



# Lecture 18: Deep Generative Models II: AE & VAE

Lan Xu  
SIST, ShanghaiTech  
Fall, 2021

# Outline

- Representation learning
  - AutoEncoder
- Variational Autoencoders (VAEs)
  - VAE objective
  - Reparametrization trick
  - Connection to Auto-Encoders

*Acknowledgement: Feifei Li et al's cs231n notes*

# Recall EM GMM

- MLE: maximizing the log-likelihood

$$\ell(\theta) = \sum_{i=1}^n \log p(x^{(i)}; \theta)$$

- ELBO: Evidence Lower Bound

$$\log p(\mathbf{x}) = \log \int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z})$$

$$= \log \int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) \frac{q(\mathbf{z})}{q(\mathbf{z})}$$

$$= \log(E_q[\frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z})}])$$

$$\geq E_q[\log p(\mathbf{x}, \mathbf{z})] - E_q[\log q(\mathbf{z})]$$

$$= ELBO$$

$$KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})) = E_q[\log q(\mathbf{z})] - E_q[\log p(\mathbf{z}|\mathbf{x})]$$

$$= E_q[\log q(\mathbf{z})] - E_q[\log p(\mathbf{z}, \mathbf{x})] + \log p(\mathbf{x})$$

$$= \log p(\mathbf{x}) - (E_q[\log p(\mathbf{z}, \mathbf{x})] - E_q[\log q(\mathbf{z})])$$

$$= \log p(\mathbf{x}) - ELBO$$



$$EBLO = E_q[\log p(\mathbf{x}, \mathbf{z})] - E_q[\log q(\mathbf{z})]$$

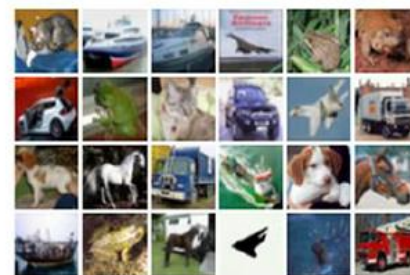
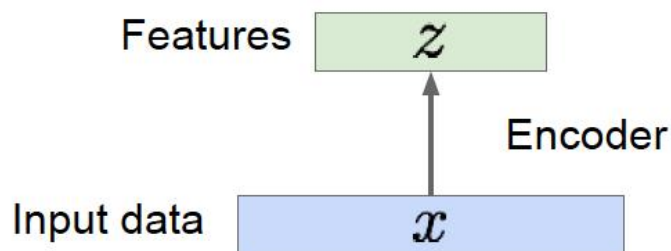
$$= E_q[\log \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{z})}] - E_q[\log \frac{q(\mathbf{z})}{p(\mathbf{z})}]$$

$$= E_q[\log p(\mathbf{x}|\mathbf{z})] - KL(q(\mathbf{z})||p(\mathbf{z}))$$

# Autoencoder

- Feature representation learning

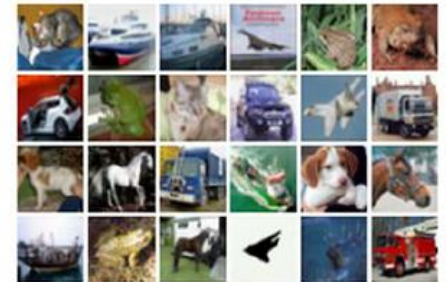
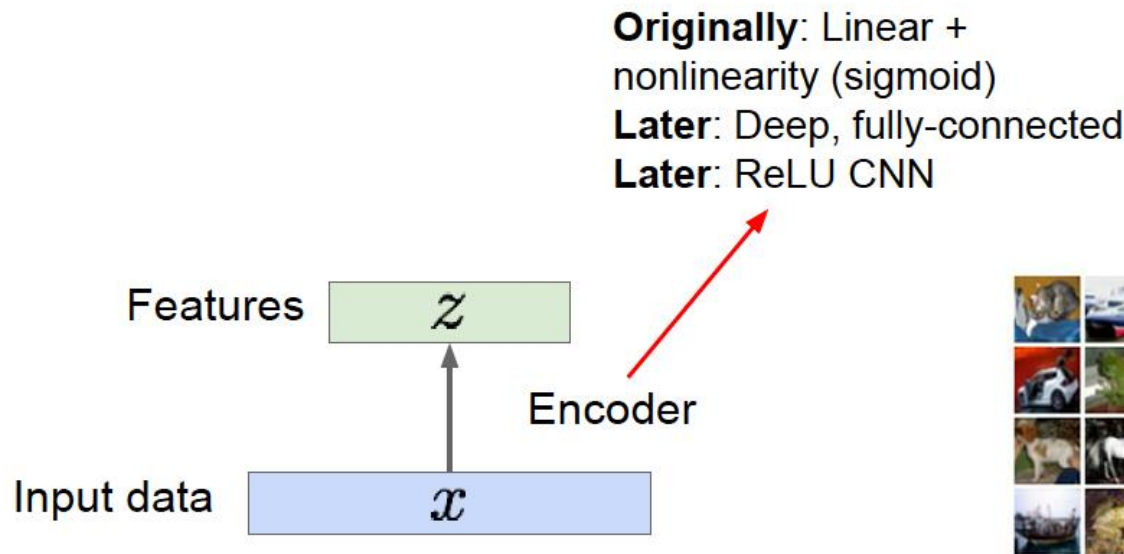
Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data



# Autoencoder

## ■ Feature representation learning

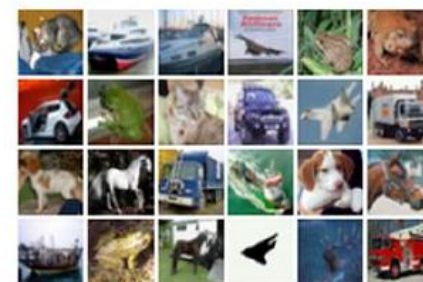
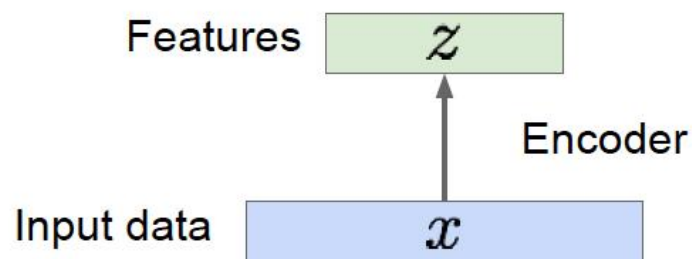
Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data



# Autoencoder

- Feature representation learning

How to learn this feature representation?

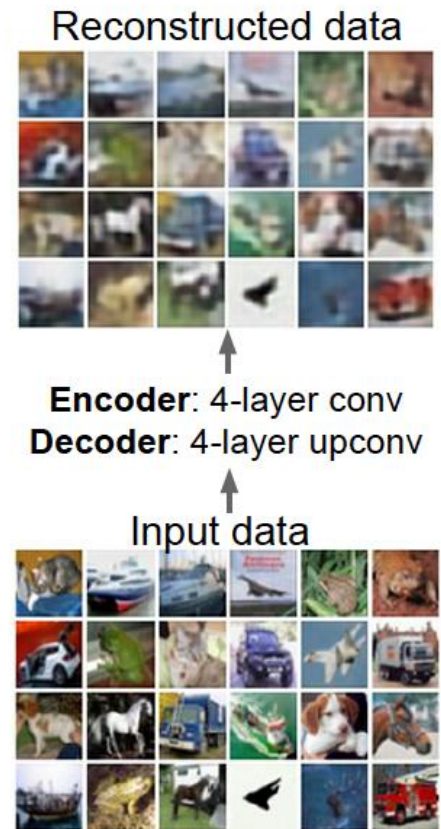
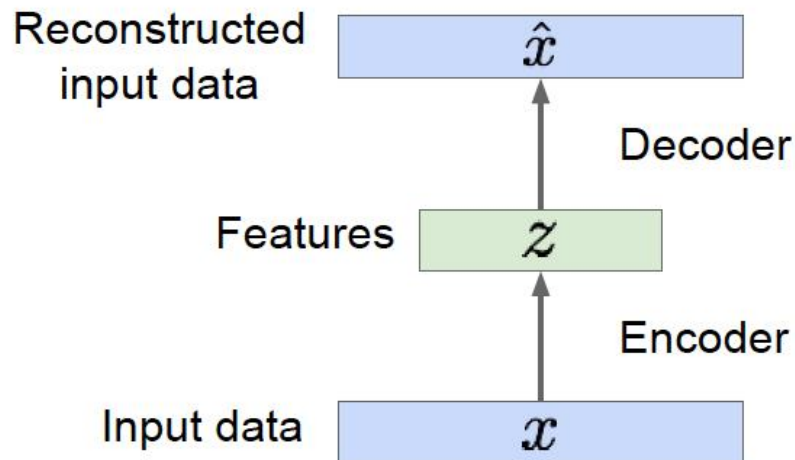


# Autoencoder

## ■ Feature representation learning

How to learn this feature representation?

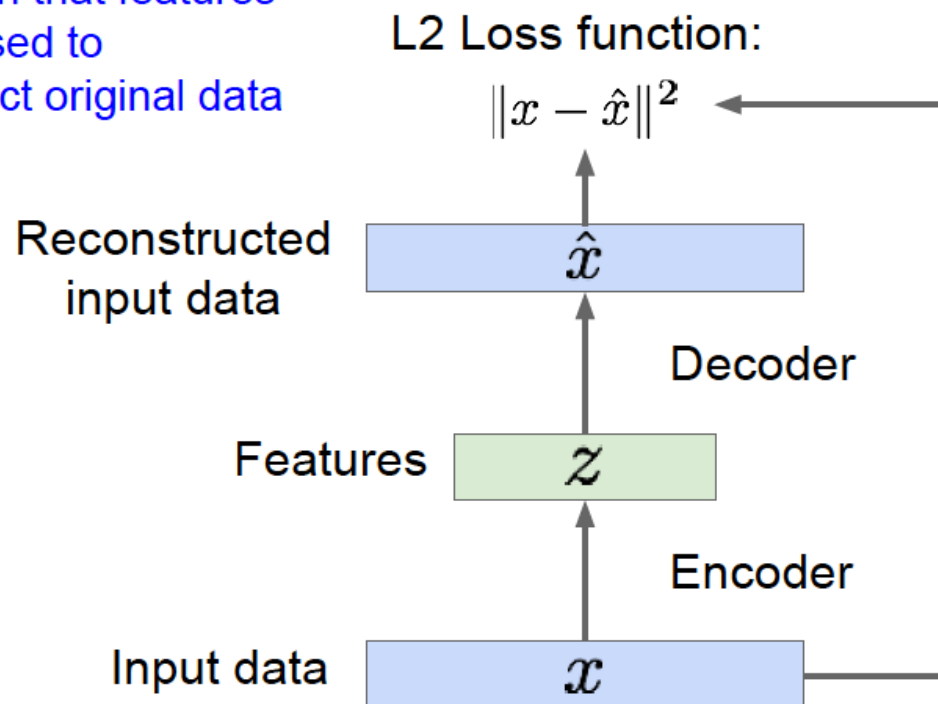
Train such that features can be used to reconstruct original data  
“Autoencoding” - encoding itself



# Autoencoder

- Feature representation learning

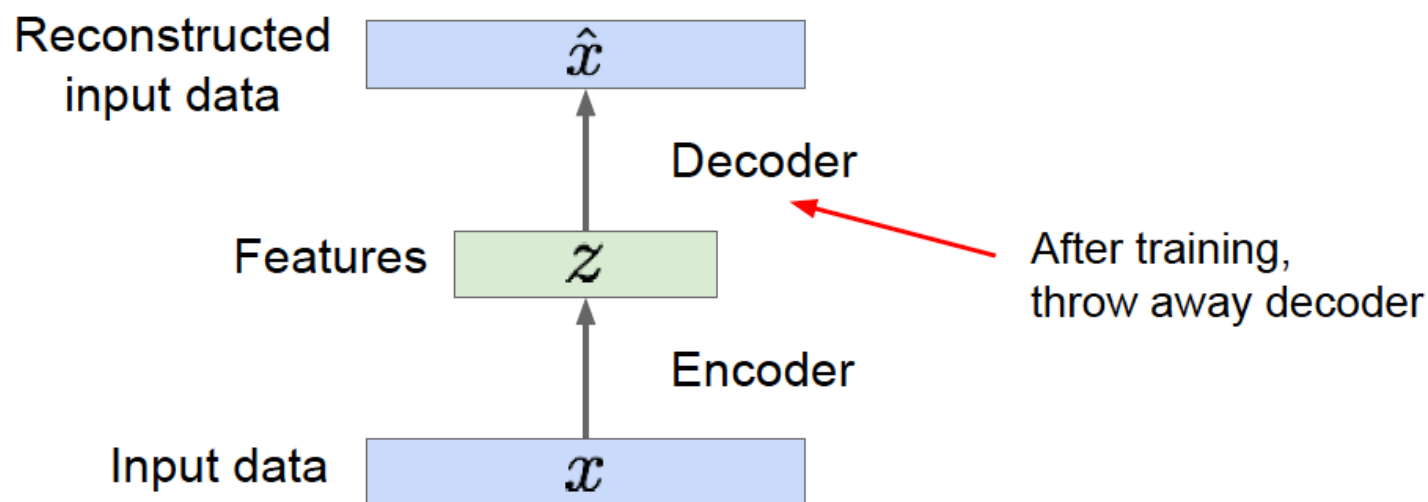
Train such that features  
can be used to  
reconstruct original data





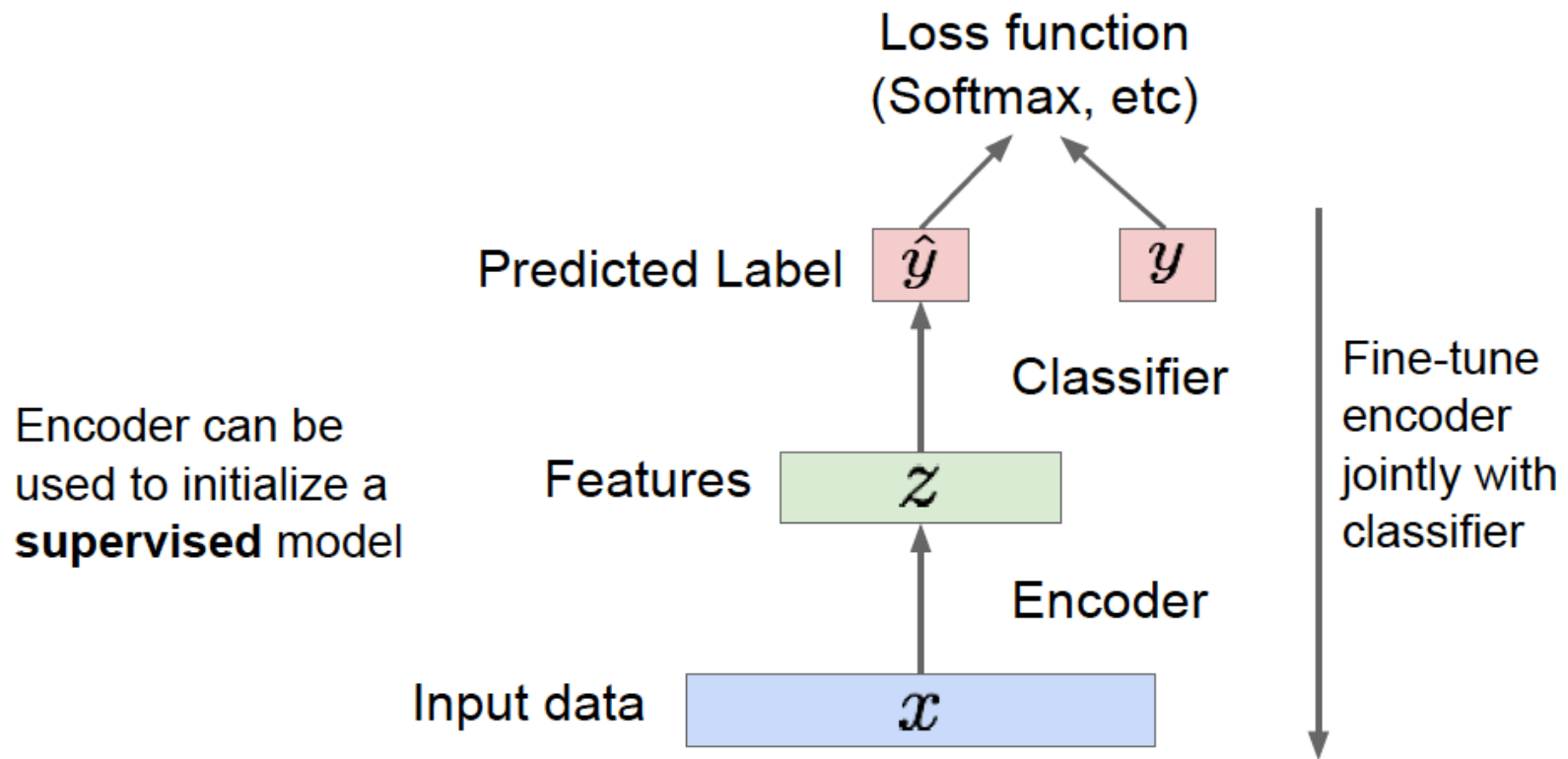
# Autoencoder

- Feature representation learning



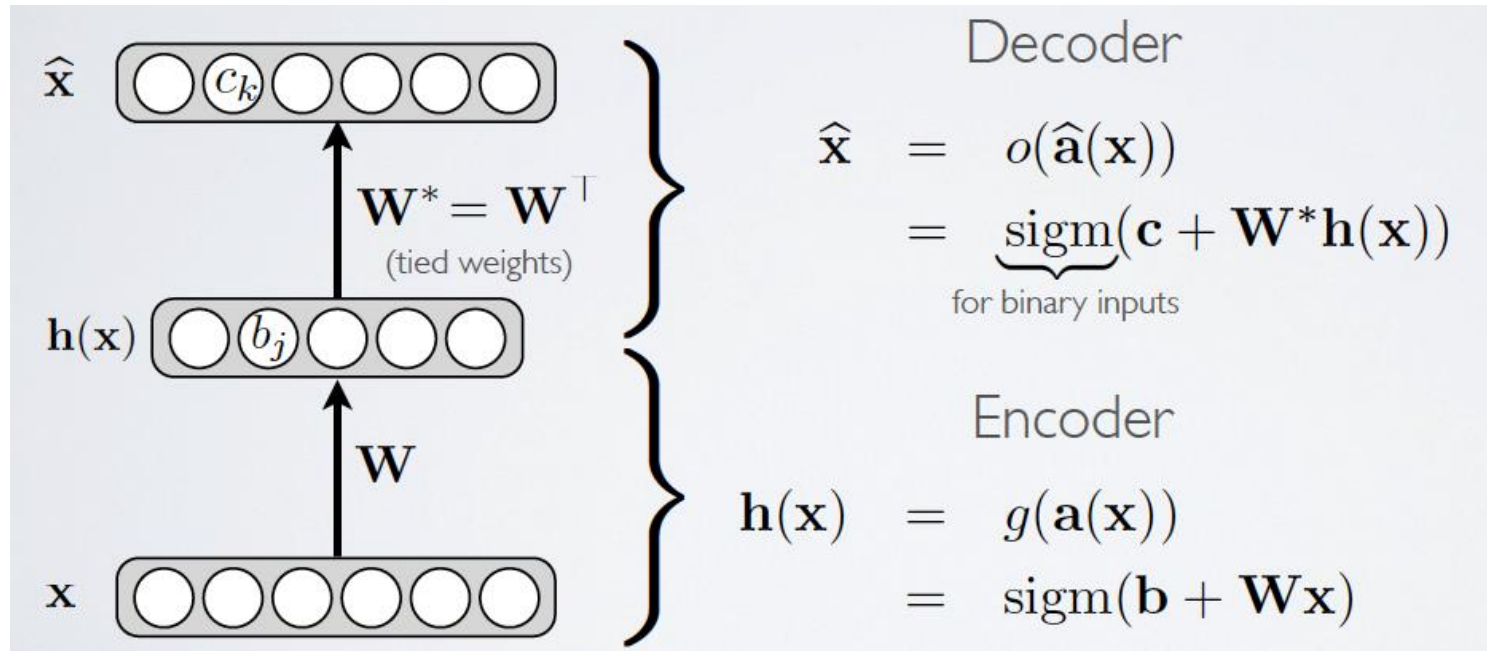
# Autoencoder

- Feature representation learning
- But not probabilistic: no way to sample new data



# Autoencoder

## ■ Binary input example



$$l(f(\mathbf{x})) = - \sum_k (x_k \log(\hat{x}_k) + (1 - x_k) \log(1 - \hat{x}_k))$$

- cross-entropy (more precisely: sum of Bernoulli cross-entropies)

# Regularization

- Regularized autoencoders: add regularization term that encourages the model to have other properties
  - Sparsity of the representation (sparse autoencoder)
  - Robustness to noise or to the missing inputs (denoising autoencoder)
  - Smallness of the derivative of the representation (contractive autoencoder)

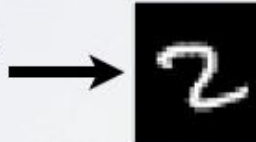
# Regularization

## ■ Undercomplete representation

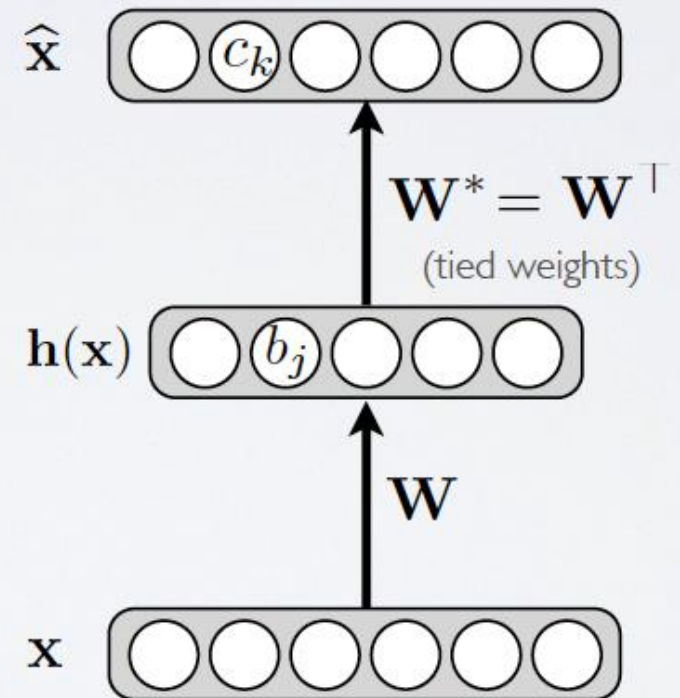
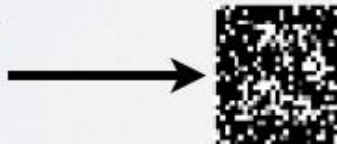
- Hidden layer is undercomplete if smaller than the input layer
  - ▶ hidden layer “compresses” the input
  - ▶ will compress well only for the training distribution

- Hidden units will be

- ▶ good features for the training distribution

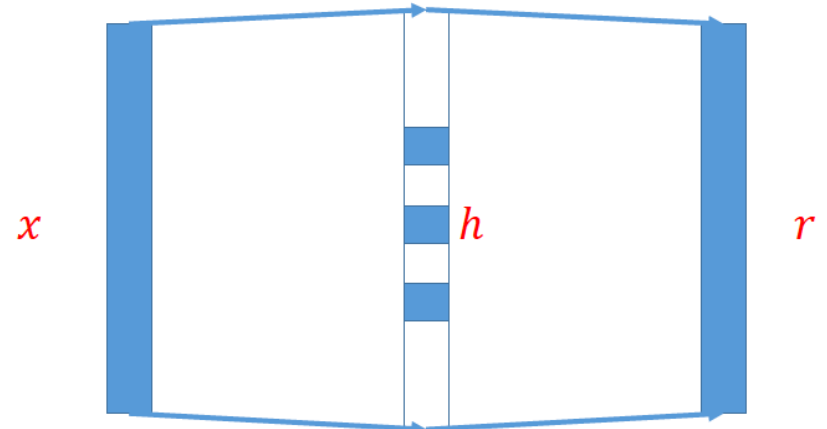


- ▶ but bad for other types of input



# Regularization

$$L_R = L(x, g(f(x))) + R(h)$$



## ■ Sparse autoencoder

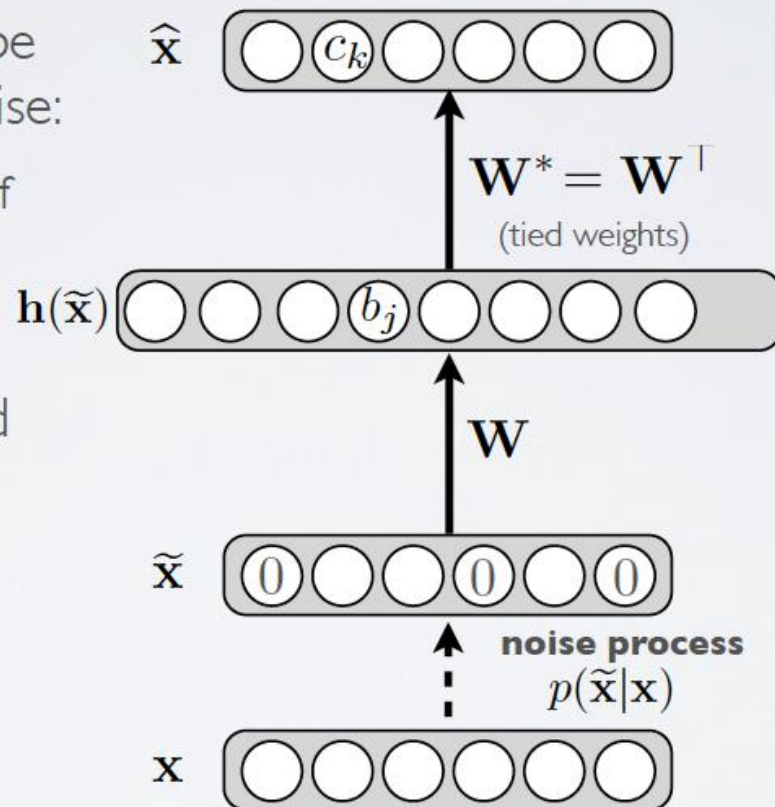
- Constrain the code to have sparsity
- Training: minimize a loss function

$$L_R = L(x, g(f(x))) + \lambda |h|_1$$

# Regularization

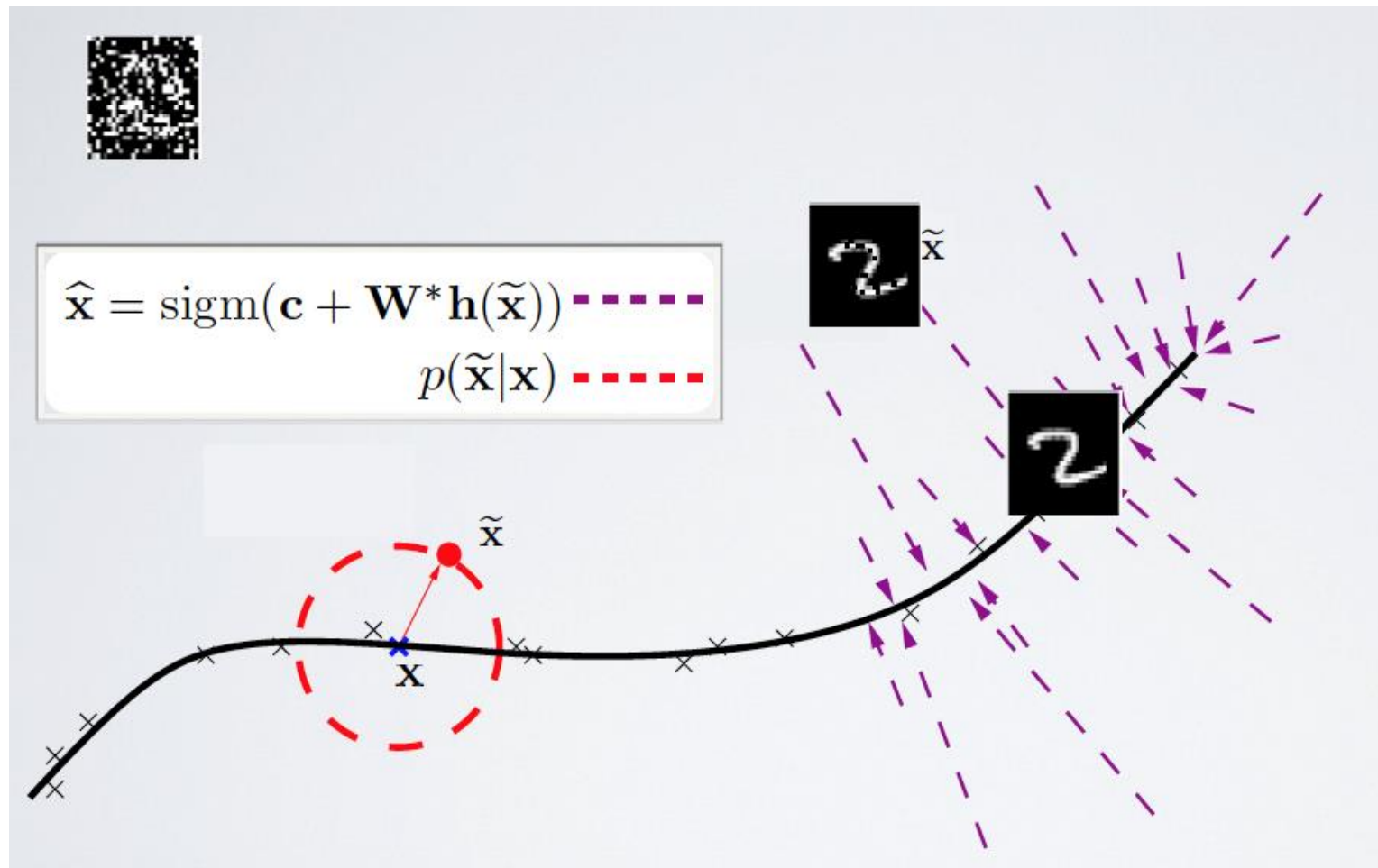
## ■ Denoising autoencoder

- Idea: representation should be robust to introduction of noise:
  - random assignment of subset of inputs to 0, with probability  $\nu$
  - Gaussian additive noise
- Reconstruction  $\hat{\mathbf{x}}$  computed from the corrupted input  $\tilde{\mathbf{x}}$
- Loss function compares  $\hat{\mathbf{x}}$  reconstruction with the **noiseless input  $\mathbf{x}$**



# Regularization

- Denoising autoencoder





# Outline

- Representation learning
  - AutoEncoder
- Variational Autoencoders (VAEs)
  - VAE objective
  - Reparametrization trick
  - Connection to Auto-Encoders

*Acknowledgement: Feifei Li et al's cs231n notes*

# Latent variable model

## ■ Data generation process

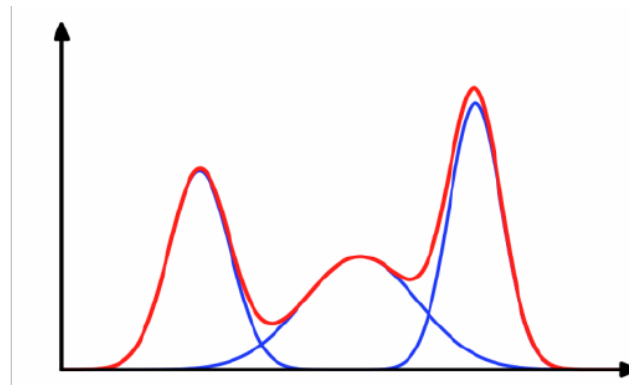
□ Latent variable  $\mathbf{z}$        $p(\mathbf{z}) = \text{something simple}$

□ A mapping from the latent space to observation  $\mathbf{x}$

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z}) d\mathbf{z} \quad \text{where} \quad p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x} | \mathbf{z})p(\mathbf{z})$$

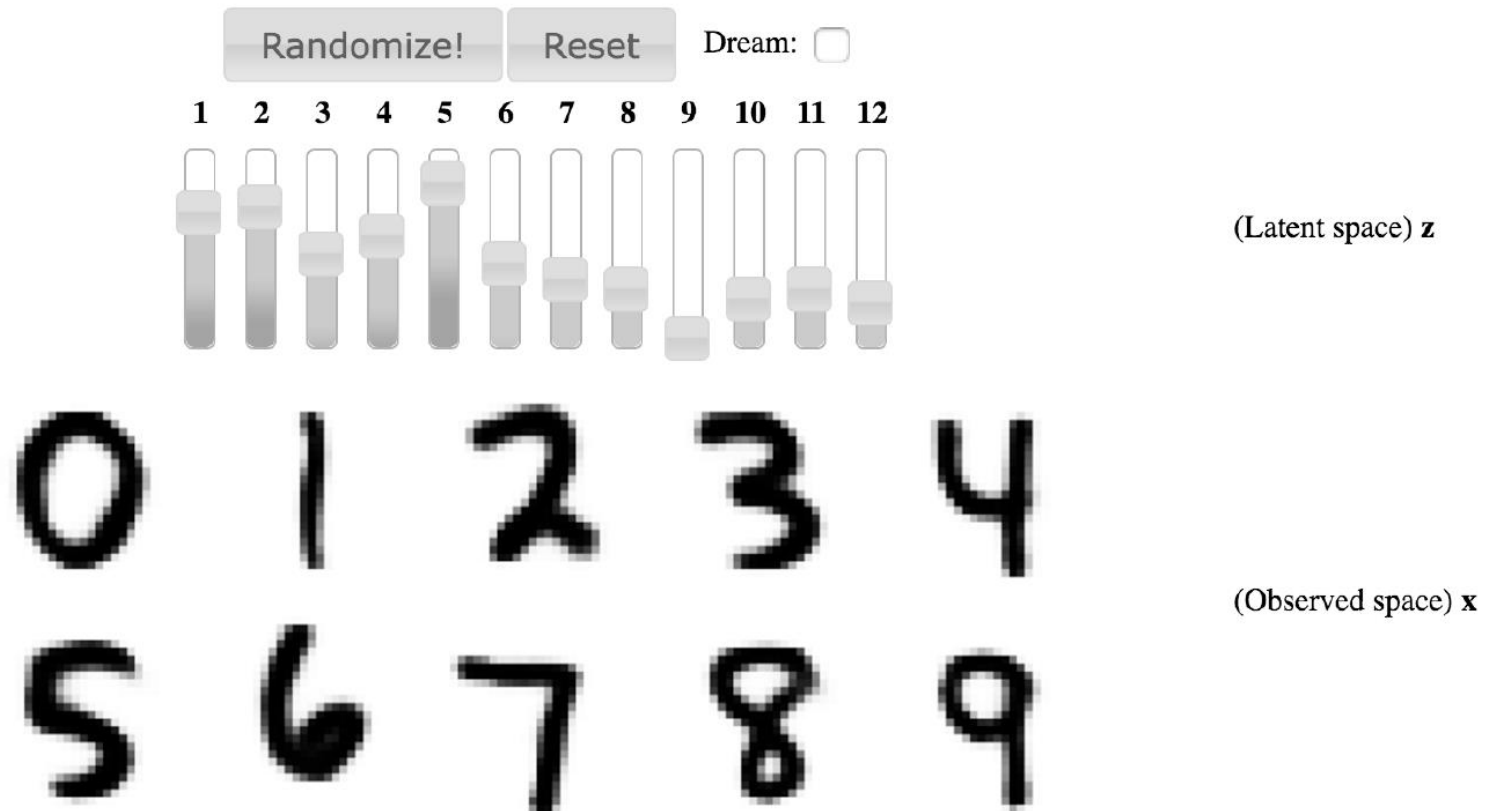
For example, a Gaussian mixture model

$$p_{\theta}(\mathbf{x}) = \sum_{k=1}^K p_{\theta}(\mathbf{z} = k) p_{\theta}(\mathbf{x} | \mathbf{z} = k)$$



# An example

- Generating hand-written digits

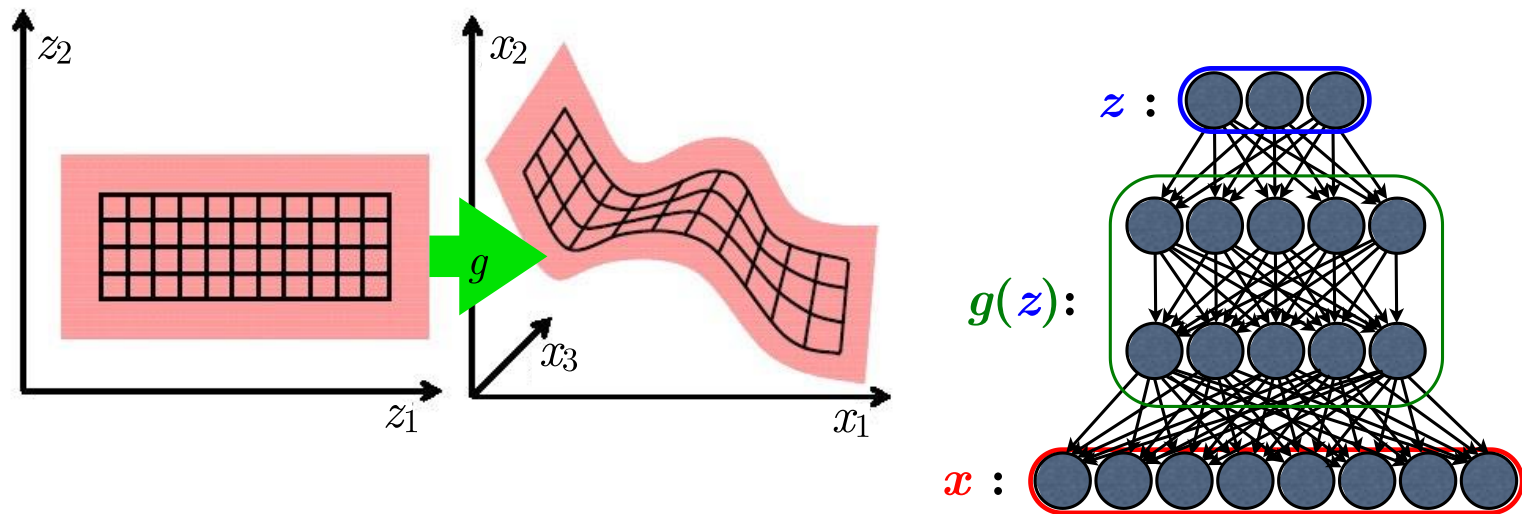


☐ [http://www.dpkingma.com/sgvb\\_mnist\\_demo/demo.html](http://www.dpkingma.com/sgvb_mnist_demo/demo.html)

# Deep Latent Variable Models

- Leverage neural networks in a latent variable model

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x} \mid \mathbf{z})p(\mathbf{z}) \quad p(\mathbf{x} \mid \mathbf{z}) = g(\mathbf{z})$$



- Can represent complicated data distribution and conditional dependencies

# An example

- $p(z) = N(0, I)$

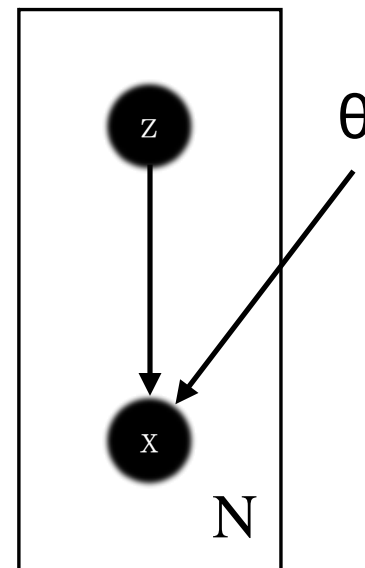
- $P_{\theta}(x|z) = N(\mu, \sigma^2)$

$\mu = f_{\theta}(z) = \text{multilayer neural net}$

- With flexible neural net,

$$p_{\theta}(x) = \int_z p_{\theta}(x|z)p(z)dz$$

- Can be arbitrarily complicated
- The multilayer network can capture complex dependencies in the data generation process



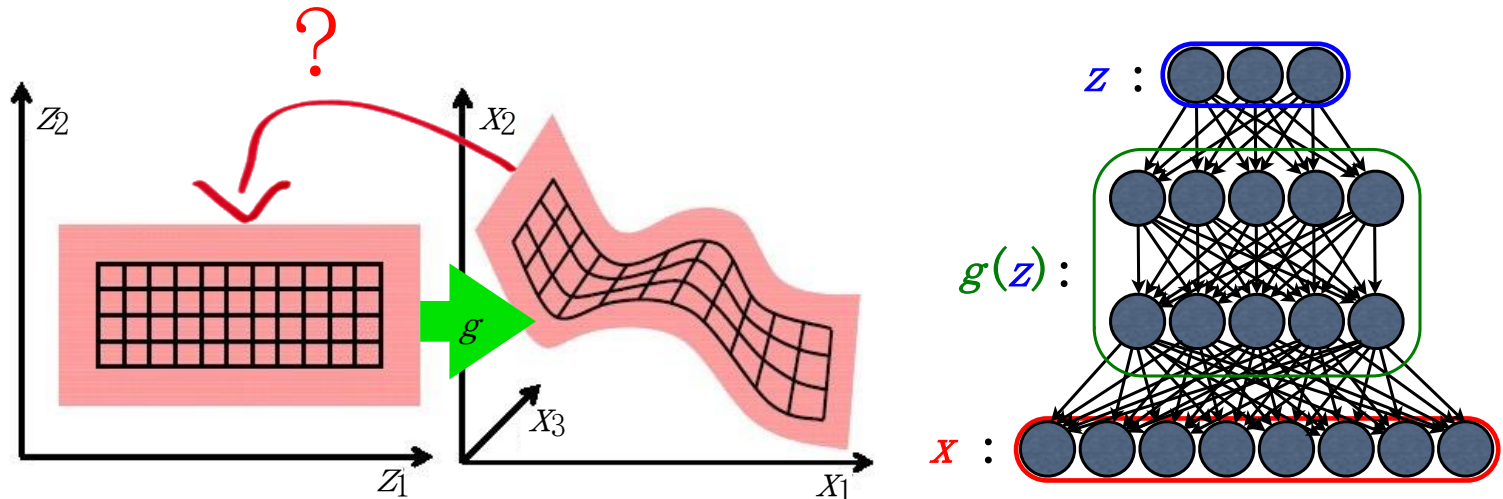
# Challenges in Deep LVM

## ■ Inference:

- Given an observation  $\mathbf{x}$ , what is the probable  $\mathbf{z}$  ?
- Computing the posterior  $p(\mathbf{z}|\mathbf{x})$  (intractable)

## ■ Learning:

- Given a large dataset of observations  $\mathbf{X} = \{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\}$
- Estimating the parameters in Deep LVM (inefficient/intractable)



# The Variational Autencoder: overview

## ■ Inference:

- Introduce a parametric model  $q_\phi(z | x)$  to approximate the true posterior  $p_\theta(z | x)$
- Variational inference or Amortized inference

$$\begin{aligned} \forall x_i \quad & \arg \min_{q_i} D_{KL}(q_i(z) || p_\theta(z|x_i)) \\ \Rightarrow \forall x_i \quad & \arg \min_{\phi} D_{KL}(q_\phi(z|x_i) || p_\theta(z|x_i)) \end{aligned}$$

- Replacing datum-wise posterior distribution by a parametric family of conditional densities.

# The Variational Autencoder: overview

## ■ Inference:

- Introduce a parametric model  $q_{\phi}(z \mid x)$  to approximate the true posterior  $p_{\theta}(z \mid x)$
- Variational inference or Amortized inference

## ■ Learning:

- Based on Maximum Likelihood

$$\max \sum_{i=1}^N \log p(x^{(i)})$$

- Direct optimization is challenging: use EM learning strategy
- Jointly learning inference model with the deep latent variable model



# VAE objective

- Recall lower bound of the data log likelihood

$$\begin{aligned}\log p_{\theta}(x) &= \log \int_z p_{\theta}(x, z) dz \\ &= \log \int_z q_{\phi}(z|x) \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} dz \\ &\geq \int_z q_{\phi}(z|x) \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} dz \quad (\text{Jensen's Inequality}) \\ &= \mathbf{E}_{q_{\phi}(z|x)} [\log p_{\theta}(x, z) - \log q_{\phi}(z|x)] = \mathcal{L}(x; \theta, \phi)\end{aligned}$$

$$\log p_{\theta}(x) = \boxed{\mathcal{L}(x; \theta, \phi)} + D_{KL}(q_{\phi}(z|x) || p_{\theta}(z|x))$$

- Learning: maximize the lower bound of data likelihood
- The evidence lower bound (ELBO)

# VAE objective

- From the EM perspective

$$\begin{aligned}\log p_{\theta}(x) &= \log \frac{p_{\theta}(x|z)p(z)}{p_{\theta}(z|x)} = \log \frac{p_{\theta}(x|z)p(z)q_{\phi}(z|x)}{p_{\theta}(z|x)q_{\phi}(z|x)} \\ &= E_z[\log p_{\theta}(x|z)] - E_z \left[ \log \frac{q_{\phi}(z|x)}{p(z)} \right] + E_z \left[ \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} \right] \\ &= E_{z \sim q_{\phi}(z|x)}[\log p_{\theta}(x|z)] - D_{KL}(q_{\phi}(z|x), p(z)) + D_{KL}(q_{\phi}(z|x), p_{\theta}(z|x))\end{aligned}$$

Data reconstruction

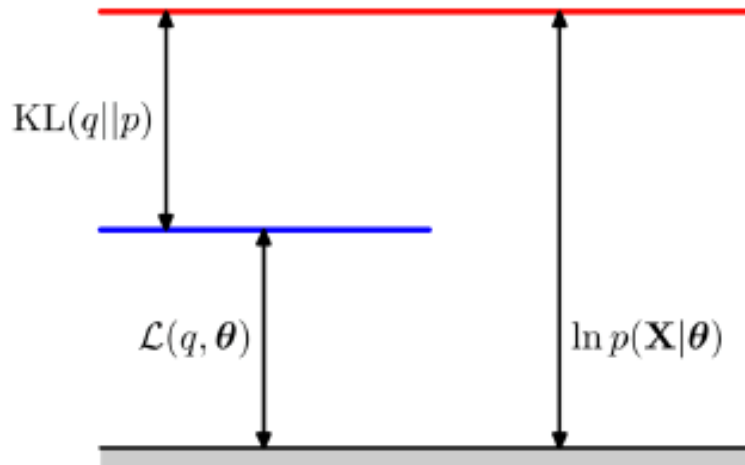
KL divergence between prior, and  
samples from the encoder network

KL divergence between encoder  
and posterior of decoder

# VAE objective

## ■ Visualizing ELBO

$$\log p_{\theta}(x) = \mathcal{L}(x; \theta, \phi) + D_{KL}(q_{\phi}(z|x) || p_{\theta}(z|x))$$



Bishop – Pattern Recognition and Machine Learning

- Note: all we have done so far is decompose the log probability of the data, we still have exact equality
- This holds for any distribution  $q$

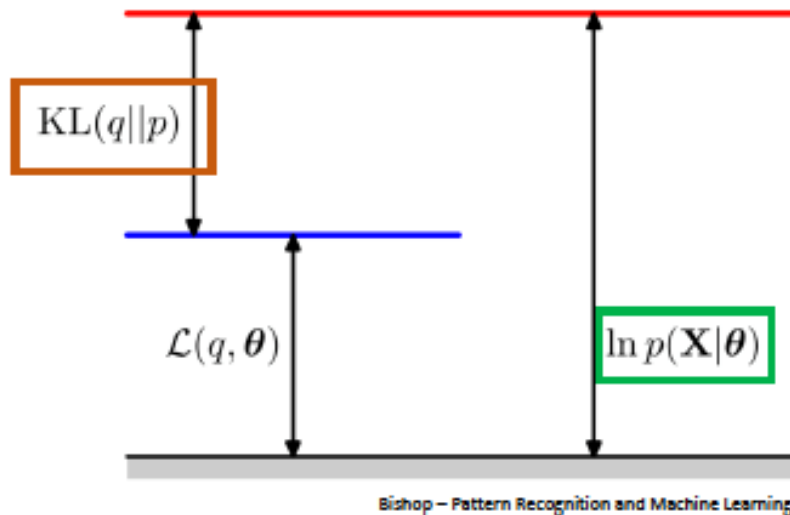
# VAE learning

## ■ EM perspective

- Expectation Maximization alternately optimizes the ELBO,  $\mathcal{L}(q, \theta)$ , with respect to  $q$  (the E step) and  $\theta$  (the M step)
- Initialize  $\theta^{(0)}$
- At each iteration  $t = 1, \dots$ 
  - **E step:** Hold  $\theta^{(t-1)}$  fixed, find  $q^{(t)}$  which maximizes  $\mathcal{L}(q, \theta^{(t-1)})$
  - **M step:** Hold  $q^{(t)}$  fixed, find  $\theta^{(t)}$  which maximizes  $\mathcal{L}(q^{(t)}, \theta)$

# EM perspective

## ■ The E step

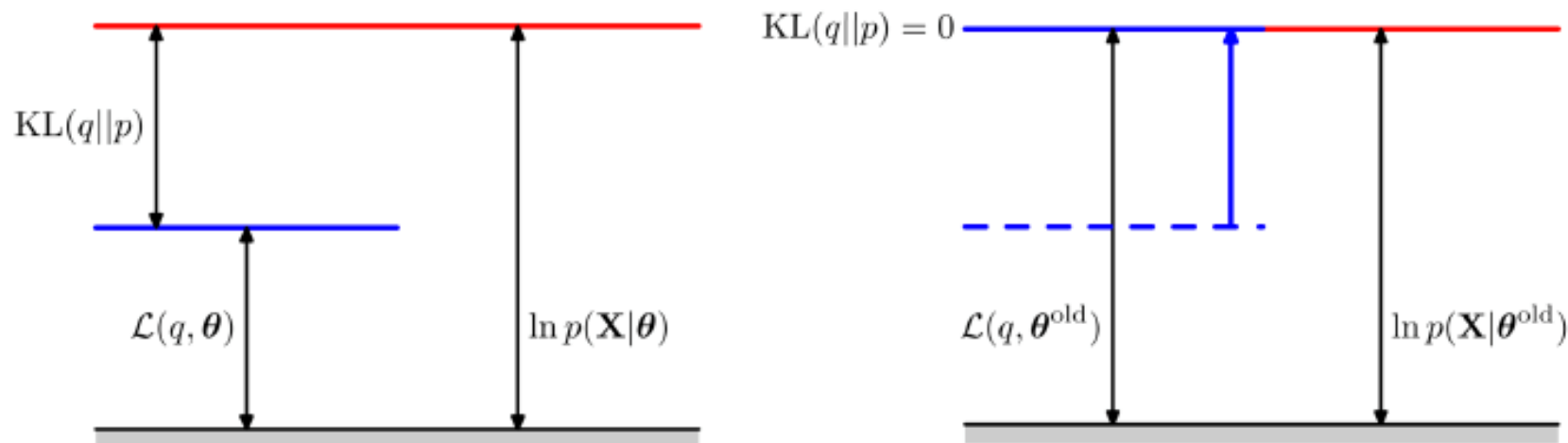


- The **first term** does not involve  $q$ , and we know the KL divergence must be non-negative
- The best we can do is to **make the KL divergence 0**
- Thus the solution is to set  $q^{(t)}(z) \leftarrow p(z|x, \theta^{(t-1)})$

- Suppose we are at iteration  $t$  of our algorithm. How do we maximize  $\mathcal{L}(q, \theta^{(t-1)})$  with respect to  $q$ ? We know that:  
$$\operatorname{argmax}_q \mathcal{L}(q, \theta^{(t-1)}) = \operatorname{argmax}_q \log p(x|\theta^{(t-1)}) - \text{KL}(q(z) || p(z|x, \theta^{(t-1)}))$$

# EM perspective

## ■ The E step

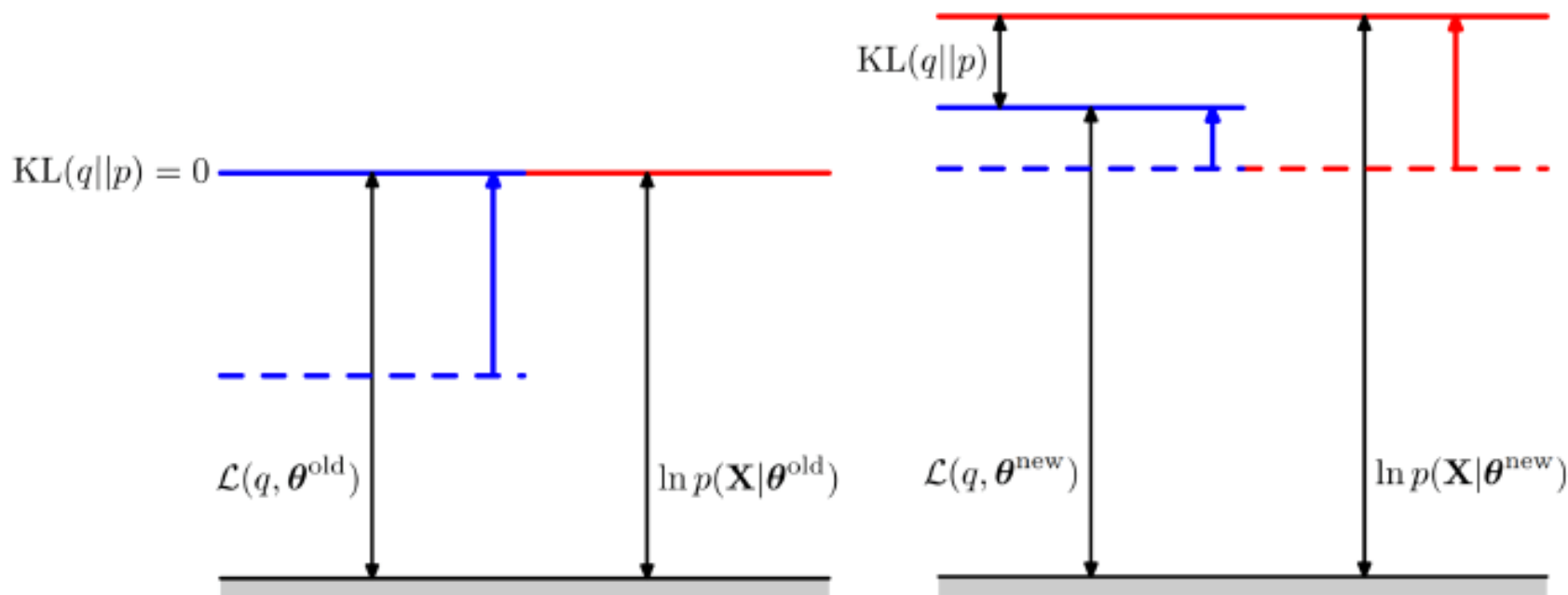


Bishop – Pattern Recognition and Machine Learning

- Suppose we are at iteration  $t$  of our algorithm. How do we maximize  $\mathcal{L}(q, \theta^{(t-1)})$  with respect to  $q$ ?  $q^{(t)}(z) \leftarrow p(z|x, \theta^{(t-1)})$

# EM perspective

## ■ The M step

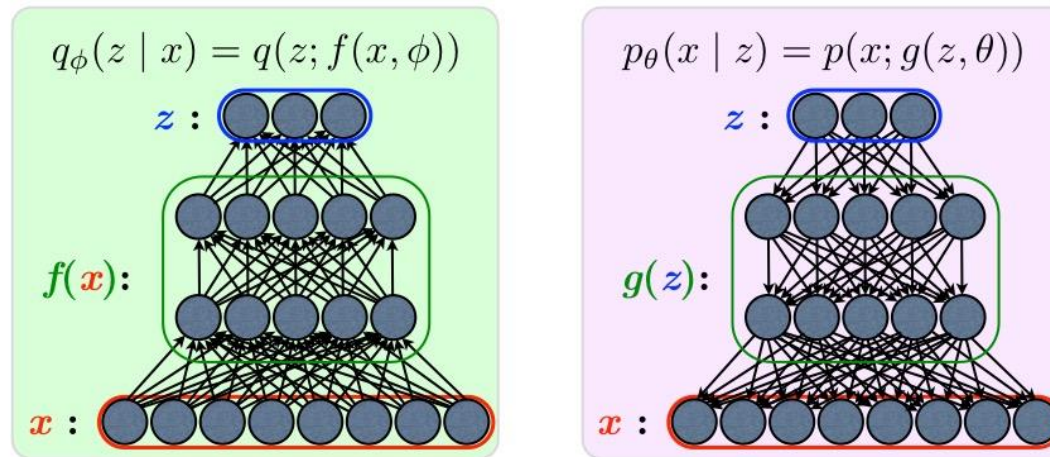


Bishop – Pattern Recognition and Machine Learning

- After applying the E step, we increase the likelihood of the data by finding better parameters according to:  $\theta^{(t)} \leftarrow \mathbf{argmax}_{\theta} \mathbb{E}_{q^{(t)}(z)} [\log p(x, z | \theta)]$

# VAE learning

- What is  $q_\phi(z|x)$  ?
  - Parametrize  $q_\phi(z|x)$  with another neural network



- Interpreting VAE objective

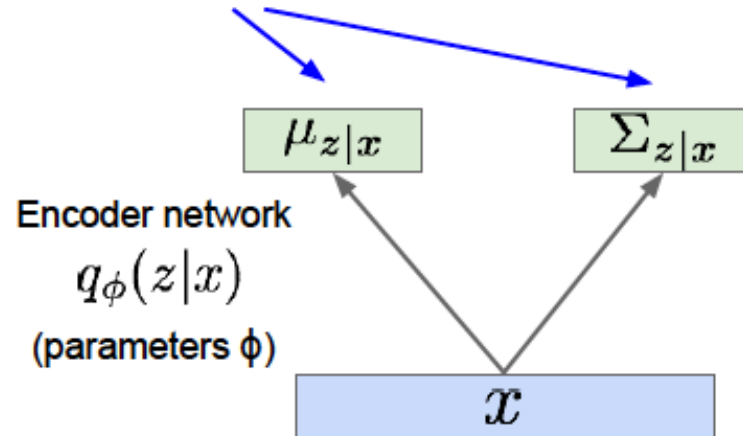
$$\begin{aligned}\mathcal{L}(\theta, \phi, x) &= \mathbb{E}_{q_\phi(z|x)} [\log p_\theta(x, z) - \log q_\phi(z|x)] \\ &= \mathbb{E}_{q_\phi(z|x)} [\log p_\theta(x|z) + \log p_\theta(z) - \log q_\phi(z|x)] \\ &= \underbrace{-D_{\text{KL}}(q_\phi(z|x) \| p_\theta(z))}_{\text{regularization term}} + \underbrace{\mathbb{E}_{q_\phi(z|x)} [\log p_\theta(x|z)]}_{\text{reconstruction term}}\end{aligned}$$



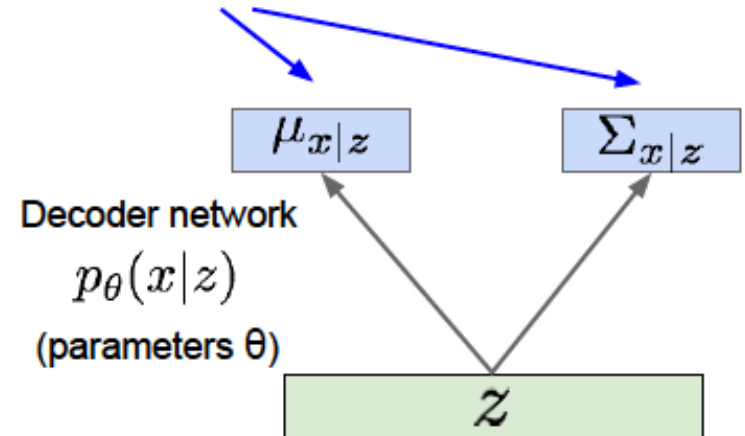
# VAE Example

- Conditionals Gaussians

Mean and (diagonal) covariance of  $z | x$



Mean and (diagonal) covariance of  $x | z$

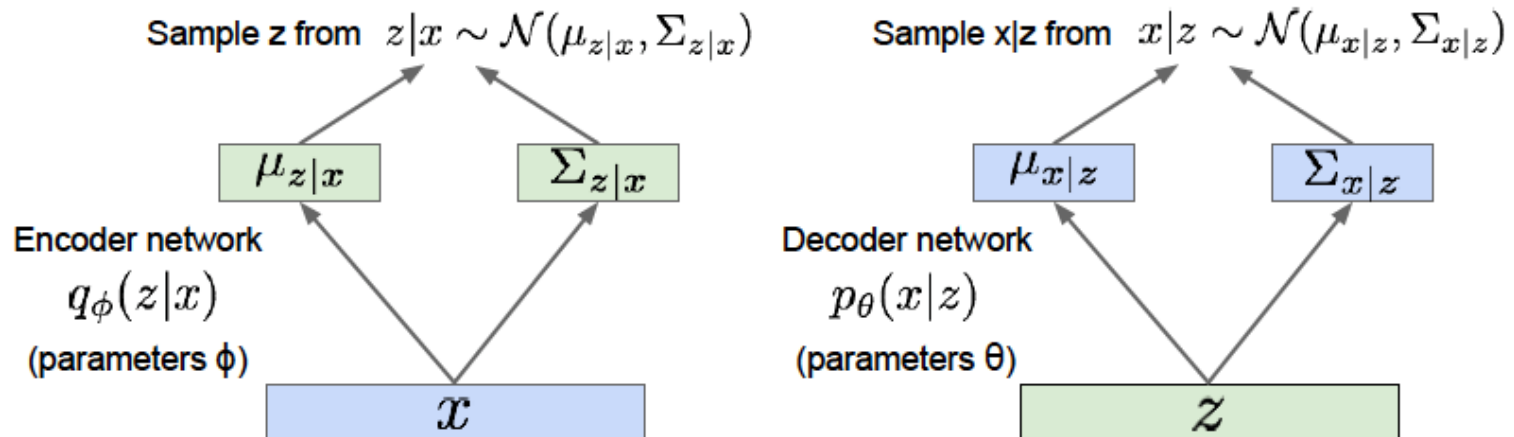


# VAE Example

- Conditionals Gaussians
- Jointly train encoder  $q$  and decoder  $p$  to maximize the Evidence Lower Bound (ELBO)

$$\log p_{\theta}(x) \geq E_{z \sim q_{\phi}(z|x)} [\log p_{\theta}(x|z)] - D_{KL} \left( q_{\phi}(z|x), p(z) \right)$$

Since we're modeling probabilistic generation of data, encoder and decoder networks are probabilistic



Encoder and decoder networks also called  
"recognition"/"inference" and "generation" networks

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

# VAE Example

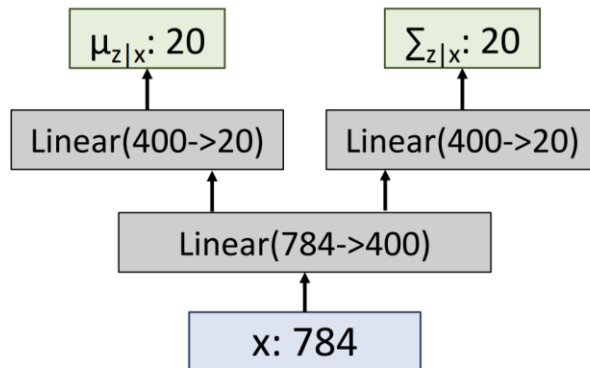
- Toy example: Fully-Connected VAE

x: 28x28 image, flattened to 784-dim vector

z: 20-dim vector

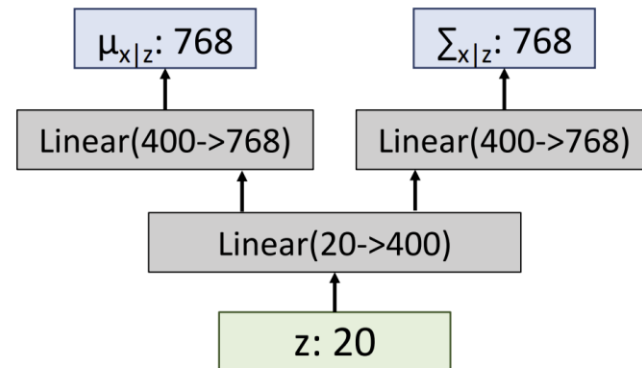
## Encoder Network

$$q_{\phi}(z | x) = N(\mu_{z|x}, \Sigma_{z|x})$$



## Decoder Network

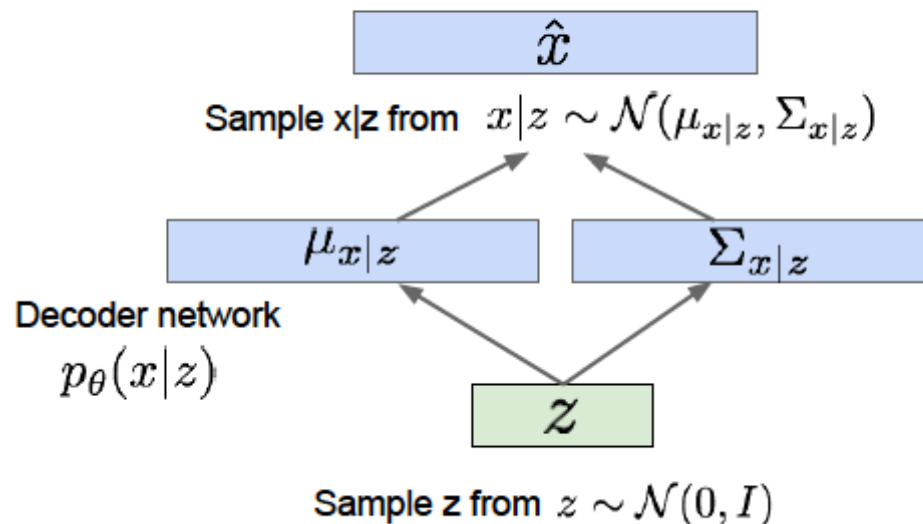
$$p_{\theta}(x | z) = N(\mu_{x|z}, \Sigma_{x|z})$$



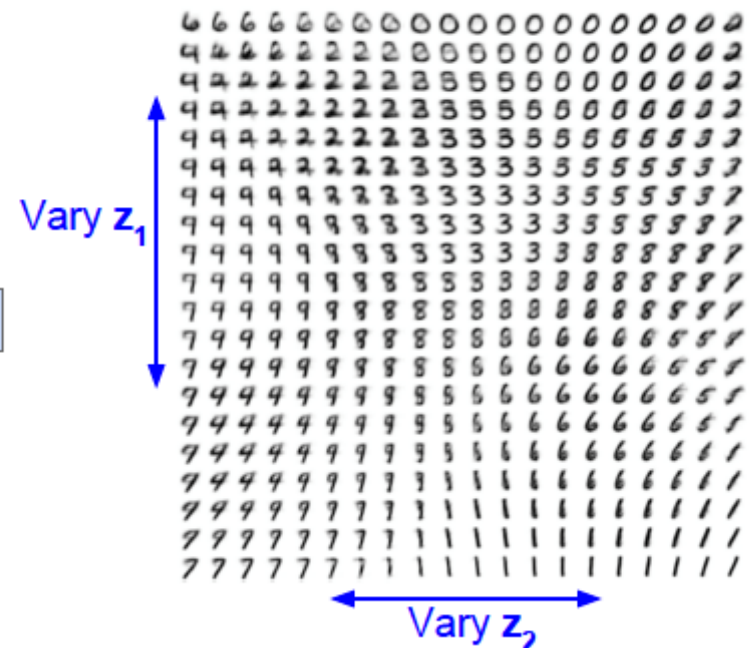
# VAE Example

## ■ Generating data

Use decoder network. Now sample  $z$  from prior!



Data manifold for 2-d  $z$

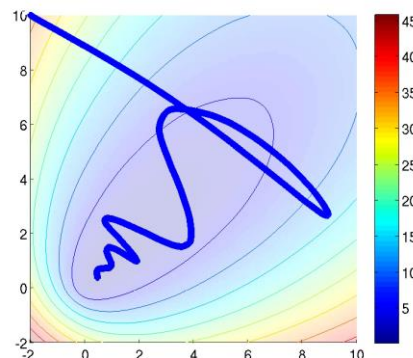


Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

# Two views of Learning VAE

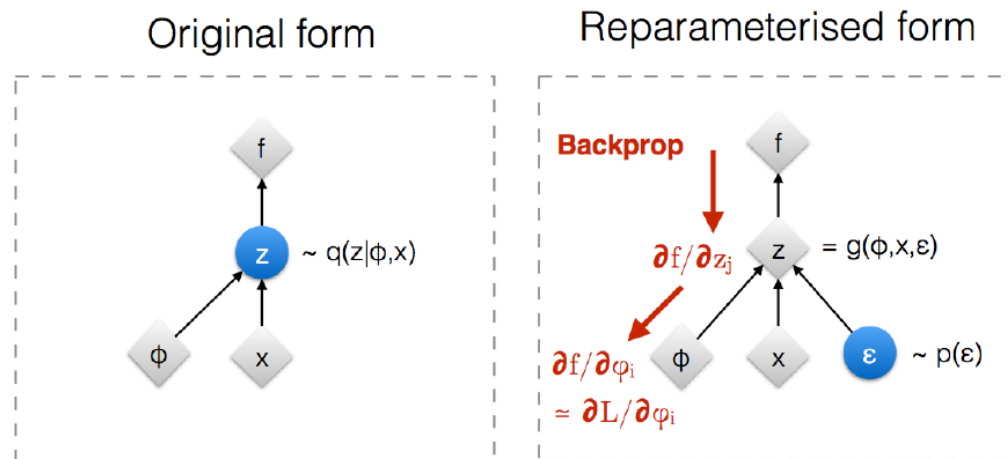
## ■ Optimization interpretation

- Stochastic gradient-based



## ■ Network interpretation

- Backpropagation



# Optimization interpretation

- Recall VAE objective

$$\mathcal{L}(x, \phi, \theta) = E_{q_{\phi}(z|x)}[\log p_{\theta}(x, z) - \log q_{\phi}(z|x)]$$

- Or rewrite as  $\mathcal{L}(x, \phi, \theta) = E_{q_{\phi}(z|x)}[f_{\phi, \theta}(x, z)]$

- Often no analytic solution to exact gradient

$$\nabla_{\phi, \theta} \mathcal{L}(x, \phi, \theta)$$

- Solution: stochastic gradient ascent
  - Requires unbiased estimates of gradient
  - Can use small minibatches or single point of data

$$\nabla_{\phi} \mathcal{L}(x, \phi, \theta) \approx \nabla_{\phi} f_{\phi, \theta}(x, z^{(i)}), \quad z^{(i)} \sim q_{\phi}(z|x)$$

High variance for gradient estimation

# Reparameterization trick

- Reparameterize  $\mathbf{z}^{(i)} \sim q_\phi(\mathbf{z}|\mathbf{x})$  using a differentiable transformation of an auxiliary noise variable  $\epsilon$

$$\mathbf{z} = g_\phi(\epsilon, \mathbf{x}) \quad \text{with} \quad \epsilon \sim q(\epsilon)$$

- Then we can write the ELBO as

$$\mathcal{L}(x, \phi, \theta) = E_{q_\phi(z|x)}[f_{\phi, \theta}(x, z)] = E_{q(\epsilon)}[f_{\phi, \theta}(x, g_\phi(\epsilon, \mathbf{x}))]$$

- And its gradient estimation with L samples

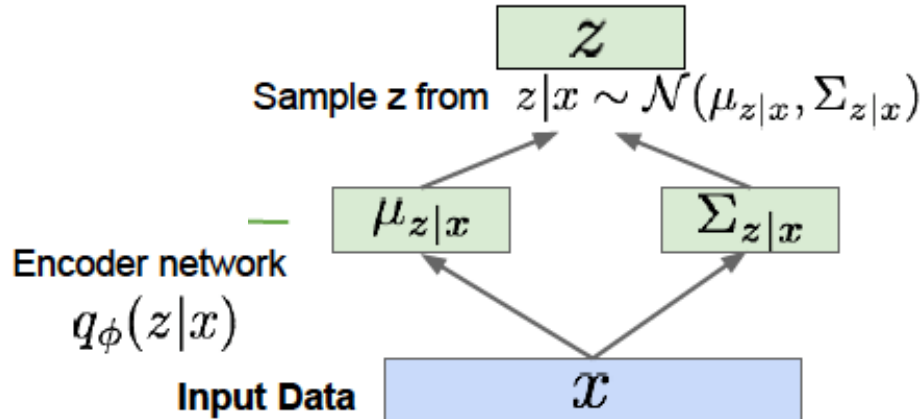
$$\nabla_\phi \mathcal{L}(x, \phi, \theta) = E_{q(\epsilon)}[\nabla_\phi f_{\phi, \theta}(x, z)] \approx \frac{1}{L} \sum_{i=1}^L \nabla_\phi f_{\phi, \theta}(x, g_\phi(\epsilon^{(i)}, x)), \quad \epsilon^{(i)} \sim q(\epsilon)$$

# VAE Example

- Univariate Gaussian  $z \sim p(z|x) = \mathcal{N}(\mu, \sigma^2)$

$$z = \mu + \sigma\epsilon \quad \epsilon \sim \mathcal{N}(0, 1)$$

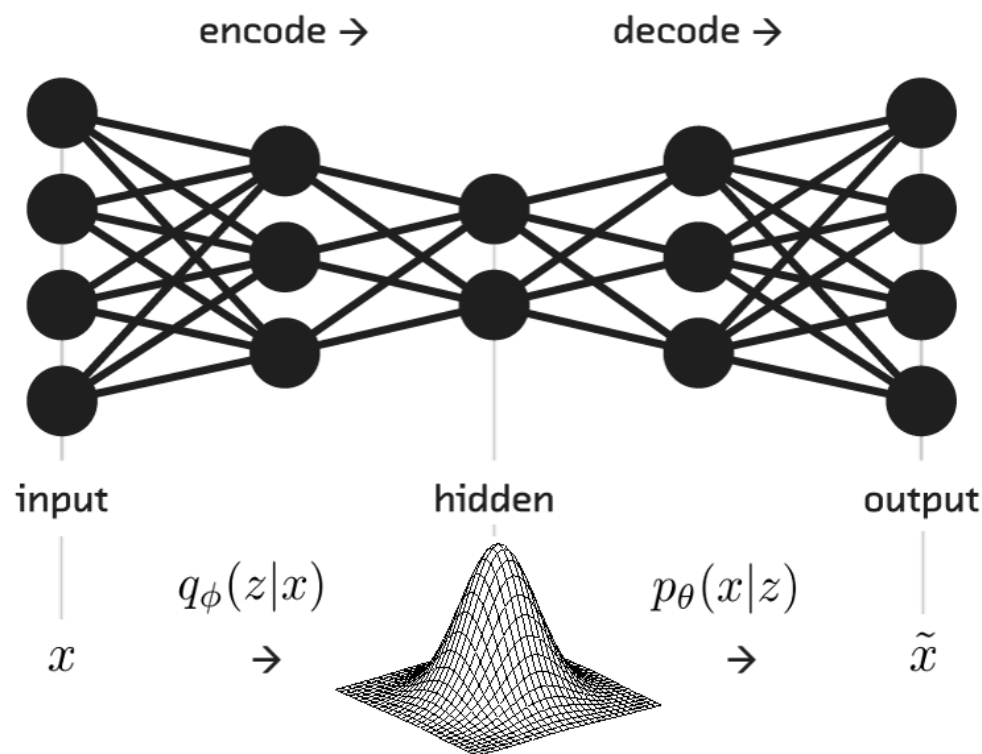
$$\mathbb{E}_{\mathcal{N}(z;\mu,\sigma^2)} [f(z)] = \mathbb{E}_{\mathcal{N}(\epsilon;0,1)} [f(\mu + \sigma\epsilon)] \simeq \frac{1}{L} \sum_{l=1}^L f(\mu + \sigma\epsilon^{(l)})$$





# Autoencoder Interpretation

- Objective  $\mathcal{L}(x, \phi, \theta) = -D_{KL}(q_{\phi}(z|x)||p_{\theta}(z)) + E_{q_{\phi}(z|x)}[\log p_{\theta}(x|z)]$   
Regularization term      Reconstruction term



# VAE Example

## ■ Learning objective

Putting it all together: maximizing the likelihood lower bound

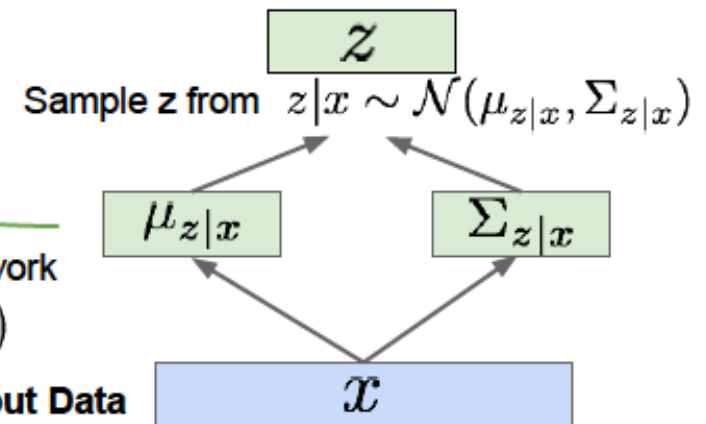
$$\underbrace{\mathbf{E}_z \left[ \log p_\theta(x^{(i)} | z) \right] - D_{KL}(q_\phi(z | x^{(i)}) || p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Make approximate posterior distribution close to prior

Encoder network

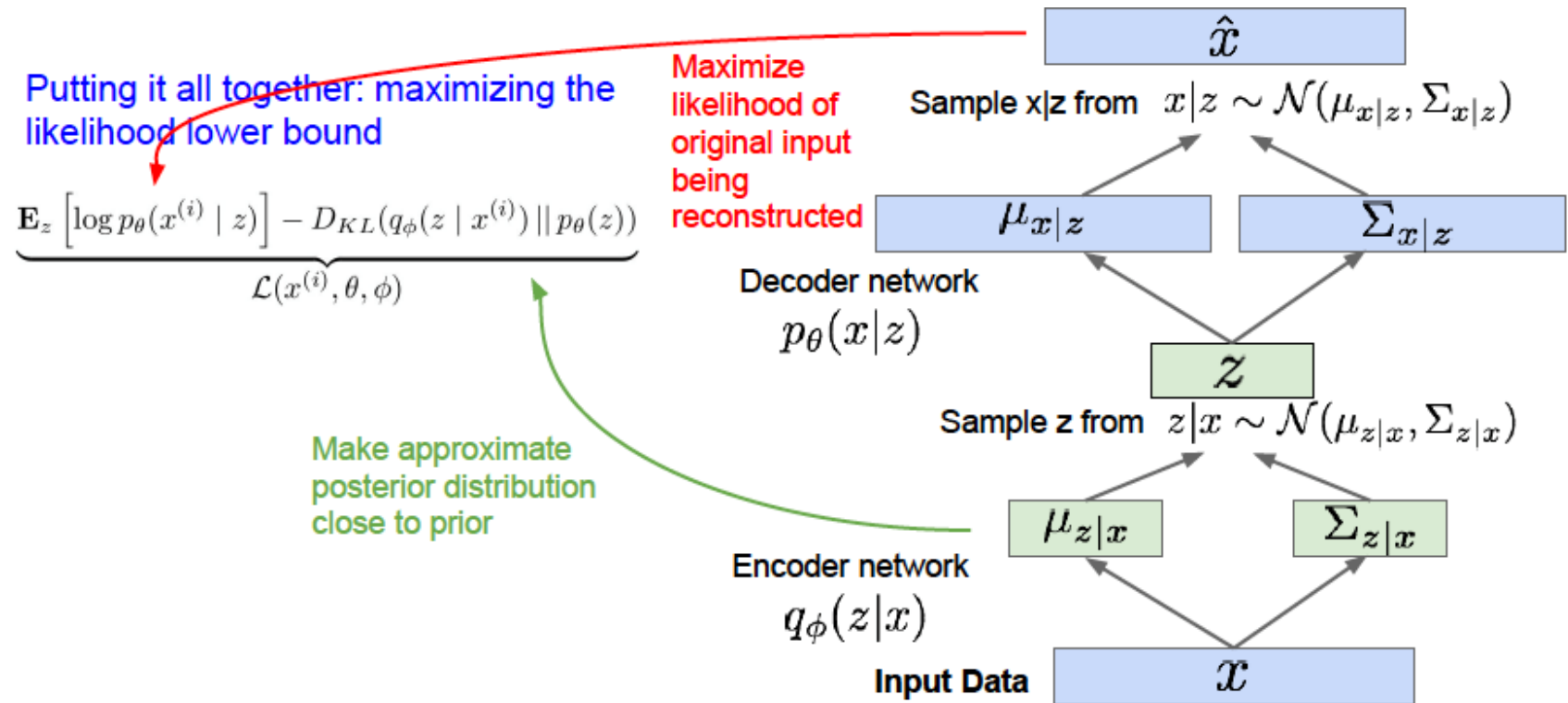
$$q_\phi(z|x)$$

Input Data



# VAE Example

## ■ Learning objective



# Variational Autoencoders

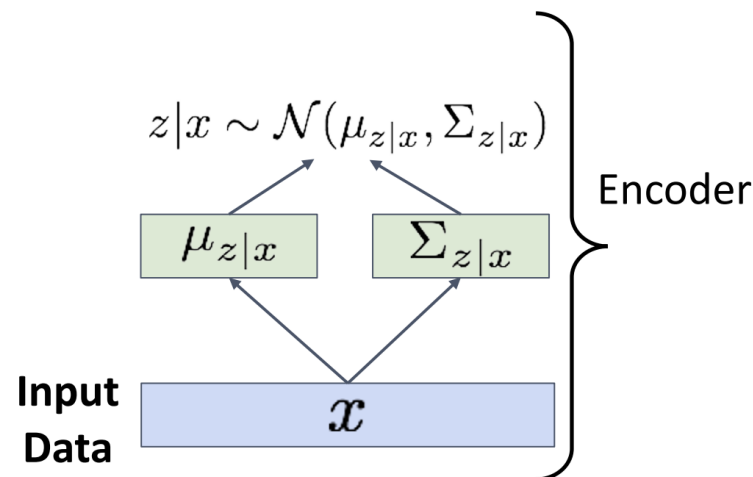
- Train by maximizing the variational lower bound

$$E_{z \sim q_\phi(z|x)} [\log p_\theta(x|z)] - \boxed{D_{KL}(q_\phi(z|x), p(z))}$$

1. Run input data through **encoder** to get a distribution over latent codes
2. **Encoder output should match the prior  $p(z)$ !**

$$\begin{aligned} -D_{KL}(q_\phi(z|x), p(z)) &= \int_z q_\phi(z|x) \log \frac{p(z)}{q_\phi(z|x)} dz \\ &= \int_z N(z; \mu_{z|x}, \Sigma_{z|x}) \log \frac{N(z; 0, I)}{N(z; \mu_{z|x}, \Sigma_{z|x})} dz \\ &= \frac{1}{2} \sum_{j=1}^J \left( 1 + \log((\Sigma_{z|x})_j^2) - (\mu_{z|x})_j^2 - (\Sigma_{z|x})_j^2 \right) \end{aligned}$$

Closed form solution when  $q_\phi$  is diagonal Gaussian and  $p$  is unit Gaussian!  
(Assume  $z$  has dimension  $J$ )

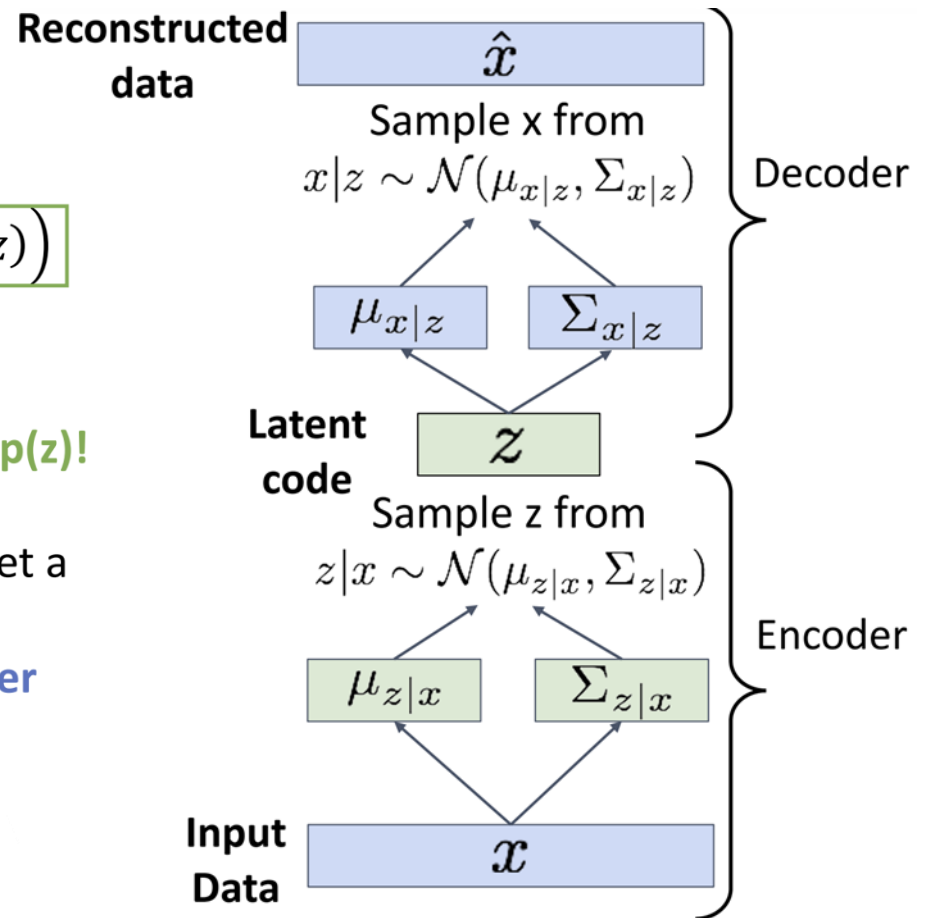


# Variational Autoencoders

- Train by maximizing the variational lower bound

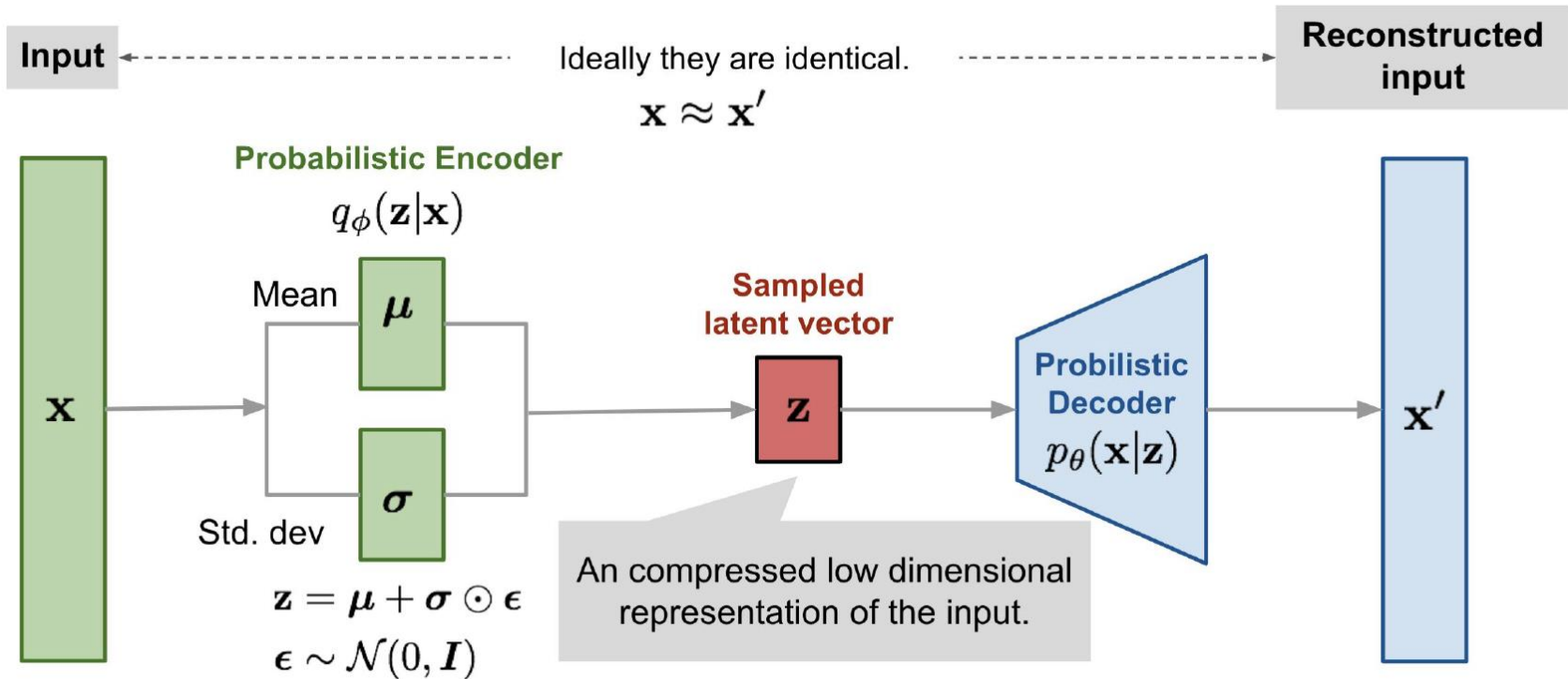
$$E_{z \sim q_\phi(z|x)} [\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x), p(z))$$

1. Run input data through **encoder** to get a distribution over latent codes
2. **Encoder output should match the prior  $p(z)$ !**
3. Sample code  $z$  from encoder output
4. Run sampled code through **decoder** to get a distribution over data samples
5. **Original input data should be likely under the distribution output from (4)!**
6. Can sample a reconstruction from (4)



# Autoencoder Interpretation

- Objective  $\mathcal{L}(x, \phi, \theta) = \underbrace{-D_{KL}(q_{\phi}(z|x)||p_{\theta}(z))}_{\text{Regularization term}} + \underbrace{E_{q_{\phi}(z|x)}[\log p_{\theta}(x|z)]}_{\text{Reconstruction term}}$



- The objective function can be represented as an Autoencoder-like **computation graph**.

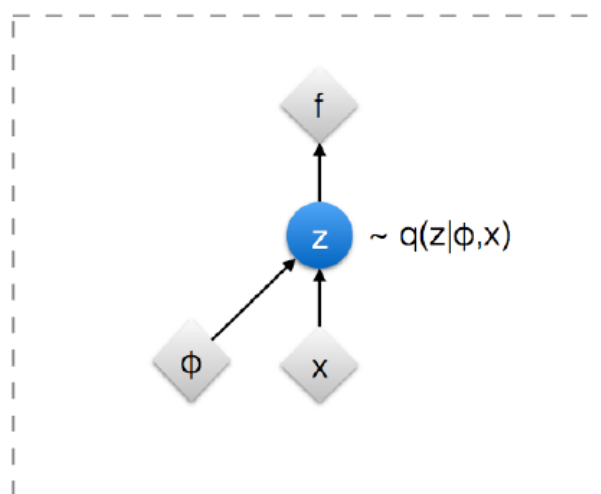
# Network interpretation

$$\mathcal{L}(x, \phi, \theta) = E_{q_{\phi}(z|x)}[f_{\phi, \theta}(x, z)]$$

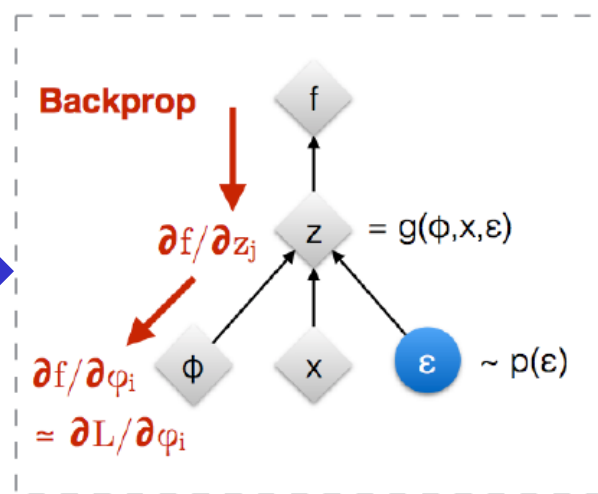


$$\mathcal{L}(x, \phi, \theta) = E_{q(\epsilon)}[f_{\phi, \theta}(x, z)] \approx \frac{1}{L} \sum_{i=1}^L f_{\phi, \theta}(x, g_{\phi}(\epsilon^{(i)}, x)), \quad \epsilon^{(i)} \sim q(\epsilon)$$

Original form



Reparameterised form



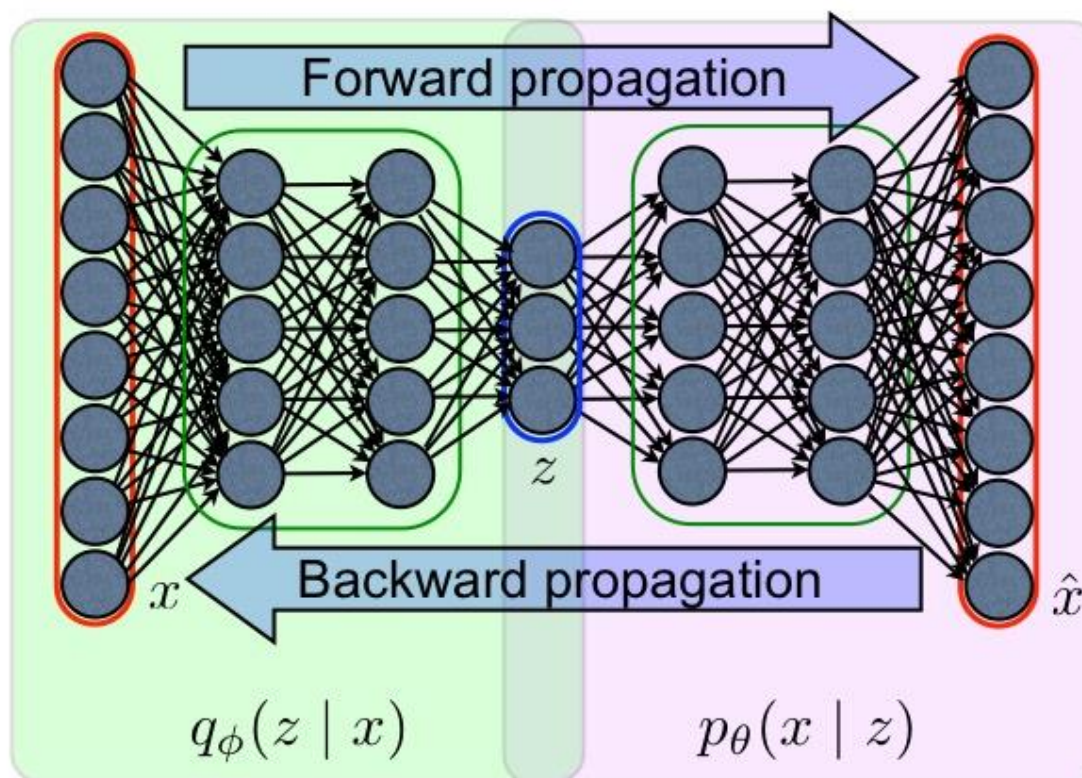
◊ : Deterministic node

● : Random node

[Kingma, 2013]  
[Bengio, 2013]  
[Kingma and Welling 2014]  
[Rezende et al 2014]

# Training with Backpropagation

- Due to **reparametrization** trick, we can simultaneously train both the **generative model** and the **inference model** by optimizing the variational bound using the gradient **backpropagation**.





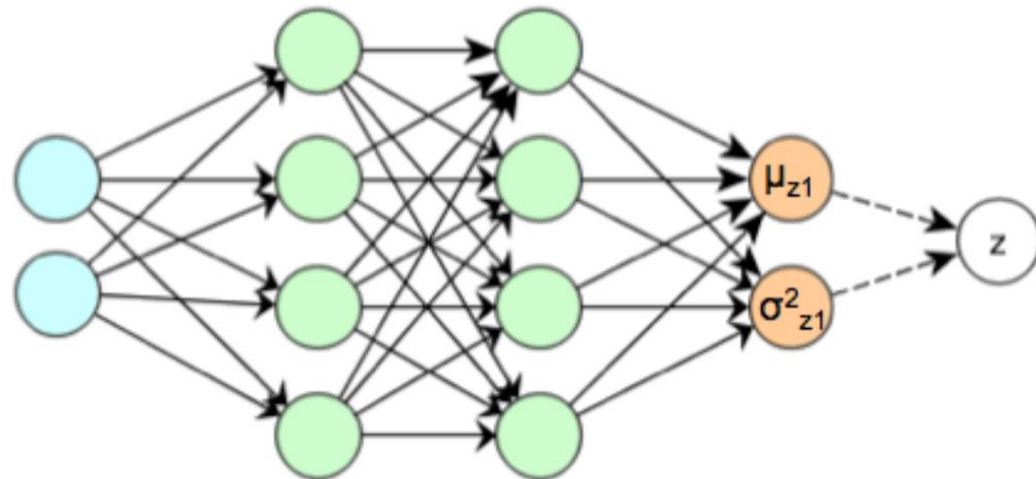
# 1D Gaussian Case

- We can compute the KL regularization in close form

Use  $N(0,1)$  as prior for  $p(z)$

$q(z|x^{(i)})$  is Gaussian with parameters  $(\mu^{(i)}, \sigma^{(i)})$  determined by NN

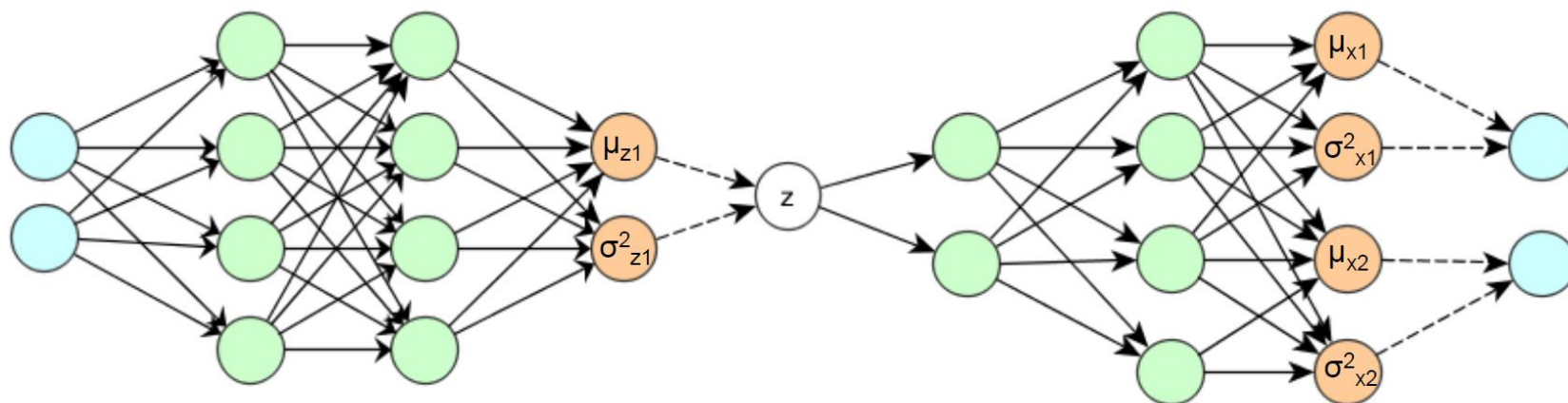
$$-D_{\text{KL}}(q(z|x^{(i)})||p(z)) = \frac{1}{2} \sum_{j=1}^J \left( 1 + \log(\sigma_{z_j}^{(i)^2}) - \mu_{z_j}^{(i)^2} - \sigma_{z_j}^{(i)^2} \right)$$



# 1D Gaussian Case

## Overall loss function for BP

Prior  $p(z) \sim N(0,1)$  and  $p, q$  Gaussian, extension to  $\dim(z) > 1$  trivial



Cost: Regularisation

$$-D_{\text{KL}}(q(z|x^{(i)})||p(z)) = \frac{1}{2} \sum_{j=1}^J \left( 1 + \log(\sigma_{z_j}^{(i)^2}) - \mu_{z_j}^{(i)^2} - \sigma_{z_j}^{(i)^2} \right)$$

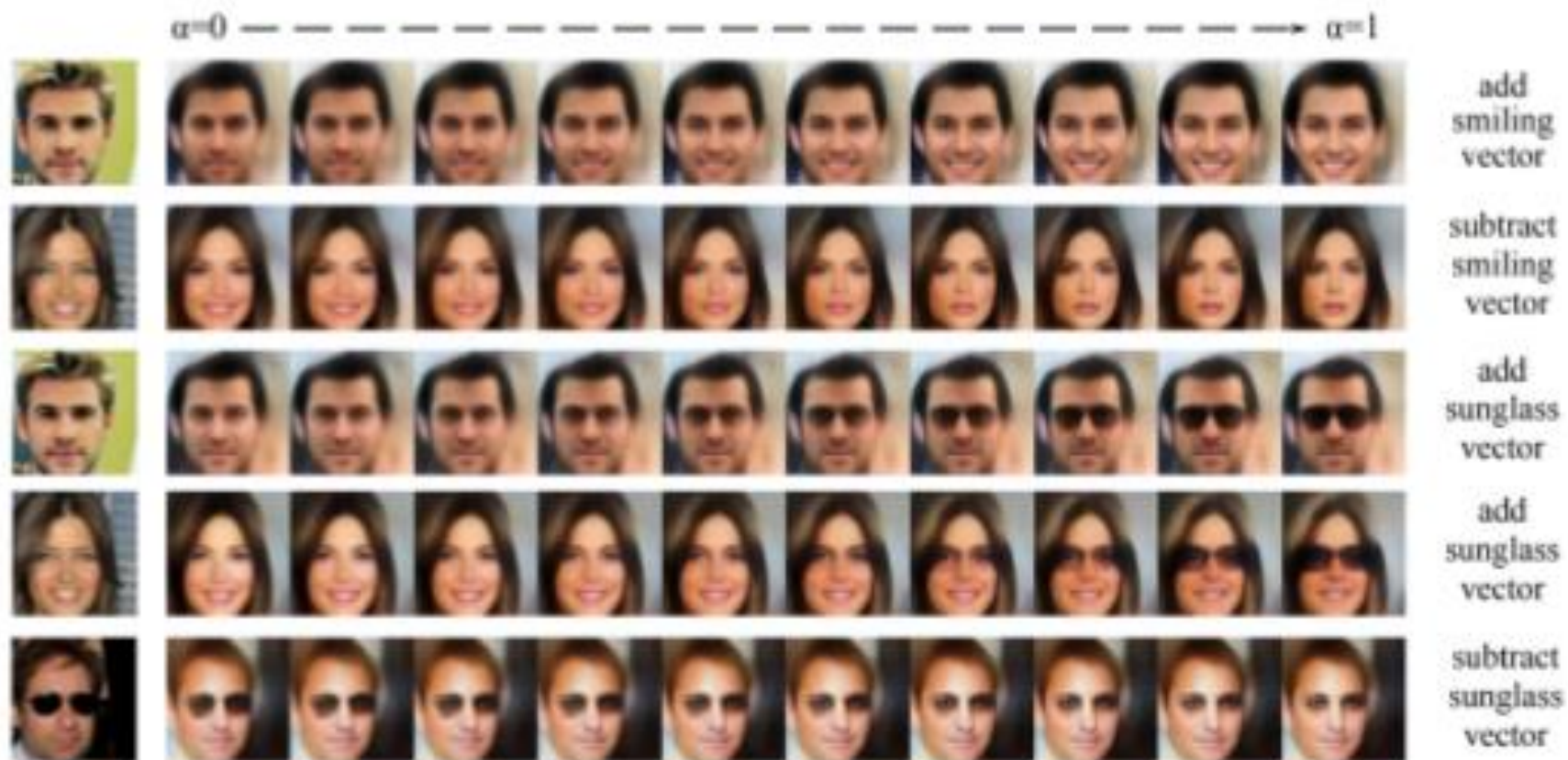
We use mini batch gradient decent to optimize the cost function over all  $x^{(i)}$  in the mini batch

Cost: Reproduction

$$-\log(p(x^{(i)}|z^{(i)})) = \sum_{j=1}^D \frac{1}{2} \log(\sigma_{x_j}^2) + \frac{(x_j^{(i)} - \mu_{x_j})^2}{2\sigma_{x_j}^2}$$

Least Square for constant variance

# Interpreting the latent space



<https://arxiv.org/pdf/1610.00291.pdf>

# Problems of VAE

## ■ Model capacity

- Note that the VAE requires 2 tractable distributions to be used:
  - The prior distribution  $p(z)$  must be easy to sample from
  - The conditional likelihood  $p(x|z, \theta)$  must be computable
- In practice this means that the 2 distributions of interest are often simple, for example uniform, Gaussian, or even isotropic Gaussian



# Problems of VAE

## ■ Blurry images



<https://blog.openai.com/generative-models/>

- The samples from the VAE look blurry
- Three plausible explanations for this
  - Maximizing the likelihood
  - Restrictions on the family of distributions
  - The lower bound approximation

# Problems of VAE

## ■ Blurry images

- Recent investigations suggest that both the simple probability distributions and the variational approximation lead to blurry images
- [Kingma & colleagues: Improving Variational Inference with Inverse Autoregressive Flow](#)
- [Zhao & colleagues: Towards a Deeper Understanding of Variational Autoencoding Models](#)
- [Nowozin & colleagues: f-gan: Training generative neural samplers using variational divergence minimization](#)

# Summary

- Autoencoders (AEs)
  - Representation learning
- Variational Autoencoders (VAEs)
  - VAE objective
  - Reparametrization trick
- Next time:
  - VAE: Vision applications
  - GAN
- Reading material
  - [http://www.cs.columbia.edu/~blei/talks/Blei\\_VI\\_tutorial.pdf](http://www.cs.columbia.edu/~blei/talks/Blei_VI_tutorial.pdf)
  - [http://www.cs.toronto.edu/~rgrosse/courses/csc421\\_2019/slides/lec17.pdf](http://www.cs.toronto.edu/~rgrosse/courses/csc421_2019/slides/lec17.pdf)
  - <https://dvl.in.tum.de/slides/adl4cv-ws18/6.Bayesian&VAE.pdf>
  - <https://arxiv.org/pdf/1312.6114.pdf>

*Acknowledgement: Feifei Li et al's cs231n notes*