Lecture 13: Deep Generative Models II: AE & VAE

Lan Xu
SIST, ShanghaiTech
Fall, 2024

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

$$\begin{aligned}
\log p(\mathbf{x}) &= \log \int_z p(\mathbf{x}, z) \\
&= \log \int_z p(\mathbf{x}, z) \frac{q(\mathbf{z})}{q(\mathbf{z})} \\
&= \log(E_q[\frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z})}]) \\
&\geq E_q[\log p(\mathbf{x}, \mathbf{z})] - E_q[\log q(\mathbf{z})] \\
&= ELBO
\end{aligned}$$

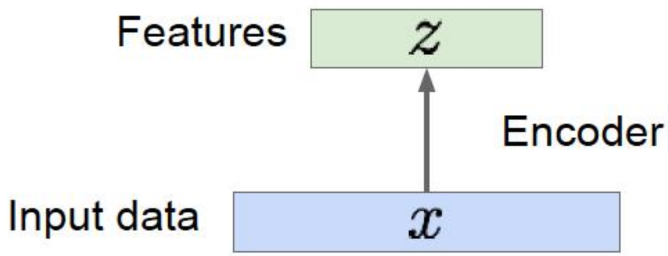
$$\begin{aligned}
KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})) &= E_q[\log q(\mathbf{z})] - E_q[\log p(\mathbf{z}|\mathbf{x})] \\
&= E_q[\log q(\mathbf{z})] - E_q[\log p(\mathbf{z}, \mathbf{x})] + \log p(\mathbf{x}) \\
&= \log p(\mathbf{x}) - (E_q[\log p(\mathbf{z}, \mathbf{x})] - E_q[\log q(\mathbf{z})]) \\
&= \log p(\mathbf{x}) - ELBO
\end{aligned}$$

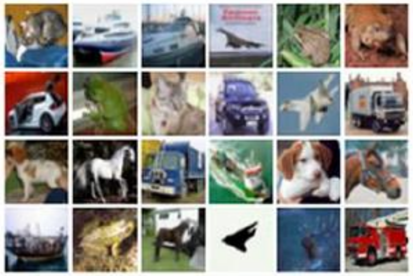
$$\begin{aligned}
EBLO &= E_q[\log p(\mathbf{x}, \mathbf{z})] - E_q[\log q(\mathbf{z})] \\
&= E_q[\log \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{z})}] - E_q[\log \frac{q(\mathbf{z})}{p(\mathbf{z})}] \\
&= E_q[\log p(\mathbf{x}|\mathbf{z})] - KL(q(\mathbf{z})||p(\mathbf{z}))
\end{aligned}$$

Autoencoder

- Feature representation learning

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

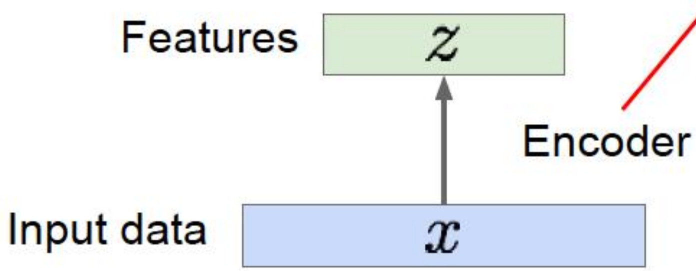


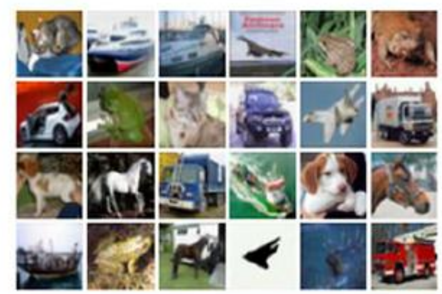


- Feature representation learning

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

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN

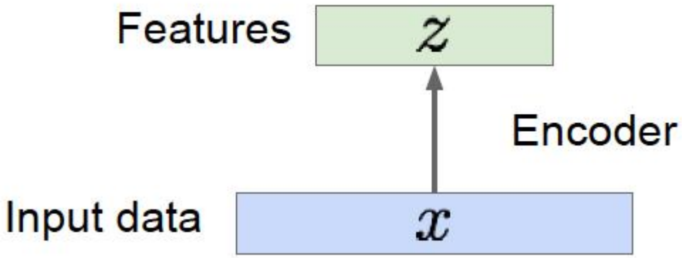


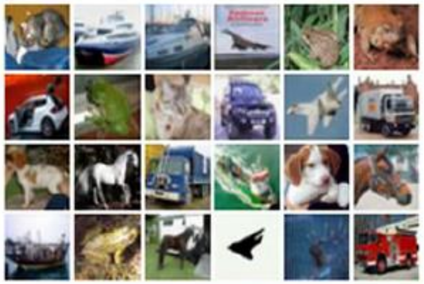


Autoencoder

- Feature representation learning

How to learn this feature representation?

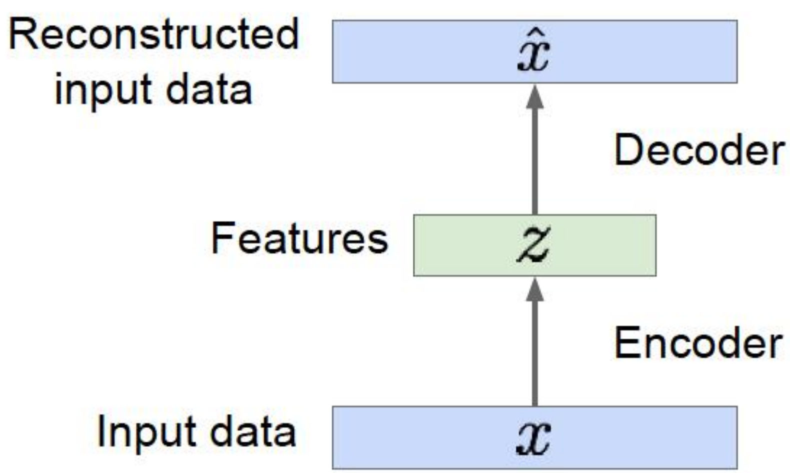




Autoencoder

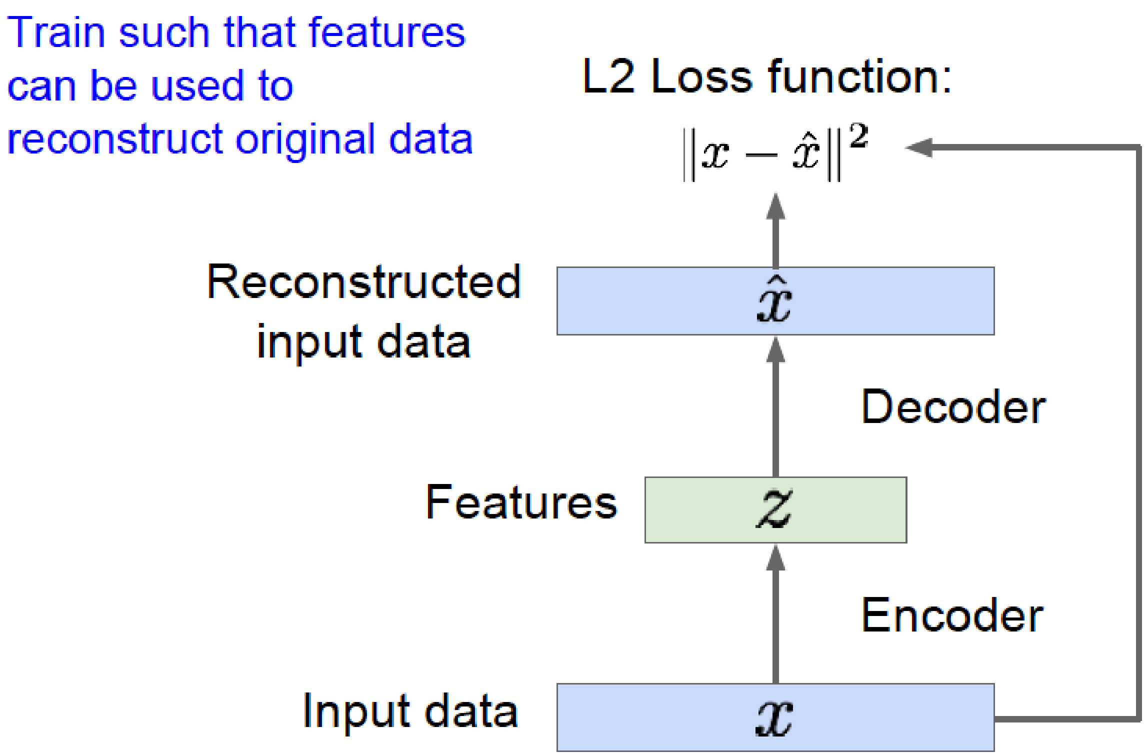
- Feature representation learning

How to learn this feature representation?
Train such that features can be used to reconstruct original data
“Autoencoding” - encoding itself



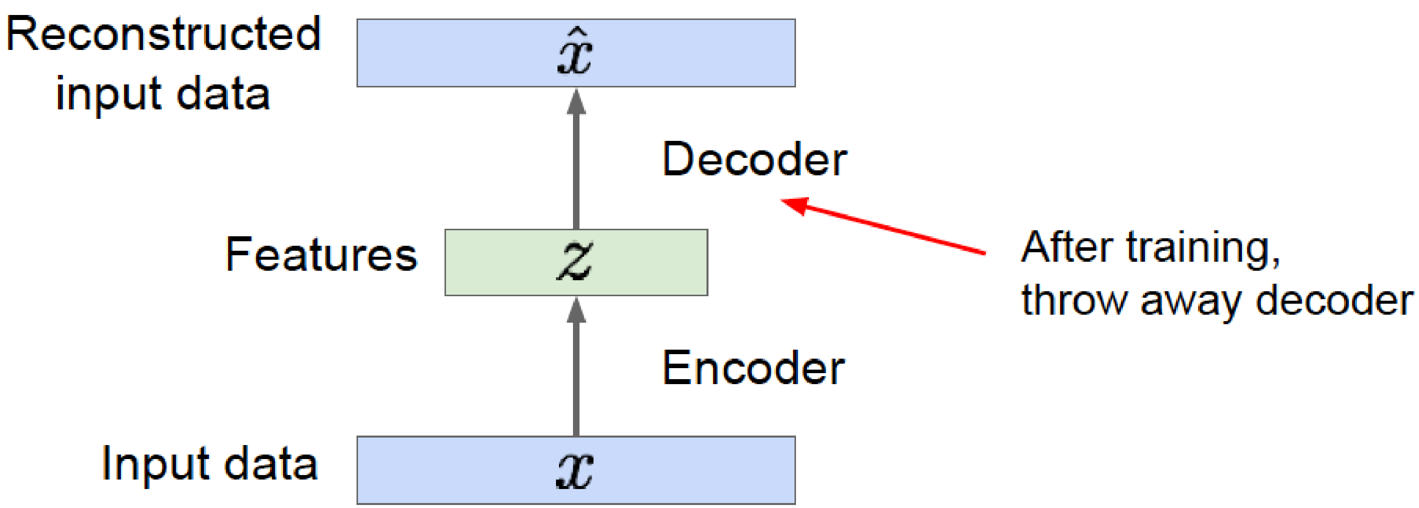
Autoencoder

- Feature representation learning



Autoencoder

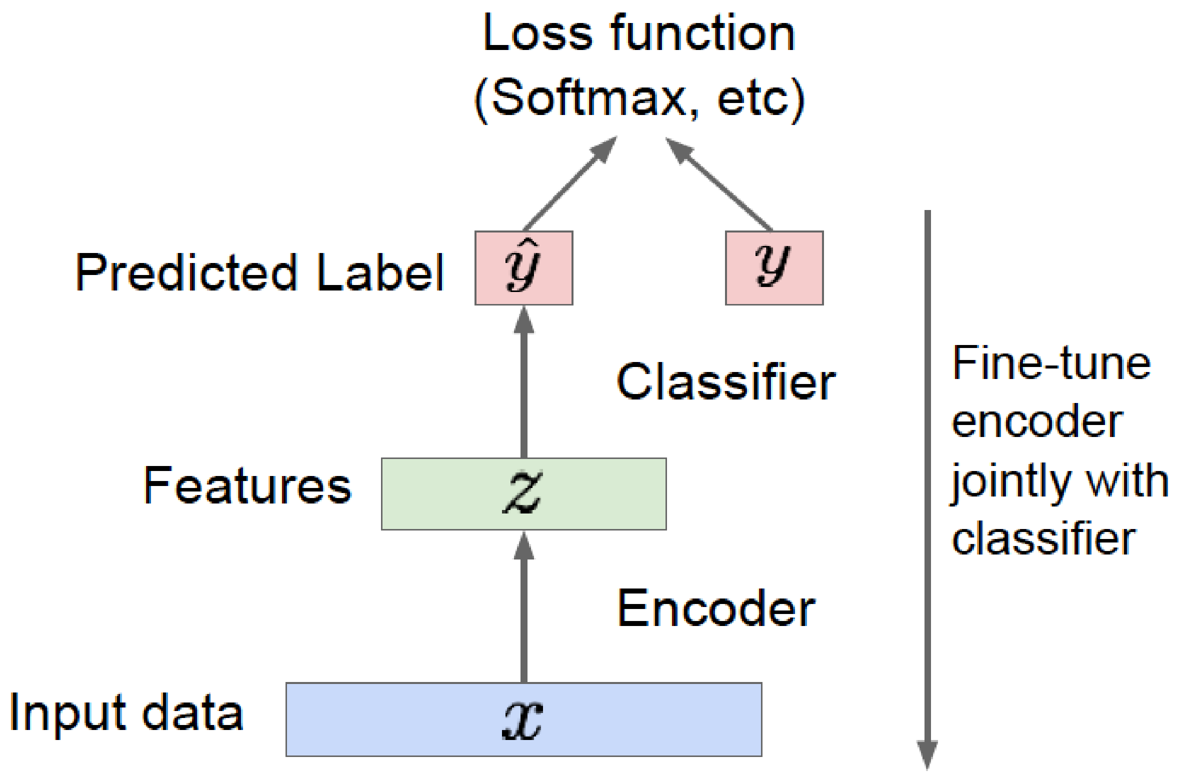
- Feature representation learning



Autoencoder

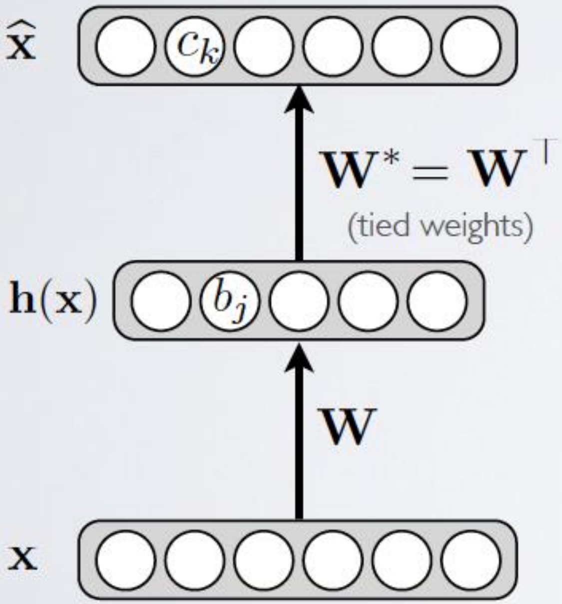
- Feature representation learning
- But not probabilistic: no way to sample new data

Encoder can be used to initialize a **supervised** model



Autoencoder

- Binary input example



Decoder

$$\begin{aligned}\widehat{\mathbf{x}} &= o(\widehat{\mathbf{a}}(\mathbf{x})) \\ &= \underbrace{\mathrm{sigm}}_{\text{for binary inputs}}(\mathbf{c} + \mathbf{W}^*\mathbf{h}(\mathbf{x}))\end{aligned}$$

Encoder

$$\begin{aligned}\mathbf{h}(\mathbf{x}) &= g(\mathbf{a}(\mathbf{x})) \\ &= \mathrm{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x})\end{aligned}$$

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

- 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 (contracitve autoencoder)

Regularization

- Undercomplete representation

• Hidden layer is undercomplete if smaller than the input layer

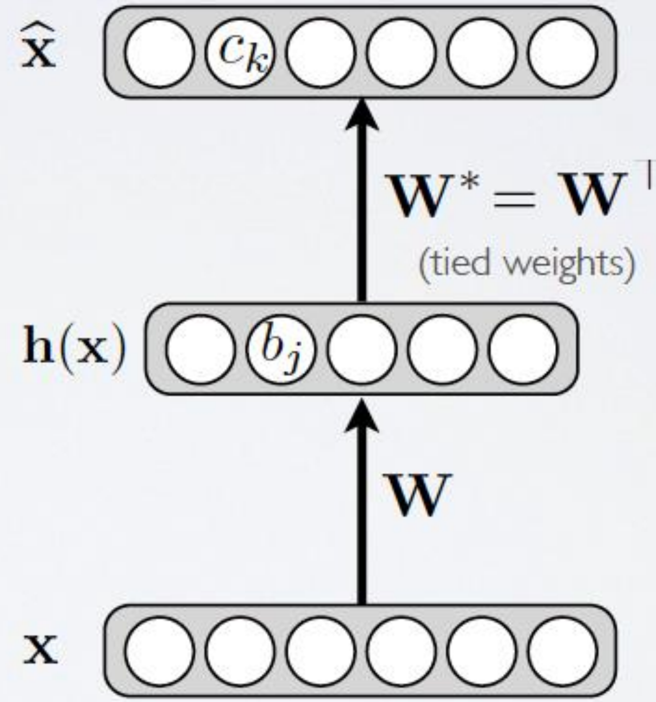
  ‣ hidden layer "compresses" the input

  ‣ will compress well only for the training distribution

• Hidden units will be

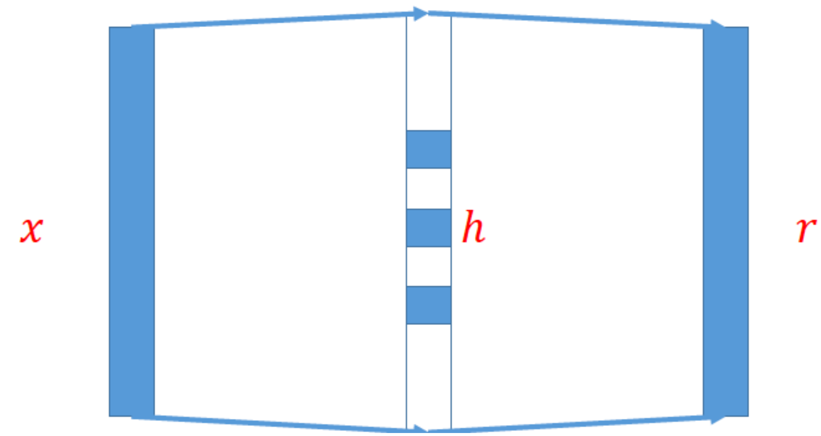
  ‣ good features for the training distribution

  ‣ but bad for other types of input



Regularization

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



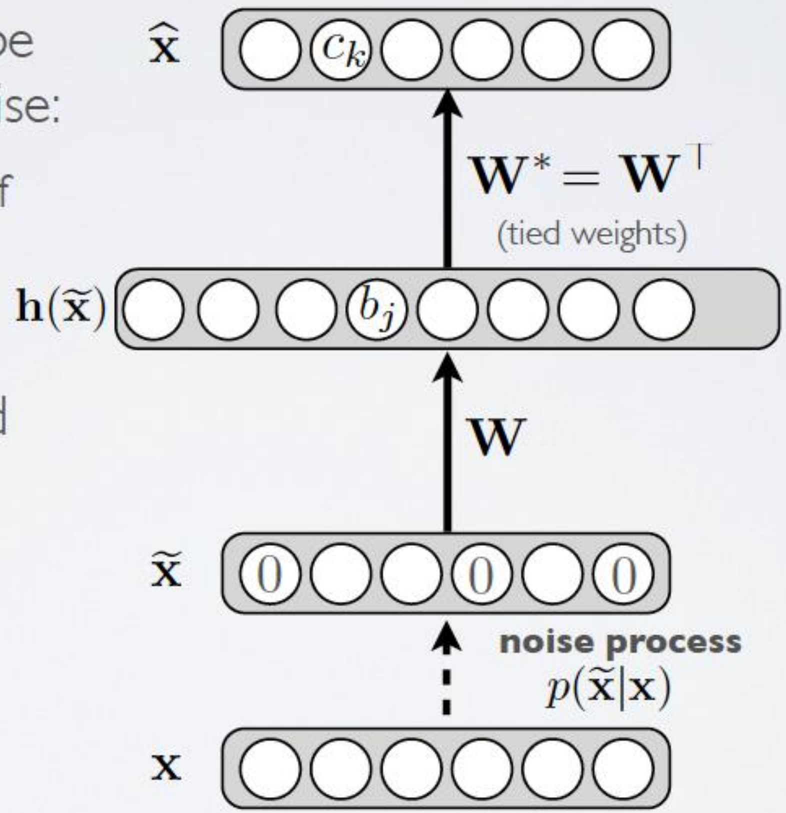
- Sparse autoencoder
  - Constrain the code to have sparsity
  - Training: minimize a loss function

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

Regularization

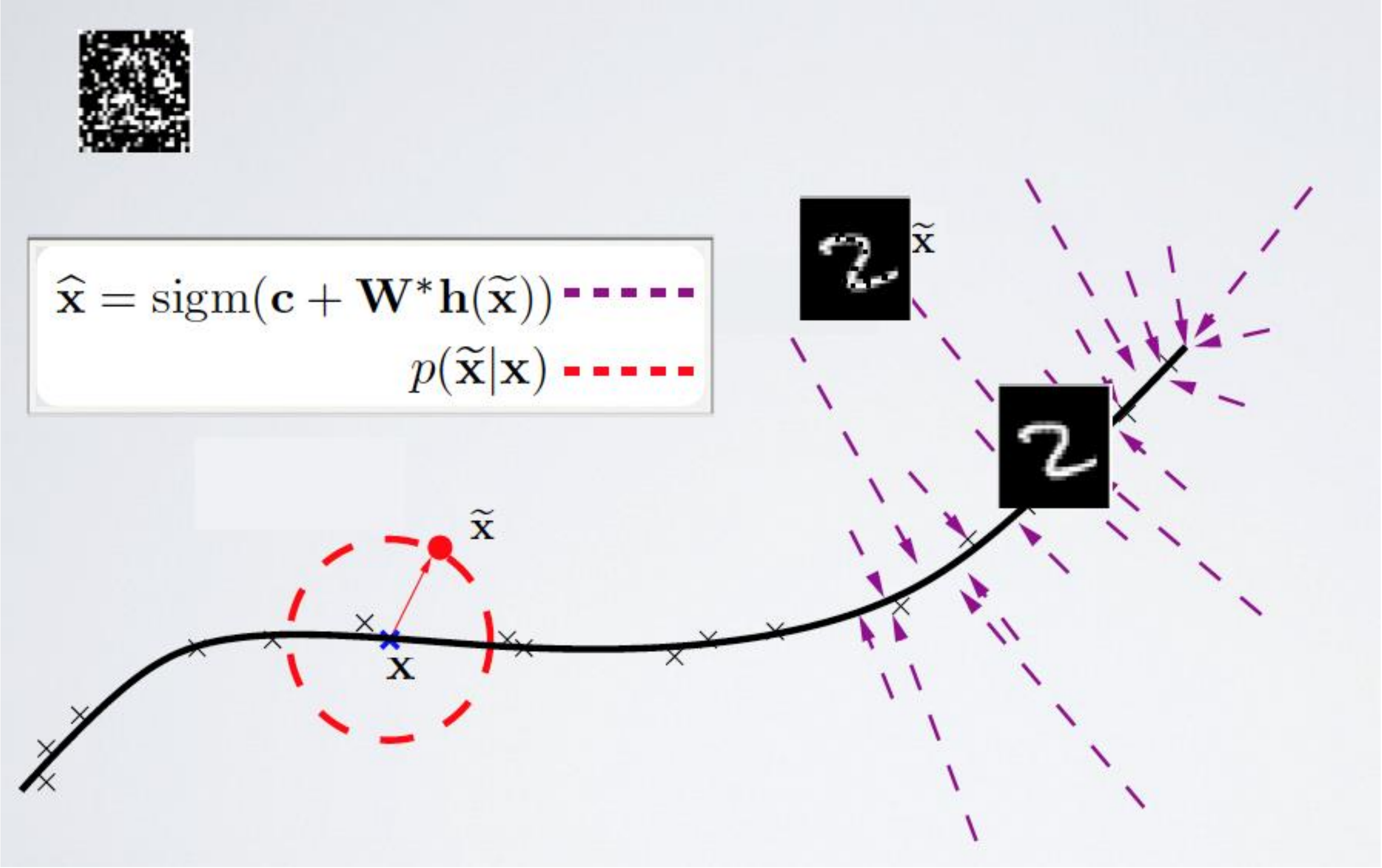
- Denoising autoencoder

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

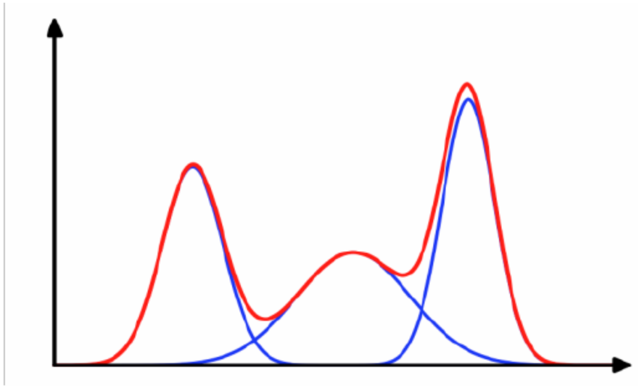
Latent variable model

- Data generation process
  - Latent variable $\boldsymbol{z}$ $\quad p(\boldsymbol{z}) = \text{something simple}$
  - A mapping from the latent space to observation $\boldsymbol{x}$

$$p(x) = \int p(x, z)\ dz \quad \text{where} \quad p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x} \mid \boldsymbol{z})p(\boldsymbol{z})$$

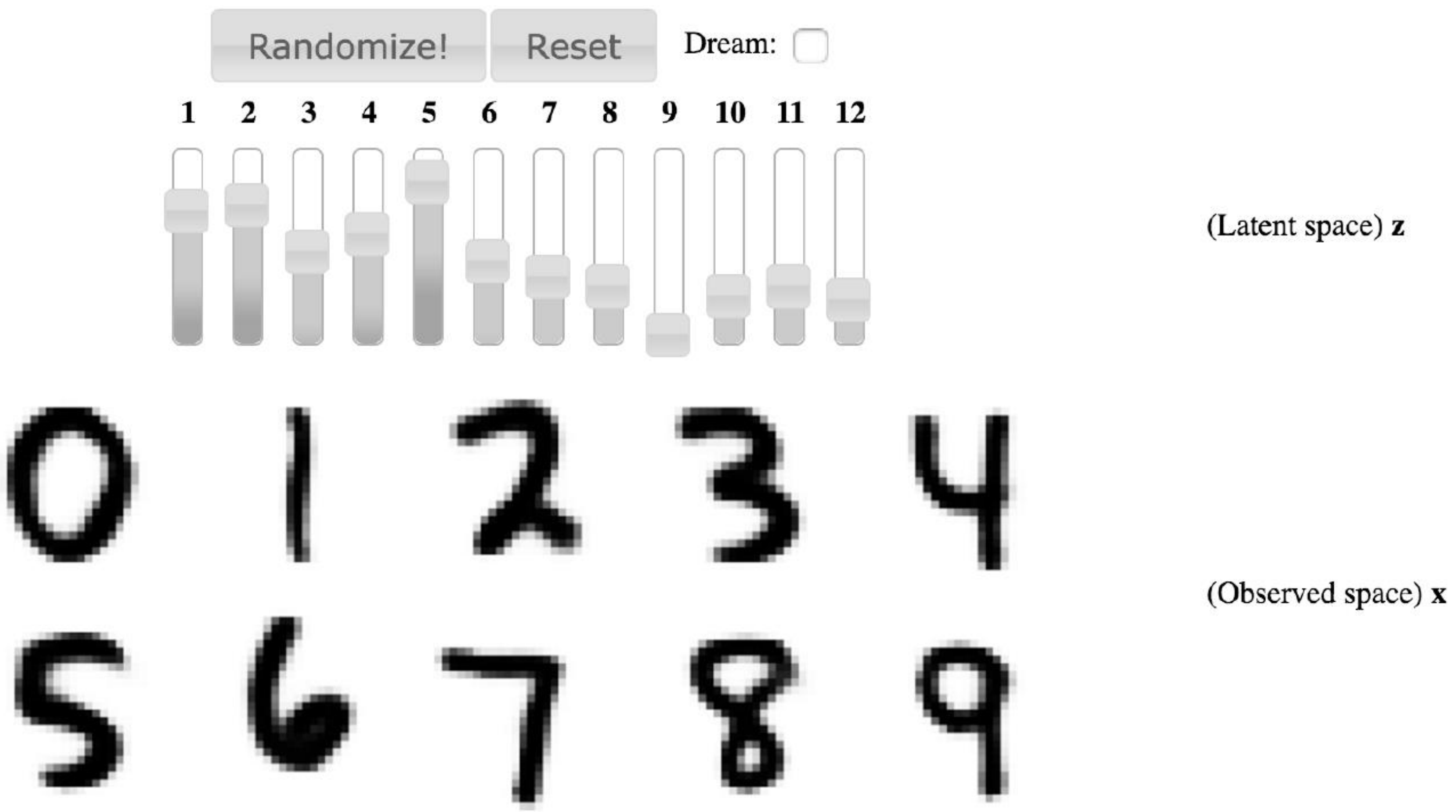
For example, a Gaussian mixture model

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



An example

- Generating hand-written digits

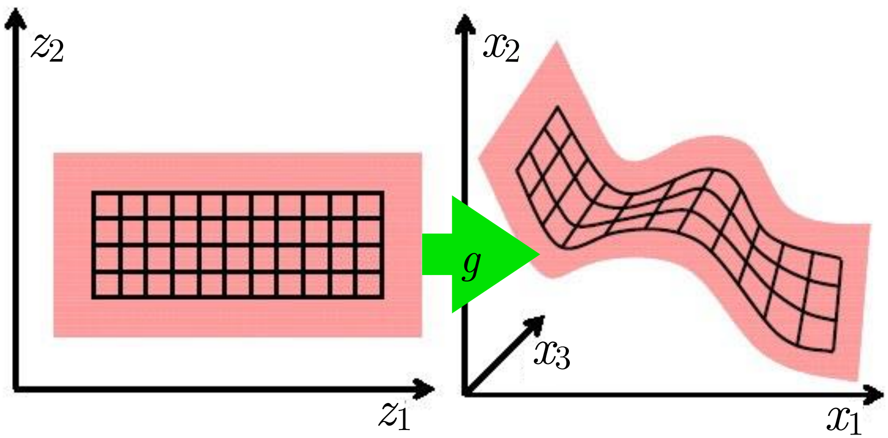


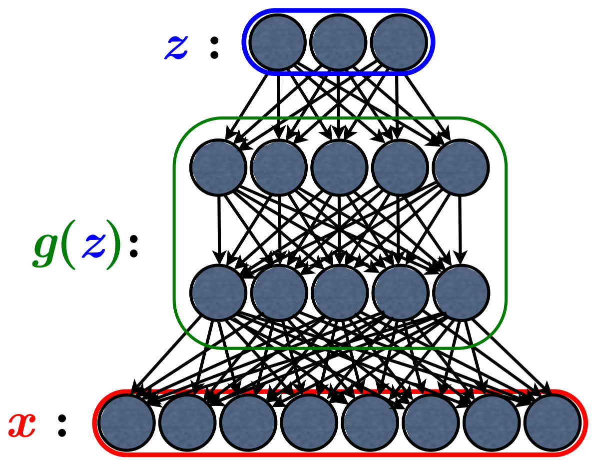
  - http://www.dpkingma.com/sgvb_mnist_demo/demo.html

Deep Latent Variable Models

- Leverage neural networks in a latent variable model

$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x} \mid \boldsymbol{z})p(\boldsymbol{z}) \qquad p(\boldsymbol{x} \mid \boldsymbol{z}) = g(\boldsymbol{z})$$





- Can represent complicated data distribution and conditional dependencies

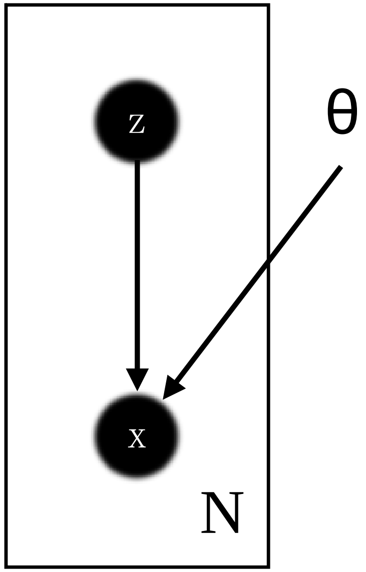
An example

- $p(z) = N(0, I)$
- $P_\theta(x|z) = N(\mu, \sigma^2)$

  $\mu = f_\theta(z) = \text{multilayer neural net}$
- With flexible neural net,

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

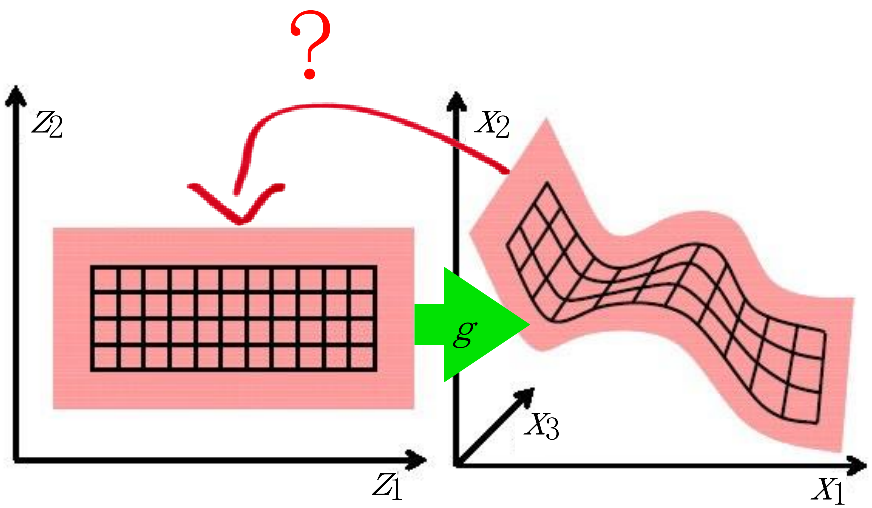
  - Can be arbitrarily complicated
  - The multilayer network can capture complex dependencies in the data generation process

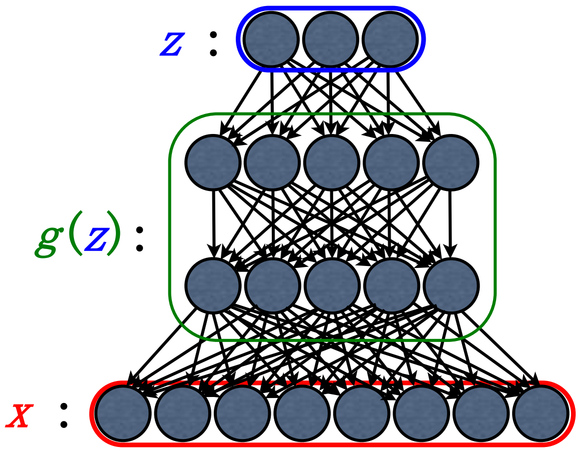


Challenges in Deep LVM

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

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





The Variational Autencoder: overview

- Auto-Encoding Variational Bayes
- ICLR2014, and ICLR 2024 Test of Time Award

Durk Kingma

Max Welling

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

$$\forall x_i \quad \arg\min_{q_i} D_{KL}(q_i(z) || p_\theta(z|x_i))$$

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

  - 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

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

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

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

VAE objective

- From the EM perspective

$$\log p_\theta(x) = \log \frac{p_\theta(x \mid z)p(z)}{p_\theta(z \mid x)} = \log \frac{p_\theta(x|z)p(z)q_\phi(z|x)}{p_\theta(z|x)q_\phi(z|x)}$$

$$= E_z[\log p_\theta(x|z)] - E_z\left[\log \frac{q_\phi(z|x)}{p(z)}\right] + E_z\left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right]$$

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

Data reconstruction

KL divergence between prior, and samples from the encoder network

KL divergence between encoder and posterior of decoder

VAE objective

- From the EM perspective

$$\log p_\theta(x) = \log \frac{p_\theta(x \mid z)p(z)}{p_\theta(z \mid x)} = \log \frac{\boxed{p_\theta(x|z)}\boxed{p(z)}\boxed{q_\phi(z|x)}}{\boxed{p_\theta(z|x)}\boxed{q_\phi(z|x)}}$$

$$= \boxed{E_z[\log p_\theta(x|z)]} - \boxed{E_z\left[\log \frac{q_\phi(z|x)}{p(z)}\right]} + \boxed{E_z\left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right]}$$

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

Data reconstruction

- How well does my decoder reconstruct a data point given the latent vector z
- We need to sample from z

KL divergence between prior, and samples from the encoder network

- Measures how good my latent distribution is with respect to my Gaussian prior

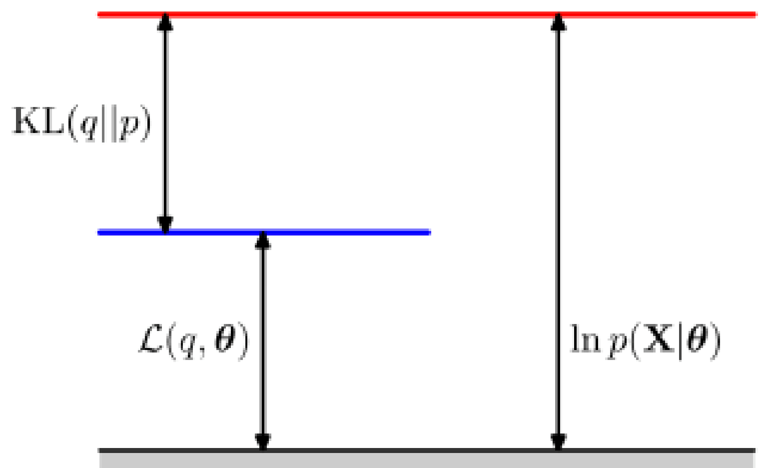
KL divergence between encoder and posterior of decoder

- Free energy, >= 0

VAE objective

- Visualizing ELBO

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



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

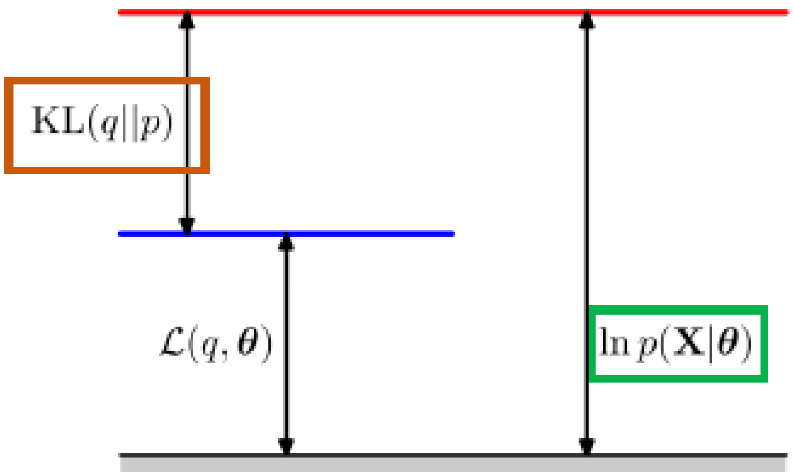
- EM perspective

  - Expectation Maximization alternately optimizes the ELBO, $\mathcal{L}(q, \theta)$, with respect to q (the E step) and θ (the M step)

  - Initialize $\theta^{(0)}$
  - At each iteration $t = 1, \ldots$
    - **E step:** Hold $\theta^{(t-1)}$ fixed, find $q^{(t)}$ which maximizes $\mathcal{L}\left(q, \theta^{(t-1)}\right)$
    - **M step:** Hold $q^{(t)}$ fixed, find $\theta^{(t)}$ which maximizes $\mathcal{L}\left(q^{(t)}, \theta\right)$

EM perspective

- The E step



Bishop – Pattern Recognition and Machine Learning

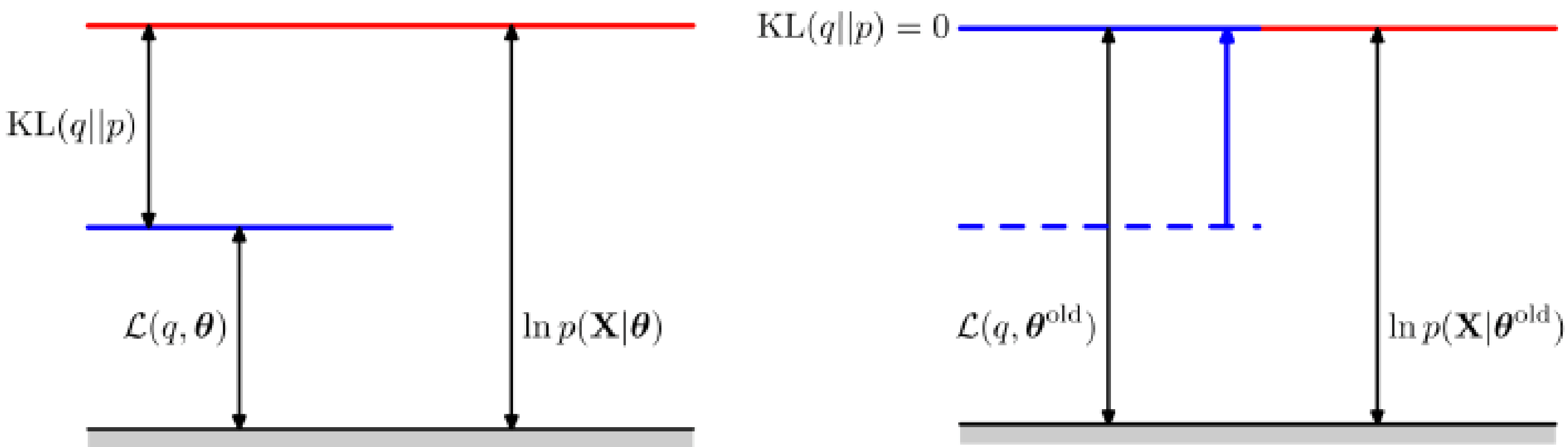
- The first term does not involve q, and we know the KL divergence must be non-negative
- The best we can do is to make the KL divergence 0
- Thus the solution is to set $\boldsymbol{q}^{(t)}(\boldsymbol{z}) \leftarrow \boldsymbol{p}\left(\boldsymbol{z}\middle|\boldsymbol{x},\boldsymbol{\theta}^{(t-1)}\right)$

- Suppose we are at iteration t of our algorithm. How do we maximize $\mathcal{L}\left(q,\theta^{(t-1)}\right)$ with respect to q? We know that:

$$\operatorname{argmax}_q \mathcal{L}\left(q,\theta^{(t-1)}\right) = \operatorname{argmax}_q \log p\left(x|\theta^{(t-1)}\right) - \mathrm{KL}\left(q(z) \,||\, p\left(z|x,\theta^{(t-1)}\right)\right)$$

EM perspective

- The E step

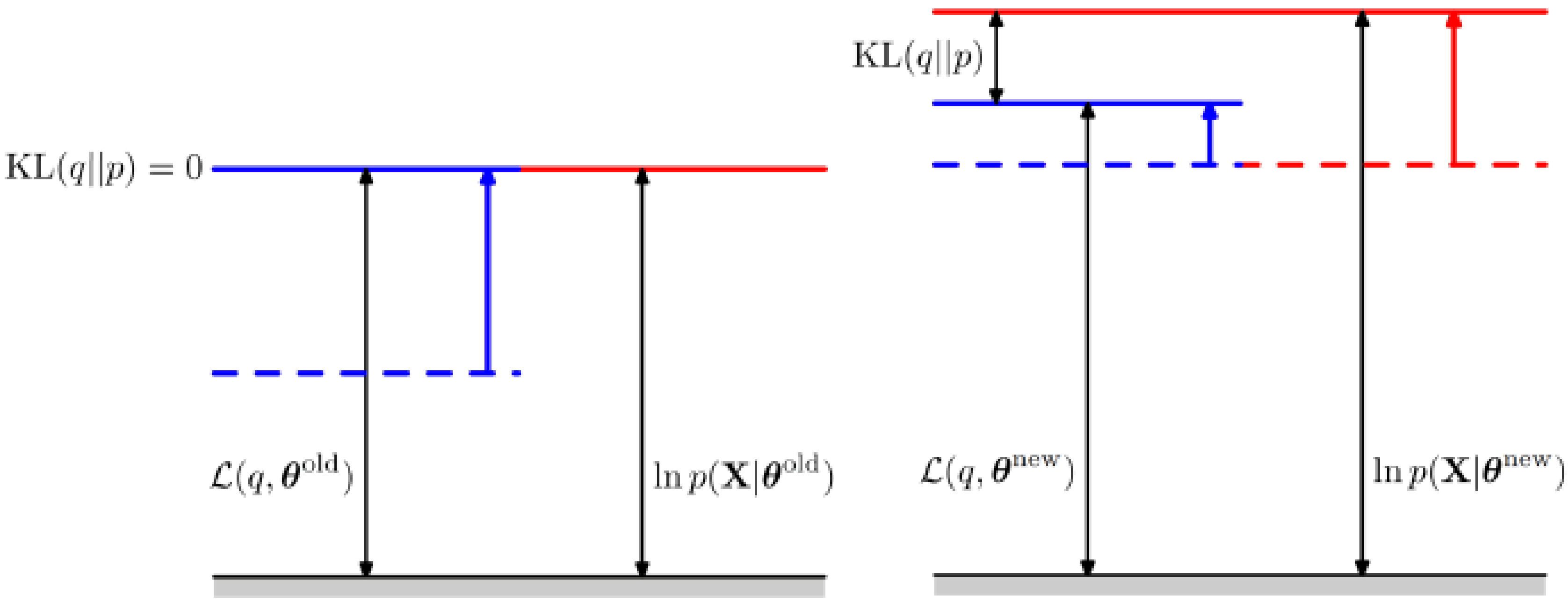


Bishop – Pattern Recognition and Machine Learning

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

EM perspective

- The M step

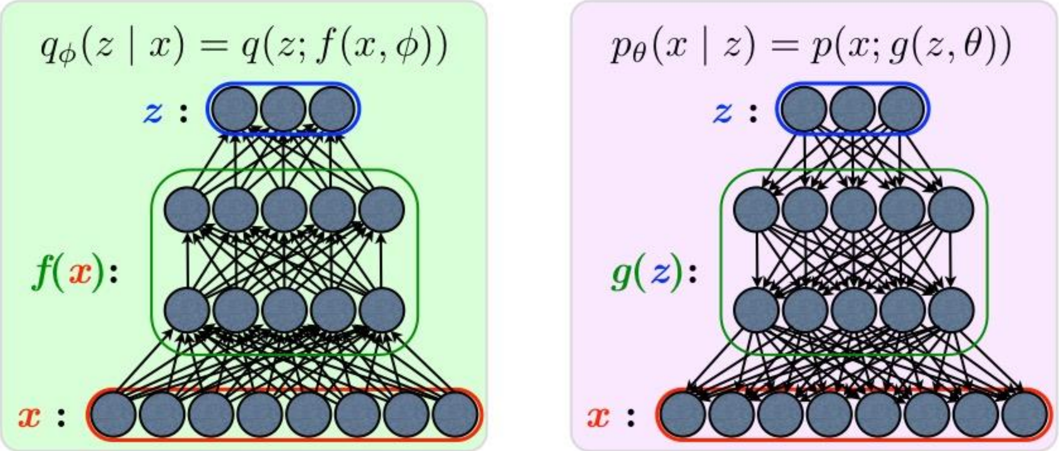


Bishop – Pattern Recognition and Machine Learning

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

VAE learning

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



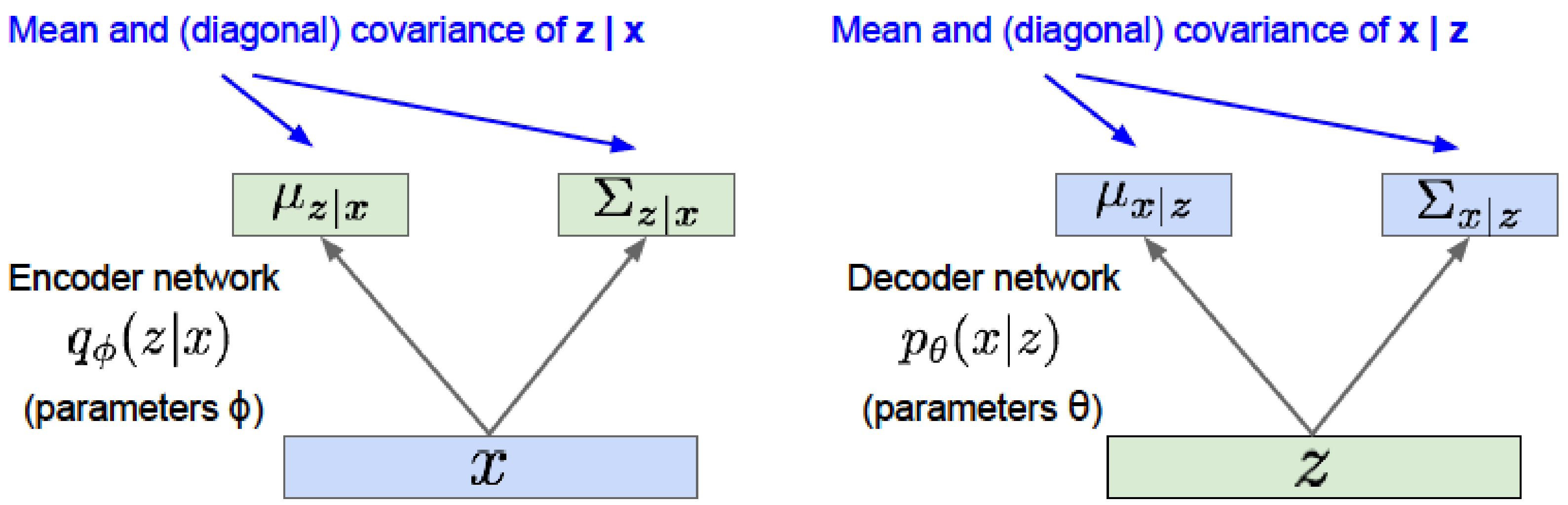
- Interpreting VAE objective

$$
\begin{aligned}
\mathcal{L}(\theta, \phi, x) &= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x, z) - \log q_\phi(z \mid x)\right] \\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z) + \log p_\theta(z) - \log q_\phi(z \mid x)\right] \\
&= -D_{\mathrm{KL}}\left(q_\phi(z \mid x) \| p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right]
\end{aligned}
$$

***regularization* term** ***reconstruction* term**

VAE Example

- Conditionals Gaussians

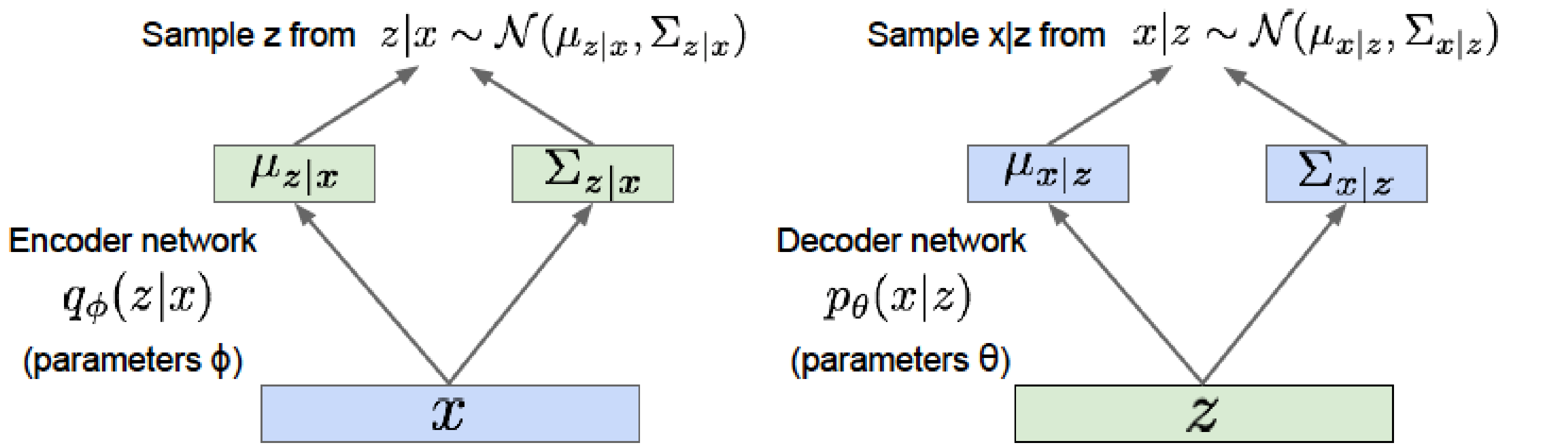


VAE Example

- 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

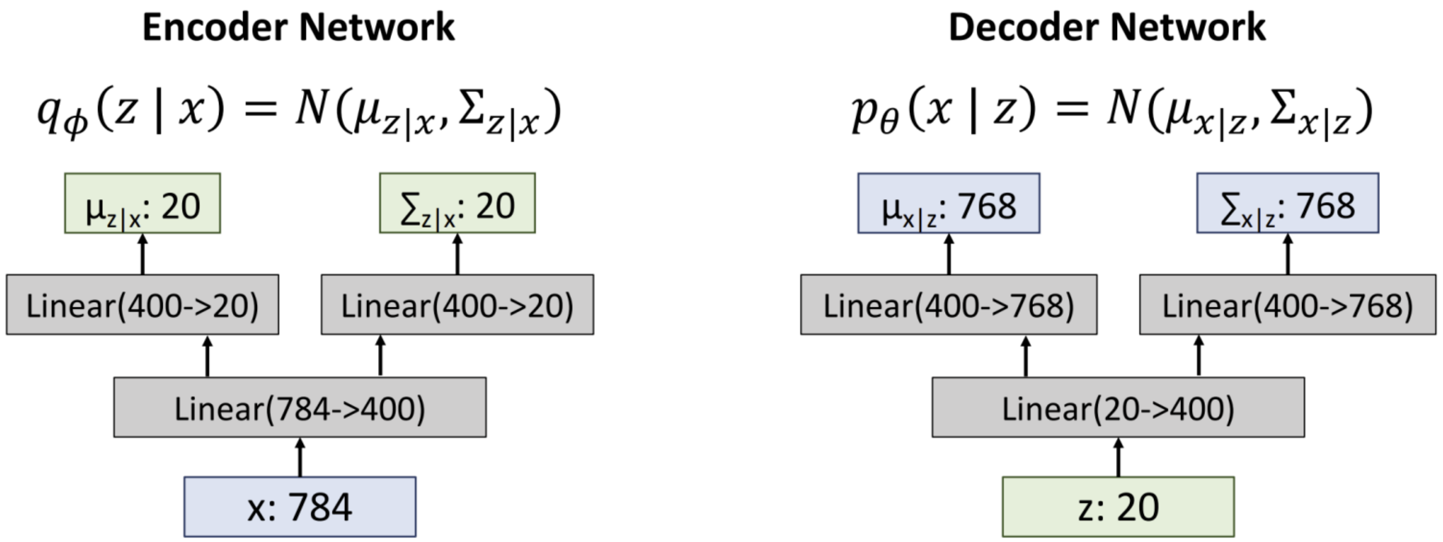
Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

VAE Example

- Toy example: Fully-Connected VAE

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

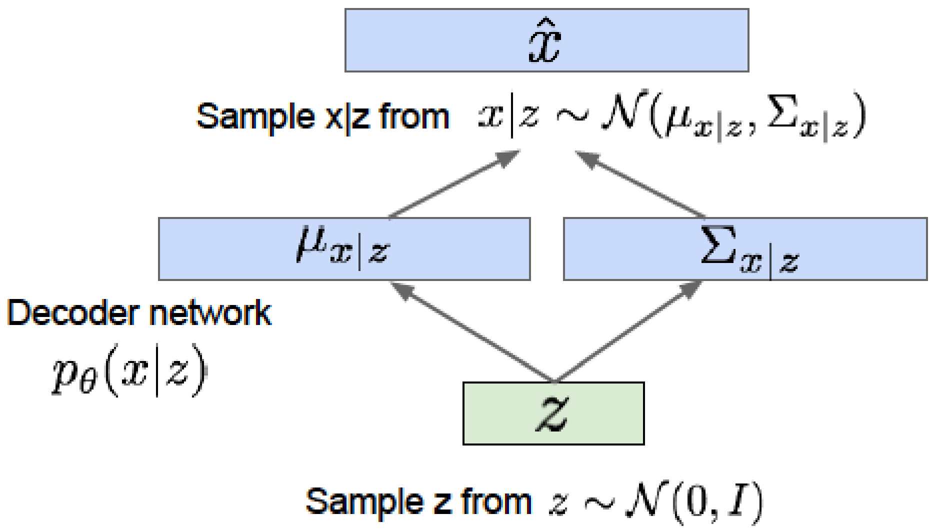
z: 20-dim vector



VAE Example

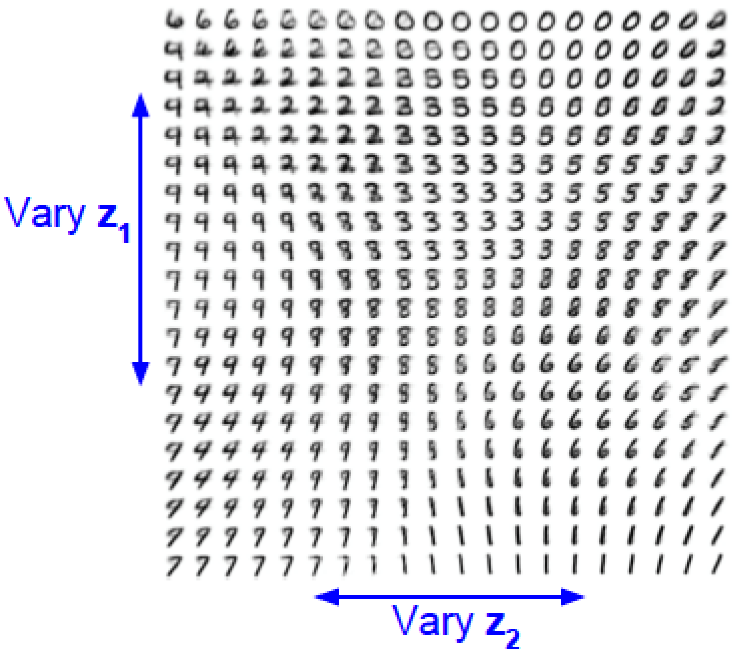
- Generating data

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



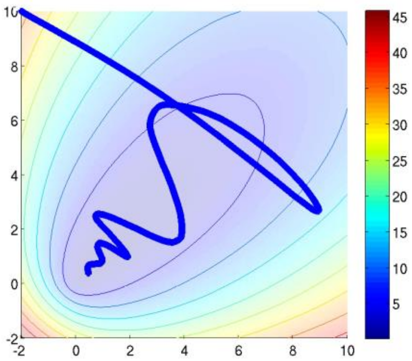
Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

Data manifold for 2-d **z**

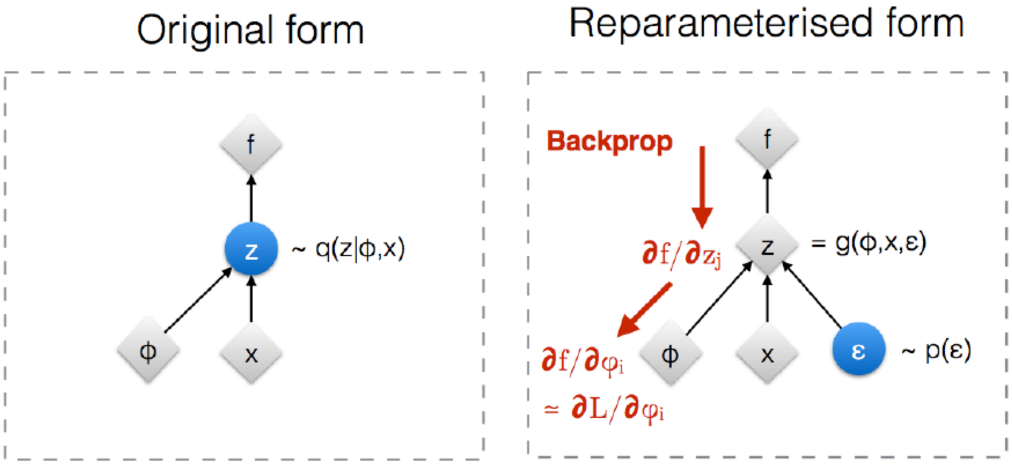


Two views of Learning VAE

- 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

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

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 $\boldsymbol{\epsilon}$

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

  - 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

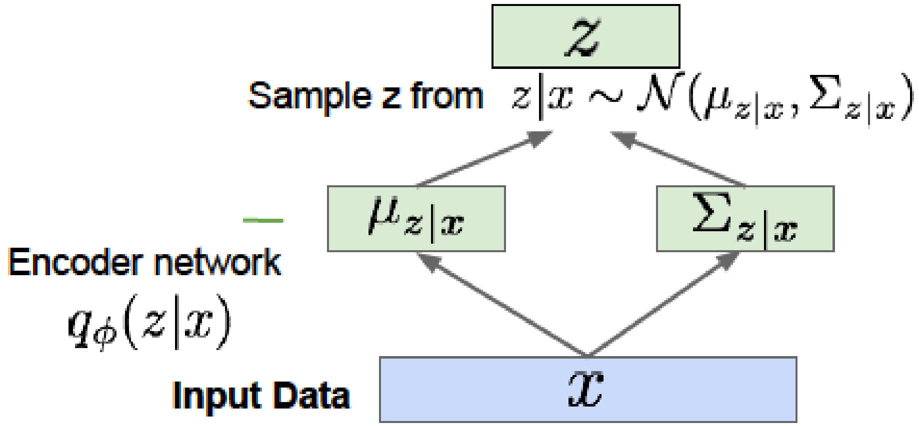
$$\nabla_\phi \mathcal{L}(x, \phi, \theta) = E_{q(\epsilon)}[\nabla_\phi f_{\phi,\theta}(x, z)] \approx \frac{1}{L} \sum_{i=1}^{L} \nabla_\phi f_{\phi,\theta}(x, g_\phi(\epsilon^{(i)}, x)), \quad \boxed{\epsilon^{(i)} \sim q(\epsilon)}$$

VAE Example

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

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

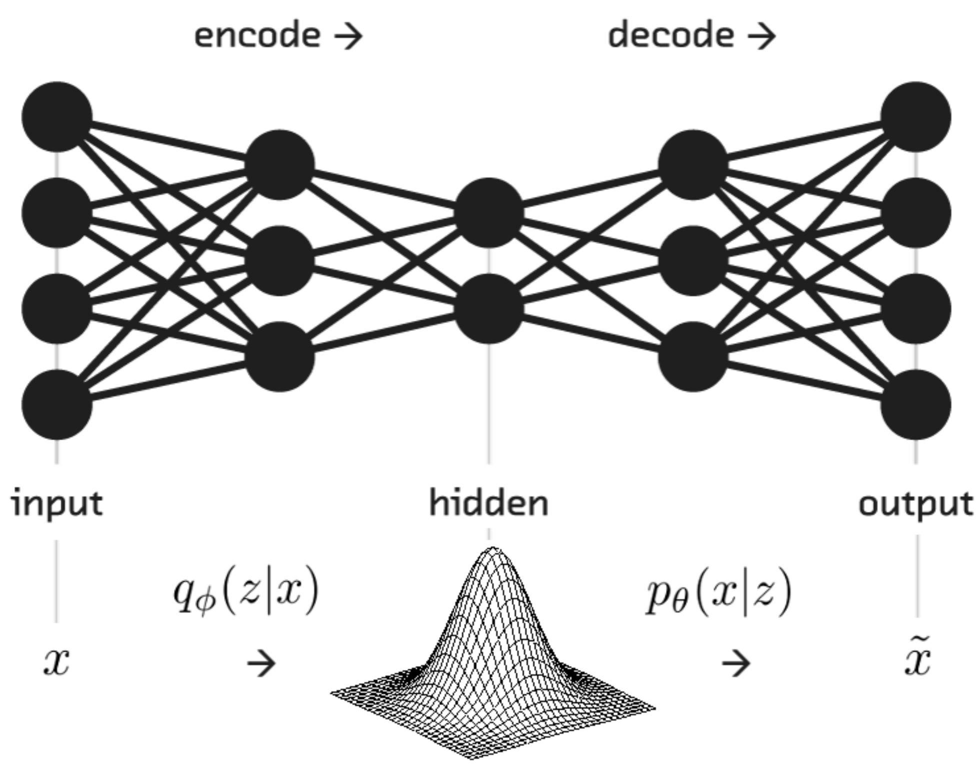
$$\mathbb{E}_{\mathcal{N}(z;\mu,\sigma^2)}\left[f(z)\right] = \mathbb{E}_{\mathcal{N}(\epsilon;0,1)}\left[f(\mu+\sigma\epsilon)\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f(\mu+\sigma\epsilon^{(l)})$$



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



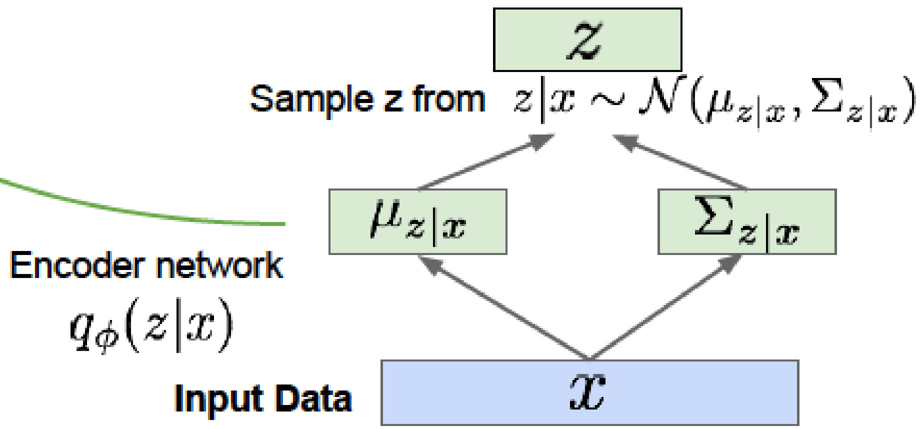
VAE Example

- Learning objective

Putting it all together: maximizing the likelihood lower bound

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

Make approximate posterior distribution close to prior



VAE Example

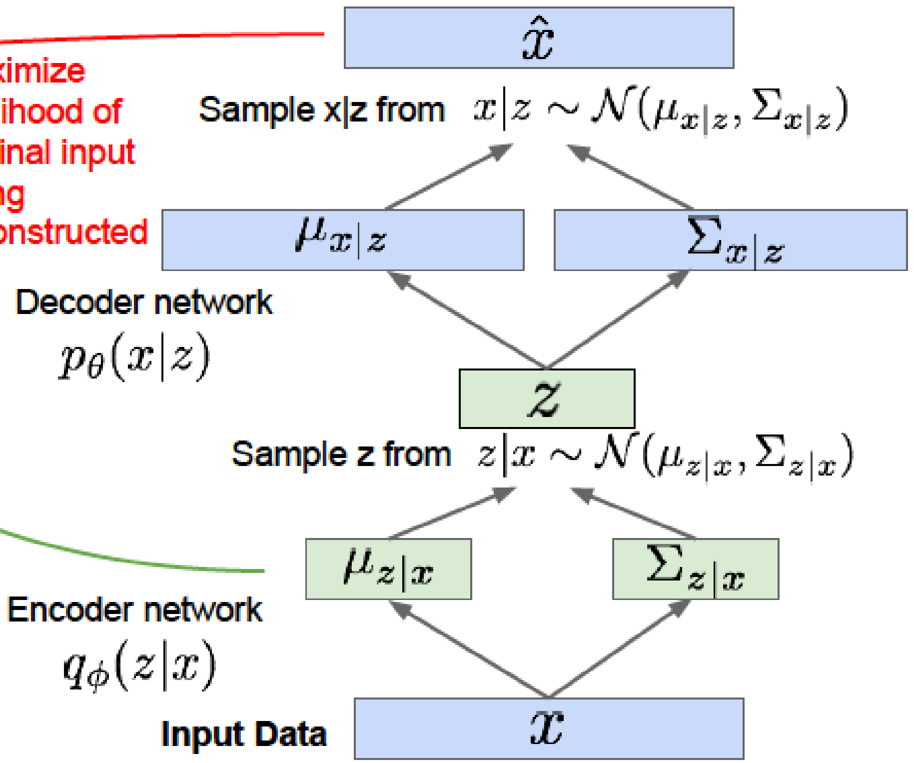
- Learning objective

Putting it all together: maximizing the likelihood lower bound

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

Maximize likelihood of original input being reconstructed

Make approximate posterior distribution close to prior



Variational Autoencoders

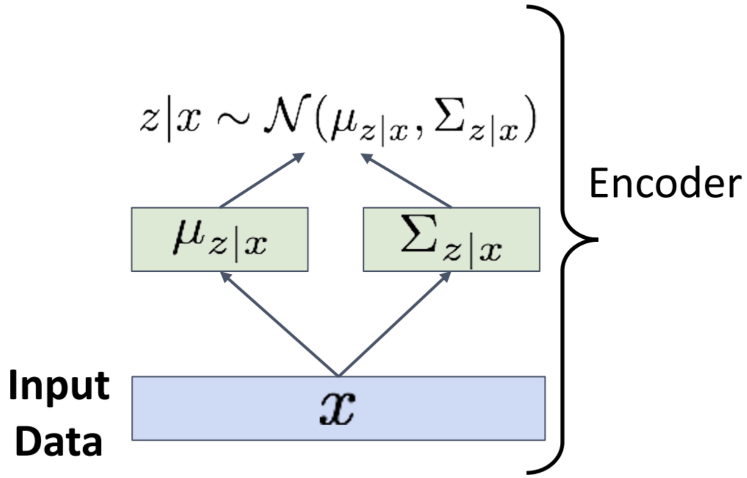
- Train by maximizing the variational lower bound

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

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

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

Closed form solution when q_ϕ is diagonal Gaussian and p is unit Gaussian!
(Assume z has dimension J)

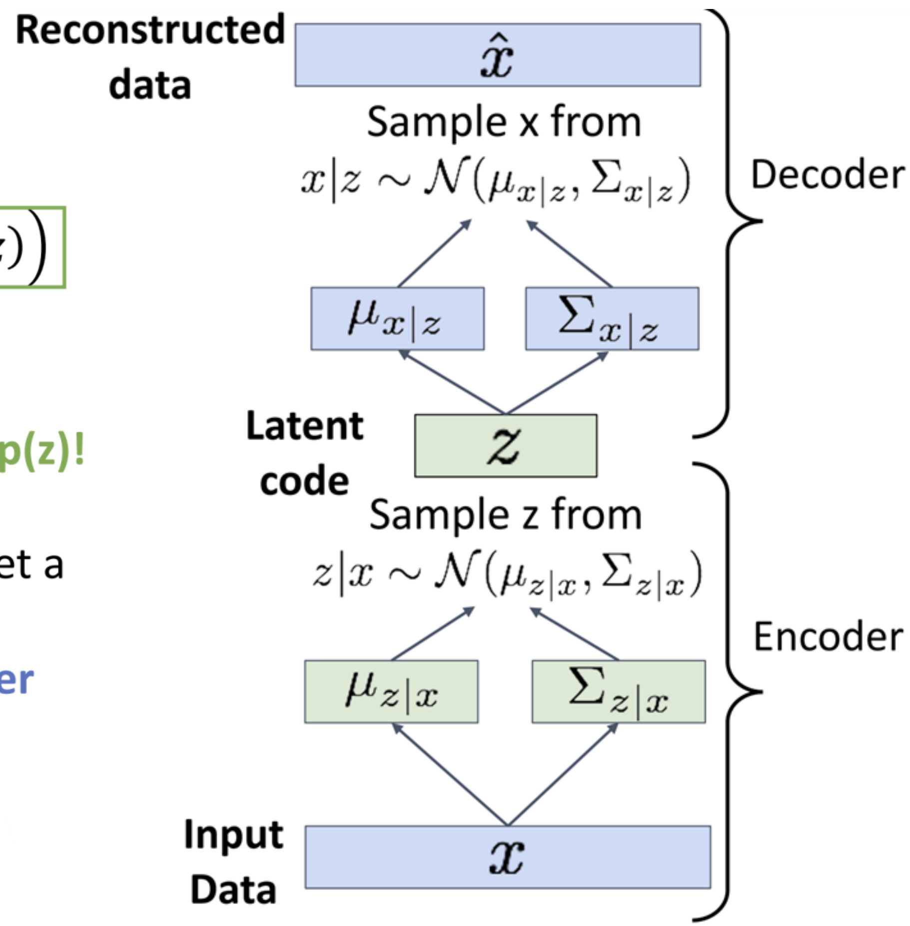


Variational Autoencoders

- Train by maximizing the variational lower bound

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

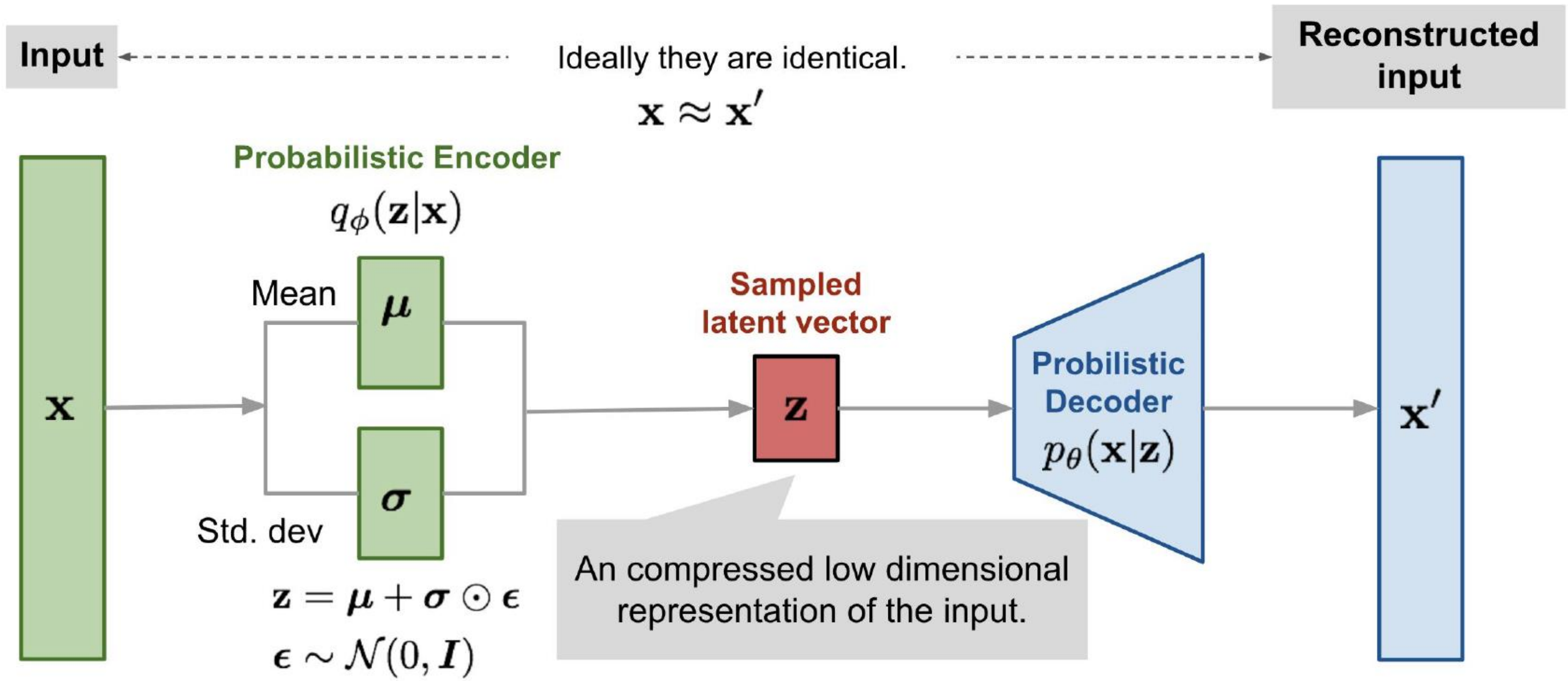
1. Run input data through **encoder** to get a distribution over latent codes
2. **Encoder output should match the prior p(z)!**
3. Sample code z from encoder output
4. Run sampled code through **decoder** to get a distribution over data samples
5. **Original input data should be likely under the distribution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



Autoencoder Interpretation

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

Regularization term Reconstruction term

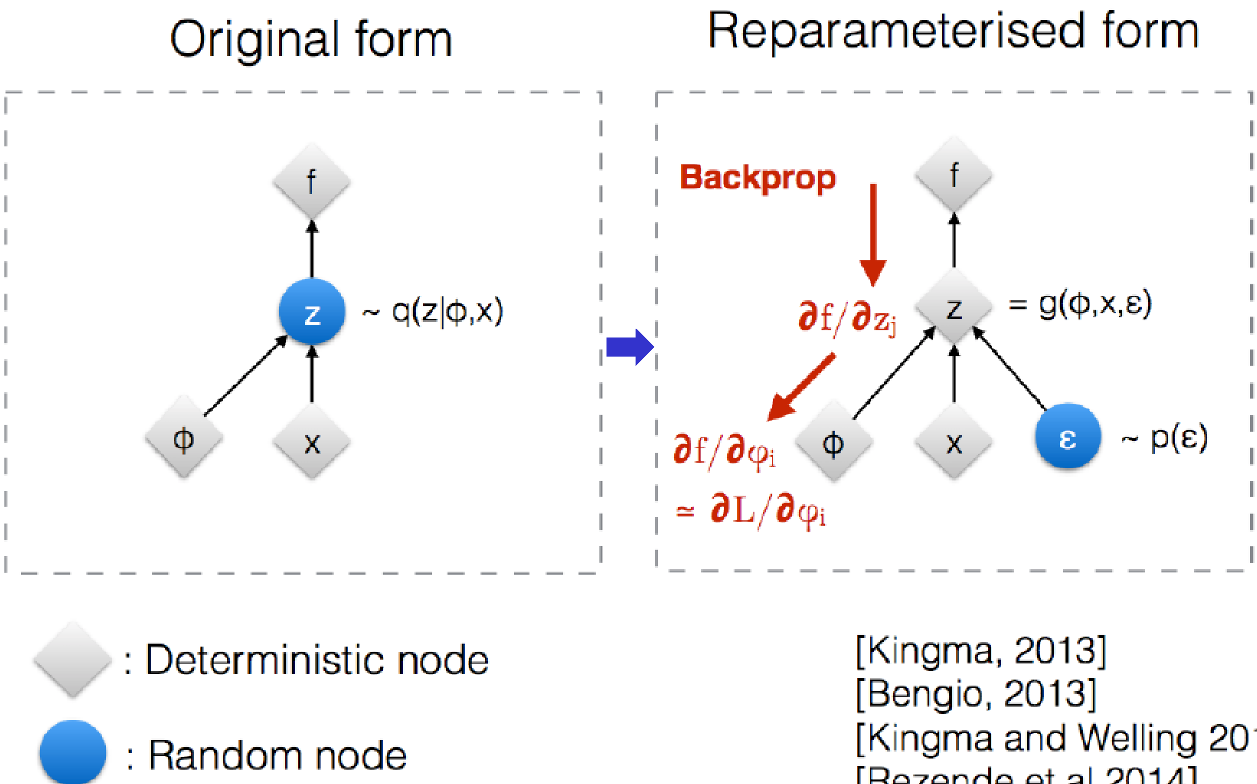


- The objective function can be represented as an Autoencoder-like computation graph.

Network interpretation

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

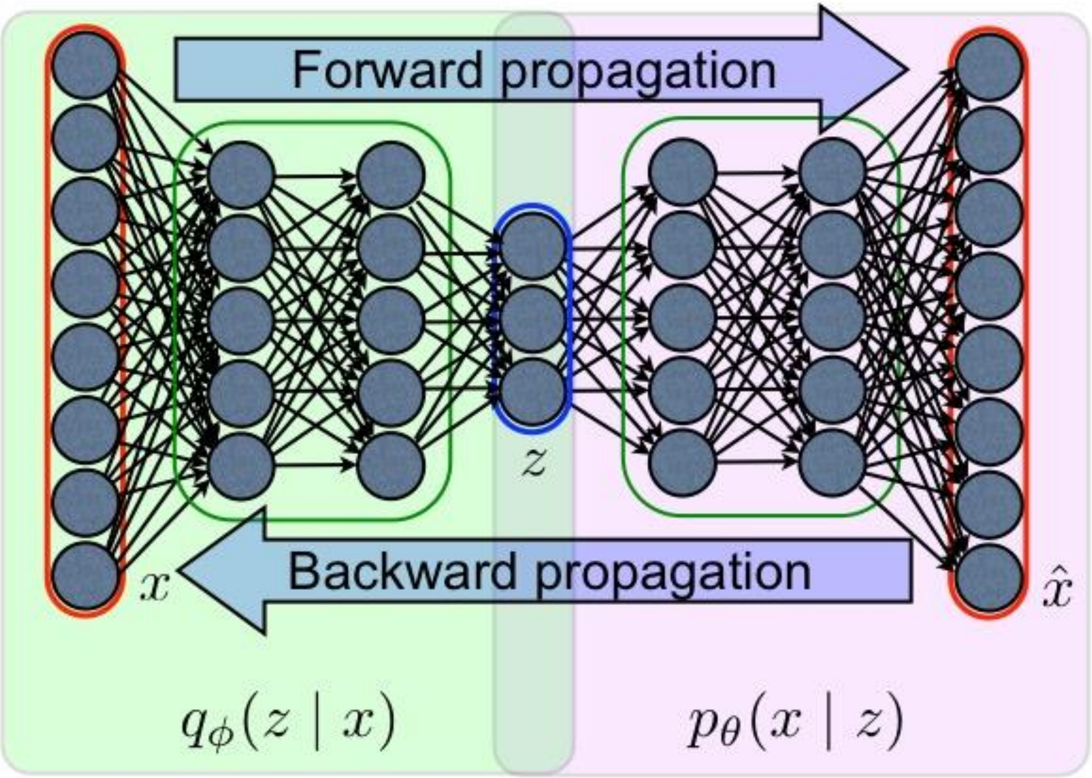
$$\mathcal{L}(x,\phi,\theta) = E_{q(\epsilon)}[f_{\phi,\theta}(x,z)] \approx \frac{1}{L}\sum_{i=1}^{L} f_{\phi,\theta}(x, g_\phi(\epsilon^{(i)}, x)), \quad \epsilon^{(i)} \sim q(\epsilon)$$



[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.



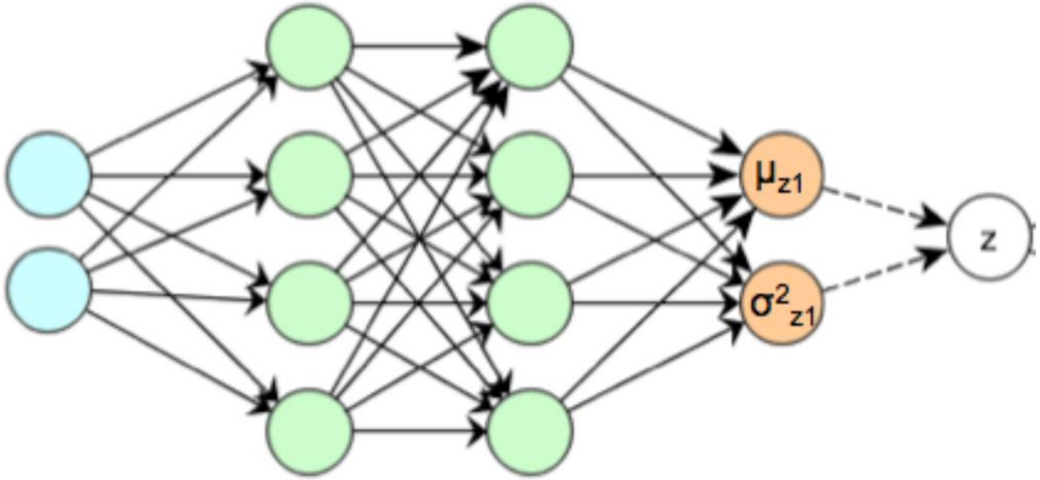
1D Gaussian Case

- 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

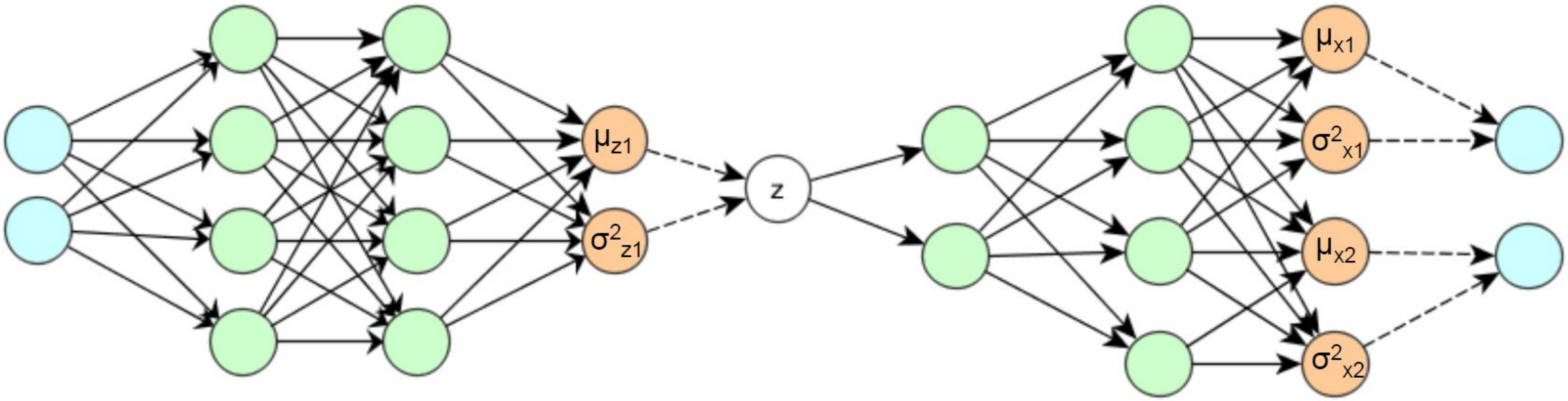
$$-D_{KL}\left(q(z|x^{(i)})||p(z)\right) = \frac{1}{2}\sum_{j=1}^{J}\left(1 + \log(\sigma_{z_j}^{(i)^2}) - \mu_{z_j}^{(i)^2} - \sigma_{z_j}^{(i)^2}\right)$$



1D Gaussian Case

- Overall loss function for BP

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



Cost: Regularisation

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

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

Cost: Reproduction

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

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 & colleages: 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.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