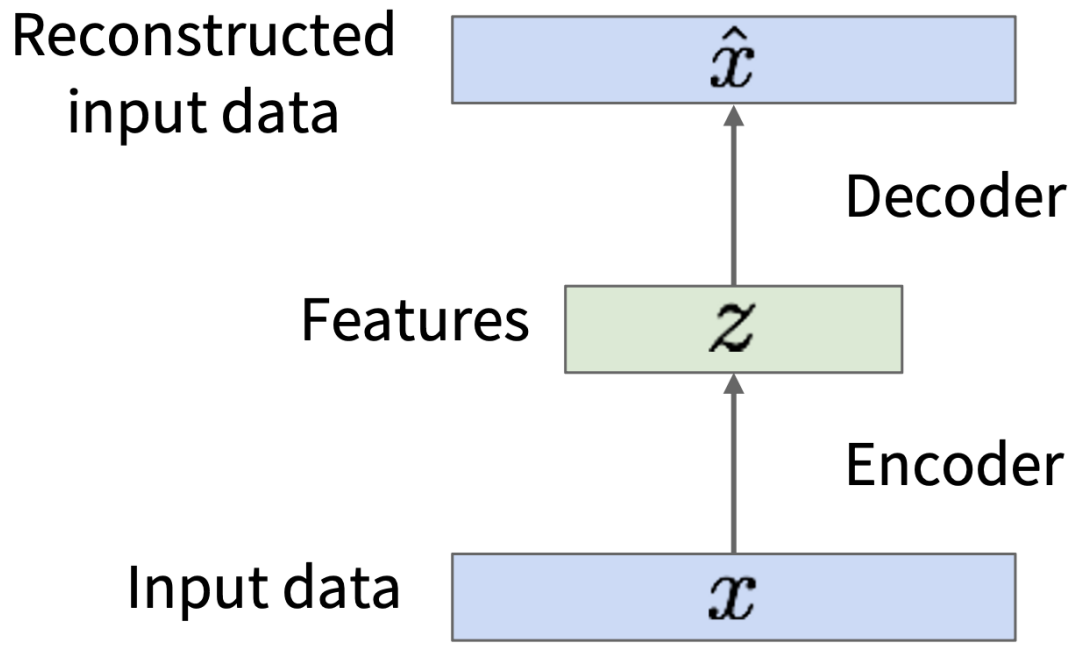


Autoencoders can reconstruct data, and can learn features to initialize a supervised model

Features capture factors of variation in training data.

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)



Autoencoders can reconstruct data, and can learn features to initialize a supervised model

Features capture factors of variation in training data.

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?

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

Probabilistic spin on autoencoders - will let us sample from the model to generate data!

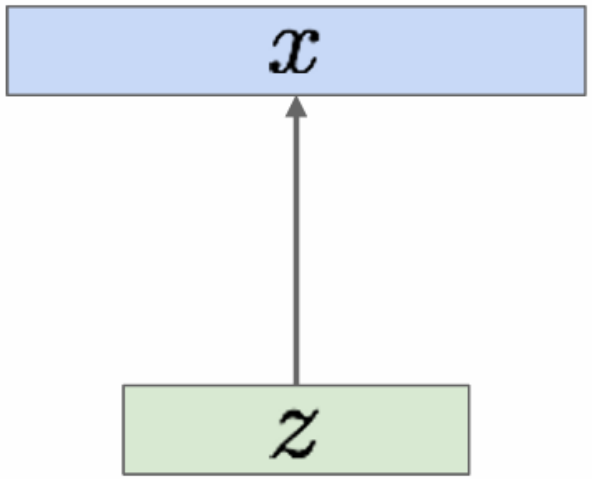
Assume training data $\{x^{(i)}\}_{i=1}^{N}$ is generated from the distribution of unobserved (latent) representation z

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

$z^{(i)} \sim p_{\theta^*}(z)$



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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

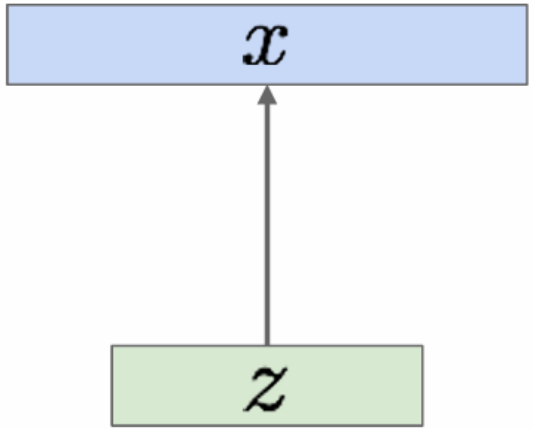
Variational Autoencoder (VAE)

Probabilistic spin on autoencoders - will let us sample from the model to generate data!

Assume training data $\{x^{(i)}\}_{i=1}^{N}$ is generated from the distribution of unobserved (latent) representation z

Sample from true conditional $p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior $z^{(i)} \sim p_{\theta^*}(z)$



Intuition (remember from autoencoders!): x is an image, z is latent factors used to generate x: attributes, orientation, etc.

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

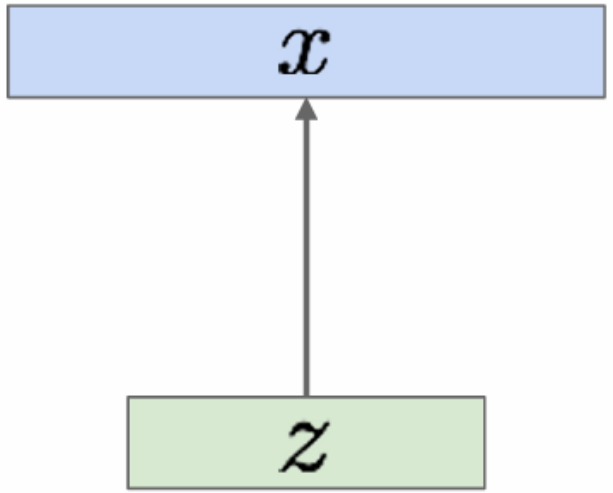
We want to estimate the true parameters θ^* of this generative model given training data x.

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

$z^{(i)} \sim p_{\theta^*}(z)$



How should we represent this model?

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

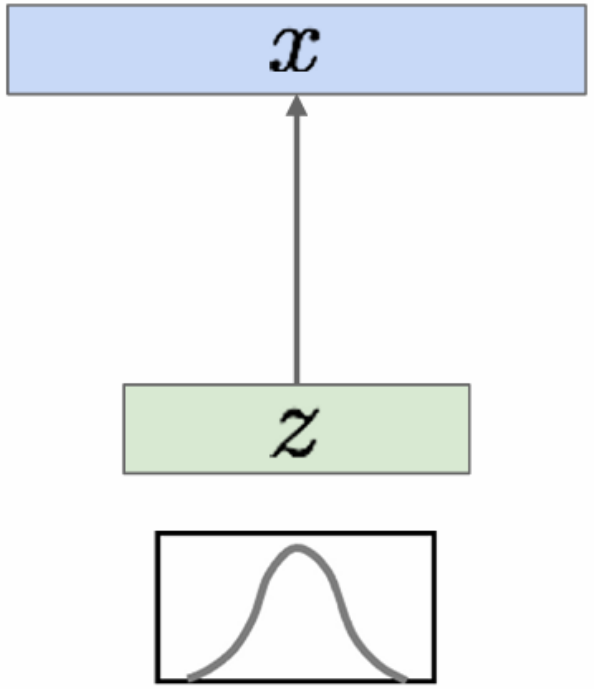
We want to estimate the true parameters θ^* of this generative model given training data x.

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

$z^{(i)} \sim p_{\theta^*}(z)$



How should we represent this model?

Choose prior p(z) to be simple, e.g. Gaussian. Reasonable for latent attributes, e.g. pose, how much smile.

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

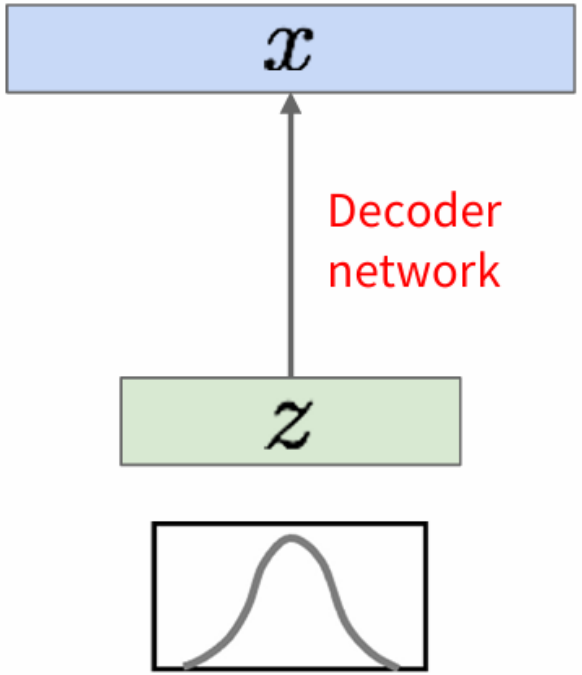
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

We want to estimate the true parameters θ^* of this generative model given training data x.

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$



Sample from true prior

$z^{(i)} \sim p_{\theta^*}(z)$

How should we represent this model?

Choose prior p(z) to be simple, e.g. Gaussian. Reasonable for latent attributes, e.g. pose, how much smile.

Conditional p(x|z) is complex (generates image) => represent with neural network

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

We want to estimate the true parameters θ^* of this generative model given training data x.

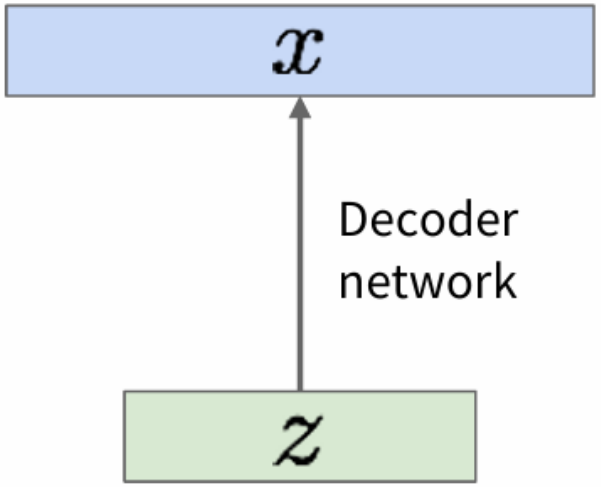
How to train the model?

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

$z^{(i)} \sim p_{\theta^*}(z)$



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

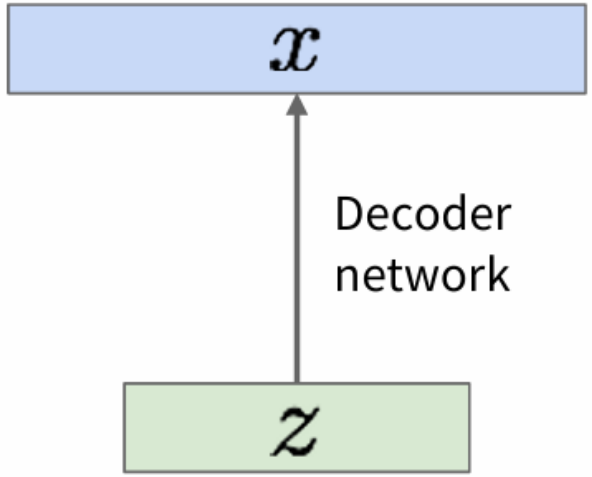
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

We want to estimate the true parameters θ^* of this generative model given training data x.

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$



Sample from true prior

$z^{(i)} \sim p_{\theta^*}(z)$

How to train the model?

Learn model parameters to maximize likelihood of training data

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

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

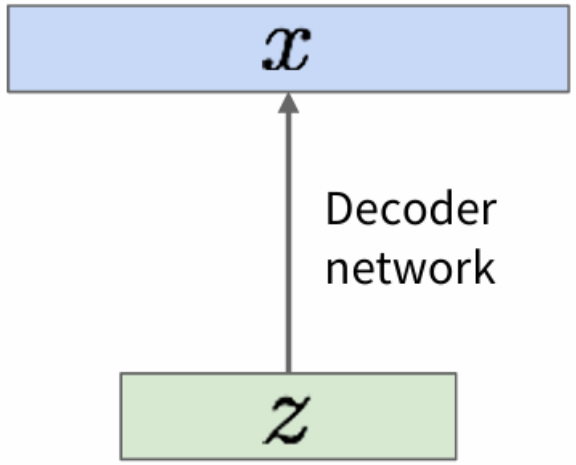
Variational Autoencoder (VAE)

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

$z^{(i)} \sim p_{\theta^*}(z)$



We want to estimate the true parameters θ^* of this generative model given training data x.

How to train the model?

Learn model parameters to maximize likelihood of training data

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

Q: What is the problem with this?

Intractable!

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

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

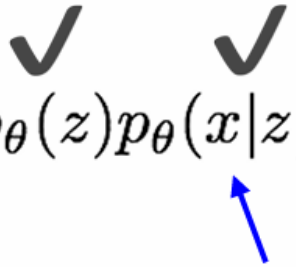
✓ $p_\theta(z)$: Simple Gaussian prior

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

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



Decoder neural network

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

😢 ✓ ✓

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

Intractable to compute p(x|z) for every z!

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

😢 ✓ ✓

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

Intractable to compute p(x|z) for every z!

$$\log p(x) \approx \log \frac{1}{k} \sum_{i=1}^{k} p(x|z^{(i)}), \text{where } z^{(i)} \sim p(z)$$

Monte Carlo estimation is too high variance

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

😢 ✓ ✓

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

Posterior density: $p_\theta(z|x) = p_\theta(x|z) p_\theta(z) / p_\theta(x)$

Intractable data likelihood

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

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

Posterior density also intractable: $p_\theta(z|x) = p_\theta(x|z)p_\theta(z)/p_\theta(x)$

Solution: In addition to modeling $p_\theta(x|z)$, learn $q_\phi(z|x)$ that approximates the true posterior $p_\theta(z|x)$.

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))$$

The expectation wrt. z (using encoder network) let us write nice KL terms

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))$$

Decoder network gives $p_\theta(x|z)$, can compute estimate of this term through sampling (need some trick to differentiate through sampling).

This KL term (between Gaussians for encoder and z prior) has nice closed-form solution!

$p_\theta(z|x)$ intractable (saw earlier), can't compute this KL term :(But we know KL divergence always >= 0.

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

We want to maximize the data likelihood

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}$$

Tractable lower bound which we can take gradient of and optimize! ($p_\theta(x|z)$ differentiable, KL term differentiable)

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}$$

Decoder: reconstruct the input data

Encoder: make approximate posterior distribution close to prior

Tractable lower bound which we can take gradient of and optimize! ($p_\theta(x|z)$ differentiable, KL term differentiable)

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

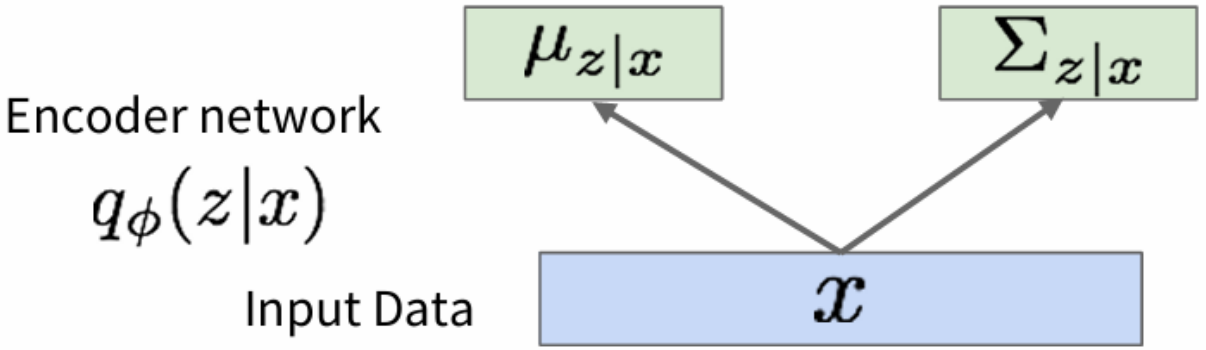
Let's look at computing the KL divergence between the estimated posterior and the prior given some data

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

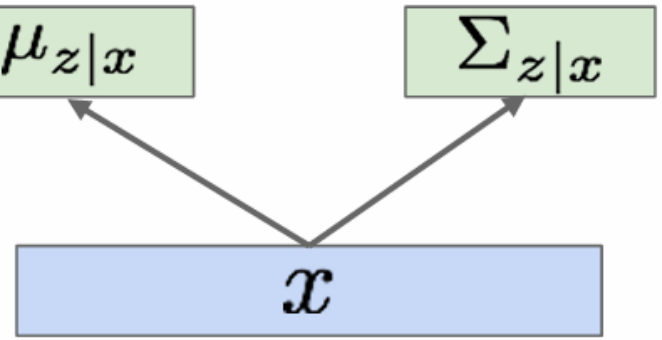
$$D_{KL}(\mathcal{N}(\mu_{z|x}, \Sigma_{z|x})||\mathcal{N}(0, I))$$

Have analytical solution

Make approximate posterior distribution close to prior

Encoder network

$q_\phi(z|x)$



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

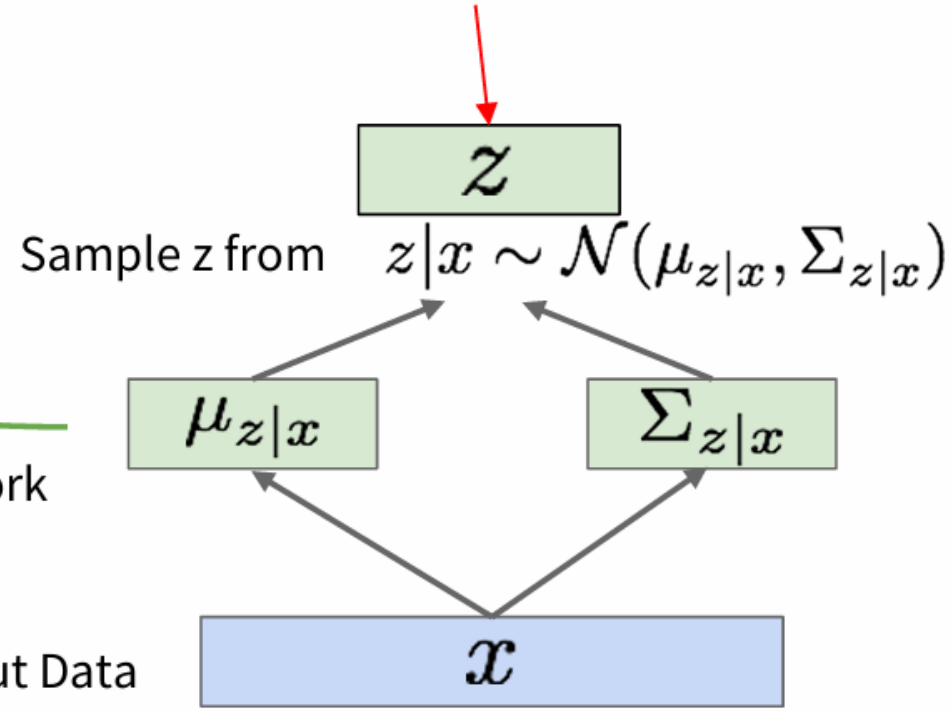
Variational Autoencoder (VAE)

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Not part of the computation graph!

Make approximate posterior distribution close to prior



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

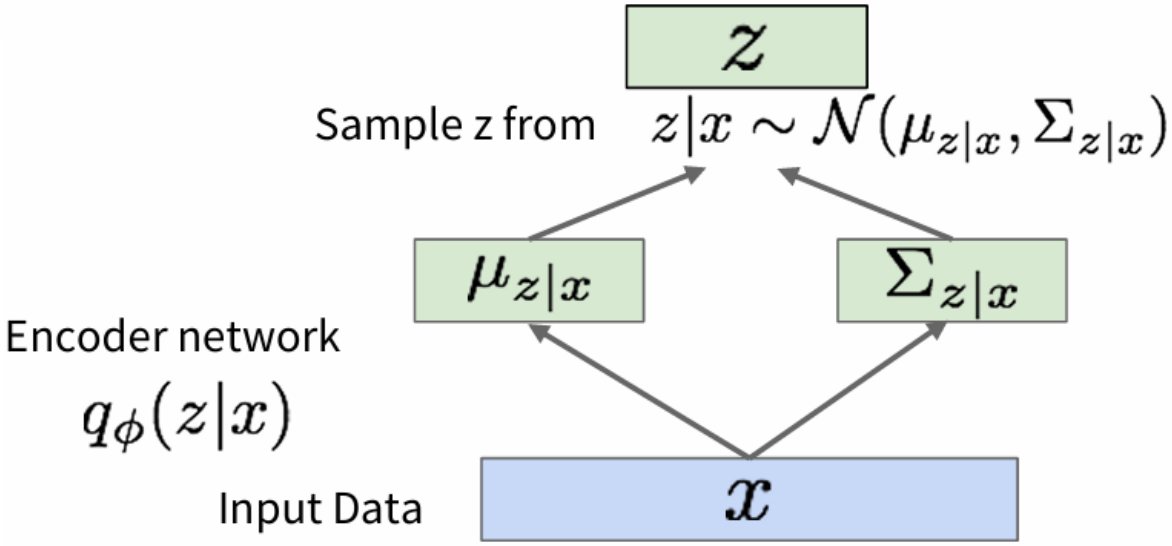
Reparameterization trick to make sampling differentiable:

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

$z = \mu_{z|x} + \epsilon \sigma_{z|x}$

Input to the graph

Part of computation graph

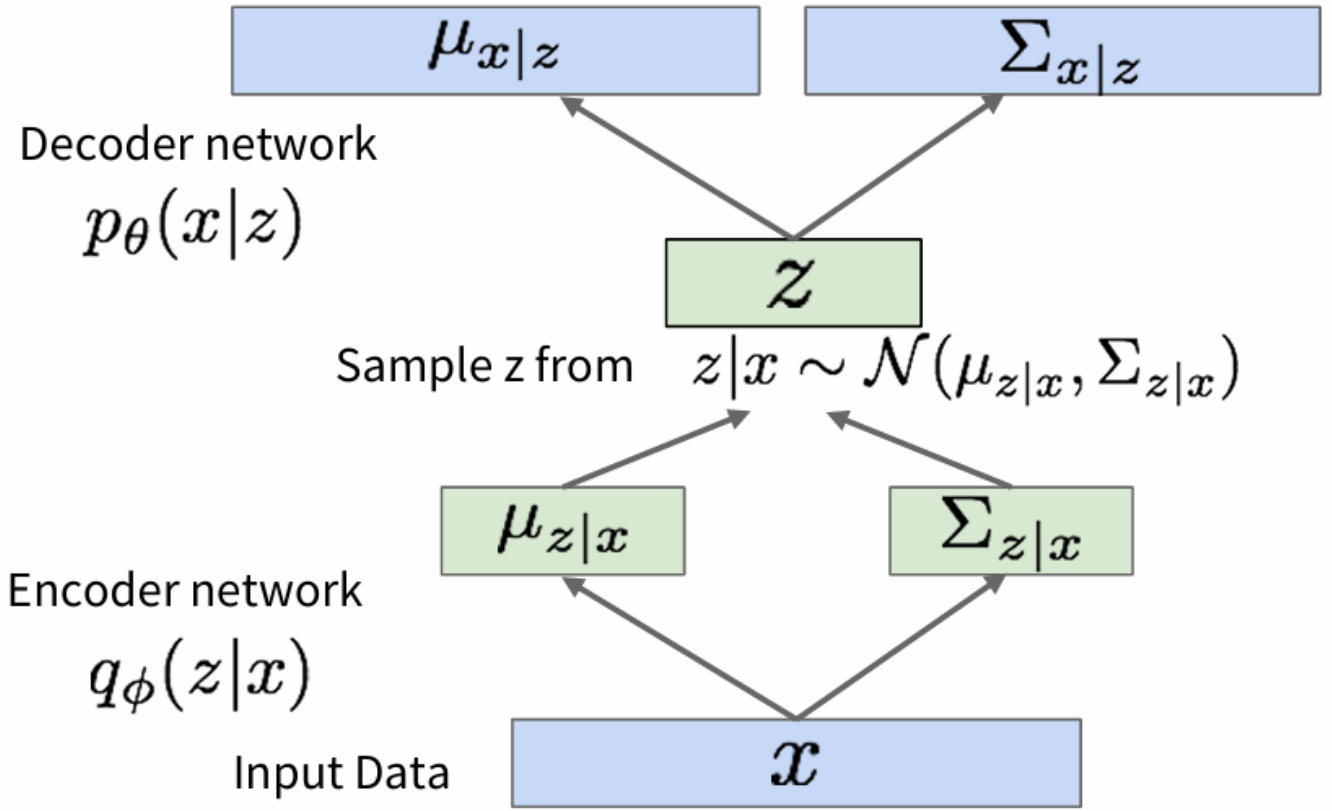


slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$



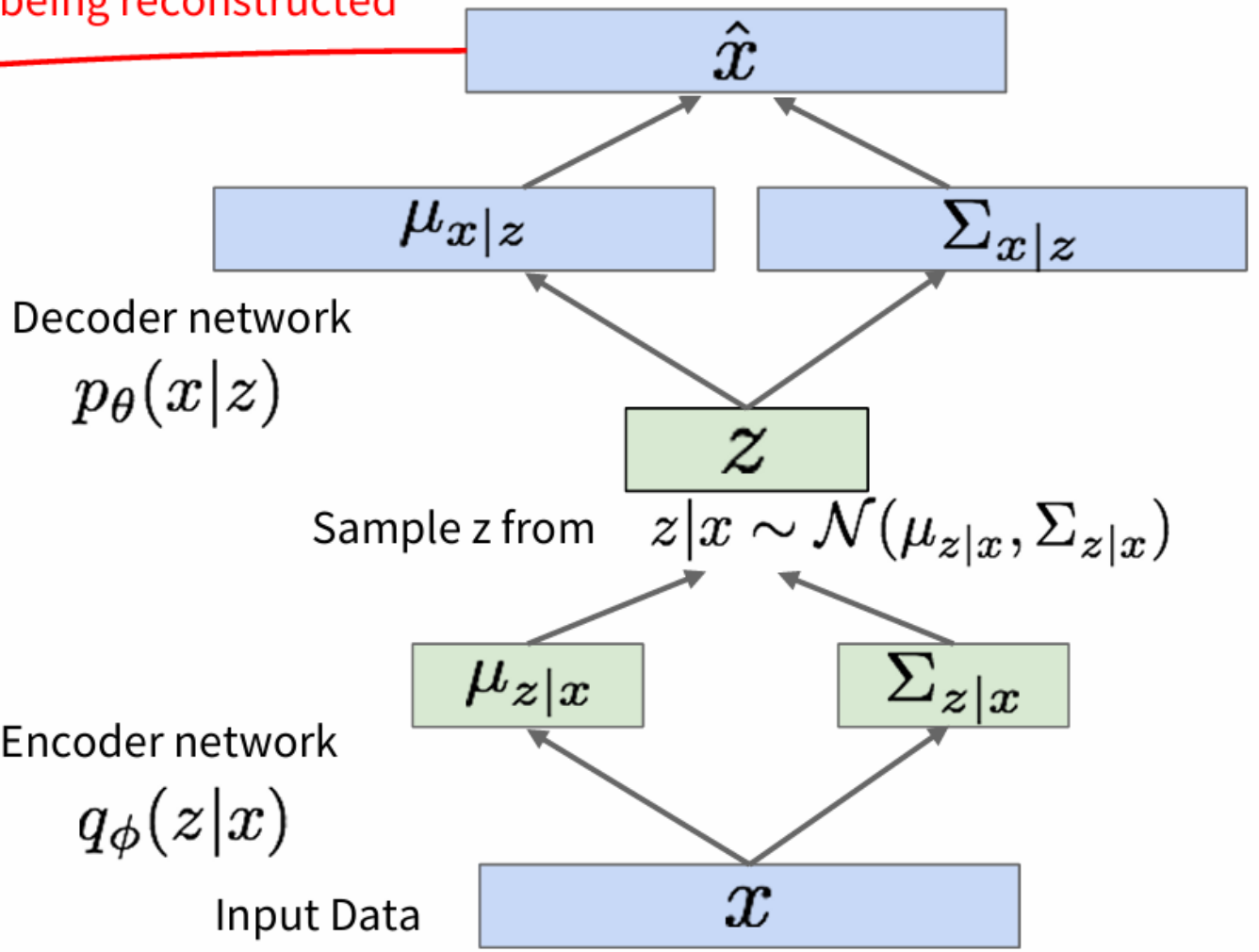
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

Putting it all together: maximizing the likelihood lower bound

Maximize likelihood of original input being reconstructed

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$



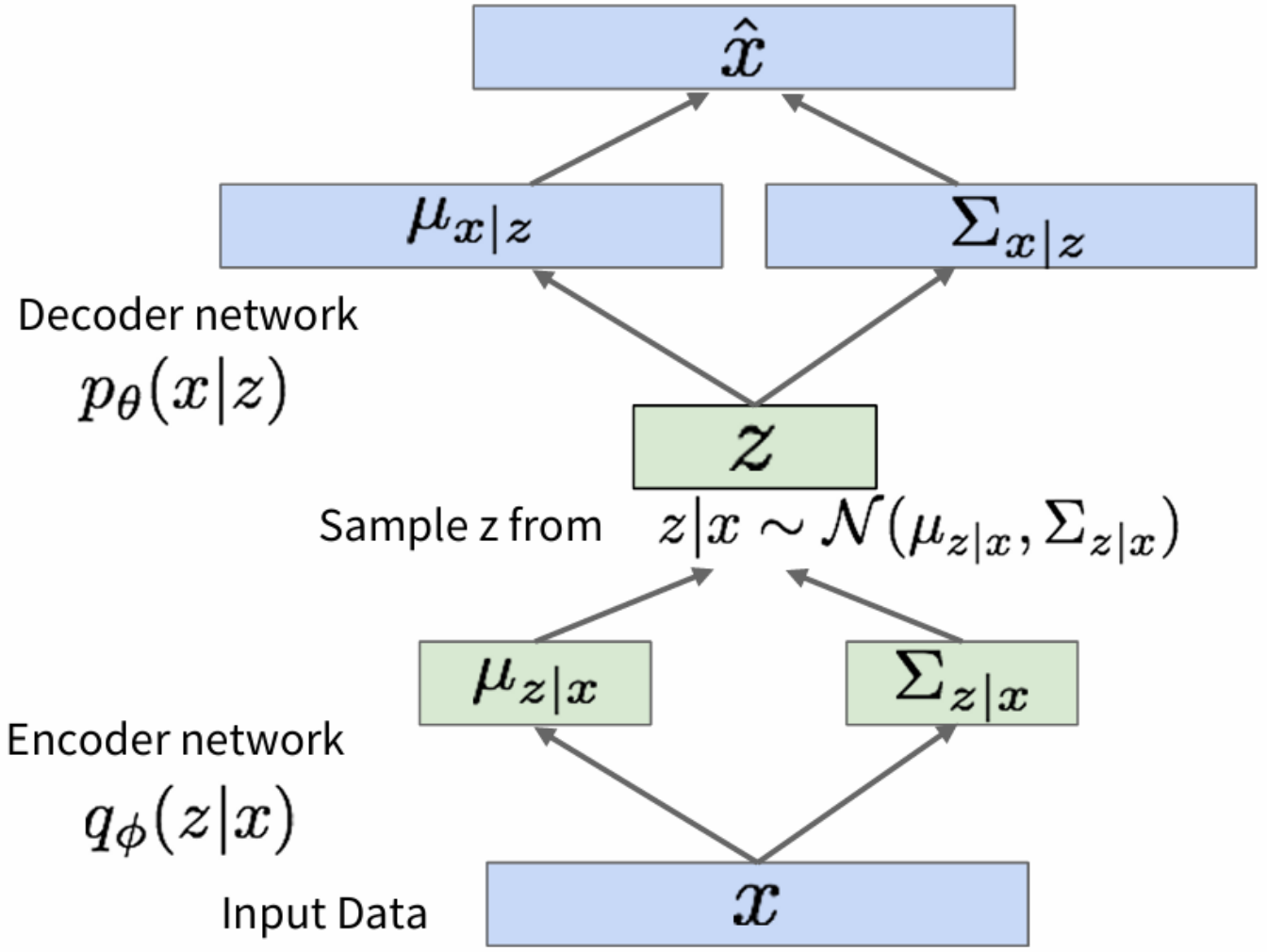
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

For every minibatch of input data: compute this forward pass, and then backprop!



slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

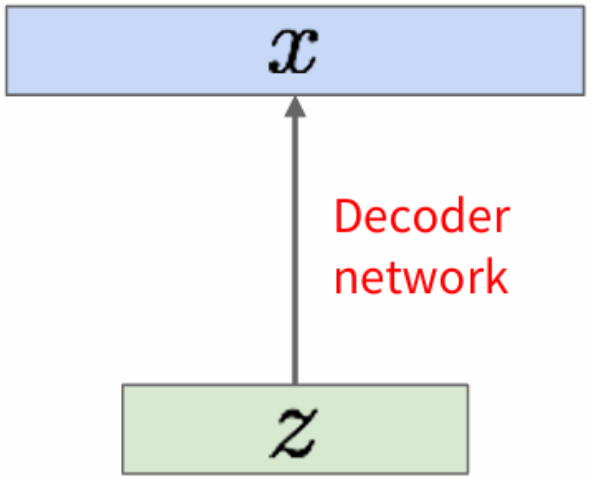
Our assumption about data generation process

Sample from true conditional

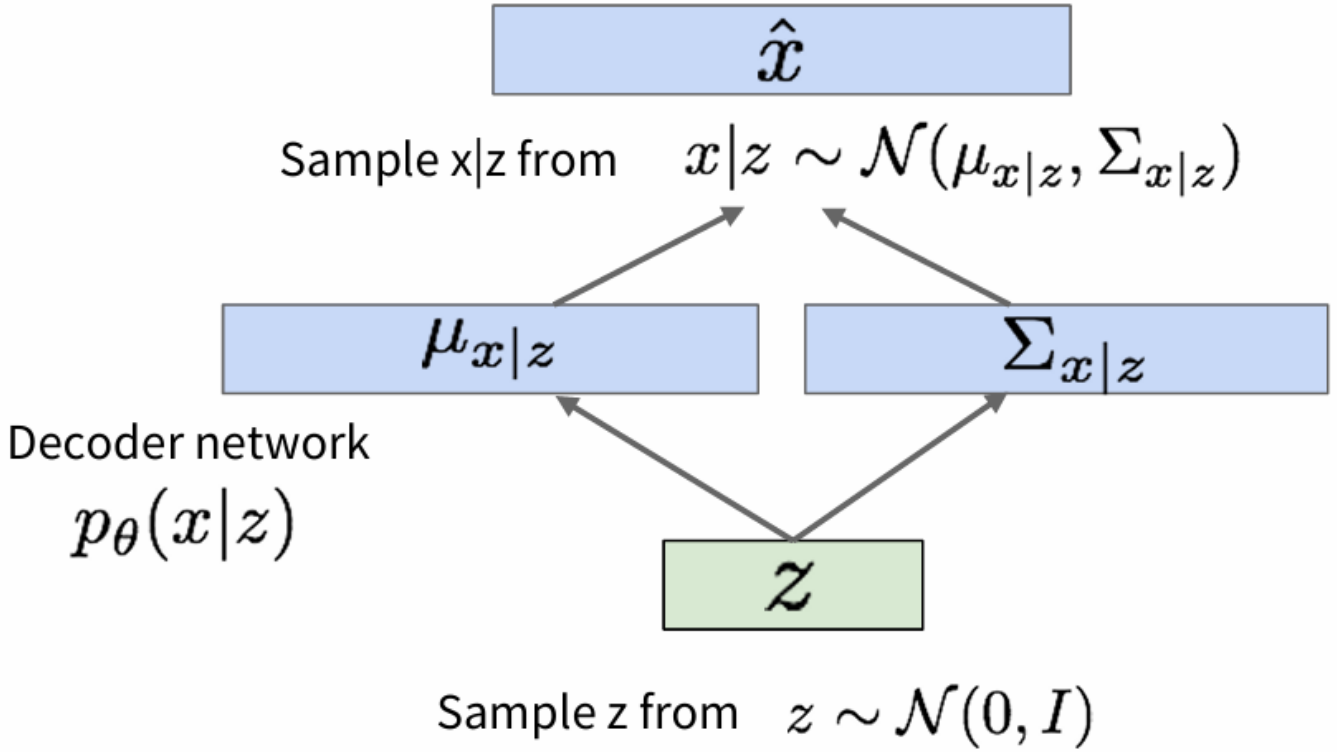
$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

$z^{(i)} \sim p_{\theta^*}(z)$



Now given a trained VAE:
use decoder network & sample z from prior!

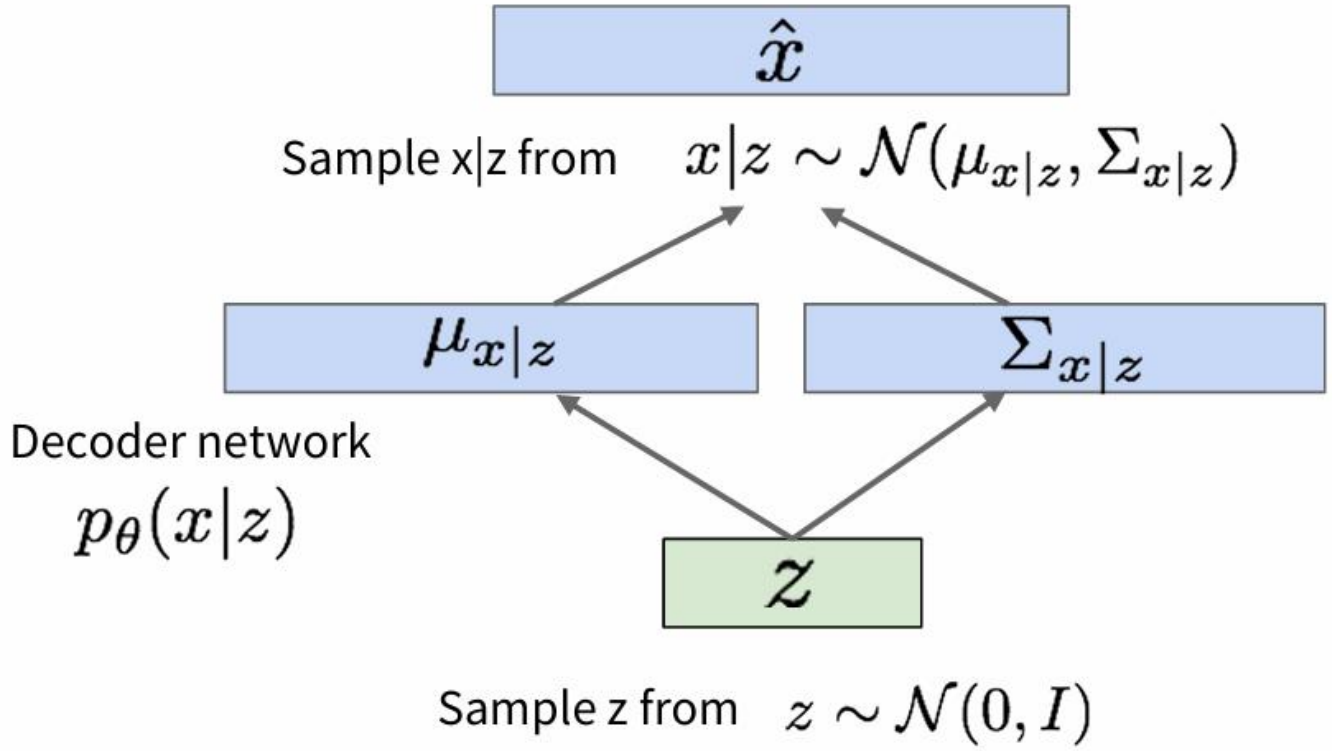


Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

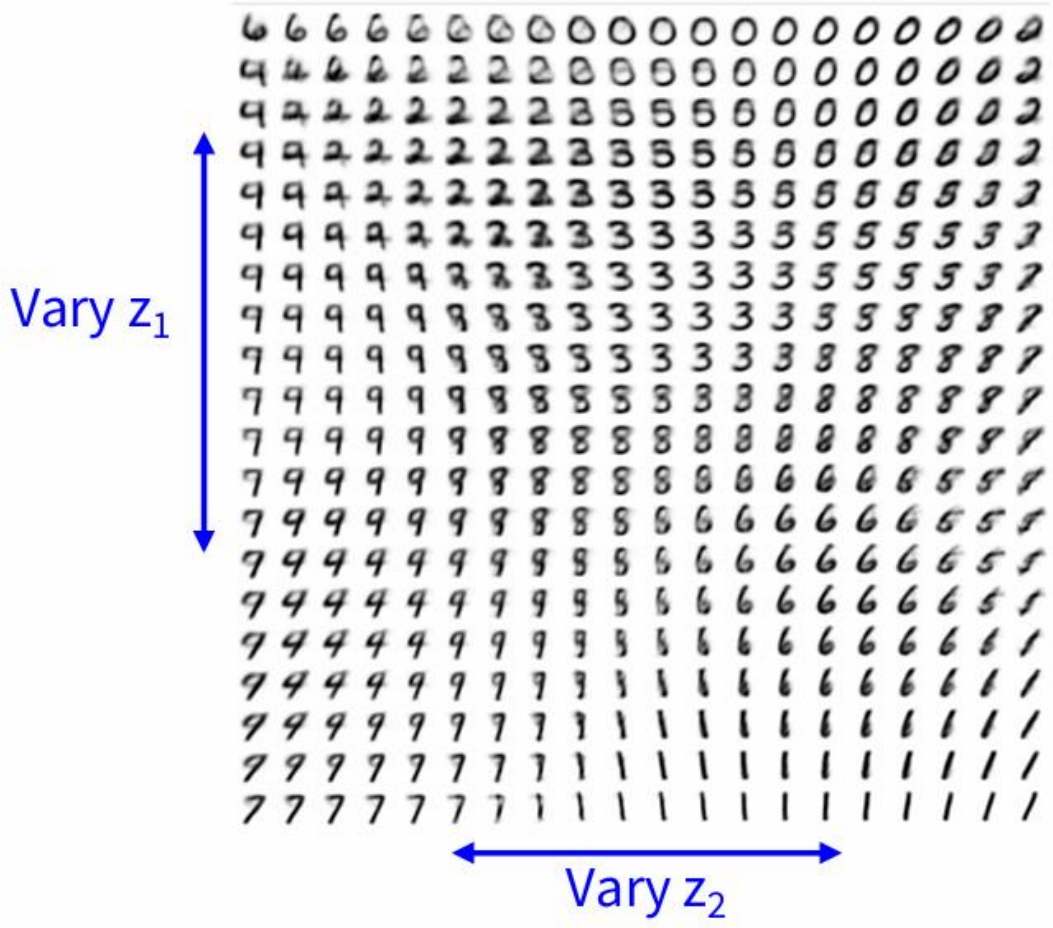
Variational Autoencoder (VAE)

Use decoder network. Now sample z from prior!



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

Data manifold for 2-d z



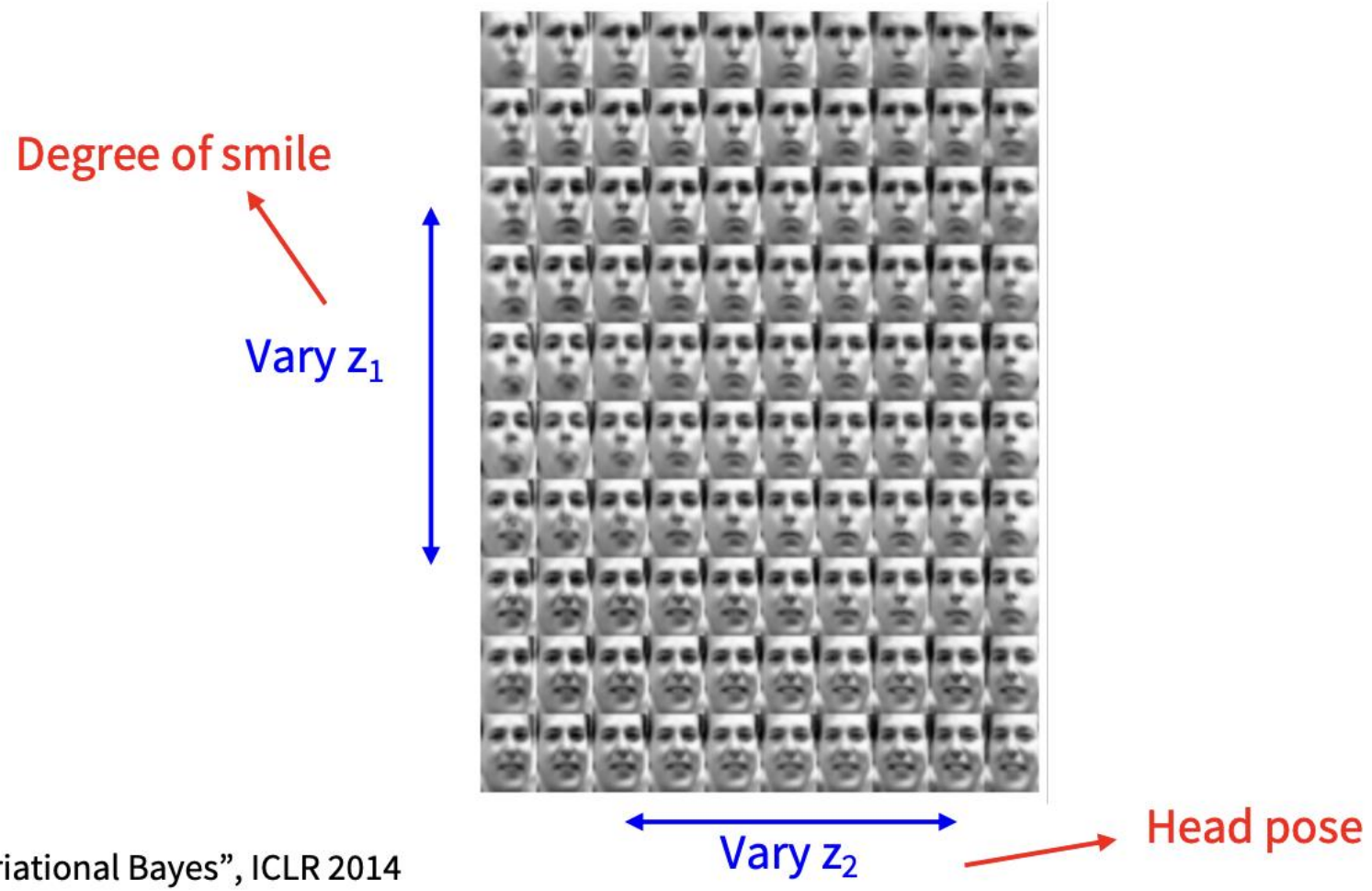
slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

- **Generating Data!**

Diagonal prior on z
=> independent
latent variables

Different dimensions
of z encode
interpretable factors
of variation



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

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

- **Generating Data!**

Diagonal prior on z
=> independent
latent variables

Different dimensions
of z encode
interpretable factors
of variation

Also good feature representation that can be computed using $q_\phi(z|x)$!

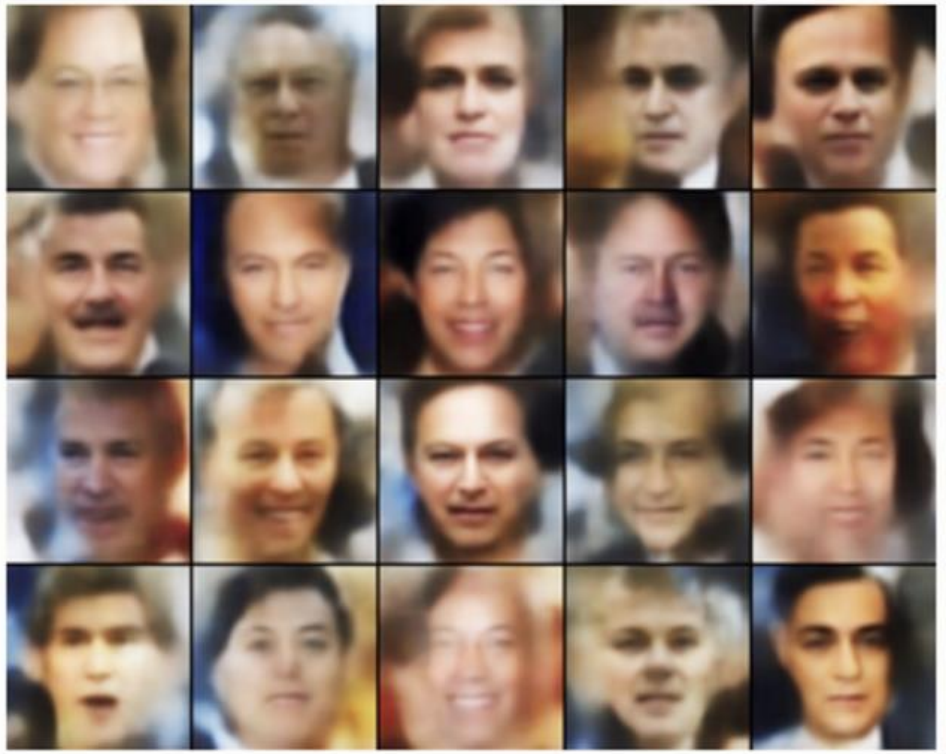
Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Variational Autoencoder (VAE)

- **Generating Data!**

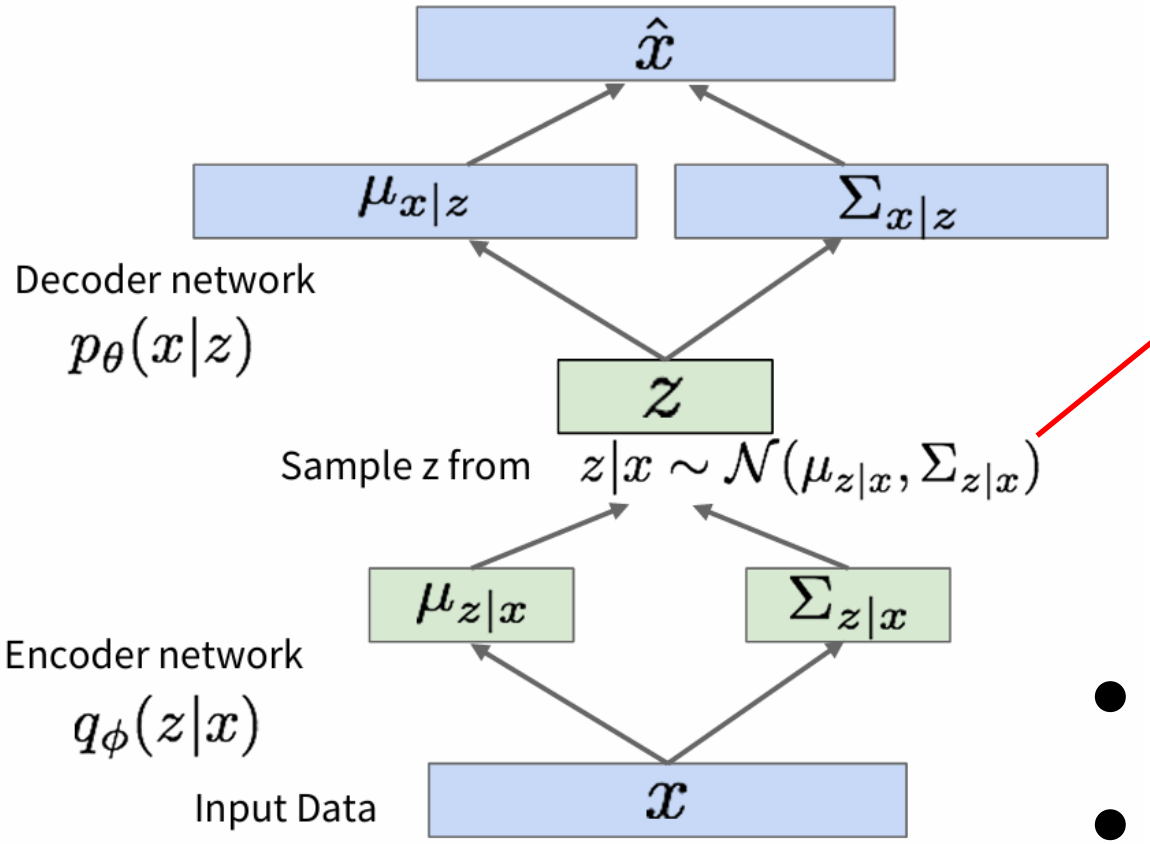
32x32 CIFAR-10



Labeled Faces in the Wild

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

VQ-VAE



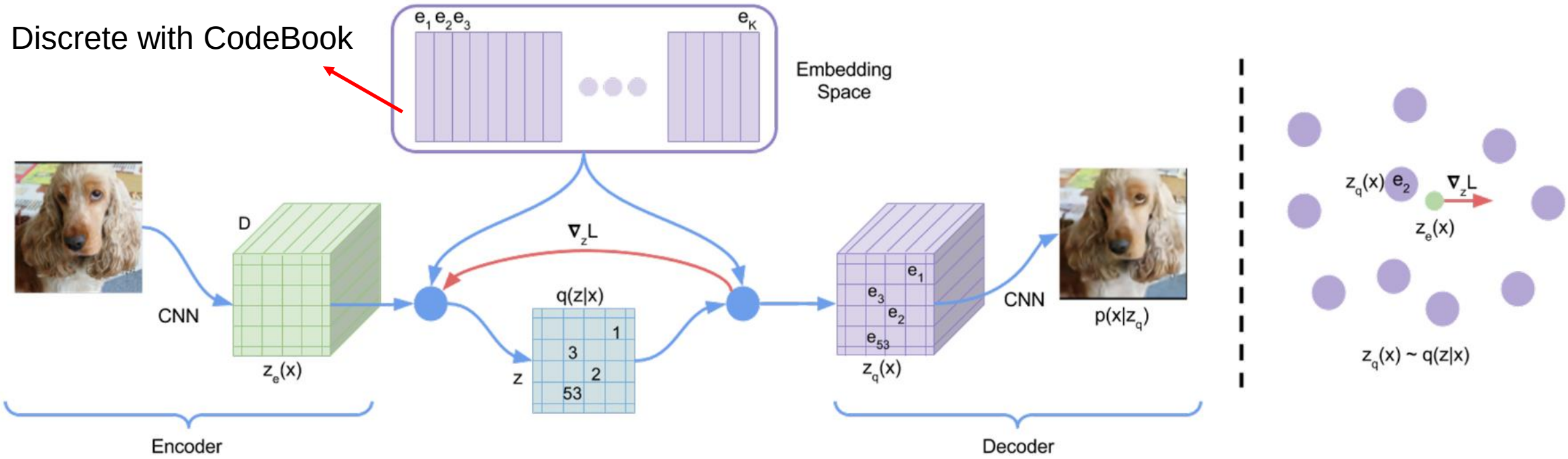
Gaussian Distribution

- Continuous Latent Space
- Blurred Images and lower quality

Vector Quantized

- Discrete latent space, alleviate posterior collapse
- Easier to interpret latent variables

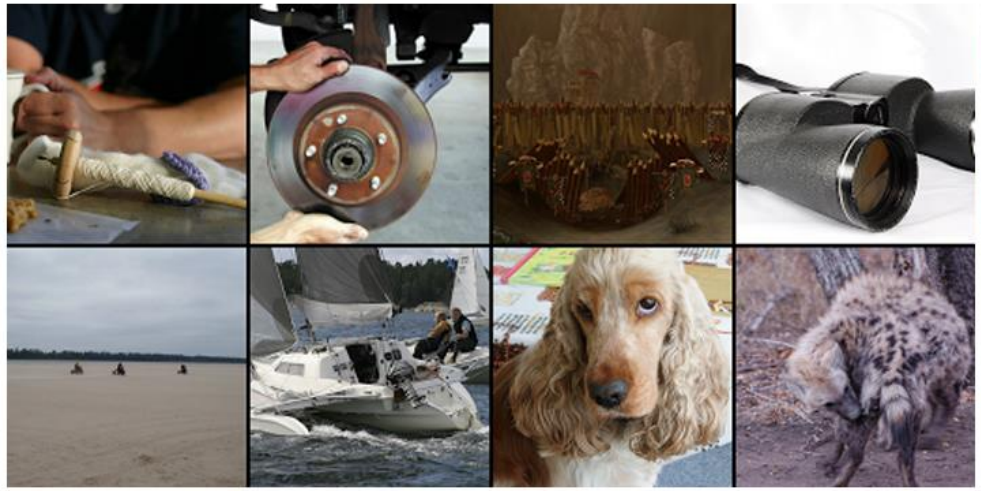
VQ-VAE



VQ-VAE

Aaron van den Oord et al. NIPS 2017

VQ-VAE



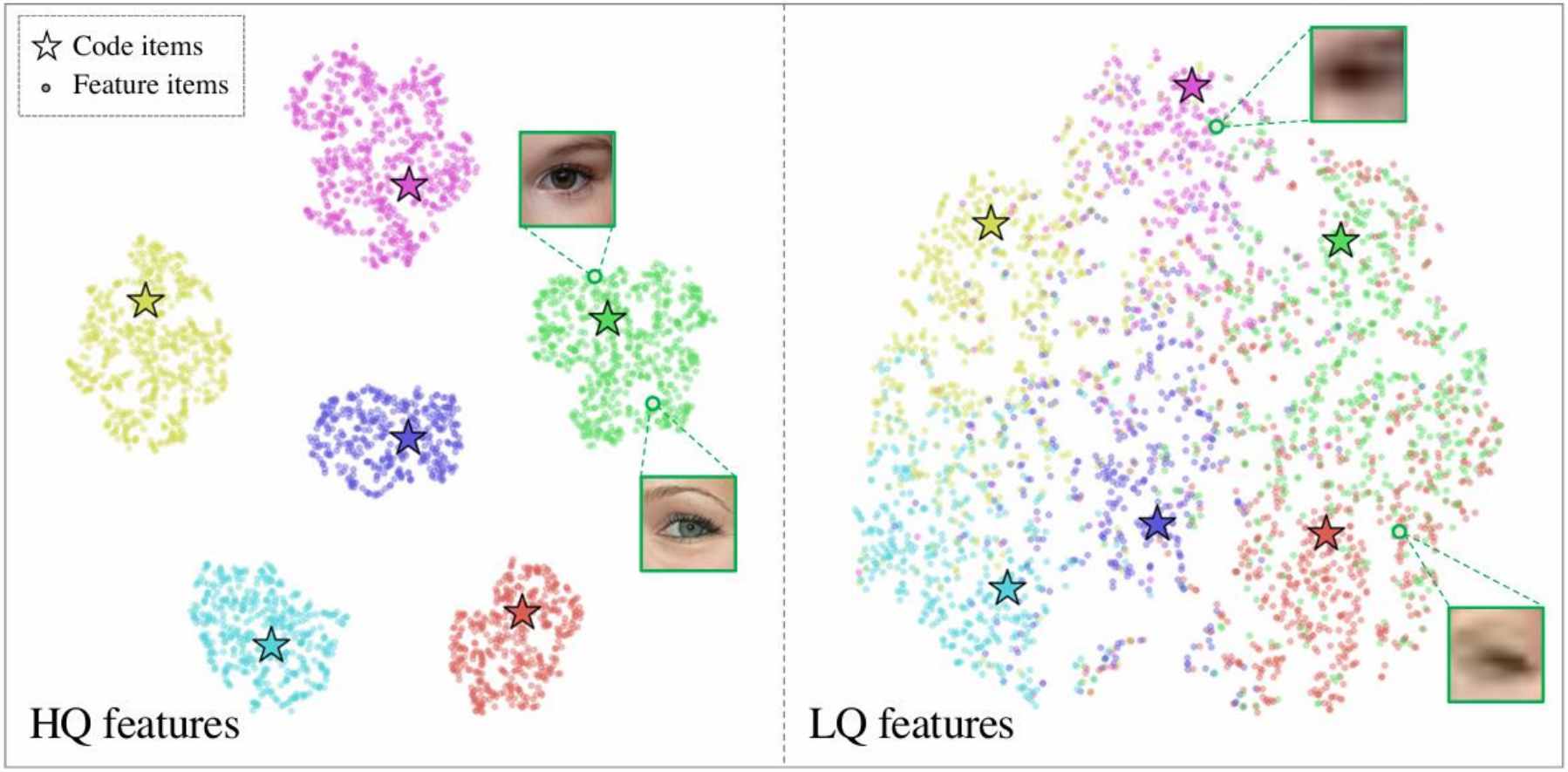
Ground Truth

Reconstructions

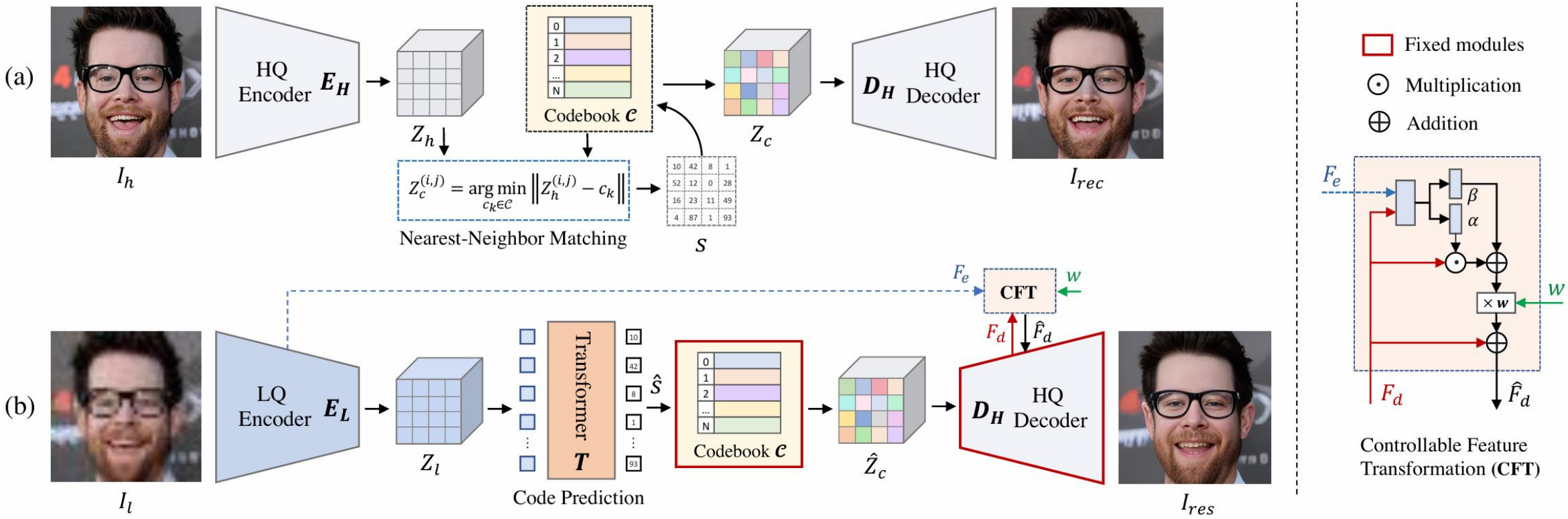
Generations with PixelCNN

Code-Former

- CodeBook lookup with KNN becomes unreliable at low resolutions.



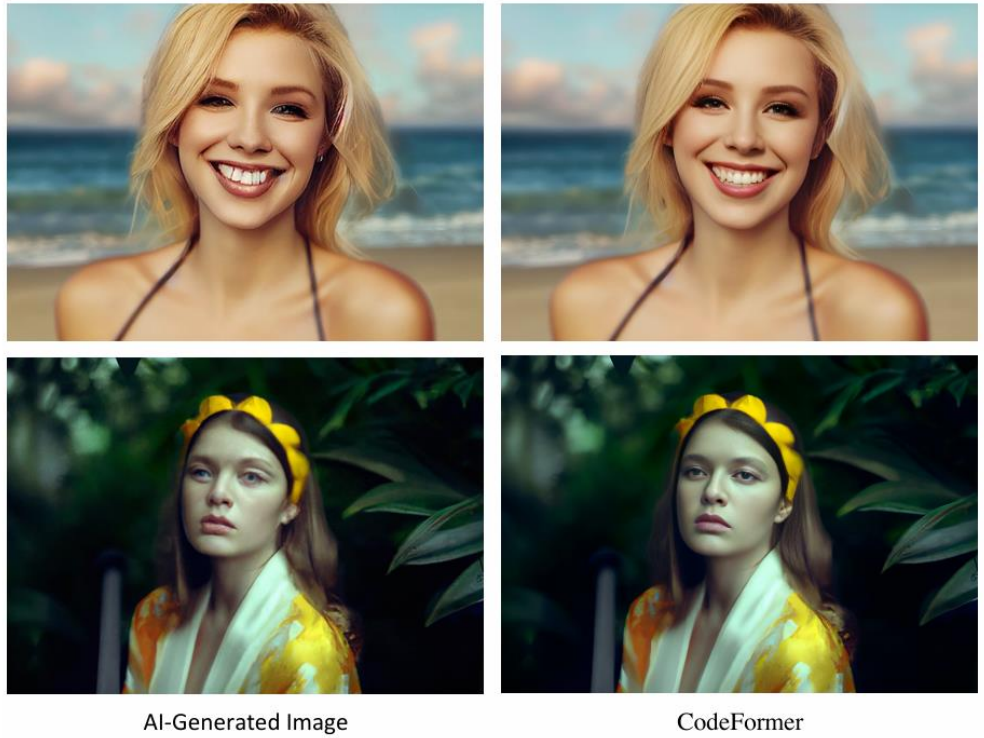
Code-Former



CodeFormer
Zhou et al. NIPS 2022

Code-Former

- Transformer-based Lookup
- Control quality and fidelity
- Fix faces in AI-generated artwork



Variational Autoencoder (VAE)

Probabilistic spin to traditional autoencoders => allows generating data
Defines an intractable density => derive and optimize a (variational) lower bound

Pros:

- Principled approach to generative models
- Interpretable latent space.
- Allows inference of $q(z|x)$, can be useful feature representation for other tasks

Cons:

- Maximizes lower bound of likelihood: okay, but not as good evaluation as PixelRNN/PixelCNN
- Samples blurrier and lower quality compared to state-of-the-art (GANs)

Active areas of research:

- More flexible approximations, e.g. richer approximate posterior instead of diagonal Gaussian, e.g., Gaussian Mixture Models (GMMs), Categorical Distributions.
- Learning disentangled representations.

slide from https://cs231n.stanford.edu/slides/2024/lecture_13.pdf

Thanks!