Chapter 14

Autoencoders

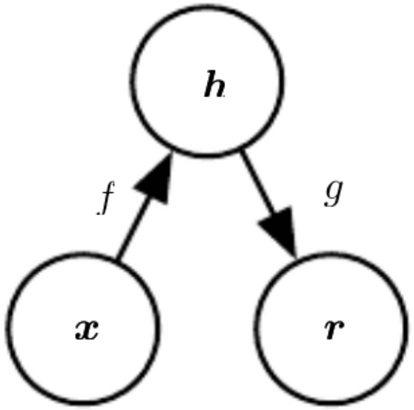
Autoencoders

- A type of neural networks trained to copy approximately its input to its output in the hopes of learning useful features
- The network of an autoencoder may be viewed as containing an encoder and a decoder, specifying deterministic or stochastic mappings

$$\text{Encoder:}\quad \boldsymbol{h} = f(\boldsymbol{x}) \text{ or } p_{\mathsf{model}}(\boldsymbol{h}|\boldsymbol{x})$$

$$\text{Decoder:}\quad \boldsymbol{r} = g(\boldsymbol{h}) \text{ or } p_{\mathsf{model}}(\boldsymbol{x}|\boldsymbol{h})$$

where the hidden layer $\boldsymbol{h}$ describes a code used to represent $\boldsymbol{x}$



- The learning is to minimize a loss function, likely with regularization

$$L(\boldsymbol{x}, g(f(\boldsymbol{x}))) + \Omega(\boldsymbol{h}, \boldsymbol{x})$$

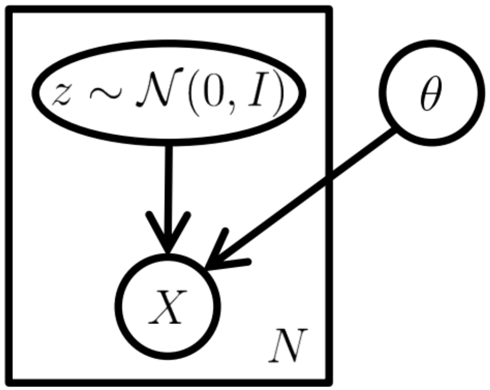
- Most learning techniques for training feedforward networks can apply
- Traditionally, autorencoders were used for dimension reduction
- However, theoretical connections between autoencoders and some modern latent variable models have brought autoencoders to the forefront of generative modeling

Variational Autoencoders (VAE)

- A probabilistic generative model with latent variables that is built on top of end-to-end trainable neural networks

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

$$p(\boldsymbol{x}|\boldsymbol{z}) = \underbrace{p(\boldsymbol{x}; o(\boldsymbol{z}; \boldsymbol{\theta}))}_{\text{Neural Networks}} = \mathcal{N}(\boldsymbol{x}; o(\boldsymbol{z}; \boldsymbol{\theta}), \sigma^2 \boldsymbol{I})$$



Training VAE

- To determine $\boldsymbol{\theta}$, we would intuitively hope to maximize the marginal distribution $p(\boldsymbol{x};\boldsymbol{\theta})$

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

- This however becomes difficult as the integration over $\boldsymbol{z}$ is in general intractable when $p(\boldsymbol{x}|\boldsymbol{z};\boldsymbol{\theta})$ is modeled by a neural network
- To circumvent this difficulty, we recall that

$$\log p(\boldsymbol{X};\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{X}, q, \boldsymbol{\theta}) + \mathsf{KL}(q(\boldsymbol{Z})||p(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}))$$

where

$$\mathcal{L}(\boldsymbol{X}, q, \boldsymbol{\theta}) = \int q(\boldsymbol{Z}) \log p(\boldsymbol{X}, \boldsymbol{Z};\boldsymbol{\theta})d\boldsymbol{Z} - \int q(\boldsymbol{Z}) \log q(\boldsymbol{Z})d\boldsymbol{Z}$$

$$\mathsf{KL}(q(\boldsymbol{Z})||p(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta})) = \int q(\boldsymbol{Z}) \log \frac{q(\boldsymbol{Z})}{p(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta})}d\boldsymbol{Z}$$

- A rearrangement gives

$$\log p(\boldsymbol{X};\boldsymbol{\theta}) - \mathsf{KL}(q(\boldsymbol{Z})||p(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta})) = \mathcal{L}(\boldsymbol{X}, q, \boldsymbol{\theta})$$

- As the equality holds for any choice of $q(\boldsymbol{Z})$, we introduce a distribution $q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')$ modeled by another neural network with parameter $\boldsymbol{\theta}'$ to obtain

$$\log p(\boldsymbol{X};\boldsymbol{\theta}) - \mathsf{KL}(q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')||p(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta})) = \mathcal{L}(\boldsymbol{X}, q, \boldsymbol{\theta})$$

- The right hand side can be spell out as

$$\begin{aligned}\mathcal{L}(\boldsymbol{X}, q, \boldsymbol{\theta}) &= E_{\boldsymbol{Z}\sim q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')} p(\boldsymbol{X}|\boldsymbol{Z};\boldsymbol{\theta}) \quad && + E_{\boldsymbol{Z}\sim q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')} p(\boldsymbol{Z}) \\ & && - E_{\boldsymbol{Z}\sim q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')} q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}') \\ &= E_{\boldsymbol{Z}\sim q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')} p(\boldsymbol{X}|\boldsymbol{Z};\boldsymbol{\theta}) \quad && - \mathsf{KL}(q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')||p(\boldsymbol{Z}))\end{aligned}$$

- Now, instead of directly maximizing the intractable $p(\boldsymbol{X};\boldsymbol{\theta})$, we attempt to maximize

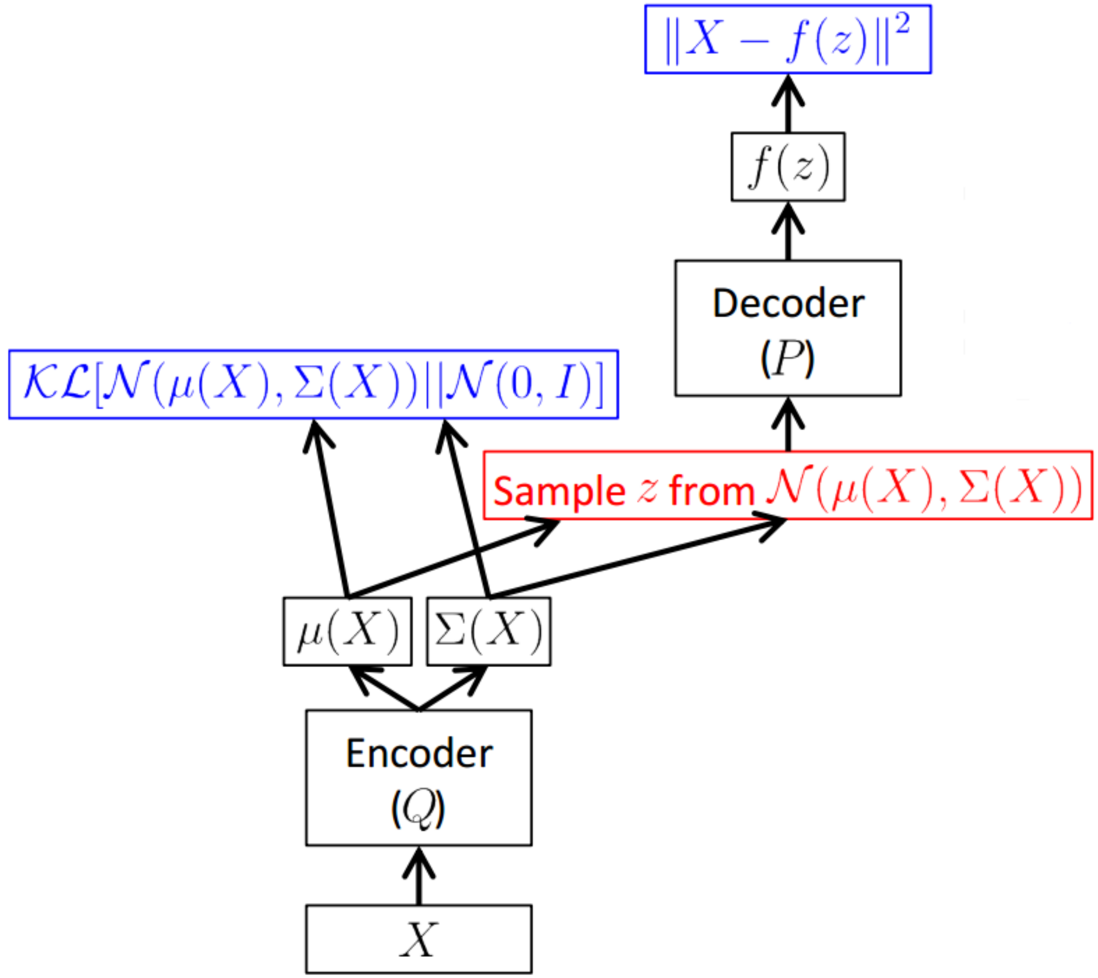
$$\log p(\boldsymbol{X};\boldsymbol{\theta}) - \mathsf{KL}(q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')||p(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}))$$

which amounts to maximizing

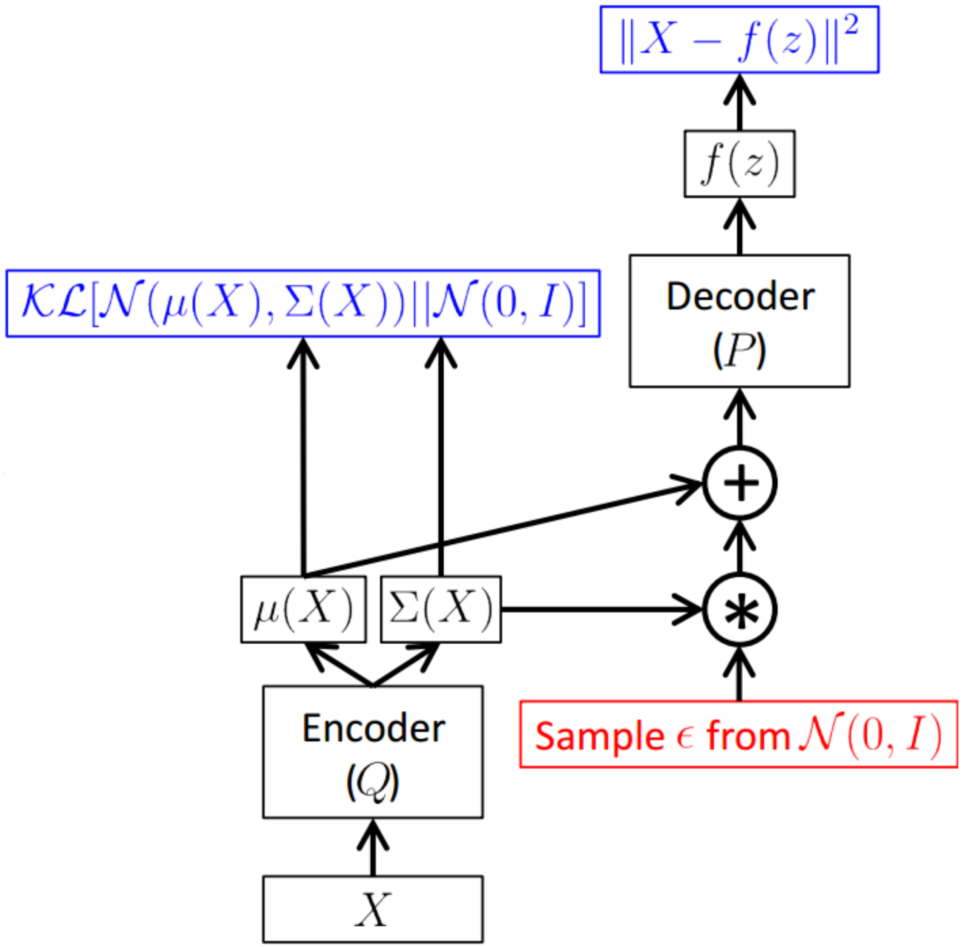
$$E_{\boldsymbol{Z}\sim q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')}p(\boldsymbol{X}|\boldsymbol{Z};\boldsymbol{\theta}) - \mathsf{KL}(q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')||p(\boldsymbol{Z}))$$

- A by-product of this training process is a stochastic encoder

$$q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}') \approx p(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta})$$



$$\underbrace{E_{\boldsymbol{Z} \sim q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')} p(\boldsymbol{X}|\boldsymbol{Z};\boldsymbol{\theta})}_{\text{Sampling needed}} - \mathrm{KL}(q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')||p(\boldsymbol{Z}))$$



$$\underbrace{E_{\boldsymbol{Z}\sim q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')} p(\boldsymbol{X}|\boldsymbol{Z};\boldsymbol{\theta})}_{\text{Re-parameterization for end-to-end training}} \quad -\mathrm{KL}(q(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta}')||p(\boldsymbol{Z}))$$