Autoencoders;
Variational Autoencoders

Neural Networks So Far

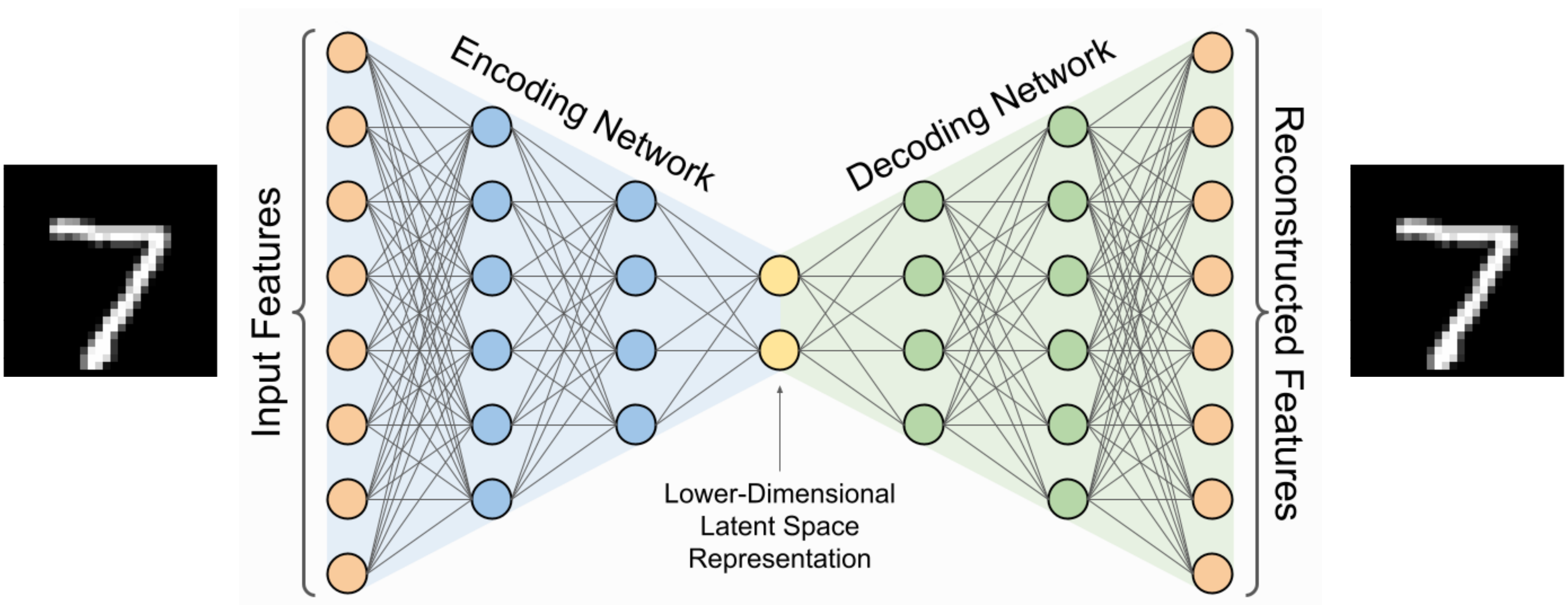
- Multilayer perceptrons
- Convolutional neural networks
- Recurrent neural networks
- Self-attention and Transformers

Today

- Autoencoders
  - dimensionality reduction
  - denoising
  - superresolution
- Variational Autoencoders
  - Generative model: synthesis of new data

Autoencoders

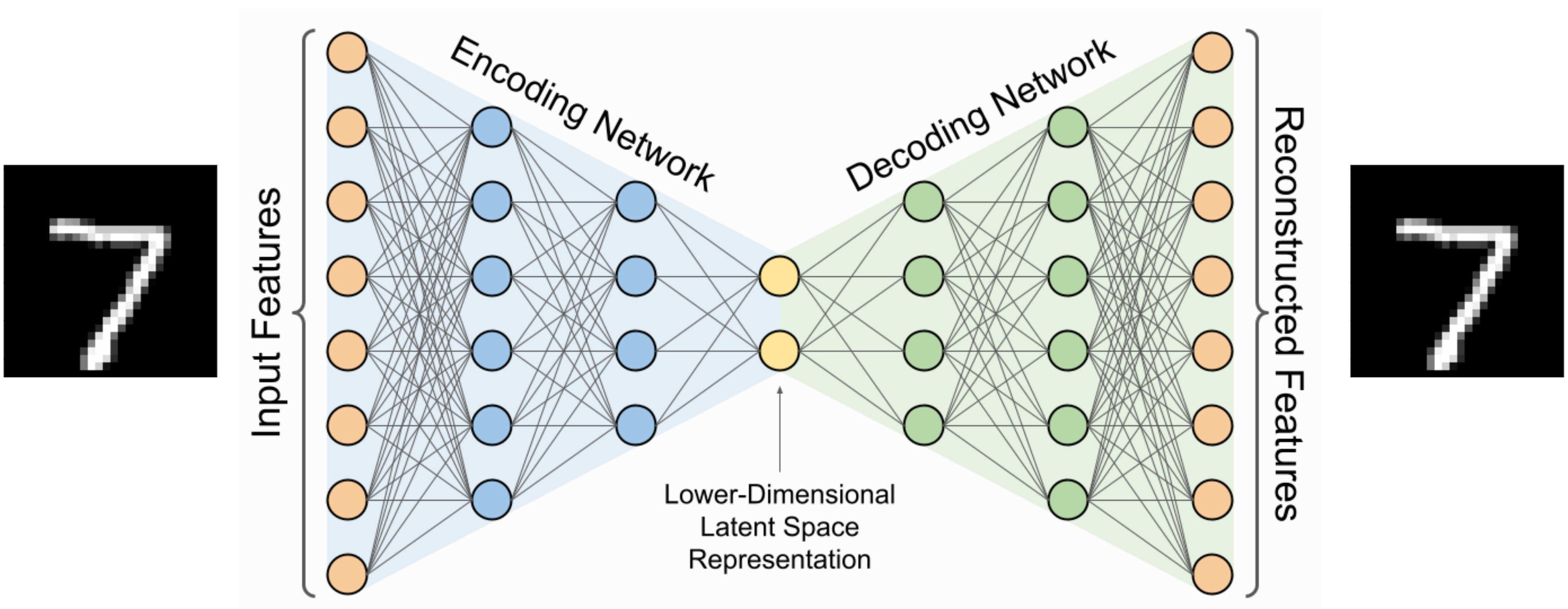
- These are neural networks for dim. reduction / feature extraction
- Idea: Try to reconstruct original data through a bottleneck
- Extracted features = bottleneck layer
- Layers need not be fully connected



Training Autoencoders

- Consider unlabeled training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$, and let $f_{\boldsymbol{\theta}}(\boldsymbol{x}) \in \mathbb{R}^d$ denote the output of the autoencoder, with weights $\boldsymbol{\theta}$.
- Objective function for learning $\boldsymbol{\theta}$?

$$\min_{\theta} \sum_{i=1}^{n} \| x_i - f_\theta(x_i) \|_2^2$$



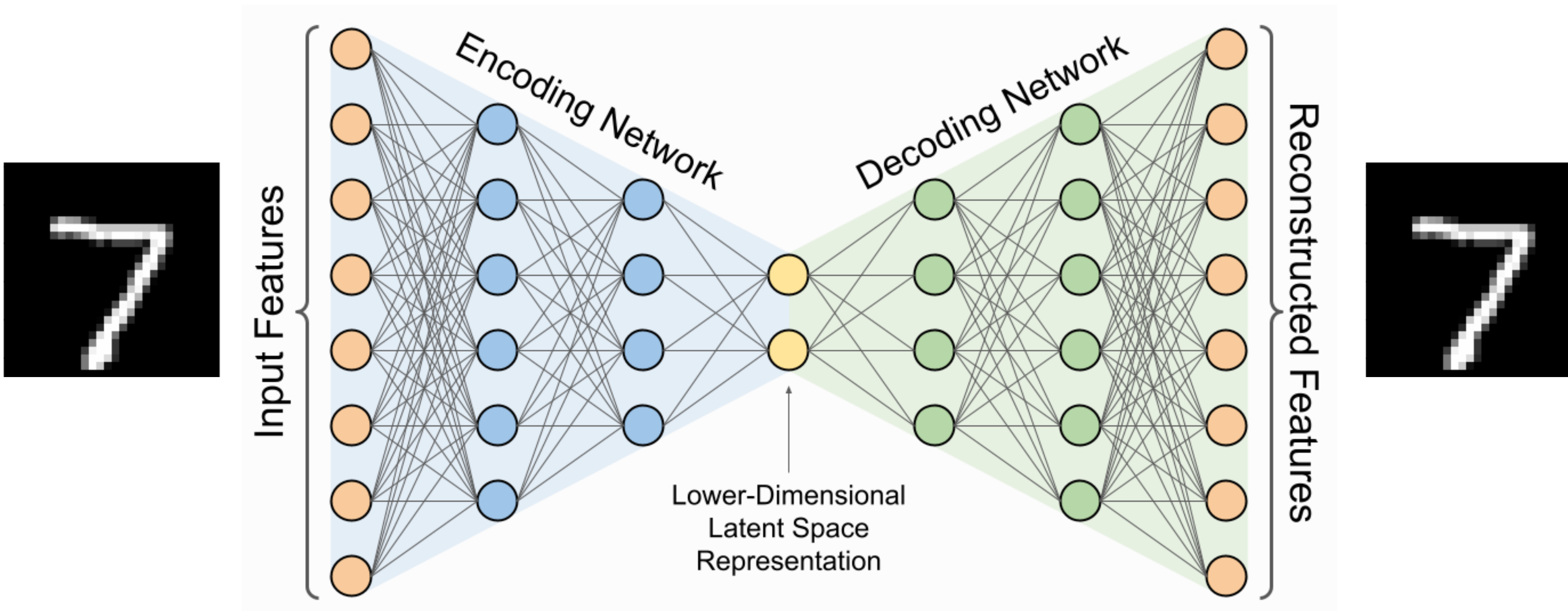
Poll

To optimize the reconstruction error, it suffices to use backpropagation as discussed previously for supervised learning with neural networks

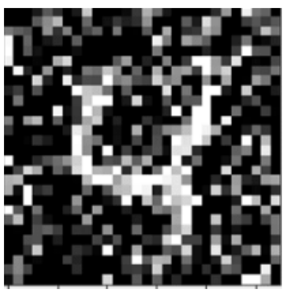
(A) True

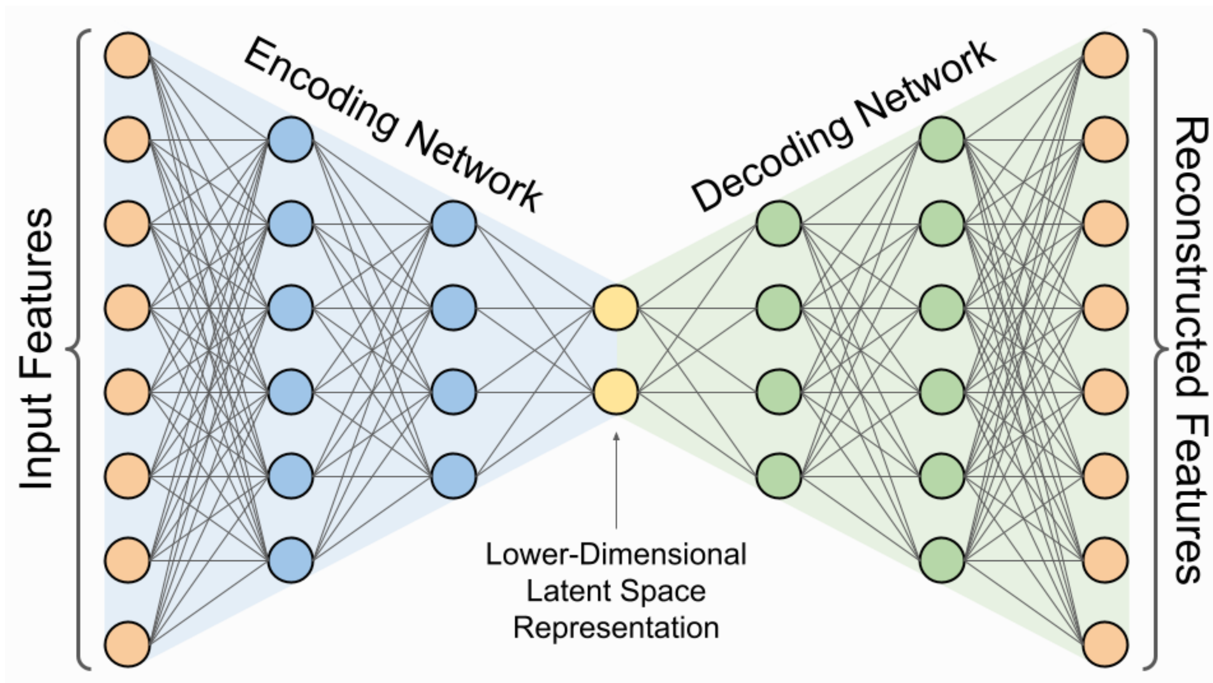
(B) False

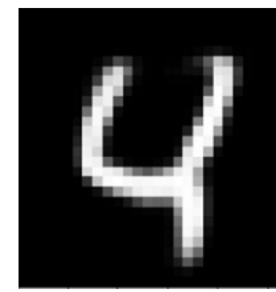
$$\sum_{i=1}^{n} \|x_i - f_\theta(x_i)\|_2^2$$



Autoencoders for Denoising

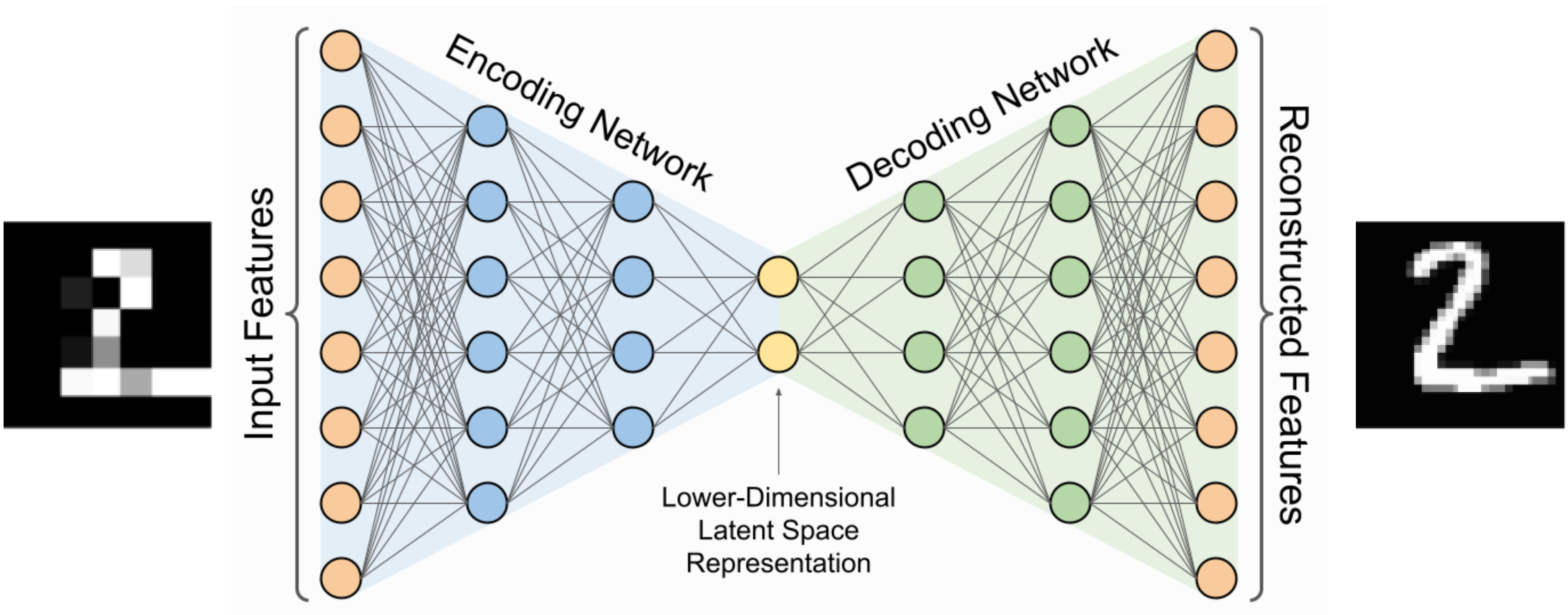






Train with pairs (noisy image, clean image)

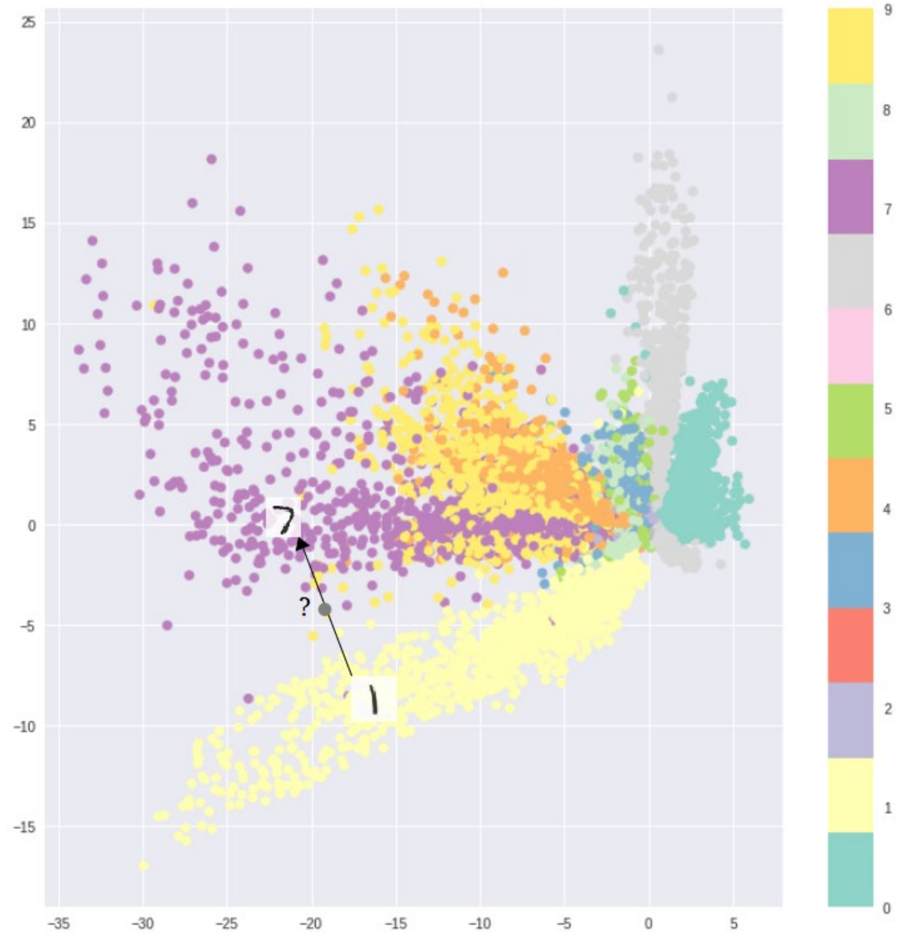
Autoencoders for Superresolution



Train with pairs (blurry image, sharp image)

Autoencoders for Simulation?

- Given: Examples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ of a probability distribution
- Goal: Simulate new instances from the distribution
- First try: Use the decoder half of an autoencoder with random inputs
- Issue: unrealistic synthetic images



https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

Variational Autoencoders

- An autoencoder that functions as a generative model
- A type of latent variable model
- Approximate maximum likelihood estimation by *variational inference*

Deep Latent Variable Models

- Given: Examples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ of a probability distribution
- Goal: Simulate new instances from the distribution
- Idea: Latent variable model

$$\boldsymbol{z} \sim p(\boldsymbol{z})$$

$x|z$

$$\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{z}; \boldsymbol{\theta})$$

where $p(\boldsymbol{x}|\boldsymbol{z}; \boldsymbol{\theta})$ is parametrized by a neural network.

DLVM Example: Binary Outputs

- Suppose

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix} \in \{0,1\}^d.$$

- DLVM:

$$z \sim N(0, I)$$

$$x_j \mid z \overset{ind}{\sim} \text{Bernoulli}(p_j), \quad \begin{bmatrix} p_1 \\ \vdots \\ p_d \end{bmatrix} = \sigma(d_\theta(z))$$

where

  - $\boldsymbol{d}_{\boldsymbol{\theta}} : \mathbb{R}^k \to \mathbb{R}^d$ is a neural network with weights $\boldsymbol{\theta}$
  - σ is a sigmoid function, such as

$$\sigma(t) = \frac{1}{1 + \exp(-t)}$$

DLVM Example: Continuous Outputs

- Suppose $\boldsymbol{x} \in \mathbb{R}^d$.
- DLVM:

$$z \sim N(0, I)$$

$$x \mid z \sim N(d_\theta(z), cI)$$

where

  - $\boldsymbol{d_\theta} : \mathbb{R}^k \to \mathbb{R}^d$ is a neural network with weights $\boldsymbol{\theta}$
  - c is a hyperparameter.

Parameter Estimation

- It is common to refer to $p(\boldsymbol{z})$ as the *prior* and $p(\boldsymbol{z}|\boldsymbol{x})$ as the *posterior*

$p(z|x;\theta)$

- Maximization of the log-likelihood

$$\log p(\boldsymbol{x};\boldsymbol{\theta}) = \log\left(\int p(\boldsymbol{x}|\boldsymbol{z};\boldsymbol{\theta})p(\boldsymbol{z})d\boldsymbol{z}\right)$$
$$= \log\left(\mathbb{E}_{p(z)}\left[p(x|z;\theta)\right]\right)$$

is intractable:

  - No analytic formula (because of the neural network)
  - No viable stochastic estimate of gradient (because of the log)

- For this reason, the posterior

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

is also intractable.

Variational Approximation

- To facilitate an approximation of the posterior, introduce a parametric family of distributions

$$q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})$$

- The idea is to approximate $p(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})$ with a member of this family:

$$q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi}) \approx p(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta}).$$

- We will also model $q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})$ with a neural network. For example:

$$(\mu, \nu) = e_\phi(x)$$

$$\sigma = \exp(\nu) \quad \text{(elementwise)}$$

$$q(z|x;\phi) = \mathcal{N}(z;\mu,\operatorname{diag}(\sigma))$$

where $\boldsymbol{e}_{\boldsymbol{\phi}} : \mathbb{R}^d \to \mathbb{R}^k$ is a neural network with weights $\boldsymbol{\phi}$

Variational Approximation

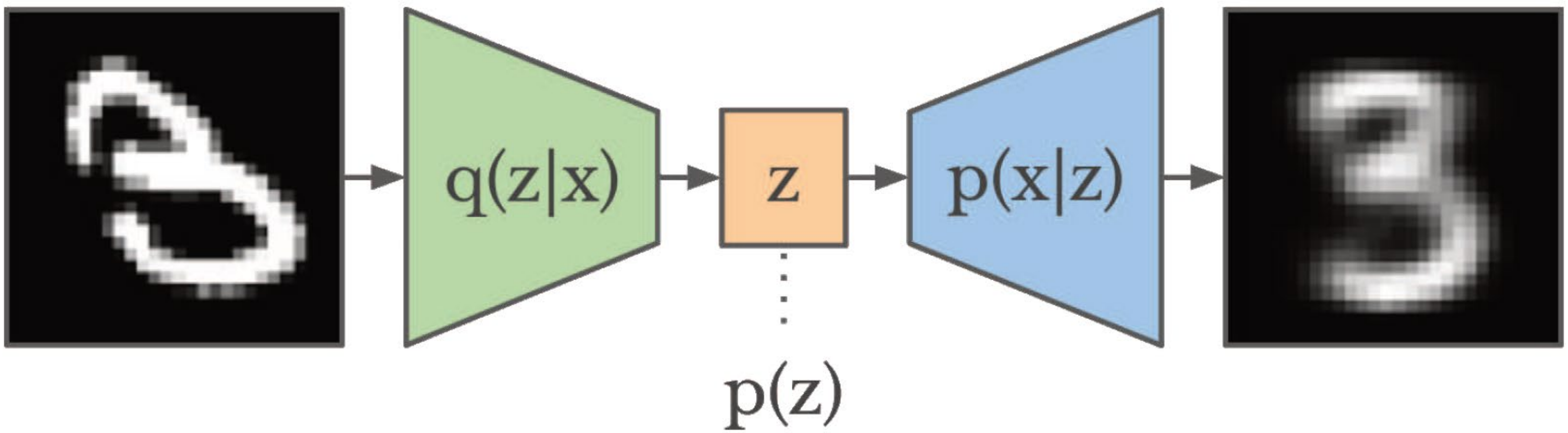


Figure credit: Murphy, *Probabilistic Machine Learning: Advanced Topics*

Variational Approximation

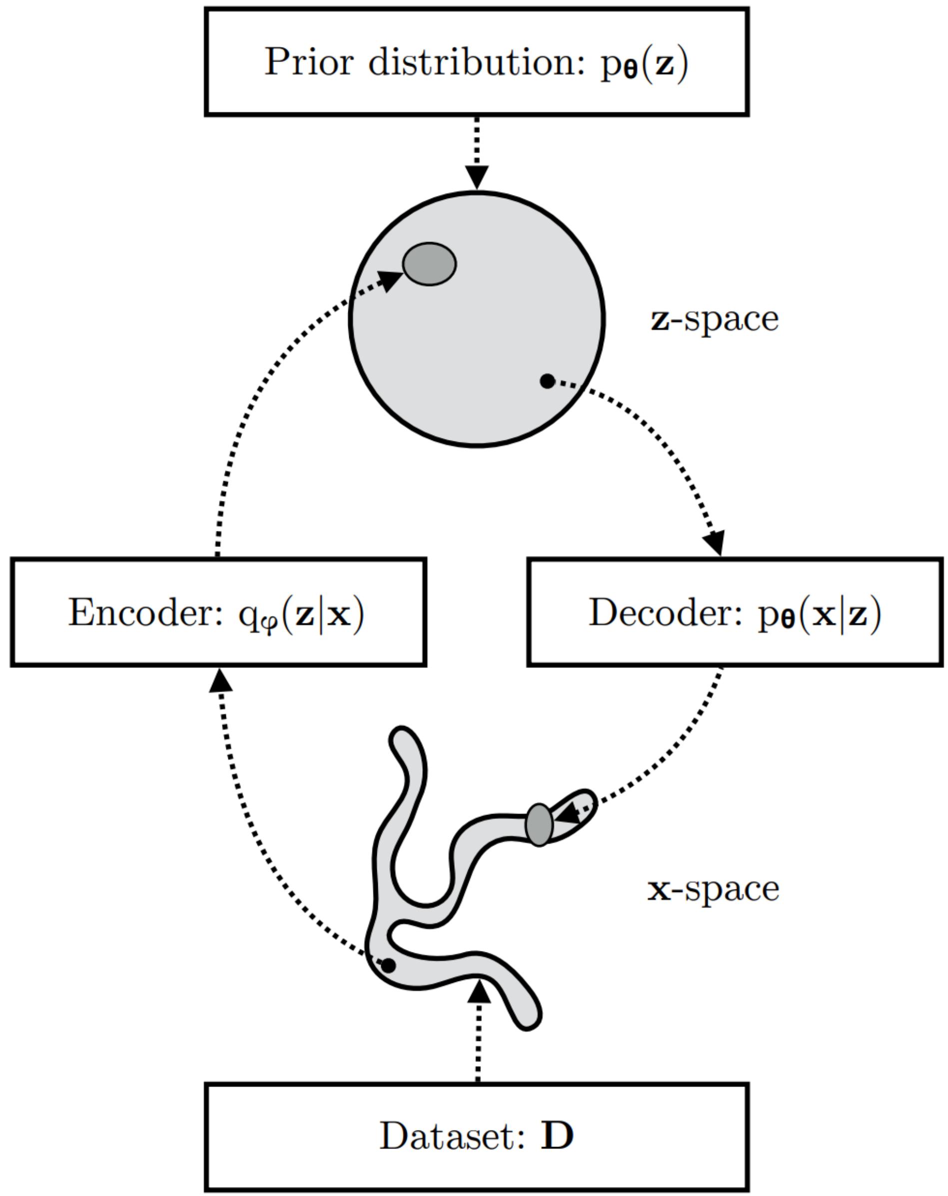


Figure credit: Kingma and Welling, An Introduction to Variational Autoencoders

Variational Inference

- So why did we introduce the approximate distribution $q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})$?
  - It is tractable (we know how to calculate it)
  - Thanks to the encoder network, it can achieve a good approximation of the posterior
  - It can be leveraged in an inference (parameter estimation) strategy known as *variational inference*

Evidence Lower Bound (ELBO)

$$
\begin{aligned}
\log p(\boldsymbol{x};\boldsymbol{\theta}) &= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}[\log p(\boldsymbol{x};\boldsymbol{\theta})] \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}\left[\log\left[\frac{p(\boldsymbol{x},\boldsymbol{z};\boldsymbol{\theta})}{p(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})}\right]\right] \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}\left[\log\left[\frac{p(\boldsymbol{x},\boldsymbol{z};\boldsymbol{\theta})}{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}\frac{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}{p(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})}\right]\right] \\
&= \underbrace{\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}\left[\log\left[\frac{p(\boldsymbol{x},\boldsymbol{z};\boldsymbol{\theta})}{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}\right]\right]}_{=:\,\mathrm{ELBO}(x;\theta,\phi)} + \underbrace{\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}\left[\log\left[\frac{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}{p(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})}\right]\right]}_{D_{KL}(q(z|x;\phi)\,\|\,p(z|x;\theta)) \geq 0}
\end{aligned}
$$

Lower bound on log-likelihood:

$$\ell(\theta;\underline{x}) = \sum_{i=1}^{n} \log p(x_i;\theta) \geq \sum_{i=1}^{n} \mathrm{ELBO}(x_i;\theta,\phi) = \underline{\ell}(\theta,\phi;\underline{x})$$

$$\underline{x} = (x_1, \ldots, x_n)$$

Variational Inference

- Basic idea is to maximize the lower bound with respect to both $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ using stochastic gradient ascent

- Need stochastic estimates of gradient wrt both $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$

- For $\boldsymbol{\theta}$, we have

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} \mathrm{ELBO}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) &= \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})} \left[\log p(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta}) - \log q(\boldsymbol{z}|\boldsymbol{x}; \boldsymbol{\phi})\right] \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})} \left[\nabla_{\boldsymbol{\theta}} (\log p(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta}) - \log q(\boldsymbol{z}|\boldsymbol{x}; \boldsymbol{\phi}))\right] \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})} \left[\nabla_{\boldsymbol{\theta}} (\log p(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta}))\right] \approx \frac{1}{M} \nabla_\theta (\log p(x, z_i; \theta)) \\
&\approx \nabla \log p(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta})
\end{aligned}$$

$z_1, \ldots, z_M \sim p(z)$

where in the last line, $\boldsymbol{z}$ is a realization of

$$\boldsymbol{z} \sim \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})}$$

Variational Inference

- For $\boldsymbol{\phi}$, let's reparametrize the probability model

$$\begin{aligned}(\boldsymbol{\mu}, \log \boldsymbol{\sigma}) &= \boldsymbol{e}_{\boldsymbol{\phi}}(\boldsymbol{x}) \\ q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi}) &= \mathcal{N}(\boldsymbol{z};\boldsymbol{\mu}, \operatorname{diag}(\boldsymbol{\sigma}))\end{aligned} \quad \Longleftrightarrow \quad \begin{aligned}\boldsymbol{\epsilon} &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \\ (\boldsymbol{\mu}, \log \boldsymbol{\sigma}) &= \boldsymbol{e}_{\boldsymbol{\phi}}(\boldsymbol{x}) \\ \boldsymbol{z} &= \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}\end{aligned}$$

- Then

$$\begin{aligned}\nabla_{\boldsymbol{\phi}} \mathrm{ELBO}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) &= \nabla_{\boldsymbol{\phi}} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})} \left[\log p(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta}) - \log q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})\right] \\ &= \nabla_{\boldsymbol{\phi}} \mathbb{E}_{p(\boldsymbol{\epsilon})} \left[\log p(\boldsymbol{x}, \boldsymbol{z}(\epsilon;\boldsymbol{\phi}); \boldsymbol{\theta}) - \log q(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\phi})\right] \\ &= \mathbb{E}_{p(\boldsymbol{\epsilon})} \left[\nabla_{\boldsymbol{\phi}} (\log p(\boldsymbol{x}, \boldsymbol{z}(\epsilon;\boldsymbol{\phi}); \boldsymbol{\theta}) - \log q(\boldsymbol{z}(\epsilon;\boldsymbol{\phi})|\boldsymbol{x};\boldsymbol{\phi}))\right] \\ &\approx \left[\nabla_{\boldsymbol{\phi}} (\log p(\boldsymbol{x}, \boldsymbol{z}(\epsilon;\boldsymbol{\phi}); \boldsymbol{\theta}) - \log q(\boldsymbol{z}(\epsilon;\boldsymbol{\phi})|\boldsymbol{x};\boldsymbol{\phi}))\right]\end{aligned}$$

where in the last line, $\boldsymbol{\epsilon}$ is a realization $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

- For additional details, see `https://arxiv.org/pdf/1906.02691.pdf`

Poll

True of False: The EM algorithm is another algorithm to maximize the ELBO

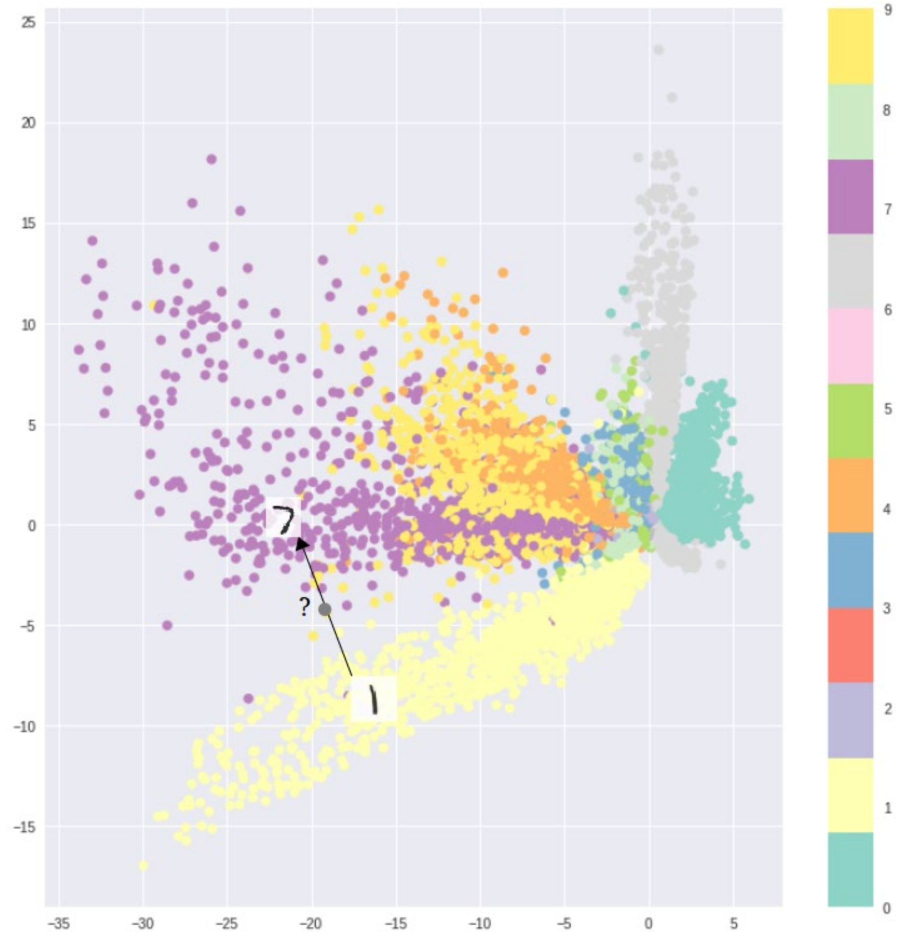
(A) True

(B) False

$$\mathrm{ELBO}(x; \theta, \phi) = \mathbb{E}_{q(z|x;\phi)}\left[\log\left(\frac{p(x,z;\theta)}{q(z|x;\phi)}\right)\right]$$

Variational Autoencoders

- Given: Examples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ of a probability distribution
- Goal: Simulate new instances from the distribution
- First try: Use the decoder half of an autoencoder with random inputs
- Issue: unrealistic synthetic images



Variational Autoencoders

- VAEs generate realistic images while the latent variable is varied continuously