EECE 571F: Deep Learning with Structures

Lecture 8 II: Auto-Encoders & Variational Auto-Encoders

Renjie Liao

University of British Columbia

Winter, Term 1, 2023

Outline

- Autoencoders
  - **Motivation & Overview**
  - Linear Autoencoders & PCA
  - Deep Autoencoders
- Denoising Autoencoders
- Variational Autoencoders
  - Motivation & Overview
  - Evidence Lower Bound (ELBO)
  - Models
  - Amortized Inference
  - Reparameterization Trick
- Graph Variational Autoencoders

Autoencoders (AEs)

- Autoencoders are feed-forward neural networks that reconstruct/predict the input

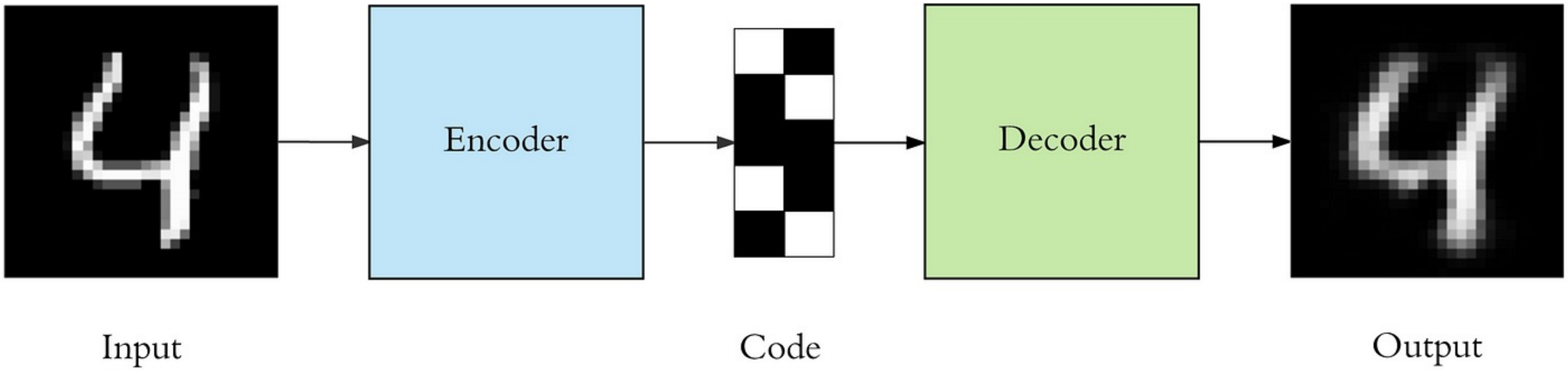


Image Credit: [1]

Autoencoders (AEs)

- Autoencoders are feed-forward neural networks that reconstruct/predict the input
- To make it non-trivial, we need a *bottleneck* (i.e. the dimension of code being much smaller compared to the input). Why?

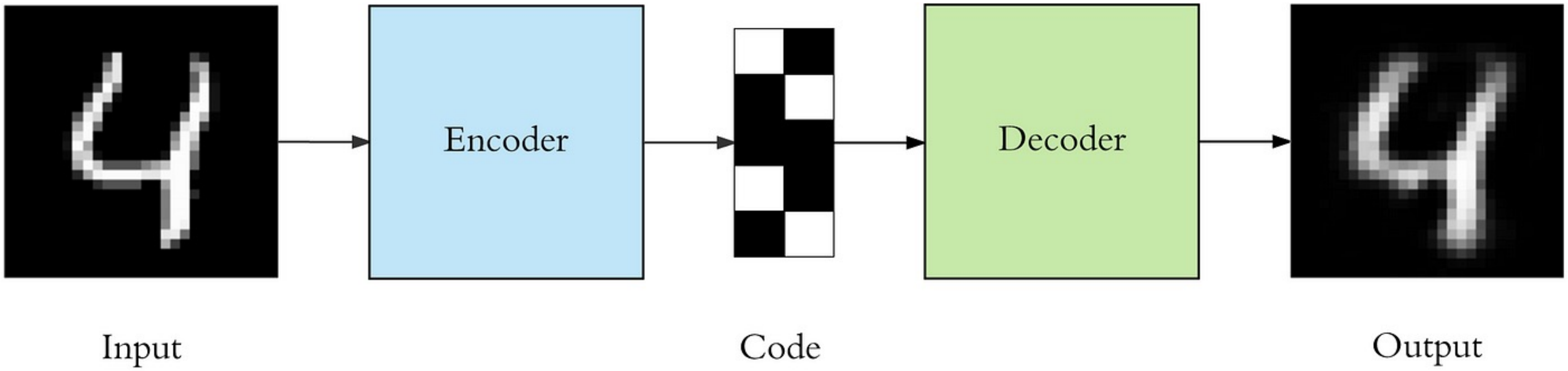


Image Credit: [1]

Autoencoders (AEs)

- Autoencoders are feed-forward neural networks that reconstruct/predict the input
- To make it non-trivial, we need a *bottleneck* (i.e. the dimension of code being much smaller compared to the input). Why? Otherwise, Encoder and Decoder can learn to just copy input (show you later).

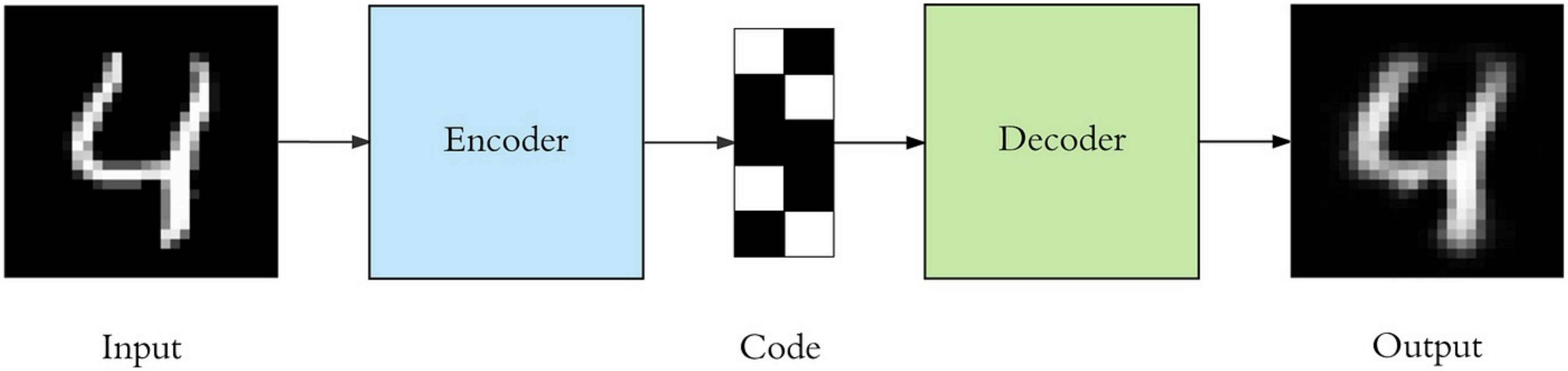


Image Credit: [1]

Autoencoders (AEs)

Why should we care?

- Dimension reduction

    e.g., visualizing high-dimension data

Autoencoders (AEs)

Why should we care?

- Dimension reduction

  e.g., visualizing high-dimension data

- Unsupervised representation learning

  e.g., if we have abundant data without annotations, learned representations will potentially be useful for downstream tasks like classification and regression

Outline

- Autoencoders
    - Motivation & Overview
    - **Linear Autoencoders & PCA**
    - Deep Autoencoders
- Denoising Autoencoders
- Variational Autoencoders
    - Motivation & Overview
    - Evidence Lower Bound (ELBO)
    - Models
    - Amortized Inference
    - Reparameterization Trick
- Graph Variational Autoencoders

Linear Autoencoders

Simplest autoencoders: a single hidden layer with linear activations

We can train them by minimizing the mean squared errors (MSE):

$$\ell(\tilde{\mathbf{x}}, \mathbf{x}) = \|\tilde{\mathbf{x}} - \mathbf{x}\|_2^2$$

The network is $\tilde{\mathbf{x}} = UV\mathbf{x}$

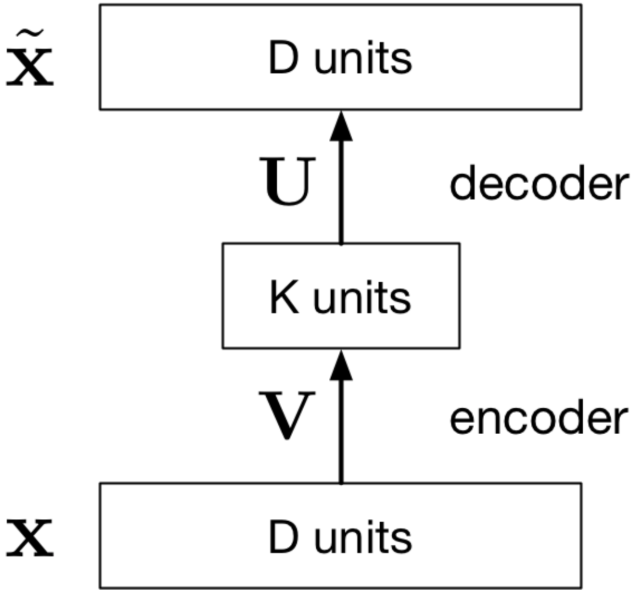
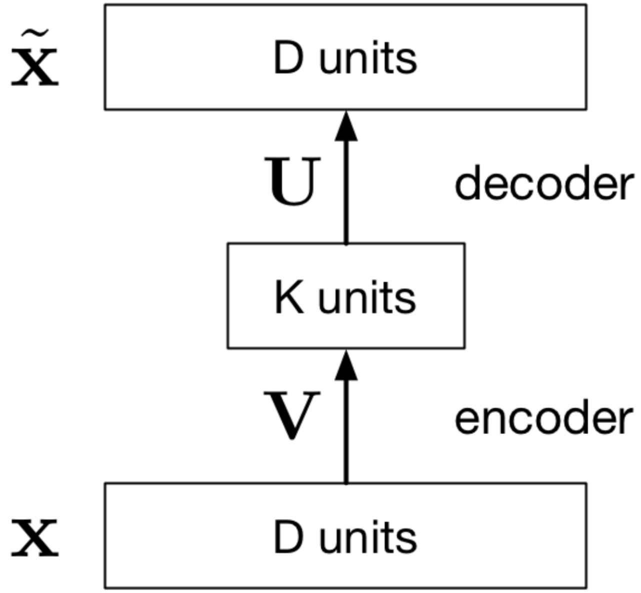


Image Credit: [2]

Linear Autoencoders

Simplest autoencoders: a single hidden layer with linear activations



We can train them by minimizing the mean squared errors (MSE):

$$\ell(\tilde{\mathbf{x}}, \mathbf{x}) = \|\tilde{\mathbf{x}} - \mathbf{x}\|_2^2$$

The network is $\tilde{\mathbf{x}} = UV\mathbf{x}$

If $K \geq D$, one can choose U and V such that $UV = I$ (copying input)

Underdetermined system of equations, possibly having infinite solutions

Image Credit: [2]

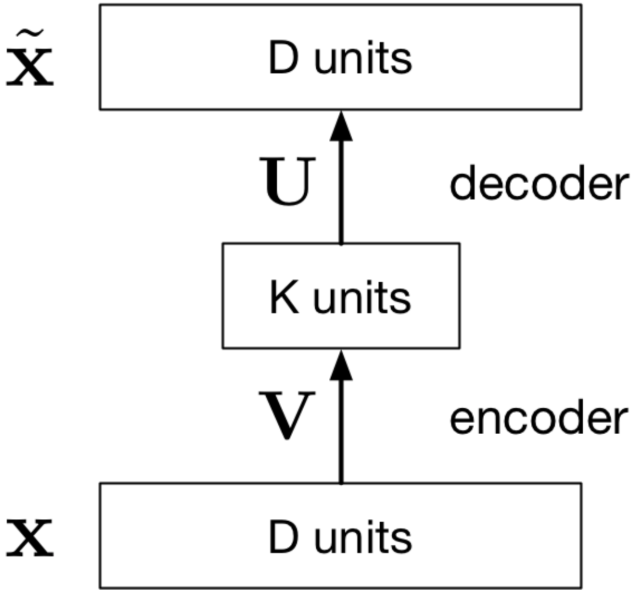
Linear Autoencoders

Simplest autoencoders: a single hidden layer with linear activations

We can train them by minimizing the mean squared errors (MSE):

$$\ell(\tilde{\mathbf{x}}, \mathbf{x}) = \|\tilde{\mathbf{x}} - \mathbf{x}\|_2^2$$

The network is $\tilde{\mathbf{x}} = UV\mathbf{x}$



If $K \geq D$, one can choose U and V such that $UV = I$ (copying input)

Underdetermined system of equations, possibly having infinite solutions

Else $K < D$, we are reducing the dimension
The reconstructed output lies in the column space of U, which is a K-dimensional subspace

Image Credit: [2]

Linear Autoencoders & Principle Component Analysis

We know linear autoencoders map D-dimensional input to a K-dimensional subspace

What is the best possible K-dimensional mapping?

Linear Autoencoders & Principle Component Analysis

We know linear autoencoders map D-dimensional input to a K-dimensional subspace

What is the best possible K-dimensional mapping?

The one that minimizes the reconstruction error!

Linear Autoencoders & Principle Component Analysis

We know linear autoencoders map D-dimensional input to a K-dimensional subspace

What is the best possible K-dimensional mapping?

The one that minimizes the reconstruction error!

To obtain it, let us first center the data, i.e., $\mathbf{x}_i = \mathbf{x}_i - \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$

Linear Autoencoders & Principle Component Analysis

We know linear autoencoders map D-dimensional input to a K-dimensional subspace

What is the best possible K-dimensional mapping?

The one that minimizes the reconstruction error!

To obtain it, let us first center the data, i.e., $\mathbf{x}_i = \mathbf{x}_i - \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$

By Pythagorean Theorem, we have:

$$\underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i\|^2}_{\text{constant}} = \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|^2}_{\text{reconstruction error}} + \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\tilde{\mathbf{x}}_i\|^2}_{\text{projected variance}}$$

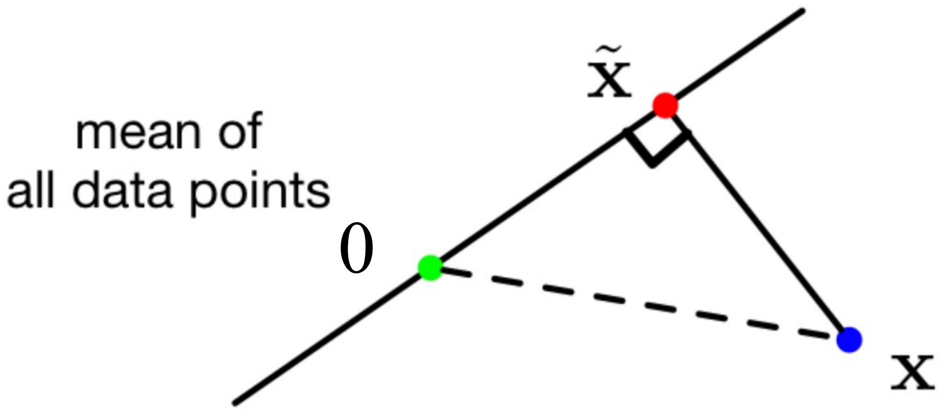


Image Credit: [2]

Linear Autoencoders & Principle Component Analysis

We know linear autoencoders map D-dimensional input to a K-dimensional subspace

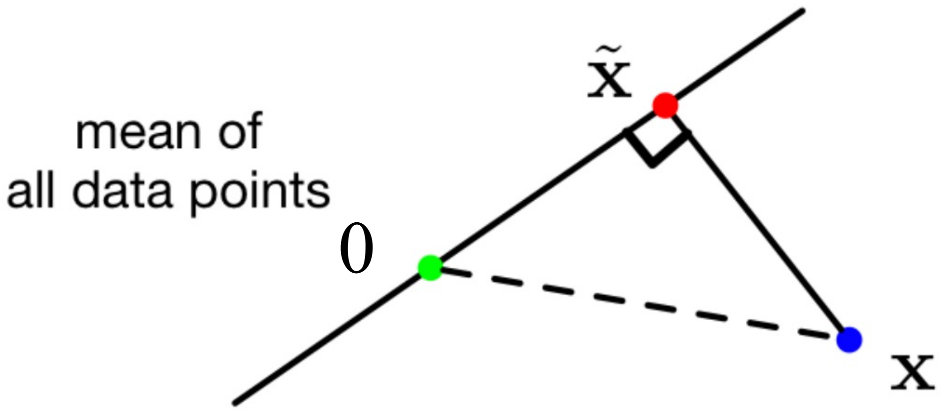
What is the best possible K-dimensional mapping?

The one that minimizes the reconstruction error!

To obtain it, let us first center the data, i.e., $\mathbf{x}_i = \mathbf{x}_i - \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$

By Pythagorean Theorem, we have:

$$\underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i\|^2}_{\text{constant}} = \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|^2}_{\text{reconstruction error}} + \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\tilde{\mathbf{x}}_i\|^2}_{\text{projected variance}}$$



Maximizing the projected variance is equivalent to minimizing the reconstruction error!

Image Credit: [2]

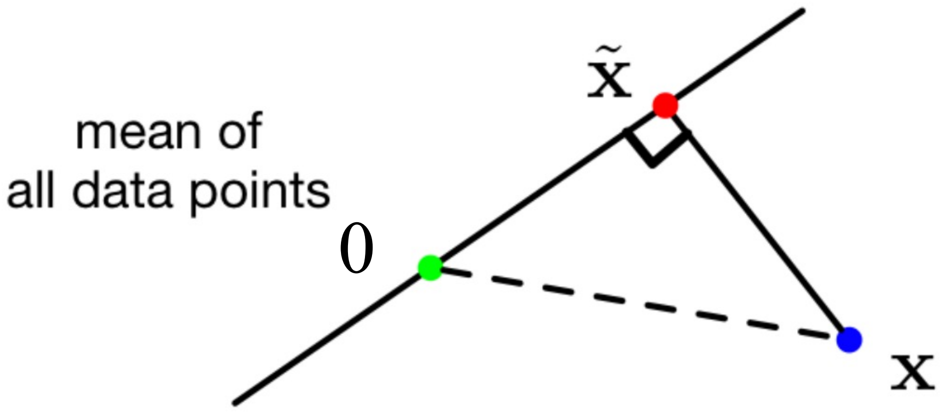
Linear Autoencoders & Principle Component Analysis

We know linear autoencoders map D-dimensional input to a K-dimensional subspace

What is the best possible K-dimensional mapping?

The one that minimizes the reconstruction error!

To obtain it, let us first center the data, i.e., $\mathbf{x}_i = \mathbf{x}_i - \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$



By Pythagorean Theorem, we have:

$$\underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i\|^2}_{\text{constant}} = \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|^2}_{\text{reconstruction error}} + \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\tilde{\mathbf{x}}_i\|^2}_{\text{projected variance}}$$

Maximizing the projected variance is equivalent to minimizing the reconstruction error!

You can maximize the variance in closed-form via *principle component analysis (PCA)*!

Image Credit: [2]

Linear Autoencoders & Principle Component Analysis

When you train a linear autoencoder, it may not give you the optimal K-dimensional mapping returned by PCA

Linear Autoencoders & Principle Component Analysis

When you train a linear autoencoder, it may not give you the optimal K-dimensional mapping returned by PCA

In fact, given $\tilde{\mathbf{x}} = UV\mathbf{x}$, the minima of the reconstruction loss $\frac{1}{N}\sum_{i=1}^{N} \|\tilde{\mathbf{x}}_i - \mathbf{x}_i\|_2^2$ is not unique!

The objective is invariant under any invertible matrix A s.t. $\tilde{\mathbf{x}} = UA^{-1}AV\mathbf{x}$

Linear Autoencoders & Principle Component Analysis

When you train a linear autoencoder, it may not give you the optimal K-dimensional mapping returned by PCA

In fact, given $\tilde{\mathbf{x}} = UV\mathbf{x}$, the minima of the reconstruction loss $\frac{1}{N}\sum_{i=1}^{N} \|\tilde{\mathbf{x}}_i - \mathbf{x}_i\|_2^2$ is not unique!

The objective is invariant under any invertible matrix A s.t. $\tilde{\mathbf{x}} = UA^{-1}AV\mathbf{x}$

One can add regularization terms [3] so that the returned minima
can exactly recover principled components!

Linear Autoencoders & Principle Component Analysis

When you train a linear autoencoder, it may not give you the optimal K-dimensional mapping returned by PCA

In fact, given $\tilde{\mathbf{x}} = UV\mathbf{x}$, the minima of the reconstruction loss $\frac{1}{N}\sum_{i=1}^{N} \|\tilde{\mathbf{x}}_i - \mathbf{x}_i\|_2^2$ is not unique!

The objective is invariant under any invertible matrix A s.t. $\tilde{\mathbf{x}} = UA^{-1}AV\mathbf{x}$

One can add regularization terms [3] so that the returned minima can exactly recover principled components!

Principle components of faces ("Eigenfaces") from CBCL dataset:

Image Credit: [2]

Linear Autoencoders & Principle Component Analysis

Principle components of digits from MNIST dataset:

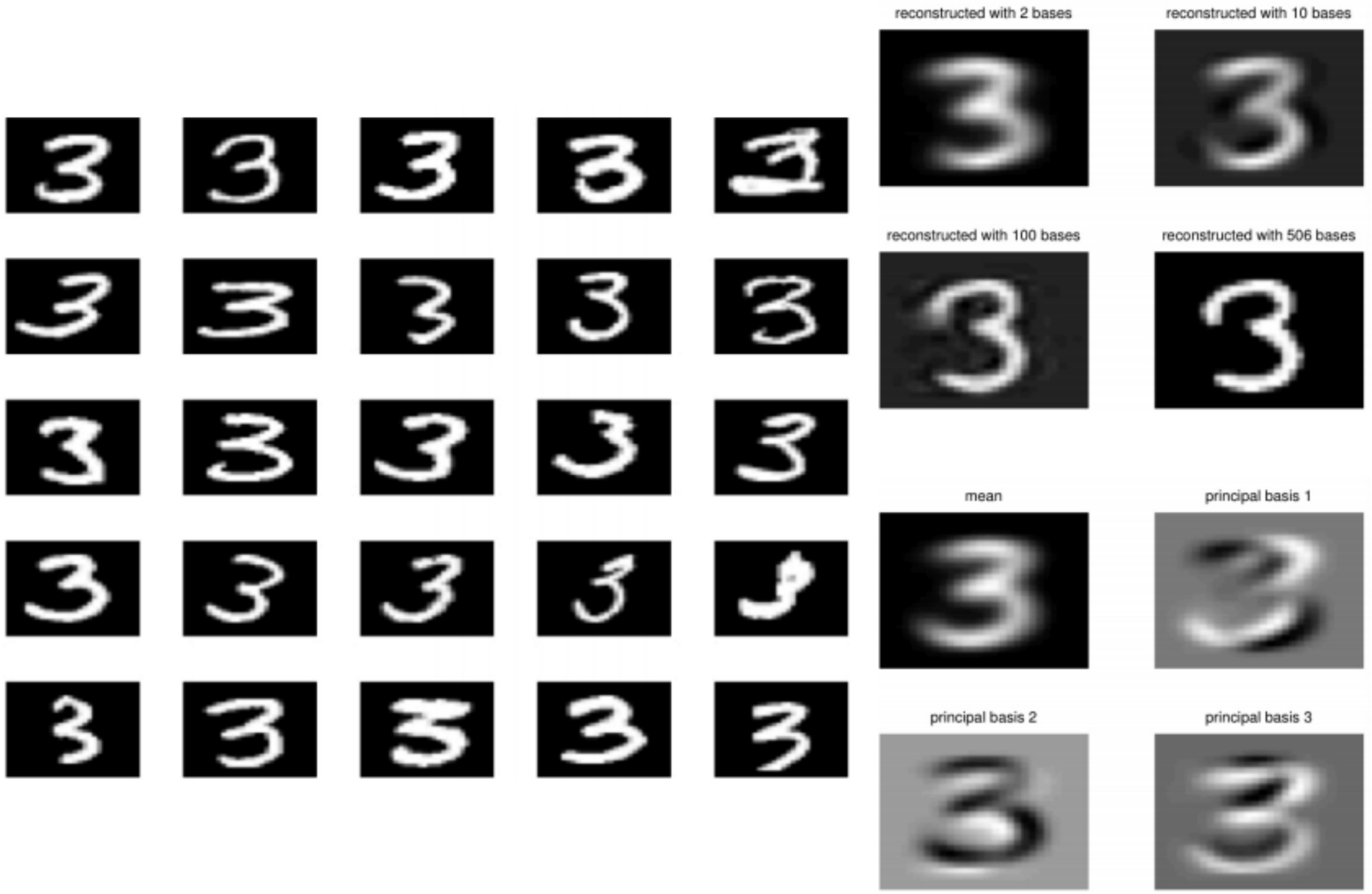


Image Credit: [2]

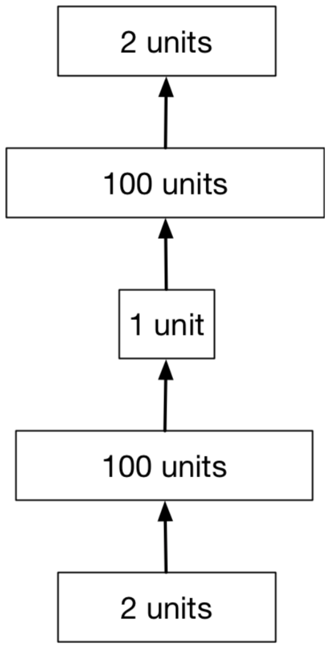
Outline

- Autoencoders
  - Motivation & Overview
  - Linear Autoencoders & PCA
  - **Deep Autoencoders**
- Denoising Autoencoders
- Variational Autoencoders
  - Motivation & Overview
  - Evidence Lower Bound (ELBO)
  - Models
  - Amortized Inference
  - Reparameterization Trick
- Graph Variational Autoencoders

Deep Autoencoders

Deep autoencoders learn to project data onto a *manifold* instead of a subspace

This is a kind of *nonlinear dimension reduction*



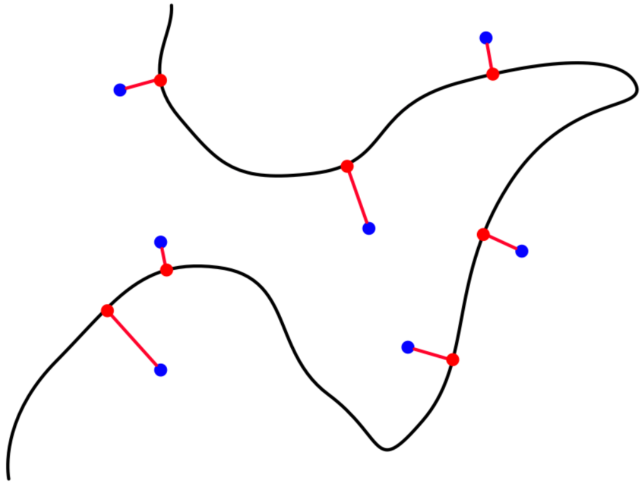
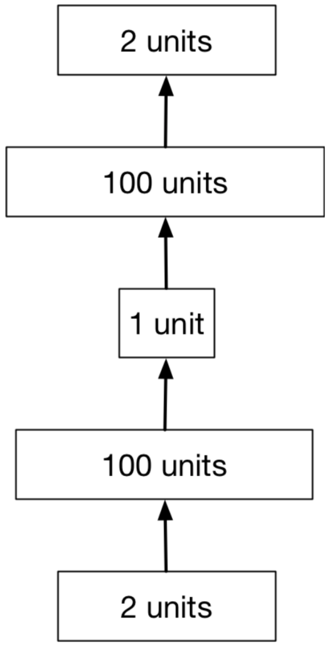


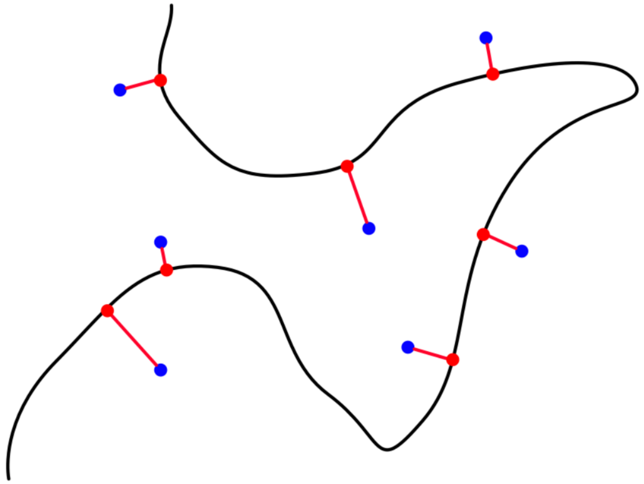
Image Credit: [2]

Deep Autoencoders

Deep autoencoders learn to project data onto a *manifold* instead of a subspace

This is a kind of *nonlinear dimension reduction*





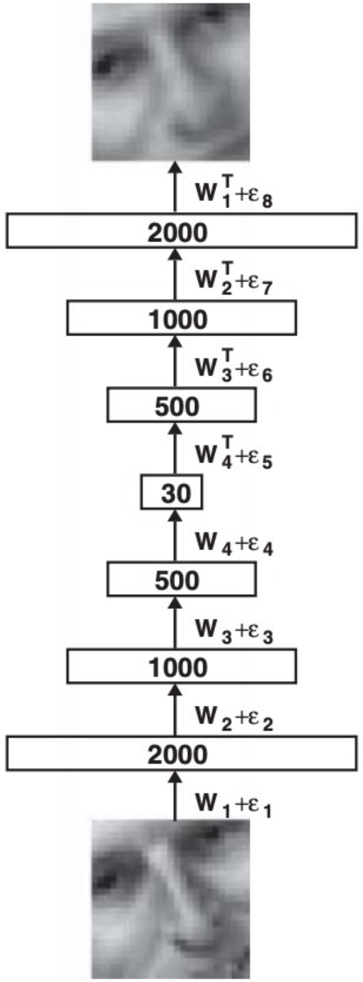


Image Credit: [2,4]

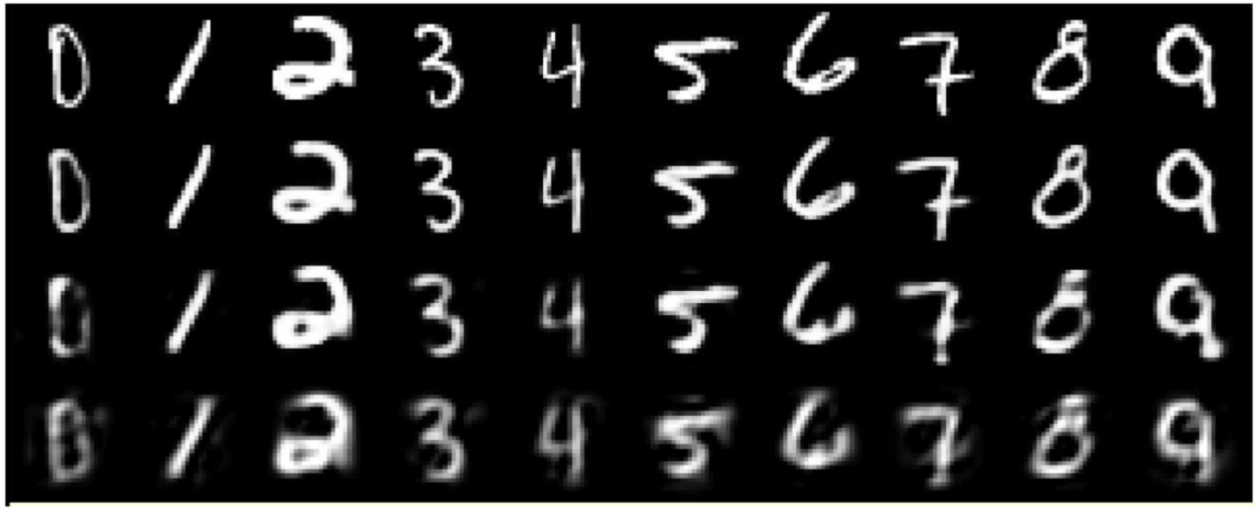
Deep Autoencoders

Deep autoencoders learn to project data onto a *manifold* instead of a subspace

This is a kind of *nonlinear dimension reduction*

Deep autoencoders can learn more powerful codes/representations compared to linear ones (PCA)

Reconstructions with various methods on MNIST dataset:



Real data

30-d deep autoencoder

30-d logistic PCA

30-d PCA

Image Credit: [4]

Outline

- Autoencoders
  - Motivation & Overview
  - Linear Autoencoders & PCA
  - Deep Autoencoders
- **Denoising Autoencoders**
- Variational Autoencoders
  - Motivation & Overview
  - Evidence Lower Bound (ELBO)
  - Models
  - Amortized Inference
  - Reparameterization Trick
- Graph Variational Autoencoders

Denoising Autoencoders (DAEs)

Reconstructing input data is not the only way to learn useful representations in an unsupervised way.

Denoising Autoencoders (DAEs)

Reconstructing input data is not the only way to learn useful representations in an unsupervised way.

We can also achieve a similar goal via **denoising**!

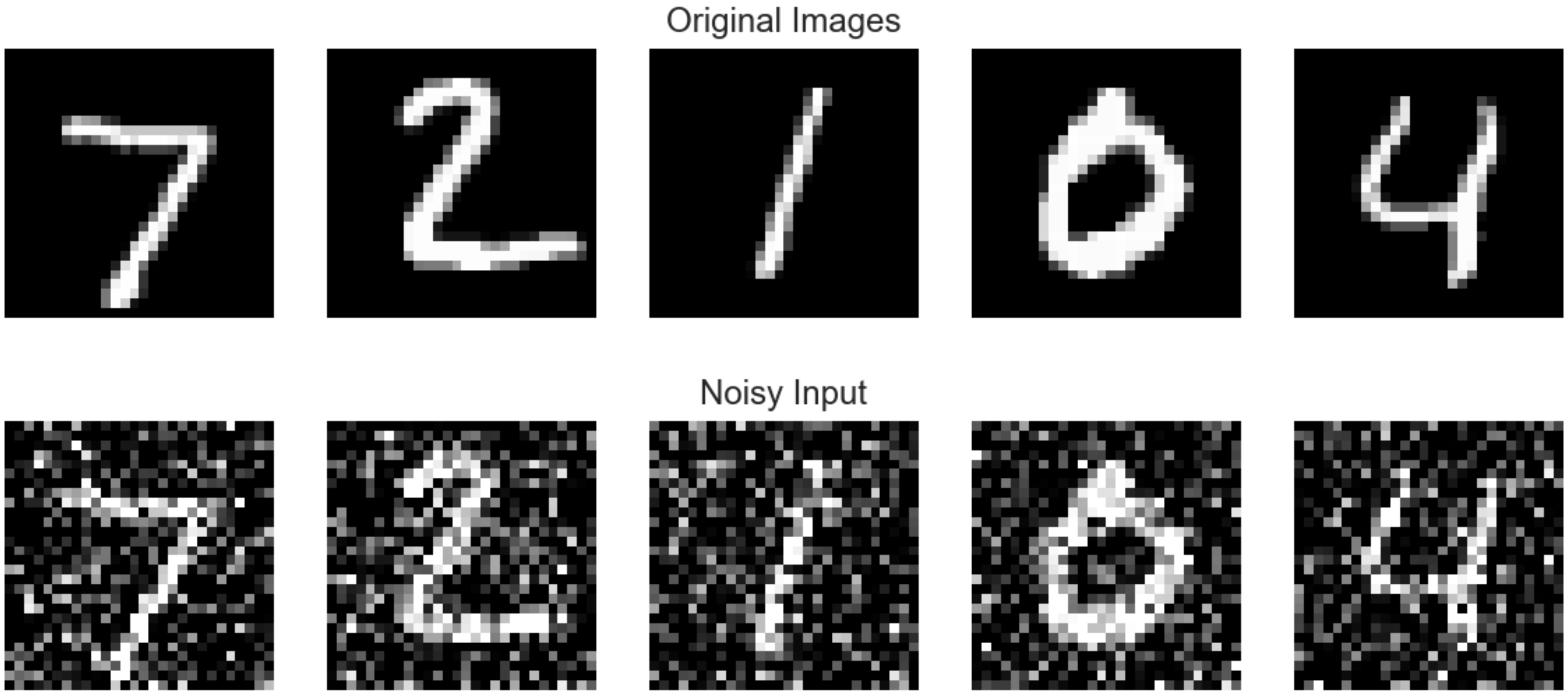


Image Credit: [1]

Denoising Autoencoders (DAEs)

Reconstructing input data is not the only way to learn useful representations in an unsupervised way.

We can also achieve a similar goal via **denoising**!

We add random noise (e.g., additive Gaussian) and force the neural network to learn useful representations so that *structures in images are preserved whereas noise is removed!*

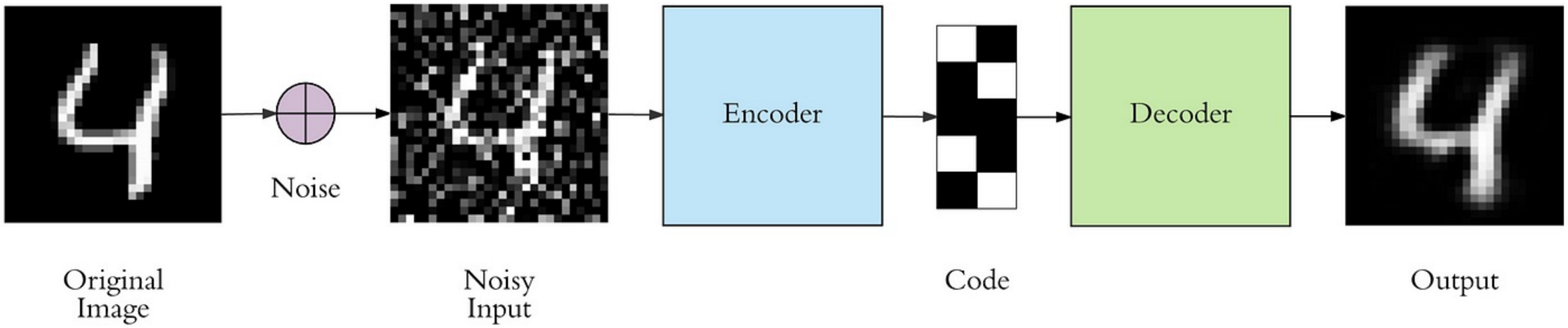


Image Credit: [1]

Denoising Autoencoders (DAEs)

DAEs can do a great job in denoising:

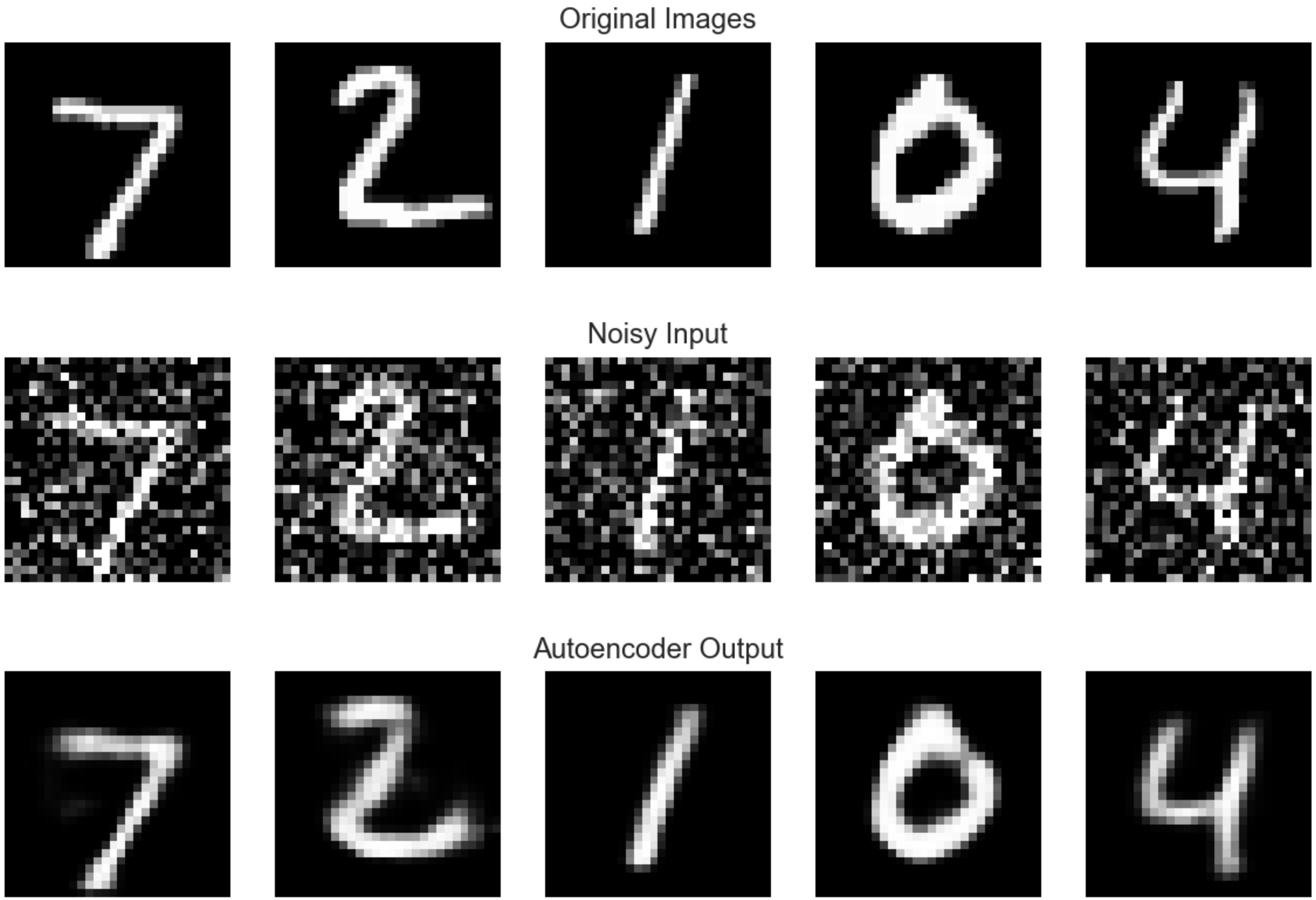
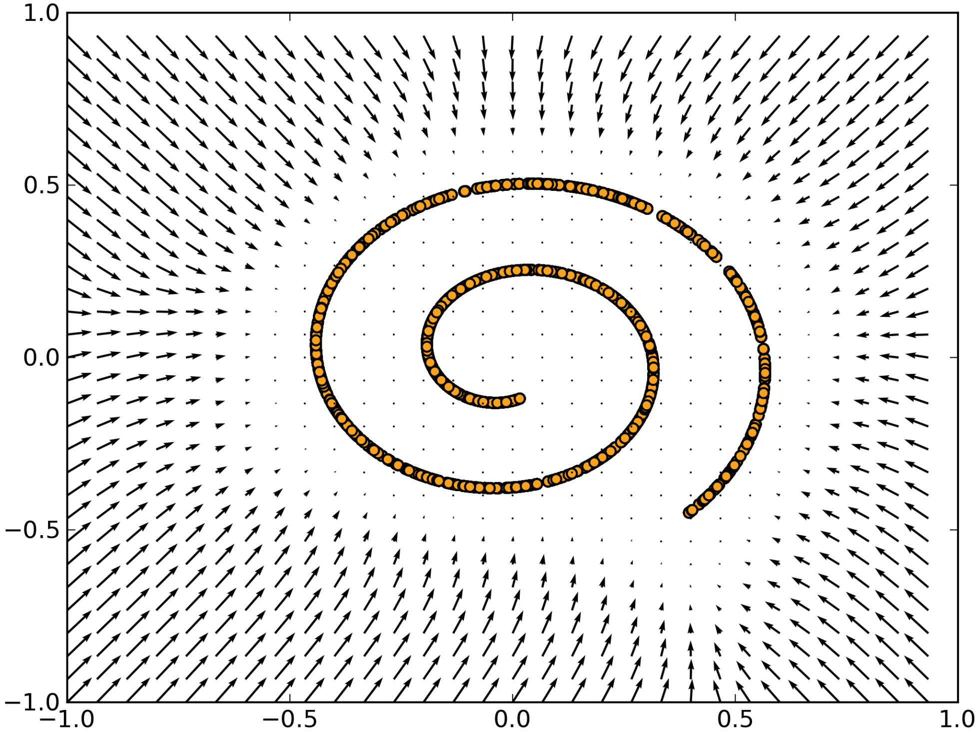


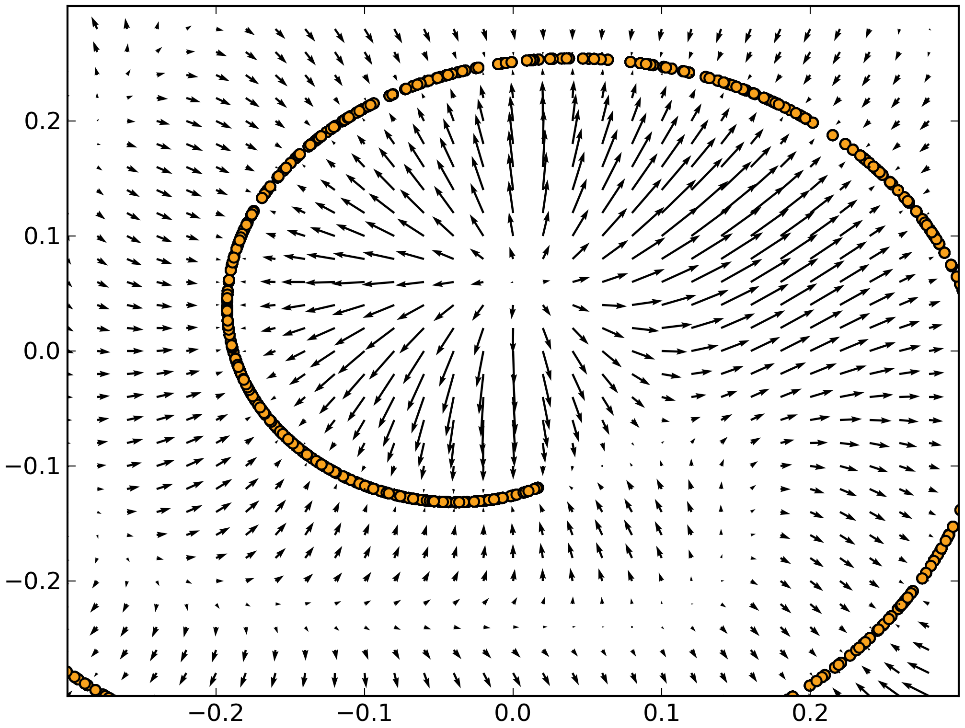
Image Credit: [1]

Denoising Autoencoders (DAEs)

DAEs can learn correct vector fields (reconstruction – noisy input) that point to data manifold (spiral):



zoom out



zoom in

Image Credit: [5]

Outline

- Autoencoders
  - Motivation & Overview
  - Linear Autoencoders & PCA
  - Deep Autoencoders
- Denoising Autoencoders
- Variational Autoencoders
  - **Motivation & Overview**
  - Evidence Lower Bound (ELBO)
  - Models
  - Amortized Inference
  - Reparameterization Trick
- Graph Variational Autoencoders

Variational Autoencoders (VAEs)

Suppose we have trained an autoencoder

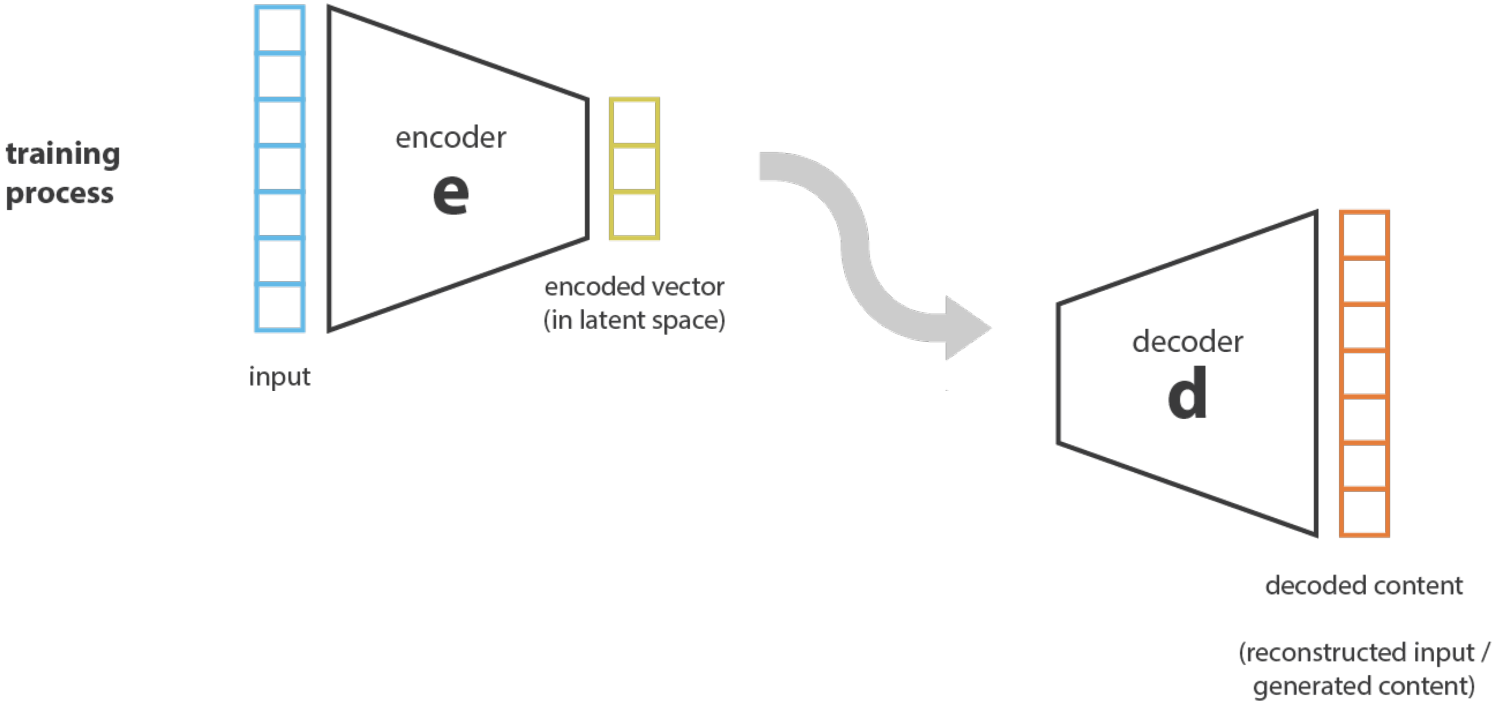


Image Credit: [6]

Variational Autoencoders (VAEs)

Suppose we have trained an autoencoder and would like to use it to generate data

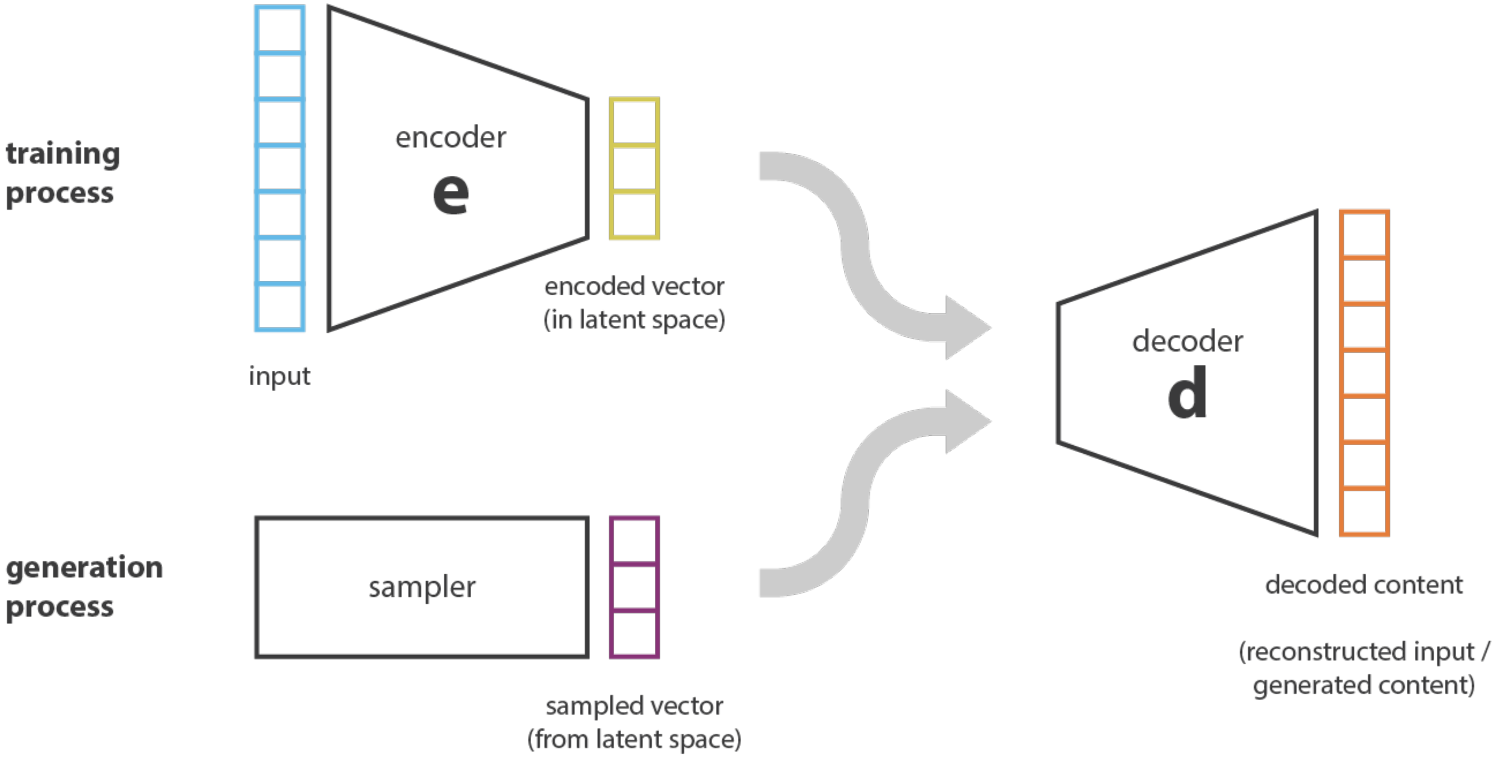


Image Credit: [6]

Variational Autoencoders (VAEs)

Suppose we have trained an autoencoder and would like to use it to generate data

What would happen?

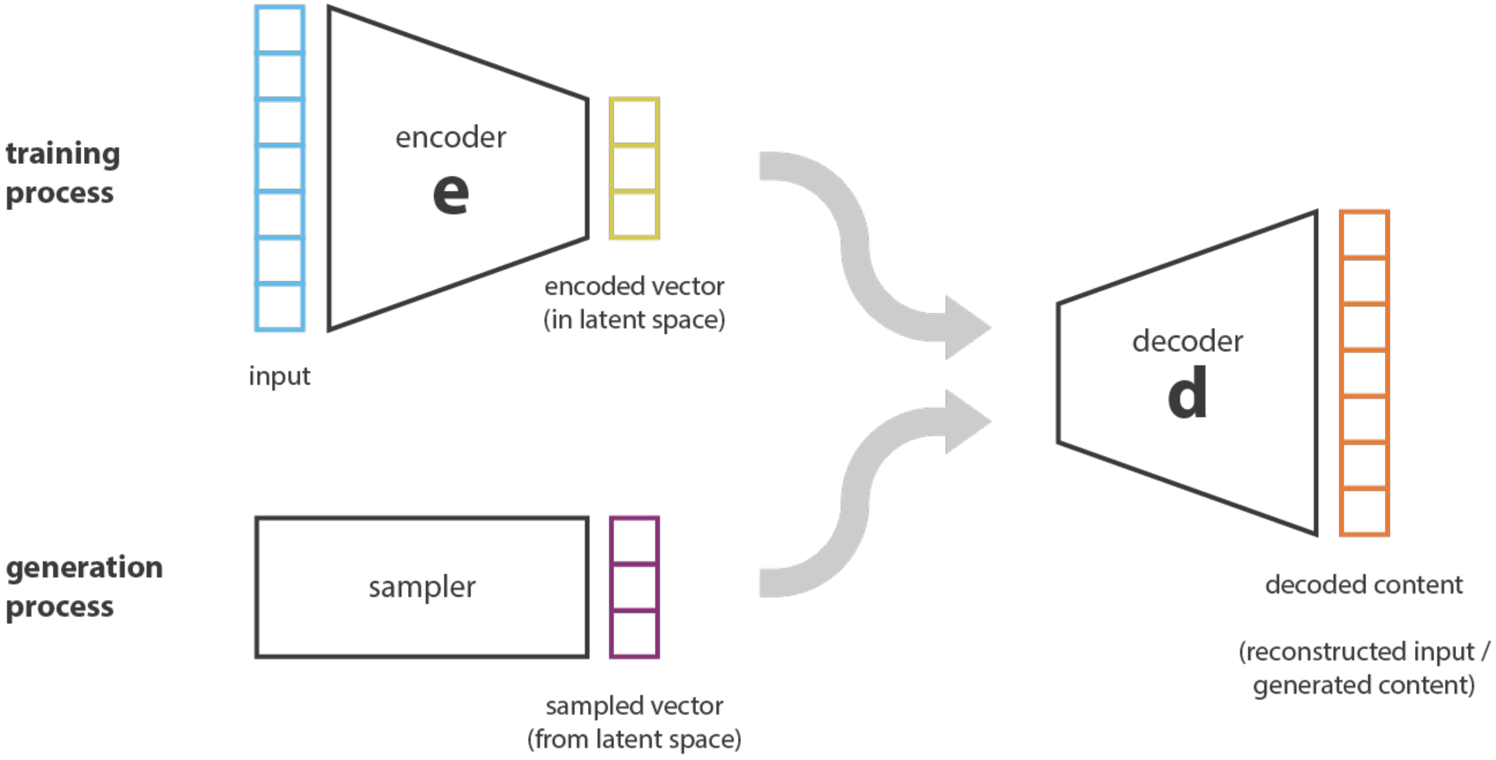


Image Credit: [6]

Variational Autoencoders (VAEs)

Suppose we have trained an autoencoder and would like to use it to generate data

What would happen? *Sampled data could be very bad if sampled latent codes are far off the manifold!*

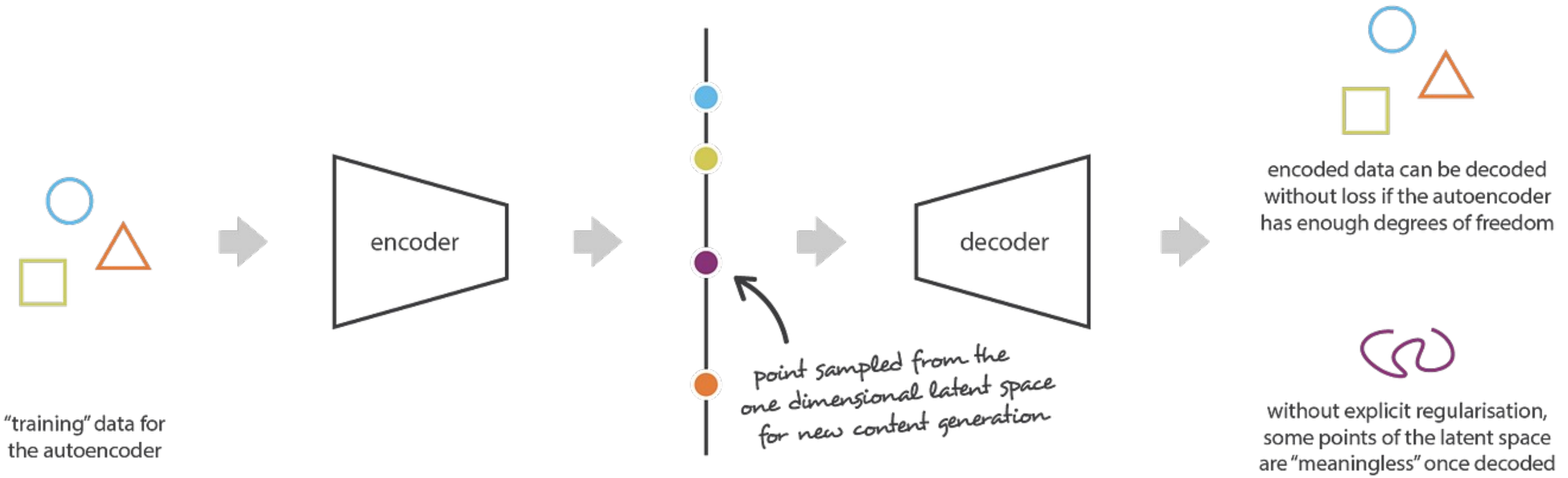


Image Credit: [6]

Variational Autoencoders (VAEs)

Suppose we have trained an autoencoder and would like to use it to generate data

What would happen? *Sampled data could be very bad if sampled latent codes are far off the manifold!*

Ideally, we hope to learn a regular latent space that similar latent codes generate similar data!

Image Credit: [6]

Variational Autoencoders (VAEs)

Suppose we have trained an autoencoder and would like to use it to generate data

What would happen? *Sampled data could be very bad if sampled latent codes are far off the manifold!*

Ideally, we hope to learn a regular latent space that similar latent codes generate similar data!

Can AEs learn such latent spaces that are good for reconstruction + generation? Yes, VAEs [7,8]!

Image Credit: [6]

Outline

- Autoencoders
    - Motivation & Overview
    - Linear Autoencoders & PCA
    - Deep Autoencoders
- Denoising Autoencoders
- Variational Autoencoders
    - Motivation & Overview
    - **Evidence Lower Bound (ELBO)**
    - Models
    - Amortized Inference
    - Reparameterization Trick
- Graph Variational Autoencoders

Maximum Likelihood

Given data $X \in \mathbb{R}^d$, Maximum Likelihood is:

$$\max_{\theta} \quad \log p_{\theta}(X)$$

Maximum Likelihood

Given data $X \in \mathbb{R}^d$, Maximum Likelihood is:

$$\max_{\theta} \quad \log p_\theta(X)$$

Variational Auto-Encoders (VAEs)

We introduce latent variable $Z \in \mathbb{R}^m$

Maximum Likelihood

Given data $X \in \mathbb{R}^d$, Maximum Likelihood is:

$$\max_{\theta} \quad \log p_\theta(X)$$

Variational Auto-Encoders (VAEs)

We introduce latent variable $Z \in \mathbb{R}^m$

$$\begin{aligned} p_\theta(X) &= \int_Z p_\theta(X, Z) dZ \\ &= \int_Z p_\theta(X|Z) p_\theta(Z) dZ \end{aligned}$$

Maximum Likelihood

Given data $X \in \mathbb{R}^d$, Maximum Likelihood is:

$$\max_{\theta} \quad \log p_\theta(X)$$

Variational Auto-Encoders (VAEs)

We introduce latent variable $Z \in \mathbb{R}^m$

$$\begin{aligned} p_\theta(X) &= \int_Z p_\theta(X, Z) dZ \\ &= \int_Z p_\theta(X|Z) p_\theta(Z) dZ \end{aligned}$$

Intractable Integration!

Evidence Lower Bound (ELBO)

Variational Approximation

$$\begin{aligned}\log p_\theta(X) &= \log\left(\frac{p_\theta(X, Z)}{p_\theta(Z|X)}\right)\\ &= \log\left(\frac{p_\theta(X, Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)\end{aligned}$$

Evidence Lower Bound (ELBO)

Variational Approximation

$$\begin{aligned}\log p_\theta(X) &= \log\left(\frac{p_\theta(X,Z)}{p_\theta(Z|X)}\right)\\ &= \log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)\end{aligned}$$

Integrating from both sides:

$$\begin{aligned}\log p_\theta(X) &= \int q_\phi(Z|X)\log p_\theta(X)dZ\\ &= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ\\ &= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)dZ + \int q_\phi(Z|X)\log\left(\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ\\ &= \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)\right] + \mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z|X)\right)\end{aligned}$$

Evidence Lower Bound (ELBO)

Variational Approximation

$$\begin{aligned}\log p_\theta(X) &= \log\left(\frac{p_\theta(X,Z)}{p_\theta(Z|X)}\right)\\ &= \log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)\end{aligned}$$

Integrating from both sides:

$$\begin{aligned}\log p_\theta(X) &= \int q_\phi(Z|X)\log p_\theta(X)dZ\\ &= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ\\ &= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)dZ + \int q_\phi(Z|X)\log\left(\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ\\ &= \underbrace{\mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)\right]}_{\text{Evidence Lower Bound (ELBO)}} + \underbrace{\mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z|X)\right)}_{\text{Kullback-Leibler (KL) Divergence}}\end{aligned}$$

Evidence Lower Bound (ELBO)

Variational Approximation

$$\log p_\theta(X) = \log\left(\frac{p_\theta(X,Z)}{p_\theta(Z|X)}\right)$$
$$= \log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)$$

Integrating from both sides:

Why is it a lower bound?

$$\log p_\theta(X) = \int q_\phi(Z|X)\log p_\theta(X)dZ$$
$$= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ$$
$$= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)dZ + \int q_\phi(Z|X)\log\left(\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ$$
$$= \underbrace{\mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)\right]}_{\text{Evidence Lower Bound (ELBO)}} + \underbrace{\mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z|X)\right)}_{\text{Kullback-Leibler (KL) Divergence}}$$

Evidence Lower Bound (ELBO)

Variational Approximation

$$\log p_\theta(X) = \log\left(\frac{p_\theta(X,Z)}{p_\theta(Z|X)}\right)$$
$$= \log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)$$

Integrating from both sides:

Why is it a lower bound? KL is nonnegative!

$$\log p_\theta(X) = \int q_\phi(Z|X)\log p_\theta(X)dZ$$
$$= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ$$
$$= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)dZ + \int q_\phi(Z|X)\log\left(\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ$$
$$= \underbrace{\mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)\right]}_{\text{Evidence Lower Bound (ELBO)}} + \underbrace{\mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z|X)\right)}_{\text{Kullback-Leibler (KL) Divergence}}$$

Evidence Lower Bound (ELBO)   Kullback-Leibler (KL) Divergence

Evidence Lower Bound (ELBO)

Variational Approximation

$$\log p_\theta(X) = \log\left(\frac{p_\theta(X,Z)}{p_\theta(Z|X)}\right)$$
$$= \log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)$$

Integrating from both sides:

Why is it a lower bound? KL is nonnegative!

Why is it called variational approximation?

$$\log p_\theta(X) = \int q_\phi(Z|X)\log p_\theta(X)dZ$$
$$= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ$$
$$= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)dZ + \int q_\phi(Z|X)\log\left(\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ$$
$$= \underbrace{\mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)\right]}_{\text{Evidence Lower Bound (ELBO)}} + \underbrace{\mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z|X)\right)}_{\text{Kullback-Leibler (KL) Divergence}}$$

Evidence Lower Bound (ELBO) Kullback-Leibler (KL) Divergence

Evidence Lower Bound (ELBO)

Variational Approximation

$$\log p_\theta(X) = \log\left(\frac{p_\theta(X,Z)}{p_\theta(Z|X)}\right)$$
$$= \log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)$$

Integrating from both sides:

$$\log p_\theta(X) = \int q_\phi(Z|X)\log p_\theta(X)dZ$$
$$= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ$$
$$= \int q_\phi(Z|X)\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)dZ + \int q_\phi(Z|X)\log\left(\frac{q_\phi(Z|X)}{p_\theta(Z|X)}\right)dZ$$
$$= \underbrace{\mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)\right]}_{\text{Evidence Lower Bound (ELBO)}} + \underbrace{\mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z|X)\right)}_{\text{Kullback-Leibler (KL) Divergence}}$$

Evidence Lower Bound (ELBO) Kullback-Leibler (KL) Divergence

Why is it a lower bound? KL is nonnegative!

Why is it called variational approximation?
We choose one distribution (function) from a family to approximate the target!

Evidence Lower Bound (ELBO)

Since true posterior $p_\theta(Z|X)$ is often unknown, KL term is intractable

Evidence Lower Bound (ELBO)

Since true posterior $p_\theta(Z|X)$ is often unknown, KL term is intractable

ELBO:

$$\begin{aligned}
\mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)\right] &= \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X|Z)p_\theta(Z)}{q_\phi(Z|X)}\right)\right] \\
&= \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(p_\theta(X|Z)\right)\right] + \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(Z)}{q_\phi(Z|X)}\right)\right] \\
&= -\mathbb{E}_{q_\phi(Z|X)}\left[-\log\left(p_\theta(X|Z)\right)\right] - \mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z)\right)
\end{aligned}$$

Evidence Lower Bound (ELBO)

Since true posterior $p_\theta(Z|X)$ is often unknown, KL term is intractable

ELBO:

$$\begin{aligned}\mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)\right] &= \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X|Z)p_\theta(Z)}{q_\phi(Z|X)}\right)\right] \\ &= \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(p_\theta(X|Z)\right)\right] + \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(Z)}{q_\phi(Z|X)}\right)\right] \\ &= \underbrace{-\mathbb{E}_{q_\phi(Z|X)}\left[-\log\left(p_\theta(X|Z)\right)\right]}_{\text{Reconstruction Error/Loss}} - \underbrace{\mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z)\right)}_{\text{Regularizer}}\end{aligned}$$

Evidence Lower Bound (ELBO)

Since true posterior $p_\theta(Z|X)$ is often unknown, KL term is intractable

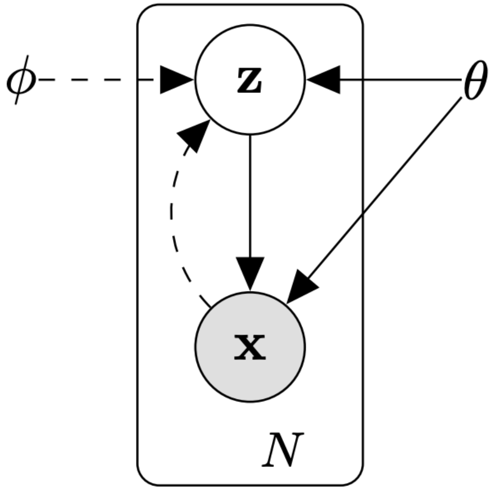
ELBO:

$$\begin{aligned}
\mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)\right] &= \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(X|Z)p_\theta(Z)}{q_\phi(Z|X)}\right)\right] \\
&= \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(p_\theta(X|Z)\right)\right] + \mathbb{E}_{q_\phi(Z|X)}\left[\log\left(\frac{p_\theta(Z)}{q_\phi(Z|X)}\right)\right] \\
&= \underbrace{-\mathbb{E}_{q_\phi(Z|X)}\left[-\log\left(p_\theta(X|Z)\right)\right]}_{\text{Reconstruction Error/Loss}} - \underbrace{\mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z)\right)}_{\text{Regularizer}}
\end{aligned}$$

Outline

- Autoencoders
  - Motivation & Overview
  - Linear Autoencoders & PCA
  - Deep Autoencoders
- Denoising Autoencoders
- Variational Autoencoders
  - Motivation & Overview
  - Evidence Lower Bound (ELBO)
  - **Models**
  - Amortized Inference
  - Reparameterization Trick
- Graph Variational Autoencoders

Variational Autoencoders

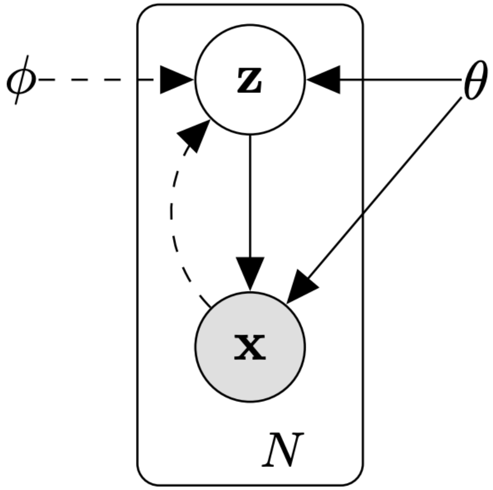


Encoder: $q_\phi(Z|X)$

Decoder: $p_\theta(X|Z)$

Prior: $p_\theta(Z)$

Variational Autoencoders



Since we typically use continuous latent variable Z, Gaussian distribution is a natural choice for the encoder:

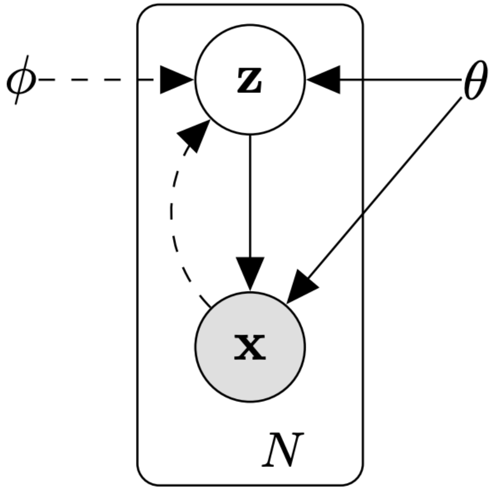
$$q_\phi(Z|X) = \mathcal{N}(Z|\mu, \sigma^2 I)$$
$$\mu = \text{EncoderNetwork}_\phi(X)$$
$$\log \sigma^2 = \text{EncoderNetwork}_\phi(X)$$

Encoder: $q_\phi(Z|X)$

Decoder: $p_\theta(X|Z)$

Prior: $p_\theta(Z)$

Variational Autoencoders



Since we typically use continuous latent variable Z, Gaussian distribution is a natural choice for the encoder:

$$q_\phi(Z|X) = \mathcal{N}(Z|\mu, \sigma^2 I)$$
$$\mu = \text{EncoderNetwork}_\phi(X)$$
$$\log \sigma^2 = \text{EncoderNetwork}_\phi(X)$$

Similarly, Gaussian distribution is often adopted for the decoder:

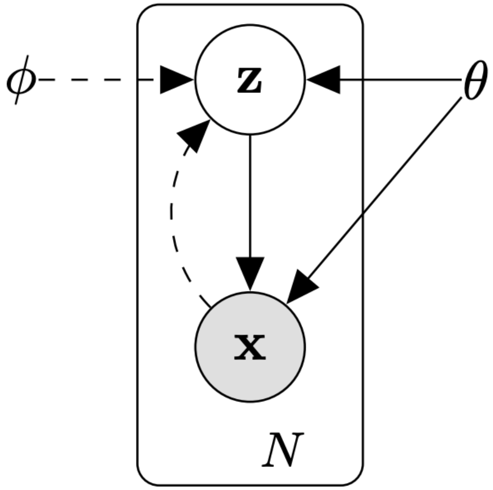
$$p_\theta(X|Z) = \mathcal{N}(X|\tilde{\mu}, \tilde{\sigma}^2 I)$$
$$\tilde{\mu} = \text{DecoderNetwork}_\theta(Z)$$
$$\log \tilde{\sigma}^2 = \text{DecoderNetwork}_\theta(Z)$$

Encoder: $q_\phi(Z|X)$

Decoder: $p_\theta(X|Z)$

Prior: $p_\theta(Z)$

Variational Autoencoders



Encoder: $q_\phi(Z|X)$

Decoder: $p_\theta(X|Z)$

Prior: $p_\theta(Z)$

Since we typically use continuous latent variable Z, Gaussian distribution is a natural choice for the encoder:

$$q_\phi(Z|X) = \mathcal{N}(Z|\mu, \sigma^2 I)$$
$$\mu = \text{EncoderNetwork}_\phi(X)$$
$$\log \sigma^2 = \text{EncoderNetwork}_\phi(X)$$

Similarly, Gaussian distribution is often adopted for the decoder:

$$p_\theta(X|Z) = \mathcal{N}(X|\tilde{\mu}, \tilde{\sigma}^2 I)$$
$$\tilde{\mu} = \text{DecoderNetwork}_\theta(Z)$$
$$\log \tilde{\sigma}^2 = \text{DecoderNetwork}_\theta(Z)$$

We often fix the prior as, e.g., standard Normal $p_\theta(Z) = \mathcal{N}(Z|\mathbf{0}, I)$

Variational Autoencoders

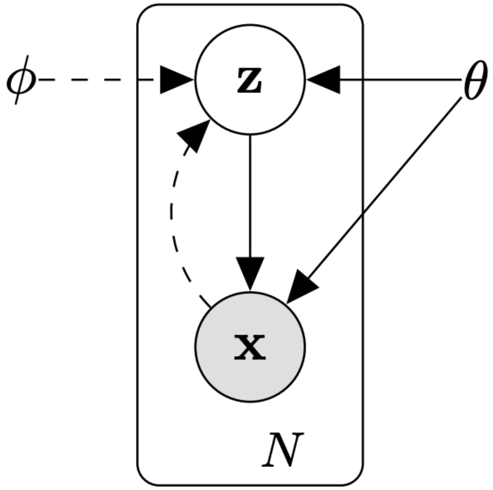
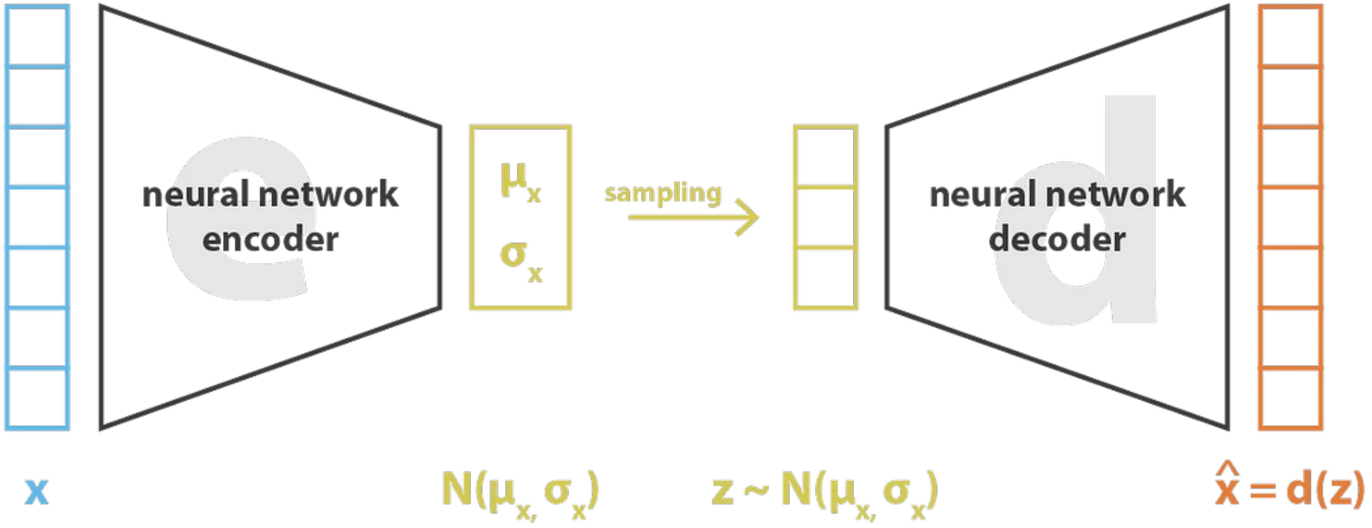


Illustration of VAEs:



Encoder: $q_\phi(Z|X)$

Decoder: $p_\theta(X|Z)$

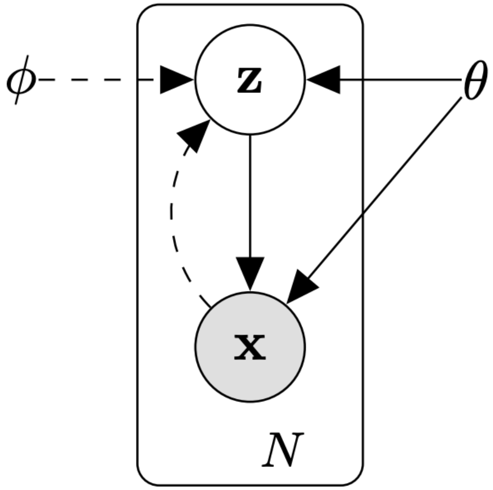
Prior: $p_\theta(Z)$

Image Credit: [6]

Outline

- Autoencoders
  - Motivation & Overview
  - Linear Autoencoders & PCA
  - Deep Autoencoders
- Denoising Autoencoders
- Variational Autoencoders
  - Motivation & Overview
  - Evidence Lower Bound (ELBO)
  - Models
  - **Amortized Inference**
  - Reparameterization Trick
- Graph Variational Autoencoders

Amortized Variational Inference



Encoder: $q_\phi(Z|X)$

Decoder: $p_\theta(X|Z)$

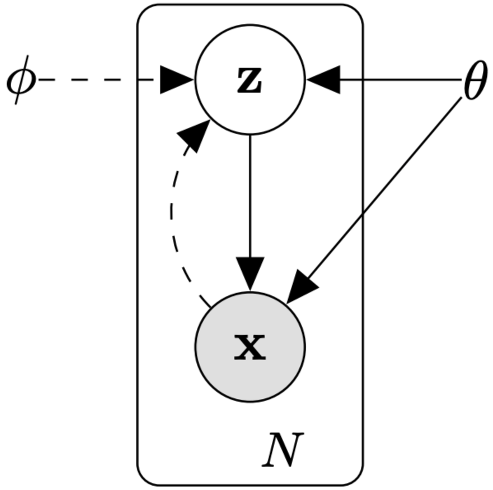
Prior: $p_\theta(Z)$

Since we typically use continuous latent variable Z, Gaussian distribution is a natural choice for the encoder:

$$q_\phi(Z|X) = \mathcal{N}(Z|\mu, \sigma^2 I)$$
$$\mu = \text{EncoderNetwork}_\phi(X)$$
$$\log \sigma^2 = \text{EncoderNetwork}_\phi(X)$$

Encoder is amortized: every X shares the same set of parameters ϕ

Amortized Variational Inference



Since we typically use continuous latent variable Z, Gaussian distribution is a natural choice for the encoder:

$$
\begin{aligned}
q_\phi(Z|X) &= \mathcal{N}(Z|\mu, \sigma^2 I) \\
\mu &= \text{EncoderNetwork}_\phi(X) \\
\log \sigma^2 &= \text{EncoderNetwork}_\phi(X)
\end{aligned}
$$

Encoder is amortized: every X shares the same set of parameters ϕ

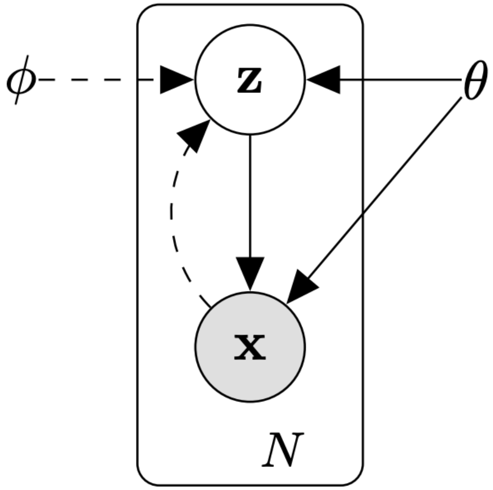
Encoder: $q_\phi(Z|X)$

Decoder: $p_\theta(X|Z)$

Prior: $p_\theta(Z)$

We thus only need to optimize ELBO over one set of parameters ϕ, whereas in traditional variational inference (VI) one needs to find the optimal variational distribution per X

Amortized Variational Inference



Encoder: $q_\phi(Z|X)$

Decoder: $p_\theta(X|Z)$

Prior: $p_\theta(Z)$

Since we typically use continuous latent variable Z, Gaussian distribution is a natural choice for the encoder:

$$
\begin{aligned}
q_\phi(Z|X) &= \mathcal{N}(Z|\mu, \sigma^2 I) \\
\mu &= \text{EncoderNetwork}_\phi(X) \\
\log \sigma^2 &= \text{EncoderNetwork}_\phi(X)
\end{aligned}
$$

Encoder is amortized: every X shares the same set of parameters ϕ

We thus only need to optimize ELBO over one set of parameters ϕ, whereas in traditional variational inference (VI) one needs to find the optimal variational distribution per X

Different X still have different encoder distributions $q_\phi(Z|X)$

Outline

- Autoencoders
  - Motivation & Overview
  - Linear Autoencoders & PCA
  - Deep Autoencoders
- Denoising Autoencoders
- Variational Autoencoders
  - Motivation & Overview
  - Evidence Lower Bound (ELBO)
  - Models
  - Amortized Inference
  - **Reparameterization Trick**
- Graph Variational Autoencoders

Reparameterization Trick

Negative ELBO: $$\mathcal{L}(\phi, \theta) = \underbrace{\mathbb{E}_{q_\phi(Z|X)}\left[-\log\left(p_\theta(X|Z)\right)\right]}_{\text{Reconstruction Error/Loss}} + \underbrace{\mathrm{KL}\left(q_\phi(Z|X) \| p_\theta(Z)\right)}_{\text{Regularizer}}$$

We want to minimize negative ELBO w.r.t. encoder parameters ϕ and decoder parameters θ

Reparameterization Trick

Negative ELBO: $\mathcal{L}(\phi, \theta) = \underbrace{\mathbb{E}_{q_\phi(Z|X)} \left[-\log \left(p_\theta(X|Z) \right) \right]}_{\text{Reconstruction Error/Loss}} + \underbrace{\mathrm{KL} \left(q_\phi(Z|X) \| p_\theta(Z) \right)}_{\text{Regularizer}}$

We want to minimize negative ELBO w.r.t. encoder parameters ϕ and decoder parameters θ

The expectation in reconstruction loss is intractable and often approximated by Monte Carlo estimation

Reparameterization Trick

Negative ELBO: $\mathcal{L}(\phi, \theta) = \underbrace{\mathbb{E}_{q_\phi(Z|X)}\left[-\log\left(p_\theta(X|Z)\right)\right]}_{\text{Reconstruction Error/Loss}} + \underbrace{\mathrm{KL}\left(q_\phi(Z|X) \| p_\theta(Z)\right)}_{\text{Regularizer}}$

We want to minimize negative ELBO w.r.t. encoder parameters ϕ and decoder parameters θ

The expectation in reconstruction loss is intractable and often approximated by Monte Carlo estimation

Once we draw samples of Z, we can get the Monte Carlo gradient of reconstruction loss w.r.t. θ via backpropagation

Reparameterization Trick

Negative ELBO: $$\mathcal{L}(\phi, \theta) = \mathbb{E}_{q_\phi(Z|X)}\left[-\log\left(p_\theta(X|Z)\right)\right] + \mathrm{KL}\left(q_\phi(Z|X) \| p_\theta(Z)\right)$$

$$\approx -\frac{1}{N} \sum_{\substack{i=1 \\ Z_i \sim q_\phi(Z|X)}}^{N} \log\left(p_\theta(X|Z_i)\right) + \mathrm{KL}\left(q_\phi(Z|X) \| p_\theta(Z)\right)$$

Reparameterization Trick

Negative ELBO:
$$\mathcal{L}(\phi,\theta) = \mathbb{E}_{q_\phi(Z|X)}\left[-\log\left(p_\theta(X|Z)\right)\right] + \mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z)\right)$$
$$\approx -\frac{1}{N}\sum_{\substack{i=1\\ Z_i\sim q_\phi(Z|X)}}^{N} \log\left(p_\theta(X|Z_i)\right) + \mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z)\right)$$

Gradient of the decoder (assuming prior is not learnable for simplicity) [7]:

$$\frac{\partial\mathcal{L}(\phi,\theta)}{\partial\theta} = \mathbb{E}_{q_\phi(Z|X)}\left[-\frac{\partial\log\left(p_\theta(X|Z)\right)}{\partial\theta}\right]$$
$$\approx -\frac{1}{N}\sum_{\substack{i=1\\ Z_i\sim q_\phi(Z|X)}}^{N} \frac{\partial\log\left(p_\theta(X|Z_i)\right)}{\partial\theta}$$

Reparameterization Trick

Negative ELBO: $\mathcal{L}(\phi, \theta) = \underbrace{\mathbb{E}_{q_\phi(Z|X)}\left[-\log\left(p_\theta(X|Z)\right)\right]}_{\text{Reconstruction Error/Loss}} + \underbrace{\mathrm{KL}\left(q_\phi(Z|X) \| p_\theta(Z)\right)}_{\text{Regularizer}}$

We want to minimize negative ELBO w.r.t. encoder parameters ϕ and decoder parameters θ

The expectation in reconstruction loss is intractable and often approximated by Monte Carlo estimation

Once we draw samples of Z, we can get the Monte Carlo gradient of reconstruction loss w.r.t. θ via backpropagation

We will use *reparameterization trick (a.k.a. pathwise derivatives)* to equivalently rewrite the expectation in reconstruction loss so that the Monte Carlo gradient w.r.t. ϕ has a low variance.

Reparameterization Trick

For any function f, we have

$$
\begin{aligned}
\mathbb{E}_{\mathcal{N}(Z|\mu,\sigma^2 I)}\left[f(Z)\right] &= \int \frac{1}{\sqrt{(2\pi)^m \prod_i \sigma_i}} \exp(-\frac{1}{2}\left\|\frac{Z-\mu}{\sigma}\right\|^2) f(Z) dZ \\
&= \int \frac{1}{\sqrt{(2\pi)^m \prod_i \sigma_i}} \exp(-\frac{1}{2}\left\|\frac{\mu+\sigma\epsilon-\mu}{\sigma}\right\|^2) f(\mu+\sigma\epsilon) d\left(\mu+\sigma\epsilon\right) \\
&= \int \frac{1}{\sqrt{(2\pi)^m}} \exp(-\frac{1}{2}\left\|\epsilon\right\|^2) f(\mu+\sigma\epsilon) d\epsilon \\
&= \mathbb{E}_{\mathcal{N}(\epsilon|0,I)}\left[f(\mu+\sigma\epsilon)\right]
\end{aligned}
$$

Change of Variable

Reparameterization Trick

For any function f, we have

$$\begin{aligned}
\mathbb{E}_{\mathcal{N}(Z|\mu,\sigma^2 I)}\left[f(Z)\right] &= \int \frac{1}{\sqrt{(2\pi)^m \prod_i \sigma_i}} \exp(-\frac{1}{2}\left\|\frac{Z-\mu}{\sigma}\right\|^2) f(Z) dZ \\
&= \int \frac{1}{\sqrt{(2\pi)^m \prod_i \sigma_i}} \exp(-\frac{1}{2}\left\|\frac{\mu+\sigma\epsilon-\mu}{\sigma}\right\|^2) f(\mu+\sigma\epsilon) d\left(\mu+\sigma\epsilon\right) \\
&= \int \frac{1}{\sqrt{(2\pi)^m}} \exp(-\frac{1}{2}\left\|\epsilon\right\|^2) f(\mu+\sigma\epsilon) d\epsilon \\
&= \mathbb{E}_{\mathcal{N}(\epsilon|0,I)}\left[f(\mu+\sigma\epsilon)\right]
\end{aligned}$$

Change of Variable

Therefore,

$$\begin{aligned}
\mathcal{L}(\phi,\theta) &= \mathbb{E}_{q_\phi(Z|X)}\left[-\log\left(p_\theta(X|Z)\right)\right] + \mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z)\right) \\
&= \mathbb{E}_{\mathcal{N}(\epsilon|0,I)}\left[-\log\left(p_\theta(X|\mu_\phi(X)+\sigma_\phi(X)\epsilon)\right)\right] + \mathrm{KL}\left(q_\phi(Z|X)\|p_\theta(Z)\right)
\end{aligned}$$

Reparameterization Trick

In original VAE,

$$q_\phi(Z|X) = \mathcal{N}(Z|\mu_\phi(X), \sigma_\phi(X)^2 I)$$
$$p_\theta(Z) = \mathcal{N}(X|0, I)$$

Reparameterization Trick

In original VAE,

$$q_\phi(Z|X) = \mathcal{N}(Z|\mu_\phi(X), \sigma_\phi(X)^2 I)$$
$$p_\theta(Z) = \mathcal{N}(X|0, I)$$

Using Gaussian integrals, we have

$$\mathrm{KL}\left(q_\phi(Z|X) \| p_\theta(Z)\right) = \frac{1}{2}\left(\mu_\phi(X)^\top \mu_\phi(X) + \sigma_\phi(X)^\top \sigma_\phi(X)\right) - \frac{1}{2}\sum_{i=1}^{m} \log \sigma_i^2 - \frac{m}{2}$$

where

$$\sigma_\phi(X) = [\sigma_1, \sigma_2, \cdots, \sigma_m]^\top$$

Reparameterization Trick

Therefore, in original VAE, we have

$$\begin{aligned}\mathcal{L}(\phi, \theta) = \ & \mathbb{E}_{\mathcal{N}(\epsilon|0,I)}\left[-\log\left(p_\theta(X|\mu_\phi(X) + \sigma_\phi(X)\epsilon)\right)\right] \\ & + \frac{1}{2}\left(\mu_\phi(X)^\top \mu_\phi(X) + \sigma_\phi(X)^\top \sigma_\phi(X)\right) - \frac{1}{2}\sum_{i=1}^{m} \log \sigma_i^2 - \frac{m}{2}\end{aligned}$$

Reparameterization Trick

Therefore, in original VAE, we have

$$\begin{aligned}\mathcal{L}(\phi,\theta) = \ & \mathbb{E}_{\mathcal{N}(\epsilon|0,I)}\left[-\log\left(p_\theta(X|\mu_\phi(X)+\sigma_\phi(X)\epsilon)\right)\right] \\ & + \frac{1}{2}\left(\mu_\phi(X)^\top\mu_\phi(X)+\sigma_\phi(X)^\top\sigma_\phi(X)\right) - \frac{1}{2}\sum_{i=1}^{m}\log\sigma_i^2 - \frac{m}{2}\end{aligned}$$

We only need *reparameterization trick* and *Monte Carlo estimation* in the first term

$$\begin{aligned}\mathcal{L}(\phi,\theta) \approx \ & -\sum_{i=1,\epsilon_i\sim\mathcal{N}(\epsilon|0,I)}^{N}\log\left(p_\theta(X|\mu_\phi(X)+\sigma_\phi(X)\epsilon_i)\right) \\ & + \frac{1}{2}\left(\mu_\phi(X)^\top\mu_\phi(X)+\sigma_\phi(X)^\top\sigma_\phi(X)\right) - \frac{1}{2}\sum_{i=1}^{m}\log\sigma_i^2 - \frac{m}{2}\end{aligned}$$

Reparameterization Trick

Therefore, in original VAE, we have

$$
\begin{aligned}
\mathcal{L}(\phi,\theta) = \ & \mathbb{E}_{\mathcal{N}(\epsilon|0,I)}\left[-\log\left(p_\theta(X|\mu_\phi(X)+\sigma_\phi(X)\epsilon)\right)\right] \\
& + \frac{1}{2}\left(\mu_\phi(X)^\top\mu_\phi(X)+\sigma_\phi(X)^\top\sigma_\phi(X)\right) - \frac{1}{2}\sum_{i=1}^{m}\log\sigma_i^2 - \frac{m}{2}
\end{aligned}
$$

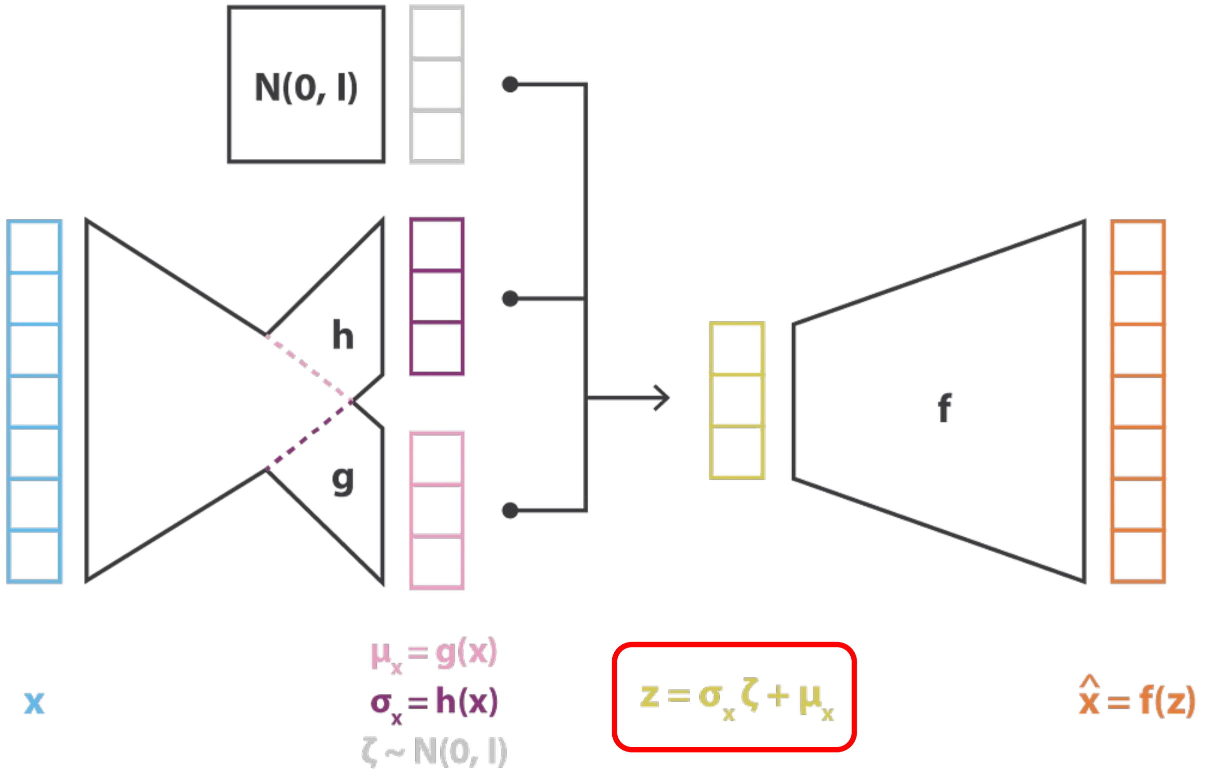
We only need *reparameterization trick* and *Monte Carlo estimation* in the first term

$$
\begin{aligned}
\mathcal{L}(\phi,\theta) \approx \ & -\sum_{i=1,\epsilon_i\sim\mathcal{N}(\epsilon|0,I)}^{N}\log\left(p_\theta(X|\mu_\phi(X)+\sigma_\phi(X)\epsilon_i)\right) \\
& + \frac{1}{2}\left(\mu_\phi(X)^\top\mu_\phi(X)+\sigma_\phi(X)^\top\sigma_\phi(X)\right) - \frac{1}{2}\sum_{i=1}^{m}\log\sigma_i^2 - \frac{m}{2}
\end{aligned}
$$

Now we can get the gradient directly!

Reparameterization Trick

In the illustration of VAEs, the latent variable is *reparameterized* as below:

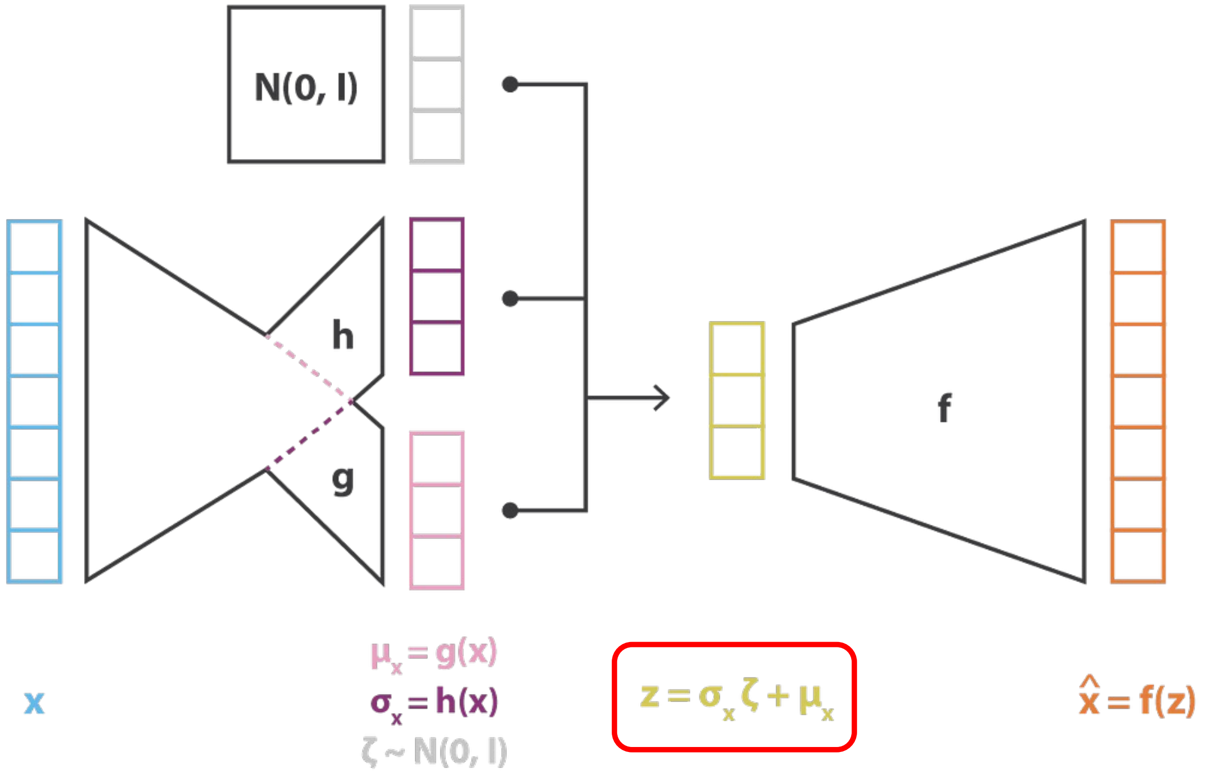


$$\text{loss} = C\,||\,x - \hat{x}\,||^2 + \mathrm{KL}[\,N(\mu_x, \sigma_x), N(0, I)\,] = C\,||\,x - f(z)\,||^2 + \mathrm{KL}[\,N(g(x), h(x)), N(0, I)\,]$$

Image Credit: [6]

Reparameterization Trick

In the illustration of VAEs, the latent variable is *reparameterized* as below:



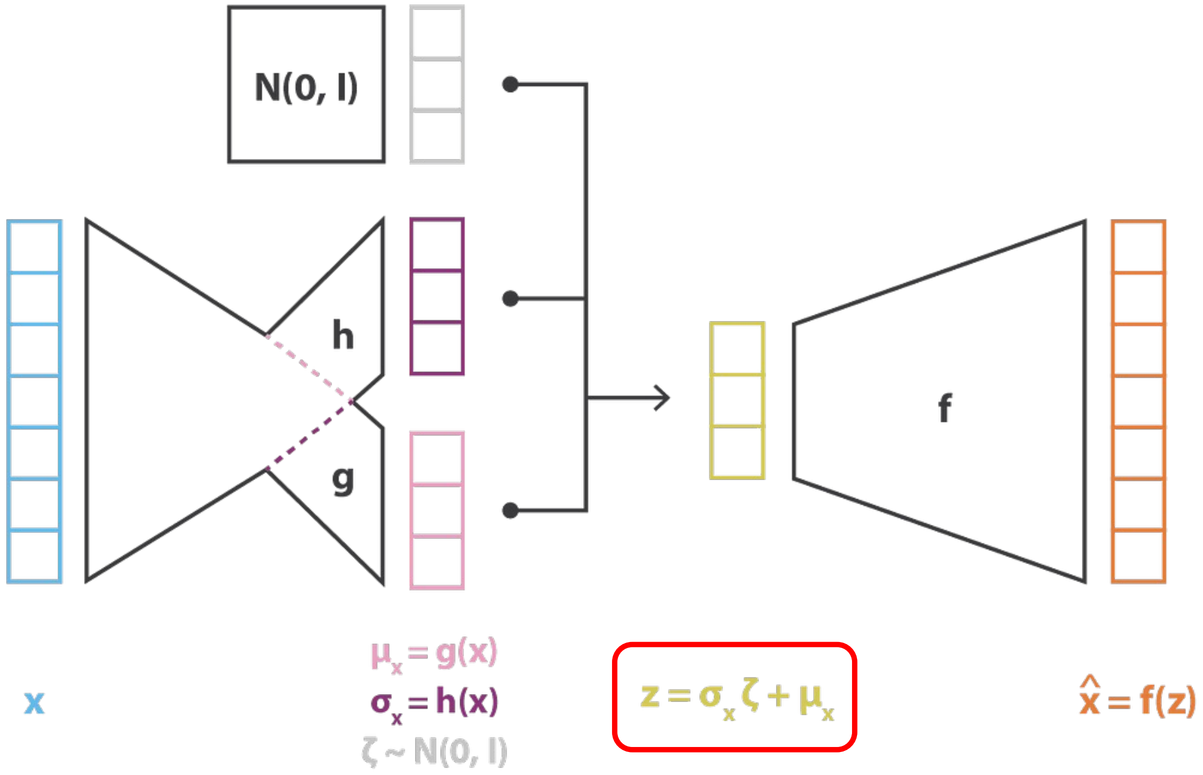
$$\text{loss} = C \, || x - \hat{x} ||^2 + KL[\, N(\mu_x, \sigma_x), N(0, I) \,] = C \, || x - f(z) ||^2 + KL[\, N(g(x), h(x)), N(0, I) \,]$$

What if we have discrete latent variables in VAEs?

Image Credit: [6]

Reparameterization Trick

In the illustration of VAEs, the latent variable is *reparameterized* as below:



$$\text{loss} = C\,\|x - \hat{x}\|^2 + \text{KL}[\,N(\mu_x, \sigma_x), N(0, I)\,] = C\,\|x - f(z)\|^2 + \text{KL}[\,N(g(x), h(x)), N(0, I)\,]$$

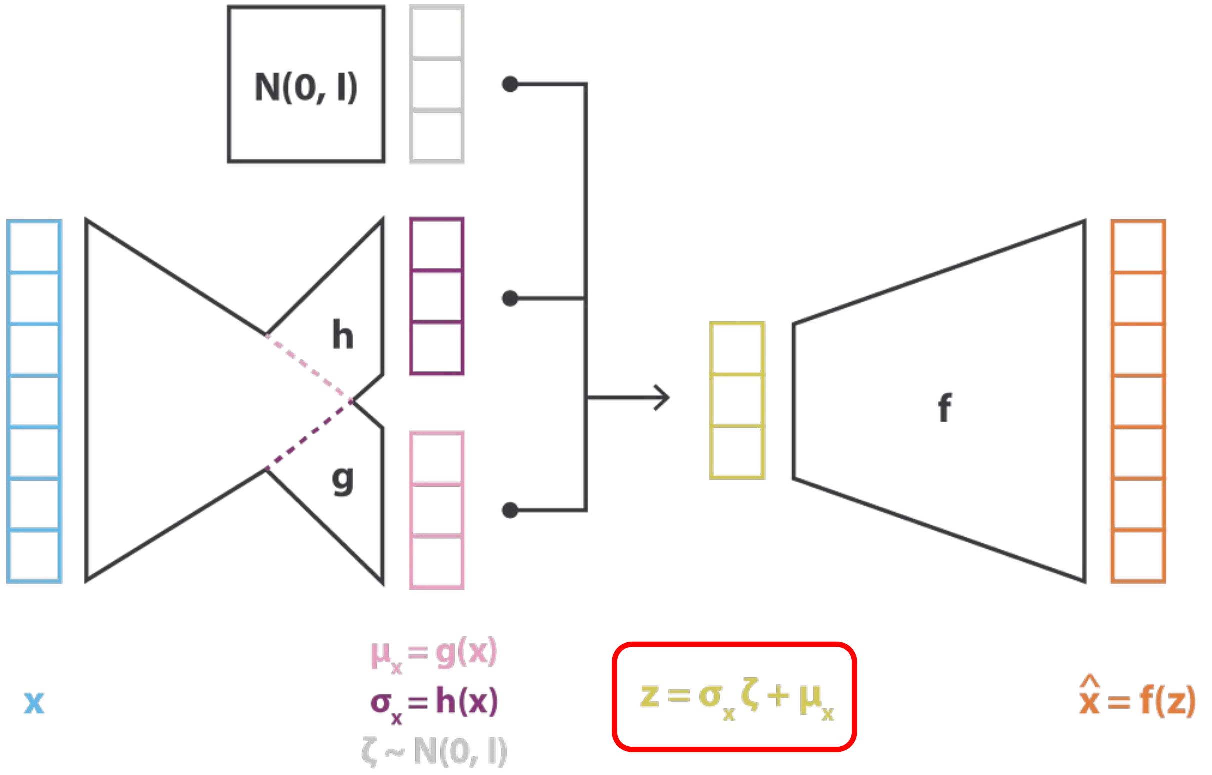
What if we have discrete latent variables in VAEs?

Reparameterization trick does not work exactly since sampling path is non-differentiable!

Image Credit: [6]

Reparameterization Trick

In the illustration of VAEs, the latent variable is *reparameterized* as below:



$$\text{loss} = C\,\|x - \hat{x}\|^2 + \mathrm{KL}[\,N(\mu_x, \sigma_x), N(0, I)\,] = C\,\|x - f(z)\|^2 + \mathrm{KL}[\,N(g(x), h(x)), N(0, I)\,]$$

What if we have discrete latent variables in VAEs?

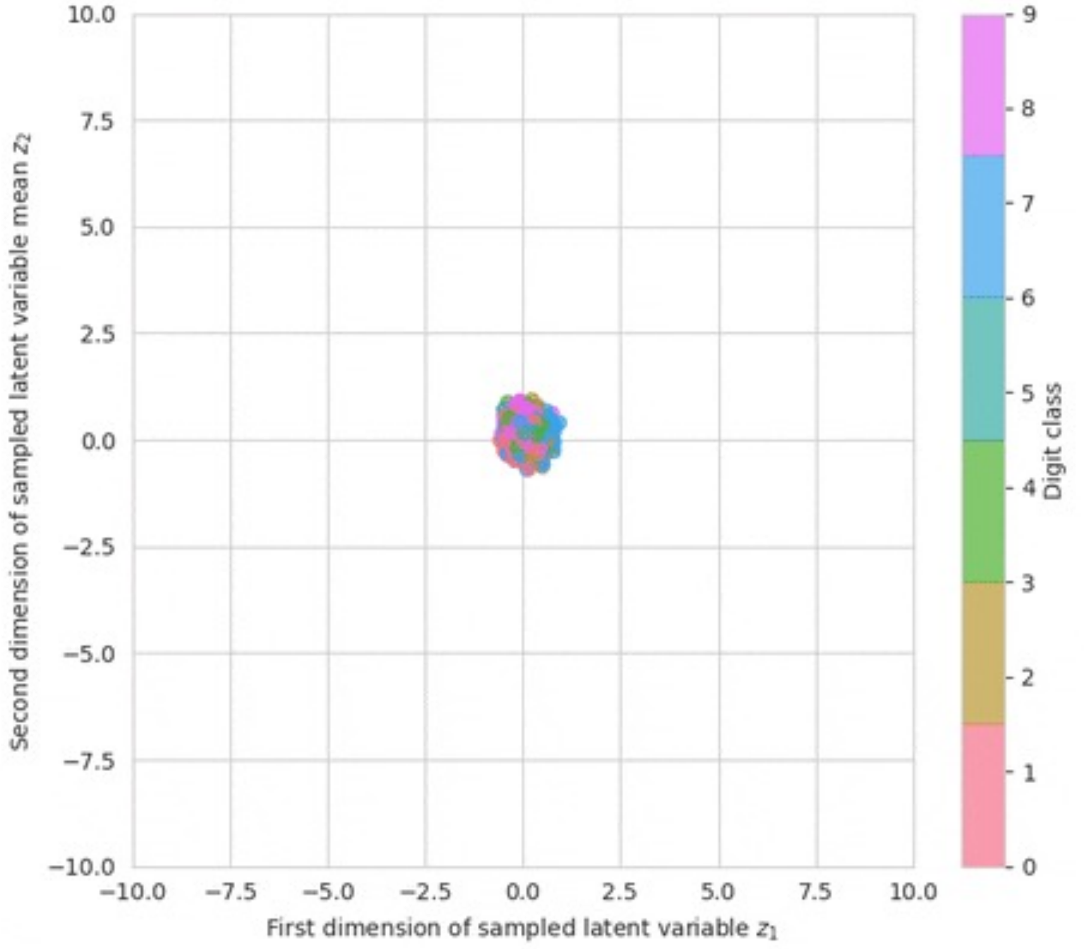
Reparameterization trick does not work exactly since sampling path is non-differentiable!

We need *Monte Carlo gradient estimation* methods which are covered by a separate lecture.

Image Credit: [6]

VAEs on MNIST

Visualize $Z \sim q_\phi(Z|X)$ during training:



Visualize $X \sim p_\theta(X|Z)$ during training:

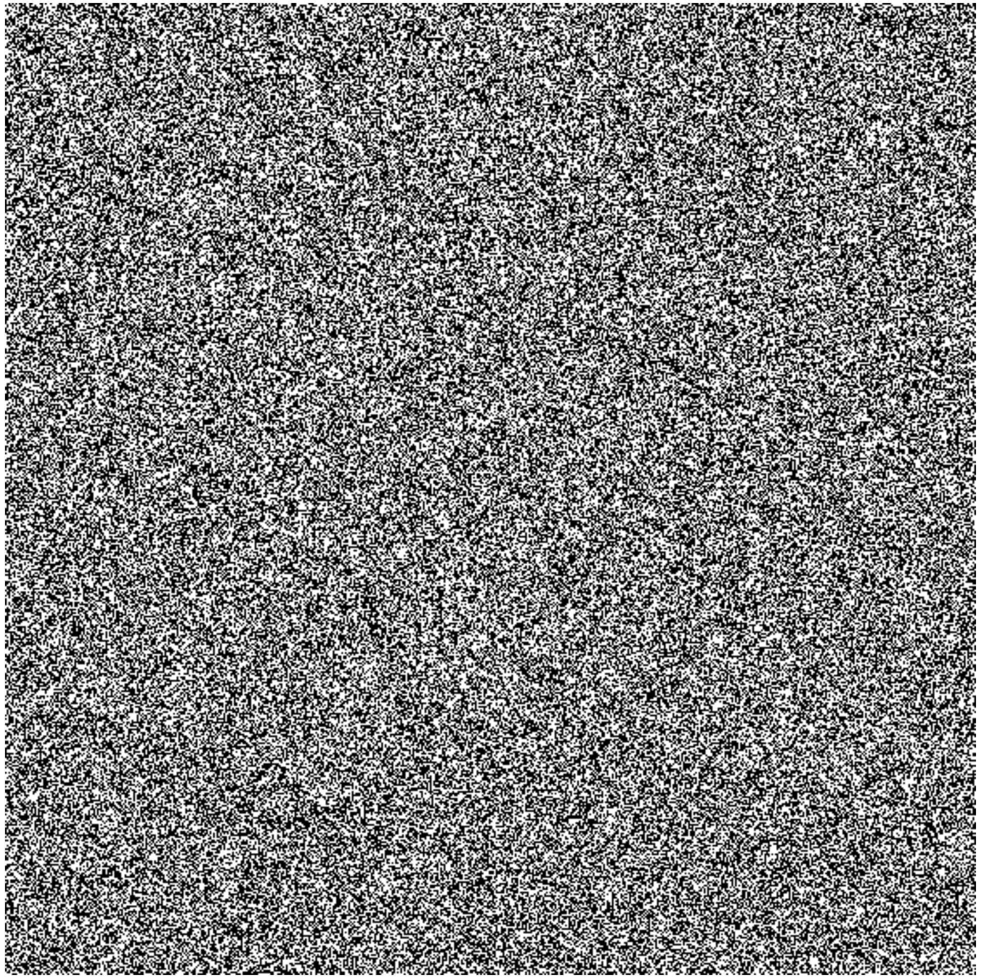


Image Credit: [9]

Outline

- Autoencoders
  - Motivation & Overview
  - Linear Autoencoders & PCA
  - Deep Autoencoders
- Denoising Autoencoders
- Variational Autoencoders
  - Motivation & Overview
  - Evidence Lower Bound (ELBO)
  - Models
  - Amortized Inference
  - Reparameterization Trick
- **Graph Variational Autoencoders**

Graph Variation Autoencoders

Graph VAEs [10, 11] generalize VAEs to graph structured data:

Node feature:	$X \in \mathbb{R}^{n \times d}$
Node latent variables:	$Z \in \mathbb{R}^{n \times m}$
Adjacency matrix:	$A \in \mathbb{R}^{n \times n}$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Encoder:

$$
\begin{aligned}
q_\phi(Z|X, A) &= \prod_i q_\phi(Z_i|X, A) \\
q_\phi(Z_i|X, A) &= \mathcal{N}(Z_i|\mu_i, \sigma_i^2 I) \\
H &= \mathrm{GNN}_\phi(X, A) \\
\mu_i, \log \sigma_i^2 &= \mathrm{Readout}_\phi(H)
\end{aligned}
$$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Encoder:

$$
\begin{aligned}
q_\phi(Z|X,A) &= \prod_i q_\phi(Z_i|X,A) \\
q_\phi(Z_i|X,A) &= \mathcal{N}(Z_i|\mu_i, \sigma_i^2 I) \\
H &= \mathrm{GNN}_\phi(X,A) \\
\mu_i, \log\sigma_i^2 &= \mathrm{Readout}_\phi(H)
\end{aligned}
$$

Prior:

$$
p(Z) = \prod_i p(Z_i) = \prod_i \mathcal{N}(Z_i|0, I)
$$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Decoder:

$$p_\theta(X, A|Z) = p_\theta(A|Z)p_\theta(X|A, Z)$$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Decoder: $$p_\theta(X, A|Z) = p_\theta(A|Z)p_\theta(X|A, Z)$$

Adjacency Matrix Decoder:
$$p_\theta(A|Z) = \prod_i \prod_j p_\theta(A_{ij}|Z)$$
$$H = \mathrm{MLP}(Z)$$
$$p_\theta(A_{ij} = 1|Z_i, Z_j) = \sigma(H_i^\top H_j)$$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Decoder:

$$p_\theta(X, A|Z) = p_\theta(A|Z)p_\theta(X|A, Z)$$

Adjacency Matrix Decoder:

$$p_\theta(A|Z) = \prod_i \prod_j p_\theta(A_{ij}|Z)$$

$$H = \mathrm{MLP}(Z)$$

$$p_\theta(A_{ij} = 1|Z_i, Z_j) = \sigma(H_i^\top H_j)$$

Node Feature Decoder:

$$p_\theta(X|A, Z) = \prod_i p_\theta(X_i|A, Z)$$

$$p_\theta(X_i|A, Z) = \mathcal{N}(X_i|\tilde{\mu}_i, \tilde{\sigma}_i^2 I)$$

$$\tilde{H} = \mathrm{GNN}_\theta(Z, A)$$

$$\tilde{\mu}_i, \log \tilde{\sigma}_i^2 = \mathrm{Readout}_\theta(\tilde{H})$$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Learning:

$$\begin{aligned}\log p_\theta(X, A) &\geq \mathrm{ELBO} \\ &= -\mathbb{E}_{q_\phi(Z|A,X)}\left[-\log\left(p_\theta(X, A|Z)\right)\right] - \mathrm{KL}\left(q_\phi(Z|X, A) \| p_\theta(Z)\right)\end{aligned}$$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Learning:

$$\begin{aligned}\log p_\theta(X, A) &\geq \text{ELBO} \\ &= -\mathbb{E}_{q_\phi(Z|A,X)}\left[-\log\left(p_\theta(X, A|Z)\right)\right] - \text{KL}\left(q_\phi(Z|X, A) \| p_\theta(Z)\right)\end{aligned}$$

Are we done?

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Learning:

$$
\begin{aligned}
\log p_\theta(X, A) &\geq \text{ELBO} \\
&= -\mathbb{E}_{q_\phi(Z|A,X)}\left[-\log\left(p_\theta(X, A|Z)\right)\right] - \text{KL}\left(q_\phi(Z|X, A)\|p_\theta(Z)\right)
\end{aligned}
$$

Are we done?

No! We hope ELBO is permutation invariant!

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Learning:

$$
\begin{aligned}
\log p_\theta(X, A) &\geq \text{ELBO} \\
&= -\mathbb{E}_{q_\phi(Z|A,X)}\left[-\log\left(p_\theta(X, A|Z)\right)\right] - \text{KL}\left(q_\phi(Z|X, A) \| p_\theta(Z)\right)
\end{aligned}
$$

Recall we use GNN as the encoder and the encoder is conditional independent

$$
\begin{aligned}
q_\phi(Z|X, A) &= \prod_i q_\phi(Z_i|X, A) \\
q_\phi(Z_i|X, A) &= \mathcal{N}(Z_i|\mu_i, \sigma_i^2 I) \\
H &= \text{GNN}_\phi(X, A) \\
\mu_i, \log\sigma_i^2 &= \text{Readout}_\phi(H)
\end{aligned}
$$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Learning:

$$\begin{aligned}\log p_\theta(X, A) &\geq \text{ELBO} \\ &= -\mathbb{E}_{q_\phi(Z|A,X)}\left[-\log\left(p_\theta(X, A|Z)\right)\right] - \text{KL}\left(q_\phi(Z|X, A) \| p_\theta(Z)\right)\end{aligned}$$

Recall we use GNN as the encoder and the encoder is conditional independent, we have

$$\begin{aligned}q_\phi(Z|X, A) &= \prod_i q_\phi(Z_i|X, A) \\ q_\phi(Z_i|X, A) &= \mathcal{N}(Z_i|\mu_i, \sigma_i^2 I) \\ H &= \text{GNN}_\phi(X, A) \\ \mu_i, \log \sigma_i^2 &= \text{Readout}_\phi(H)\end{aligned}$$

Encoder is permutation invariant!

$$\forall P \in \Pi \qquad q_\phi(Z|PAP^\top, PX) = q_\phi(Z|A, X)$$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Learning:

$$\log p_\theta(X, A) \geq \text{ELBO}$$
$$= -\mathbb{E}_{q_\phi(Z|A,X)}\left[-\log\left(p_\theta(X, A|Z)\right)\right] - \text{KL}\left(q_\phi(Z|X, A) \| p_\theta(Z)\right)$$

Similarly, recall we use GNN as the decoder and the decoder is conditional independent, we have

Decoder is permutation invariant!

$$\forall P \in \Pi \qquad p_\theta(PAP^\top, PX|PZ) = p_\theta(X, A|Z)$$

Graph Variation Autoencoders

Graph VAEs [10, 11]:

Learning:

$$\log p_\theta(X, A) \geq \text{ELBO}$$
$$= -\mathbb{E}_{q_\phi(Z|A,X)}\left[-\log\left(p_\theta(X, A|Z)\right)\right] - \text{KL}\left(q_\phi(Z|X, A) \| p_\theta(Z)\right)$$

Similarly, recall we use GNN as the decoder and the decoder is conditional independent, we have

Decoder is permutation invariant!

$$\forall P \in \Pi \qquad p_\theta(PAP^\top, PX|PZ) = p_\theta(X, A|Z)$$

And prior is standard multivariate Normal, which is permutation invariant.

Therefore, the ELBO is permutation invariant!

Graph Variation Autoencoders

Graph VAEs [10, 11]:

If you use a permutation invariant encoder or decoder, ELBO is not longer invariant.

How to approximately achieve permutation-invariance?

Graph Variation Autoencoders

Graph VAEs [10, 11]:

If you use a permutation invariant encoder or decoder, ELBO is not longer invariant.

How to approximately achieve permutation-invariance?

- Sample a few random permutations
  (e.g., importance sampling, special permutations from domain knowledge)

$$
\begin{aligned}
\log\left(\sum_{P\in\Pi} p_\theta(PX, PAP^\top)\right) &\geq \log\left(\sum_{P\in S} p_\theta(PX, PAP^\top)\right) \\
&= \log\left(\sum_{P\in S} \exp\left(\log p_\theta(PX, PAP^\top)\right)\right) \\
&\geq \log\left(\sum_{P\in S} \exp\left(\text{ELBO}\right)\right)
\end{aligned}
$$

References

[1] https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

[2] https://www.cs.toronto.edu/~rgrosse/courses/csc321_2017/slides/lec20.pdf

[3] Bao, X., Lucas, J., Sachdeva, S. and Grosse, R.B., 2020. Regularized linear autoencoders recover the principal components, eventually. Advances in Neural Information Processing Systems, 33, pp.6971-6981.

[4] Hinton, G.E. and Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. science, 313(5786), pp.504-507.

[5] Alain, G. and Bengio, Y., 2014. What regularized auto-encoders learn from the data-generating distribution. The Journal of Machine Learning Research, 15(1), pp.3563-3593.

[6] https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

[7] Kingma, D.P. and Welling, M., 2013. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

[8] Rezende, D.J., Mohamed, S. and Wierstra, D., 2014, June. Stochastic backpropagation and approximate inference in deep generative models. In International conference on machine learning (pp. 1278-1286). PMLR.

[9] https://jaan.io/what-is-variational-autoencoder-vae-tutorial/

[10] Kipf, T.N. and Welling, M., 2016. Variational graph auto-encoders. arXiv preprint arXiv:1611.07308.

[11] Simonovsky, M. and Komodakis, N., 2018, October. Graphvae: Towards generation of small graphs using variational autoencoders. In International conference on artificial neural networks (pp. 412-422). Springer, Cham.

Questions?