DSC190: Machine Learning with Few Labels

Variational inference
Self-supervised Learning

Zhiting Hu

Lecture 4, October 5, 2021

UC San Diego
HALICIOĞLU DATA SCIENCE INSTITUTE

Logistics

- Course project
  - Suggested projects
  - Define your own project, typically by picking a published paper and make extensions on top of it.

- In-class paper presentation
  - Pick any ML/AI paper you like

- Office hours **this week**
  - Thursday, 3-4:30pm

Outline

- Variational inference (cont'd)
    - Stochastic VI
    - Black-box VI
        - Computing Gradients of Expectations
    - Variational autoencoders (VAEs)

- Self-supervised learning (next lecture)

Images generated by VAEs (Razavi et al., 2019)

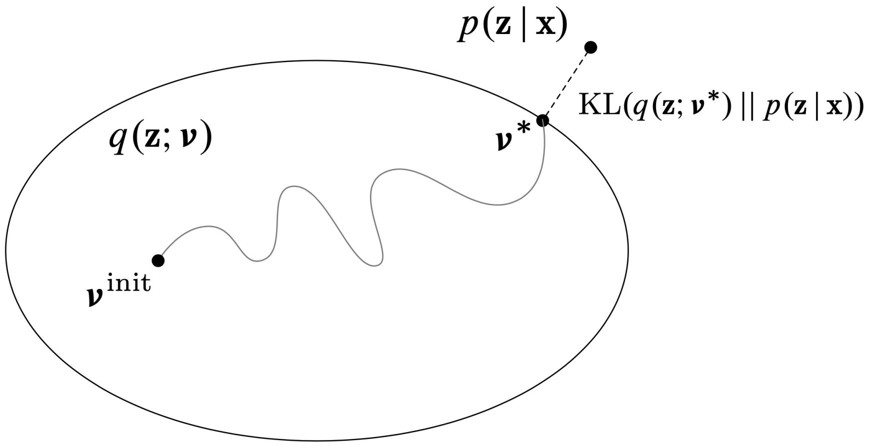
Variational Inference

- Observed variables $\boldsymbol{x}$, latent variables $\boldsymbol{z}$
- Variational (Bayesian) inference, a.k.a. **variational Bayes**, is used to approximately infer the **posterior distribution** over the latent variables

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

Variational Inference

- We often cannot compute posteriors, and so we need to approximate them, using variational methods.
- In variational Bayes, we'd like to find an approximation within some family that minimizes the KL divergence to the posterior, but we can't directly minimize this
- Therefore, we defined the ELBO, which we can maximize, and this is equivalent to minimizing the KL divergence.



Evidence Lower Bound (ELBO)

$$\ell(\theta; \boldsymbol{x}) = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z}|\theta)}{q(\boldsymbol{z}|\boldsymbol{x})}\right] + \mathrm{KL}\big(q(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z}|\boldsymbol{x}, \theta)\big)$$

Variational Inference

- We defined a family of approximations called "mean field" approximations, in which there are no dependencies between latent variables

$$q(\mathbf{z}) = q(z_1, \ldots, z_m) = \prod_{j=1}^{m} q(z_j)$$

- We optimize the ELBO with coordinate ascent updates to iteratively optimize each local variational approximation under mean field assumptions

$$q^*(z_j) \propto \exp\left\{ \mathbb{E}_{q_{-j}}[\log p(\boldsymbol{x}, \boldsymbol{z})] \right\}$$

  - The optimal solution for factor $q(z_j)$ is obtained simply by considering the log of the joint distribution over all observed and latent variables and then taking the expectation with respect to all of the other factors $q(z_k)$, $k \neq j$, then taking exponential and normalizing

Simple example:

- Consider a univariate Gaussian distribution $p(x) = \mathcal{N}(x|\mu, \tau^{-2})$, given a dataset $\mathcal{D} = \{x_1, \ldots, x_N\}$:

$$p(\mathcal{D}|\mu,\tau) = \left(\frac{\tau}{2\pi}\right)^{N/2} \exp\left\{-\frac{\tau}{2}\sum_{n=1}^{N}(x_n-\mu)^2\right\}$$

$$\begin{aligned} p(\mu|\tau) &= \mathcal{N}\left(\mu|\mu_0, (\lambda_0\tau)^{-1}\right) \\ p(\tau) &= \mathrm{Gam}(\tau|a_0, b_0) \end{aligned}$$

  - $\mathrm{Gam}(\tau|a_0, b_0) = \frac{1}{\Gamma(\mathrm{a})} b^a \lambda^{a-1} \exp(-b\lambda)$: gamma distribution

- For this simple problem the posterior distribution can be found exactly. But we use it as an example for tutorial anyway

$$q^*(z_j) \propto \exp\left\{ \mathbb{E}_{q_{-j}}[\log p(\boldsymbol{x}, \boldsymbol{z})] \right\}$$

Simple example:

$$p(\mathcal{D}|\mu, \tau) = \left(\frac{\tau}{2\pi}\right)^{N/2} \exp\left\{-\frac{\tau}{2}\sum_{n=1}^{N}(x_n - \mu)^2\right\}$$

$$\begin{aligned} p(\mu|\tau) &= \mathcal{N}\left(\mu|\mu_0, (\lambda_0\tau)^{-1}\right) \\ p(\tau) &= \mathrm{Gam}(\tau|a_0, b_0) \end{aligned}$$

- Introduce the factorized variational approximation: $q(\mu, \tau) = q_\mu(\mu)q_\tau(\tau)$
- Solution to q_μ:

$$\begin{aligned} \ln q_\mu^\star(\mu) &= \mathbb{E}_\tau\left[\ln p(\mathcal{D}|\mu, \tau) + \ln p(\mu|\tau)\right] + \mathrm{const} \\ &= -\frac{\mathbb{E}[\tau]}{2}\left\{\lambda_0(\mu - \mu_0)^2 + \sum_{n=1}^{N}(x_n - \mu)^2\right\} + \mathrm{const.} \end{aligned}$$

  - We can see q_μ^* is a Gaussian $\mathcal{N}(x|\mu_N, \lambda_N^{-1})$:

$$\begin{aligned} \mu_N &= \frac{\lambda_0\mu_0 + N\overline{x}}{\lambda_0 + N} \\ \lambda_N &= (\lambda_0 + N)\mathbb{E}[\tau] \end{aligned}$$

$$q^*(z_j) \propto \exp\left\{ \mathbb{E}_{q_{-j}}[\log p(\boldsymbol{x}, \boldsymbol{z})] \right\}$$

Simple example:

$$p(\mathcal{D}|\mu, \tau) = \left(\frac{\tau}{2\pi}\right)^{N/2} \exp\left\{-\frac{\tau}{2}\sum_{n=1}^{N}(x_n - \mu)^2\right\}$$

$$\begin{aligned} p(\mu|\tau) &= \mathcal{N}\left(\mu|\mu_0, (\lambda_0\tau)^{-1}\right) \\ p(\tau) &= \mathrm{Gam}(\tau|a_0, b_0) \end{aligned}$$

- Introduce the factorized variational approximation: $q(\mu, \tau) = q_\mu(\mu) q_\tau(\tau)$
- Solution to q_τ:

$$\begin{aligned} \ln q_\tau^\star(\tau) &= \mathbb{E}_\mu\left[\ln p(\mathcal{D}|\mu, \tau) + \ln p(\mu|\tau)\right] + \ln p(\tau) + \text{const} \\ &= (a_0 - 1)\ln\tau - b_0\tau + \frac{N}{2}\ln\tau \\ &\quad - \frac{\tau}{2}\mathbb{E}_\mu\left[\sum_{n=1}^{N}(x_n - \mu)^2 + \lambda_0(\mu - \mu_0)^2\right] + \text{const} \end{aligned}$$

  - We can see q_τ^* is a gamma distribution $\mathrm{Gam}(\tau|a_N, b_N)$:

$$\begin{aligned} a_N &= a_0 + \frac{N}{2} \\ b_N &= b_0 + \frac{1}{2}\mathbb{E}_\mu\left[\sum_{n=1}^{N}(x_n - \mu)^2 + \lambda_0(\mu - \mu_0)^2\right] \end{aligned}$$

VI with coordinate ascent

Example: Bayesian mixture of Gaussians

- Treat the mean μ_k and cluster proportion π as latent variables

$$\mu_k \sim \mathcal{N}(0, \tau^2) \text{ for } k = 1, \ldots, K$$

$$\pi \sim Dirichlet(\boldsymbol{\alpha})$$

- For each data $i = 1, \ldots, n$

$$z_i \sim \text{Cat}(\pi).$$

$$x_i \sim \mathcal{N}(\mu_{z_i}, \sigma^2).$$

- We have
  - observed variables $x_{1:n}$
  - latent variables $\mu_{1:k}$, π and $z_{1:n}$
  - Hyper-parameters $\{\tau^2, \sigma^2\}$

VI with coordinate ascent

Example: Bayesian mixture of Gaussians

Assume mean-field $q(\mu_{1:K}, \pi, z_{1:n}) = \prod_k q(\mu_k) q(\pi) \prod_i q(z_i)$

- Initialize the global variational distributions $q(\mu_k)$ and $q(\pi)$
- **Repeat**:
  - **For** each data example $i \in \{1,2, \dots, D\}$
    - Update the local variational distribution $q(z_i)$
  - **End for**
  - Update the global variational distributions $q(\mu_k)$ and $q(\pi)$
- **Until** ELBO converges

- What if we have millions of data examples? This could be very slow.

Stochastic VI

Example: Bayesian mixture of Gaussians

Assume mean-field $q(\mu_{1:K}, \pi, z_{1:n}) = \prod_k q(\mu_k) q(\pi) \prod_i q(z_i)$

- Initialize the global variational distributions $q(\mu_k)$ and $q(\pi)$
- Repeat:
  - Sample a data example $i \in \{1,2, \dots, D\}$
  - Update the local variational distribution $q(z_i)$
  - Update the global variational distributions $q(\mu_k)$ and $q(\pi)$ with **natural gradient ascent**
- Until ELBO converges

- (Setting natural gradient $= 0$ gives the traditional mean-field update)

[Hoffman et al., Stochastic Variational Inference, 2013]

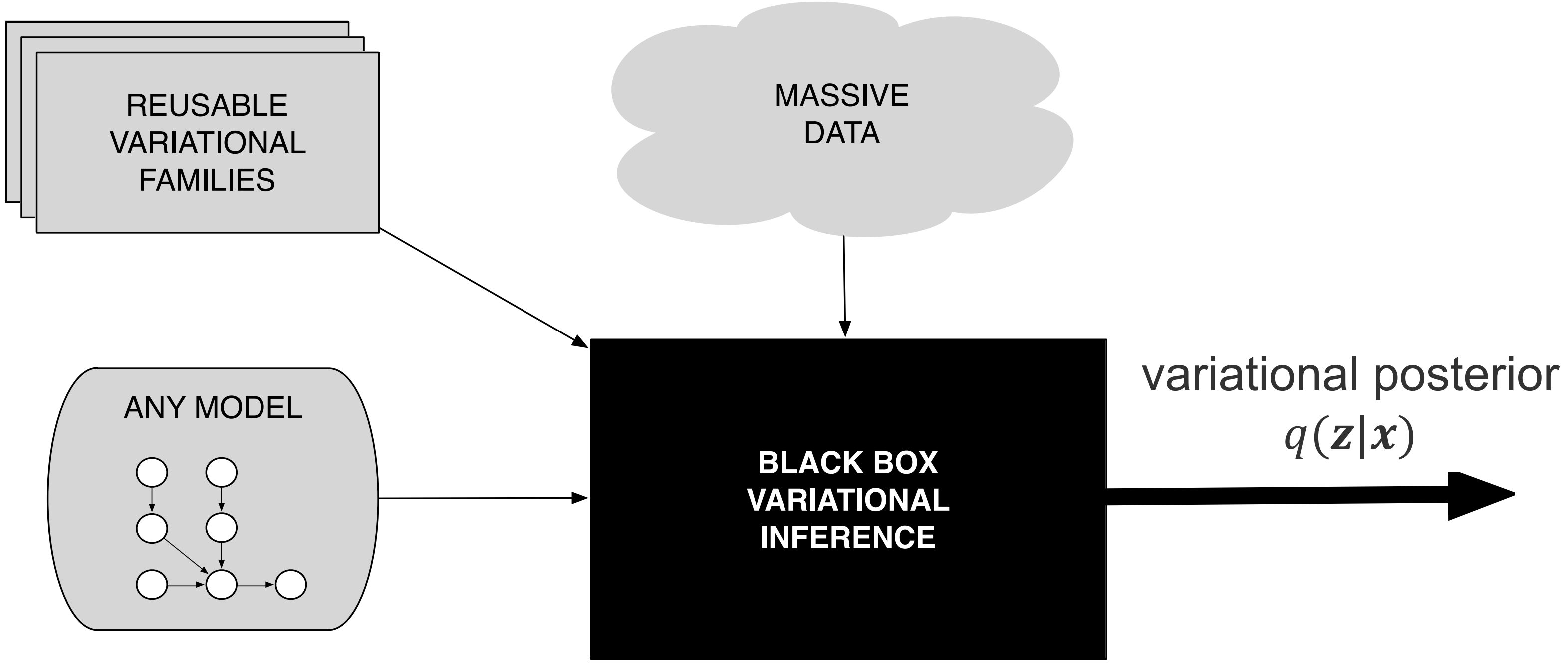
Black-box Variational Inference (BBVI)

- We have derived variational inference specific for Bayesian Gaussian (mixture) models

- There are innumerable models

- Can we have a solution that does not entail model-specific work?

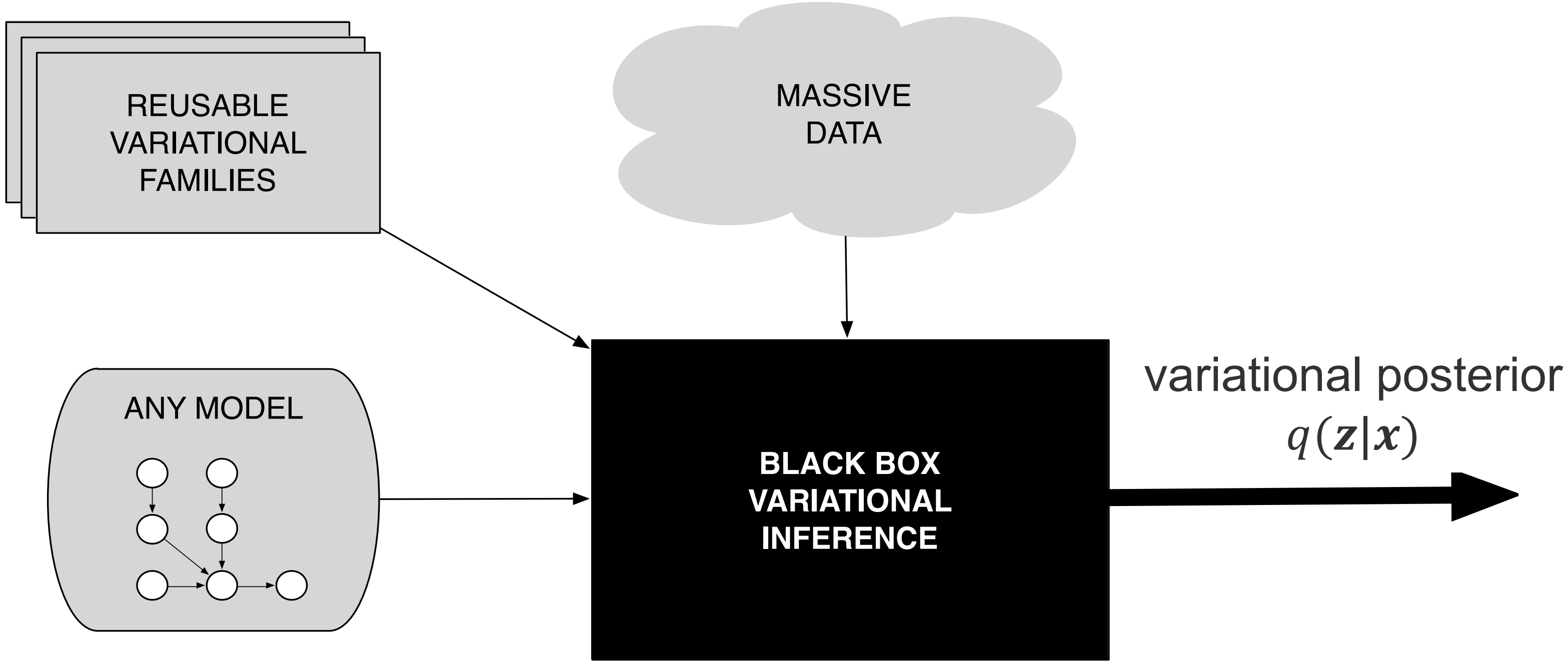
Black-box Variational Inference (BBVI)



- Easily use variational inference with **any model**
- Perform inference with **massive data**
- **No mathematical work** beyond specifying the model

(Courtesy: Blei et al., 2018)

Black-box Variational Inference (BBVI)



- Sample from $q(.)$
- Form noisy gradients (without model-specific computation)
- Use stochastic optimization

(Courtesy: Blei et al., 2018)

Black-box Variational Inference (BBVI)

- Probabilistic model: $\boldsymbol{x}$ -- observed variables, $\boldsymbol{z}$ -- latent variables
- Variational distribution $q_\lambda(\boldsymbol{z}|\boldsymbol{x})$ with parameters λ, e.g.,
  - Gaussian mixture distribution:
    - "A Gaussian mixture model is a universal approximator of densities, in the sense that any smooth density can be approximated with any specific nonzero amount of error by a Gaussian mixture model with enough components." (Deep Learning book, pp.65)
  - Deep neural networks

- ELBO:

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\lambda})}[\log p(\boldsymbol{x}, \boldsymbol{z})] - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\lambda})}[\log q(\boldsymbol{z}|\lambda)]$$

- Want to compute the gradient w.r.t variational parameters λ

[Ranganath et al.,14]

The General Problem: Computing Gradients of Expectations

- When the objective function $\mathcal{L}$ is defined as an expectation of a (differentiable) test function $f_\lambda(\boldsymbol{z})$ w.r.t. a probability distribution $q_\lambda(\boldsymbol{z})$

$$\mathcal{L} = \mathbb{E}_{q_\lambda(\boldsymbol{z})}[f_\lambda(\boldsymbol{z})]$$

- Computing exact gradients w.r.t. the parameters λ is often unfeasible
- Need stochastic gradient estimates
  - The score function estimator (a.k.a log-derivative trick, REINFORCE)
  - The reparameterization trick (a.k.a the pathwise gradient estimator)

Computing Gradients of Expectations w/ score function

- Loss: $\mathcal{L} = \mathbb{E}_{q_{\lambda}(\boldsymbol{z})}[f_{\lambda}(\boldsymbol{z})]$
- Log-derivative trick: $\nabla_{\lambda} q_{\lambda} = q_{\lambda} \nabla_{\lambda} \log q_{\lambda}$
- Gradient w.r.t. λ:

$$\nabla_{\lambda}\mathcal{L} = \mathbb{E}_{q_{\lambda}(\boldsymbol{z})}[f_{\lambda}(\boldsymbol{z}) \underline{\nabla_{\lambda} \log q_{\lambda}(\boldsymbol{z})} + \nabla_{\lambda} f_{\lambda}(\boldsymbol{z})]$$

  - score function: the gradient of the log of a probability distribution
- Compute noisy unbiased gradients with Monte Carlo samples from q_{λ}

$$\nabla_{\lambda}\mathcal{L} \approx \frac{1}{S}\sum_{s=1}^{S} f_{\lambda}(\boldsymbol{z}_s)\nabla_{\lambda} \log q_{\lambda}(\boldsymbol{z}_s) + \nabla_{\lambda} f_{\lambda}(\boldsymbol{z}_s) \qquad \text{where } \boldsymbol{z}_s \sim q_{\lambda}(\boldsymbol{z})$$

- Pros: generally applicable to any distribution $q(z|\lambda)$
- Cons: empirically has high variance → slow convergence
  - To reduce variance: Rao-Blackwellization, control variates, importance sampling, ...

Computing Gradients of Expectations w/ reparametrization trick

- Loss: $\mathcal{L} = \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{z})}[f_\lambda(\boldsymbol{z})]$
- Assume that we can express the distribution $q_{\boldsymbol{\lambda}}(\boldsymbol{z})$ with a transformation

$$\begin{aligned} \epsilon &\sim s(\epsilon) \\ z &= t(\epsilon, \lambda) \end{aligned} \quad \Leftrightarrow \quad z \sim q(z|\lambda)$$

  - E.g.,

$$\begin{aligned} \epsilon &\sim Normal(0, 1) \\ z &= \epsilon\sigma + \mu \end{aligned} \quad \Leftrightarrow \quad z \sim Normal(\mu, \sigma^2)$$

- Reparameterization gradient

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\epsilon} \sim s(\boldsymbol{\epsilon})}[f_\lambda(\boldsymbol{z}(\boldsymbol{\epsilon}, \lambda))]$$

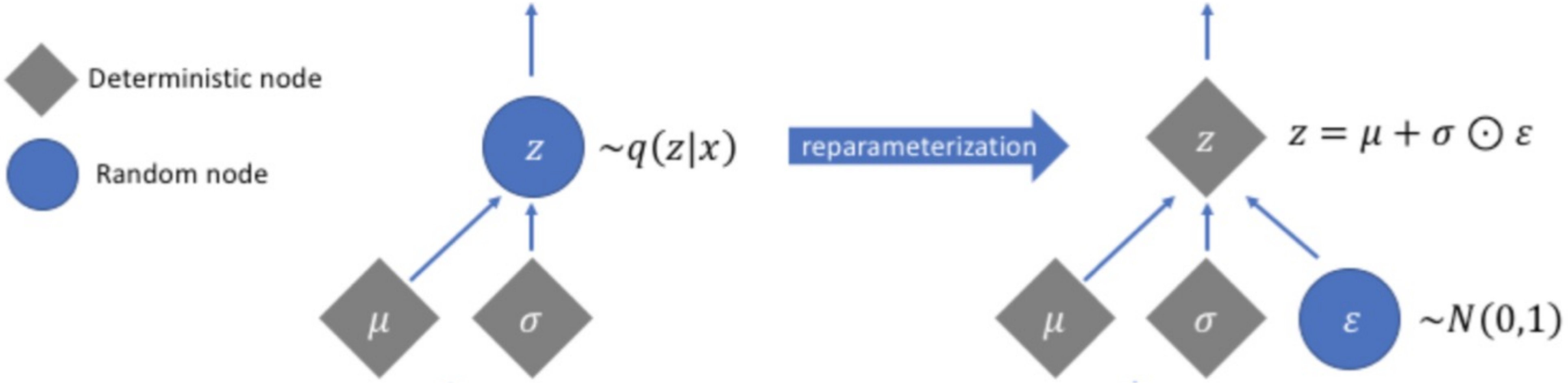
$$\nabla_\lambda \mathcal{L} = \mathbb{E}_{\boldsymbol{\epsilon} \sim \boldsymbol{s}(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{z}} f_\lambda(\boldsymbol{z})\, \nabla_\lambda t(\epsilon, \lambda)]$$

- Pros: empirically, lower variance of the gradient estimate
- Cons: Not all distributions can be reparameterized

Reparameterization trick

- Reparametrizing Gaussian distribution

$$\begin{aligned} \epsilon &\sim Normal(0,1) \\ z &= \epsilon\sigma + \mu \end{aligned} \iff z \sim Normal(\mu, \sigma^2)$$



[Courtesy: Tansey, 2016]

Reparameterization trick

- Reparametrizing Gaussian distribution

$$\begin{array}{l}\epsilon \sim Normal(0,1) \\ z = \epsilon\sigma + \mu\end{array} \iff z \sim Normal(\mu, \sigma^2)$$

- Other reparameterizable distributions: $\begin{array}{l}\epsilon \sim Uniform(\epsilon) \\ z = F^{-1}(\epsilon)\end{array} \iff z \sim q(z)$
  - Tractable inverse CDF F^{-1}:
    - Exponential, Cauchy, Logistic, Rayleigh, Pareto, Weibull, Reciprocal, Gompertz, Gumbel, Erlang
  - Location-scale:
    - Laplace, Elliptical, Student's t, Logistic, Uniform, Triangular, Gaussian
  - Composition:
    - Log-Normal (exponentiated normal) Gamma (sum of exponentials) Dirichlet (sum of Gammas) Beta, Chi-Squared, F

[Courtesy: Tansey, 2016]

Computing Gradients of Expectations: Summary

- Loss: $\mathcal{L} = \mathbb{E}_{q_{\lambda}(\mathbf{z})}[f_{\lambda}(\mathbf{z})]$

- **Score gradient**

$$\nabla_{\lambda}\mathcal{L} = \mathbb{E}_{q_{\lambda}(\mathbf{z})}[f_{\lambda}(\mathbf{z})\nabla_{\lambda}\log q_{\lambda}(\mathbf{z}) + \nabla_{\lambda}f_{\lambda}(\mathbf{z})]$$

  - Pros: generally applicable to any distribution $q(z|\lambda)$
  - Cons: empirically has high variance → slow convergence

- **Reparameterization gradient**

$$\nabla_{\lambda}\mathcal{L} = \mathbb{E}_{\boldsymbol{\epsilon}\sim \boldsymbol{s}(\boldsymbol{\epsilon})}[\nabla_{\mathbf{z}}f_{\lambda}(\mathbf{z})\ \nabla_{\lambda}t(\epsilon, \lambda)]$$

  - Pros: empirically, lower variance of the gradient estimate
  - Cons: Not all distributions can be reparameterized

Recall: Black-box Variational Inference (BBVI)

- Probabilistic model: $\boldsymbol{x}$ -- observed variables, $\boldsymbol{z}$ -- latent variables
- Variational distribution $q_\lambda(\boldsymbol{z}|\boldsymbol{x})$ with parameters λ, e.g.,
  - Gaussian mixture distribution:
    - "A Gaussian mixture model is a universal approximator of densities, in the sense that any smooth density can be approximated with any specific nonzero amount of error by a Gaussian mixture model with enough components." (Deep Learning book, pp.65)
  - Deep neural networks

$$\mathcal{L}(\lambda) \triangleq \mathrm{E}_{q_\lambda(z)}[\log p(x, z) - \log q(z)]$$

- ELBO:

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\lambda})}[\log p(\boldsymbol{x}, \boldsymbol{z})] - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\lambda})}[\log q(\boldsymbol{z}|\lambda)]$$

- Want to compute the gradient w.r.t variational parameters λ

[Ranganath et al.,14]

BBVI with the score gradient

- ELBO:

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\lambda})}[\log p(\boldsymbol{x}, \boldsymbol{z})] - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\lambda})}[\log q(\boldsymbol{z}|\lambda)]$$

- Gradient w.r.t. λ (using the log-derivative trick)

$$\nabla_\lambda \mathcal{L} = \mathrm{E}_q[\nabla_\lambda \log q(z|\lambda)(\log p(x, z) - \log q(z|\lambda))]$$

- Compute noisy unbiased gradients of the ELBO with Monte Carlo samples from the variational distribution

$$\nabla_\lambda \mathcal{L} \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_\lambda \log q(z_s|\lambda)(\log p(x, z_s) - \log q(z_s|\lambda)),$$

$$\text{where } z_s \sim q(z|\lambda).$$

[Ranganath et al.,14]

BBVI with the reparameterization gradient

- ELBO:

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\lambda})}[\log p(\boldsymbol{x}, \boldsymbol{z})] - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{\lambda})}[\log q(\boldsymbol{z}|\lambda)]$$

- Gradient w.r.t. λ

$$\begin{aligned} \epsilon &\sim s(\epsilon) \\ z &= t(\epsilon, \lambda) \end{aligned} \quad \Leftrightarrow \quad z \sim q(z|\lambda)$$

$$\nabla_\lambda \mathcal{L} = \mathrm{E}_{\epsilon \sim s(\epsilon)}[\nabla_z [\log p(x, z) - \log q(z)] \nabla_\lambda t(\epsilon, \lambda)]$$

Variational Auto-Encoders (VAEs)

VAEs are a combination of the following ideas:

- Variational Inference
  - ELBO
- Variational distribution parametrized as neural networks
- Reparameterization trick

[Courtesy: Dhruv, CS 4803]

Variational Auto-Encoders (VAEs)

- Model $p_\theta(\boldsymbol{x}, \boldsymbol{z}) = p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$
  - $p_\theta(\boldsymbol{x}|\boldsymbol{z})$: a.k.a., generative model, generator, (probabilistic) decoder, …
  - $p(\boldsymbol{z})$: prior, e.g., Gaussian
- Assume variational distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$
  - E.g., a Gaussian distribution parameterized as **deep neural networks**
  - a.k.a, recognition model, inference network, (probabilistic) encoder, …
- ELBO:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = \mathrm{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}, \boldsymbol{z})] - \mathrm{H}(q_\phi(\boldsymbol{z}|\boldsymbol{x}))$$
$$= \mathrm{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z}))$$

Reconstruction

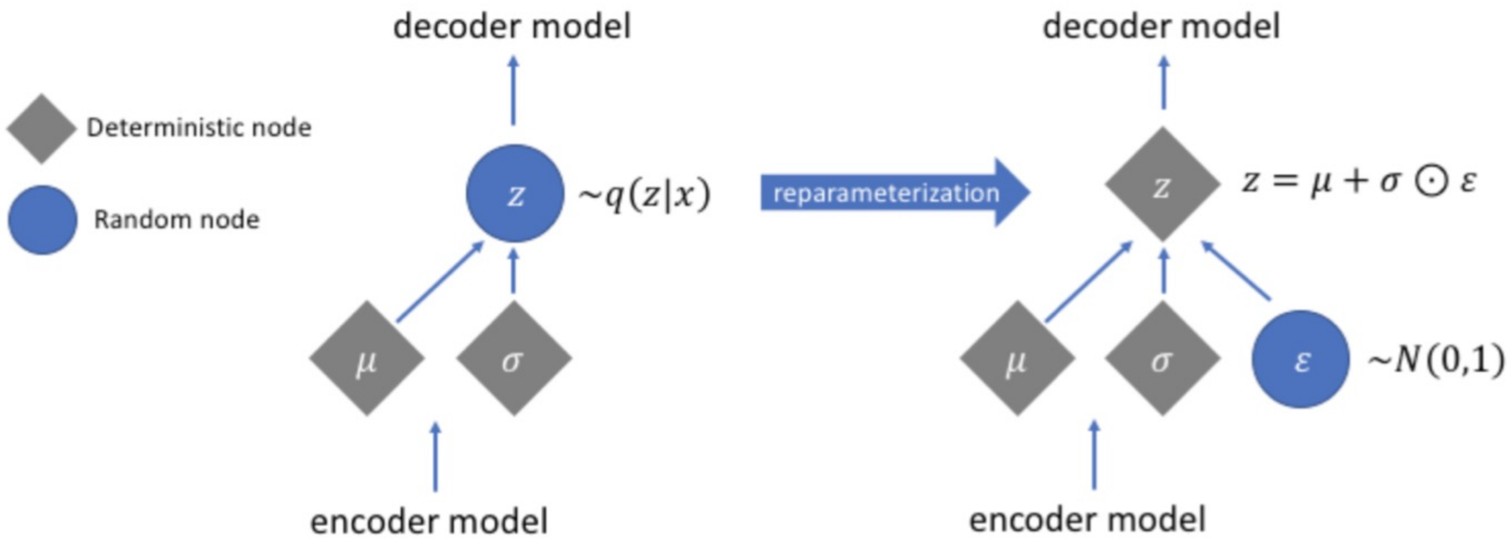
Divergence from prior
(KL divergence between two Guassians
has an analytic form)

Variational Auto-Encoders (VAEs)

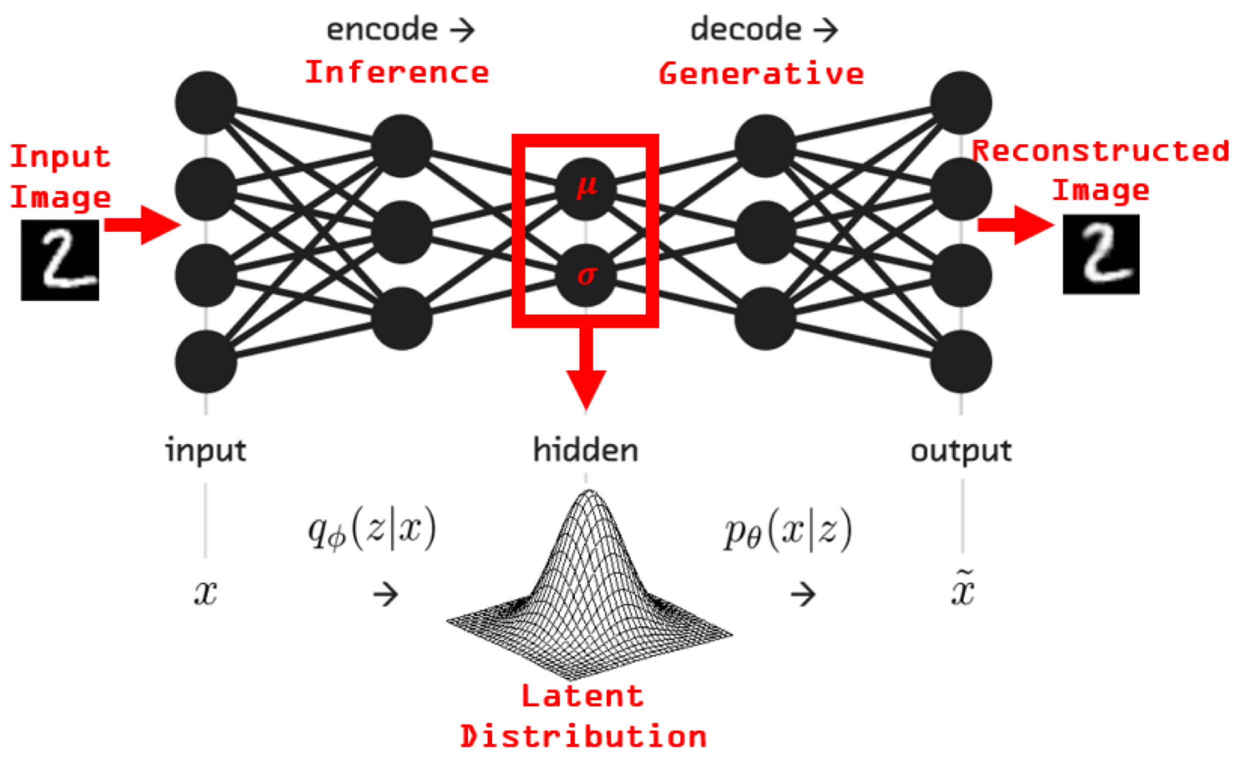
- ELBO:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = \mathrm{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}, \boldsymbol{z})] - \mathrm{H}(q_\phi(\boldsymbol{z}|\boldsymbol{x}))$$
$$= \mathrm{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z}))$$

- Reparameterization:
  - $[\boldsymbol{\mu}; \boldsymbol{\sigma}] = f_\phi(\boldsymbol{x})$ (a neural network)
  - $\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{1})$

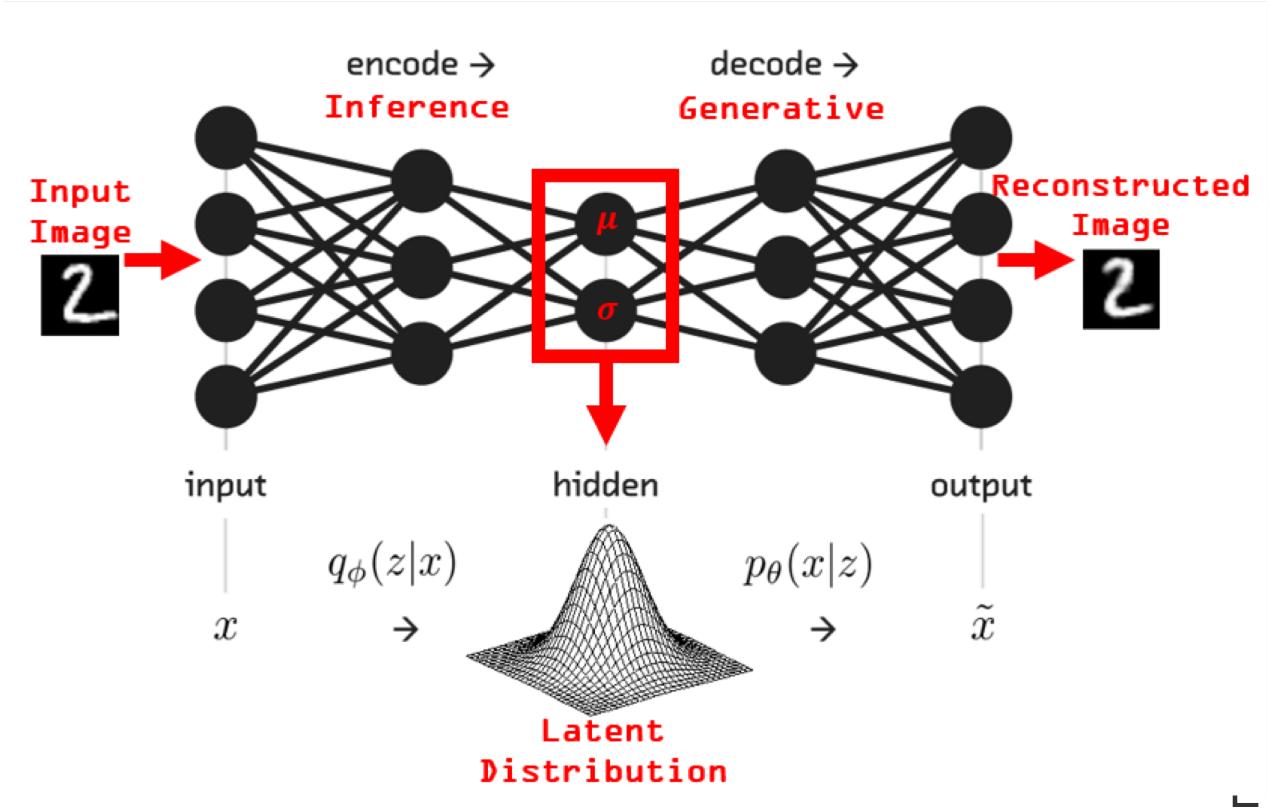


Example: VAEs for images



[https://www.kaggle.com/rvislaywade/visualizing-mnist-using-a-variational-autoencoder]

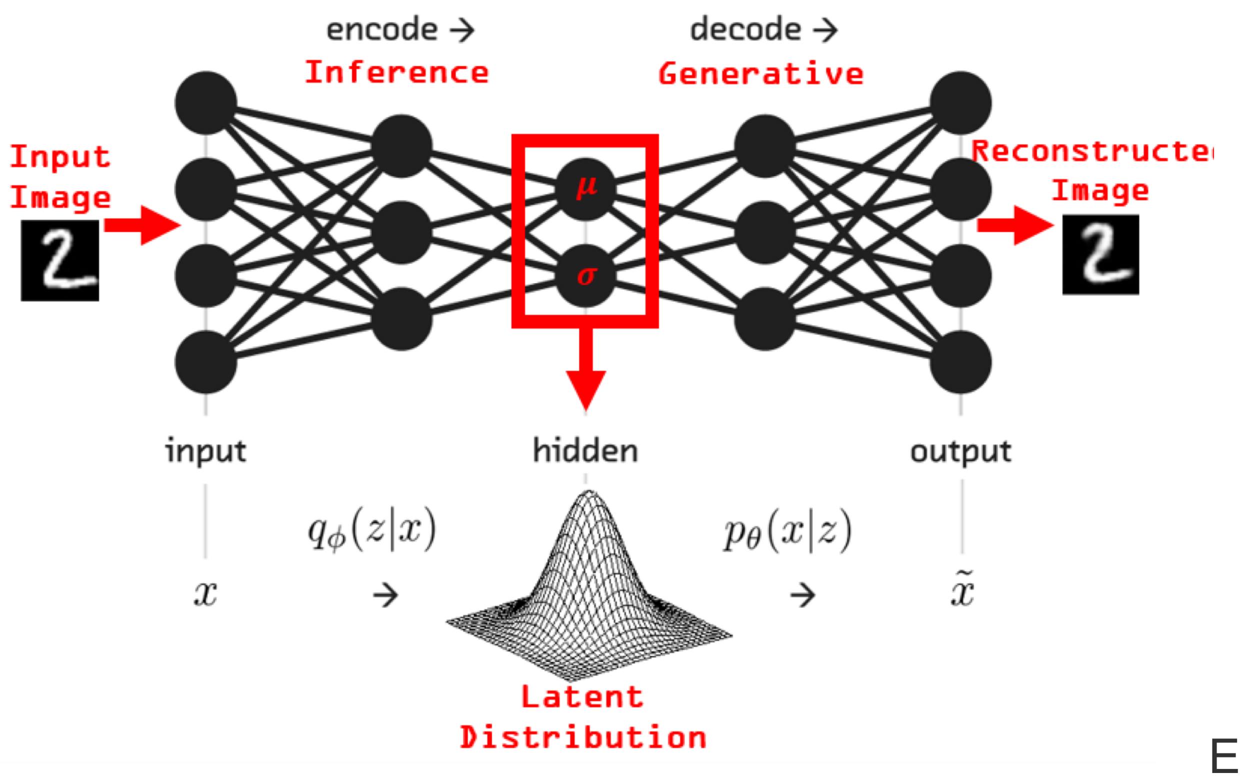
Example: VAEs for images

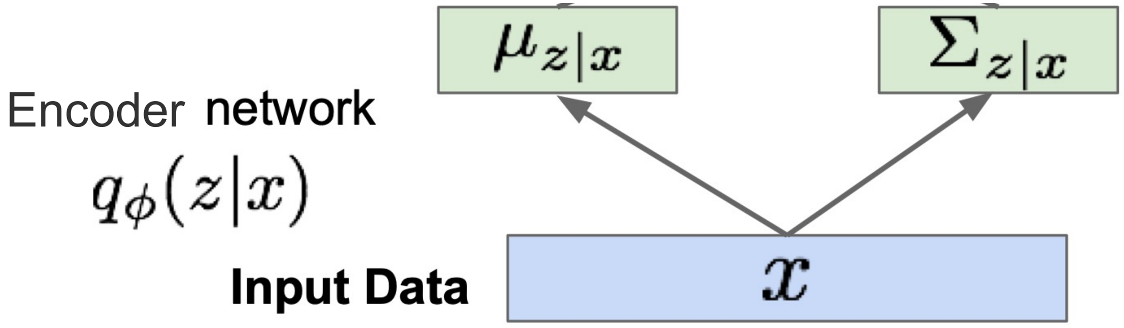


Input Data x

[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

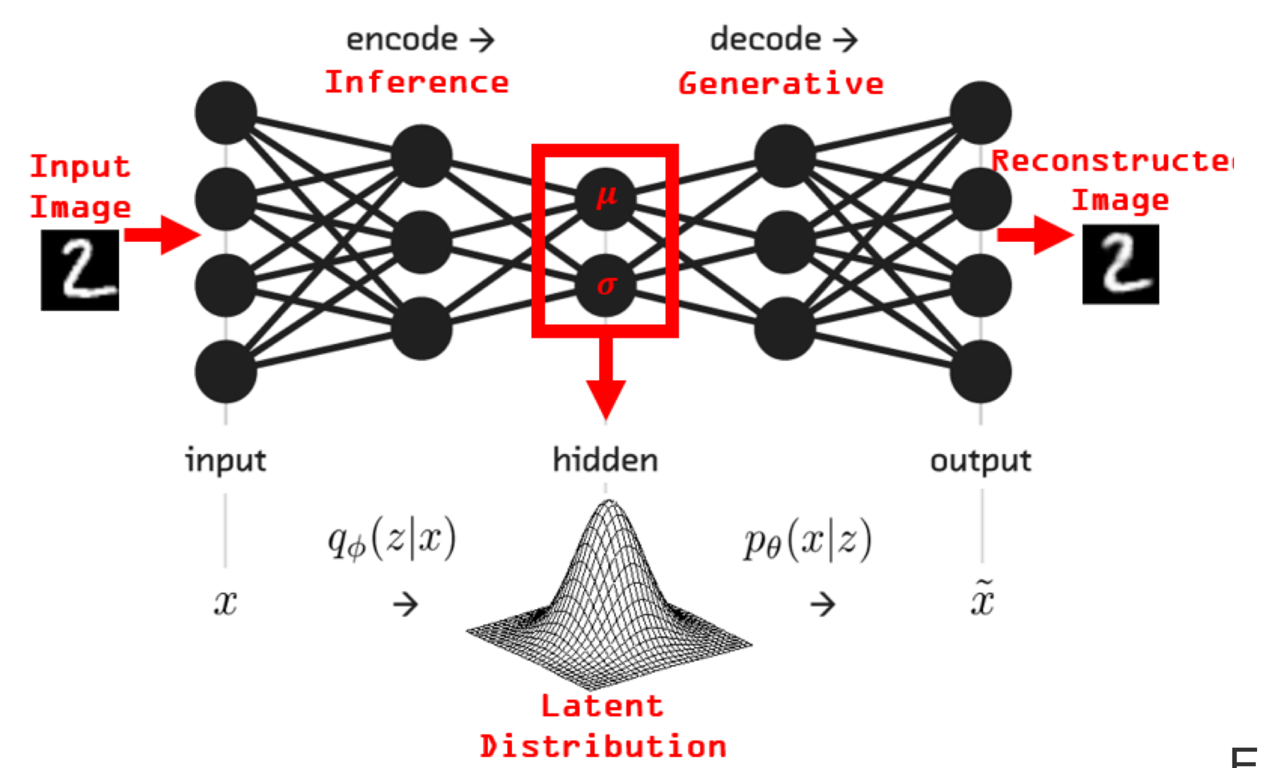
Example: VAEs for images

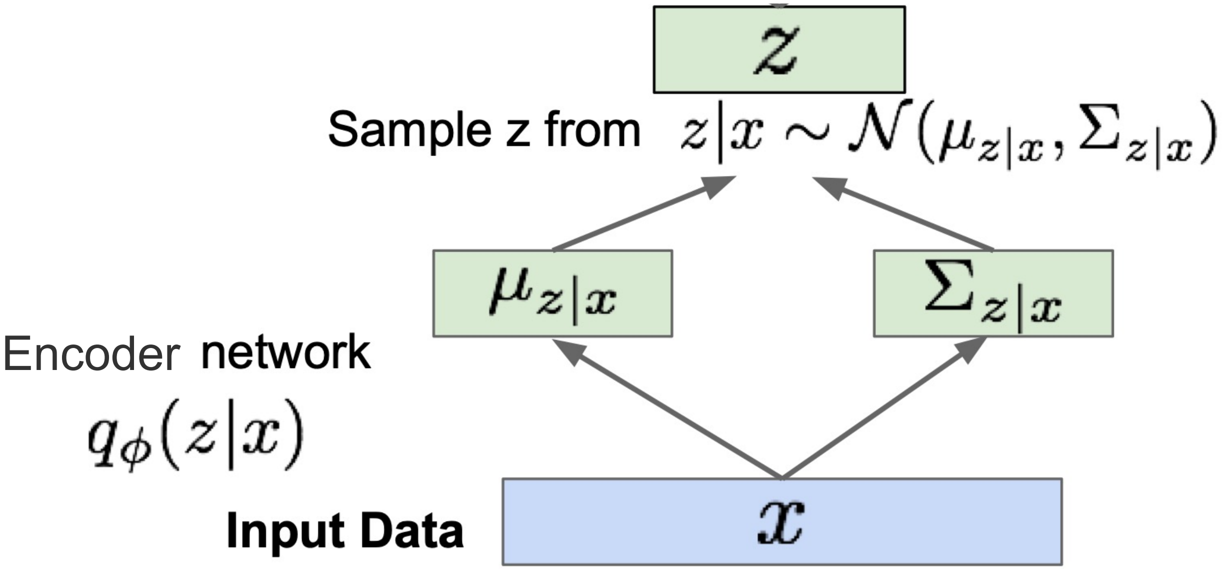




[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

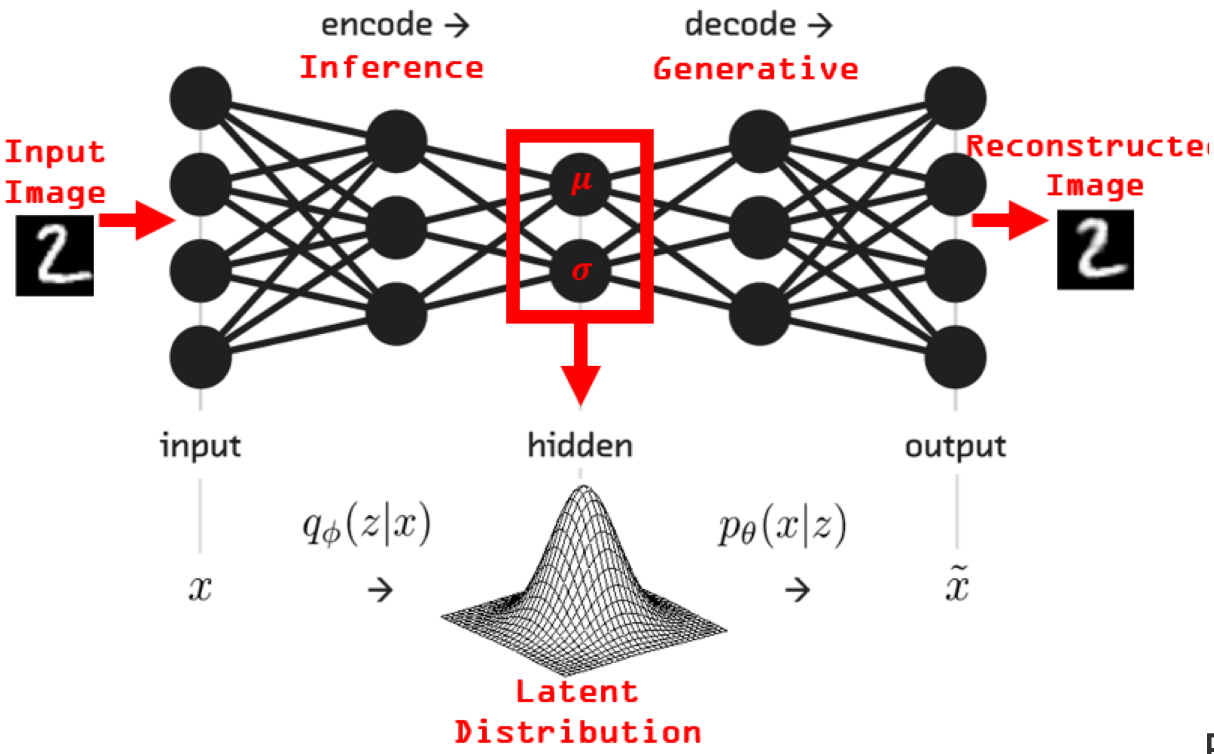
Example: VAEs for images

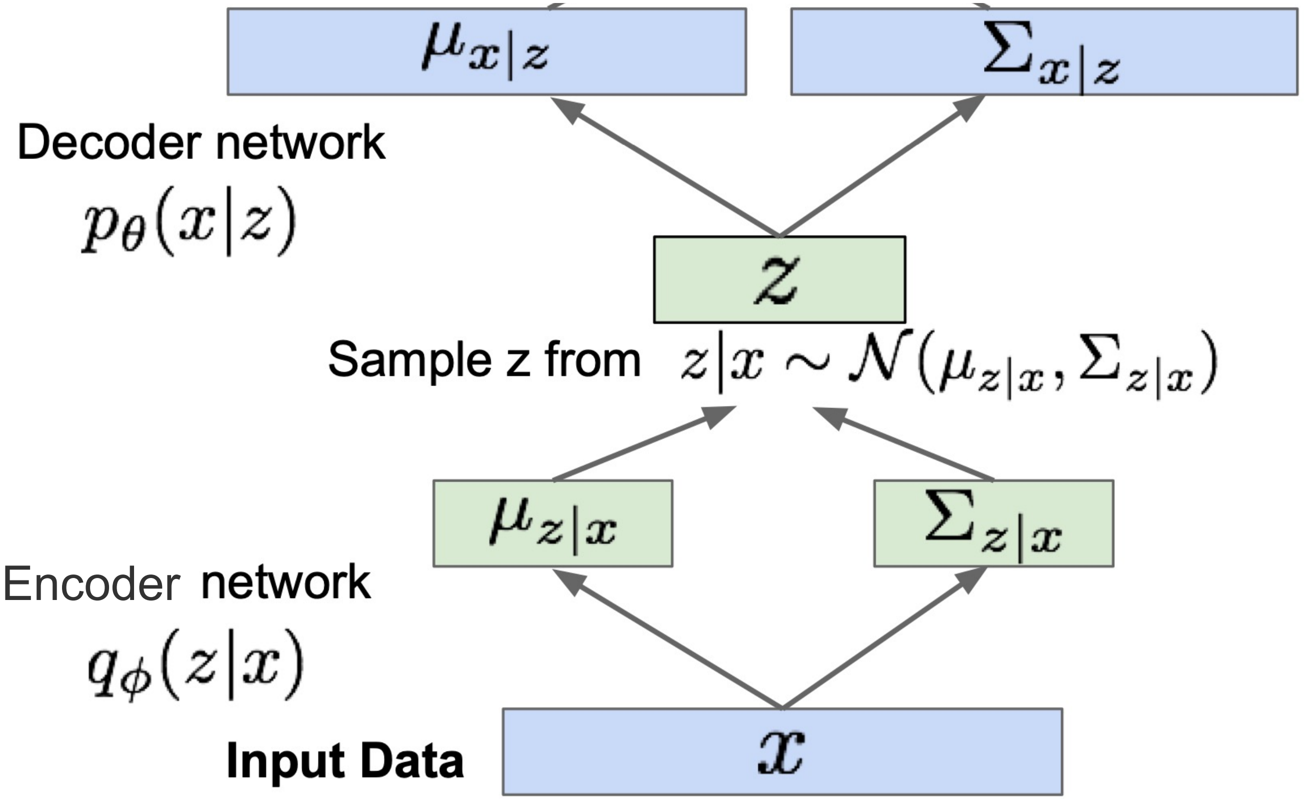




[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

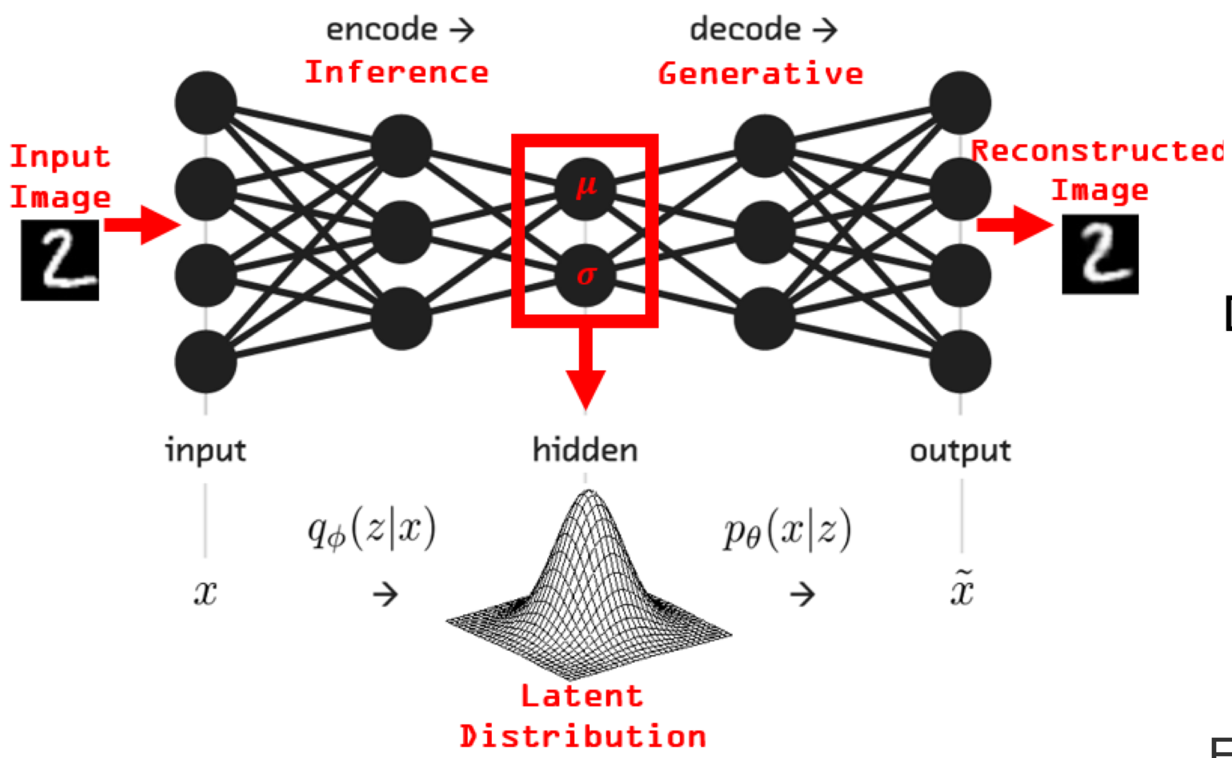
Example: VAEs for images

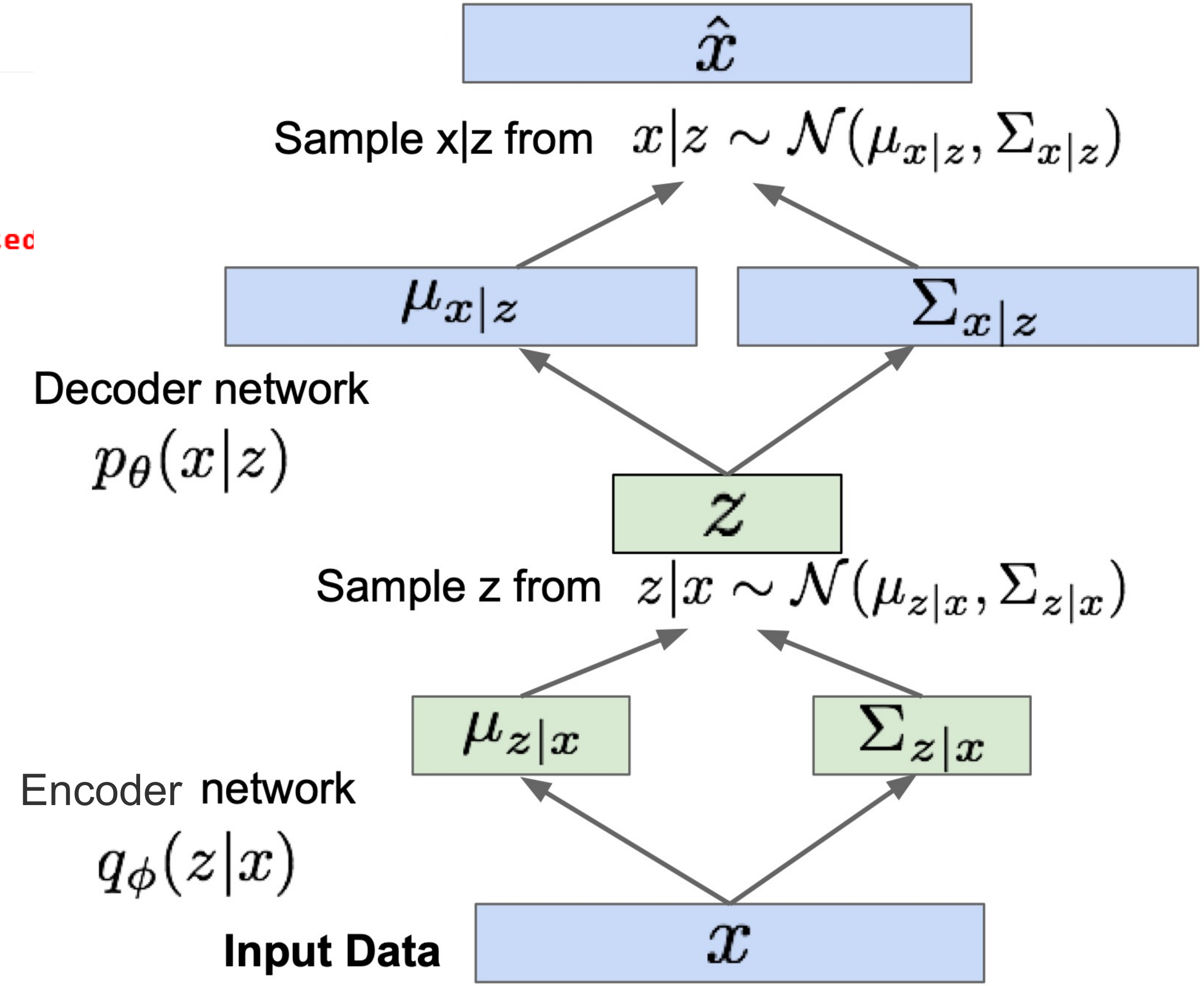




[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

Example: VAEs for images



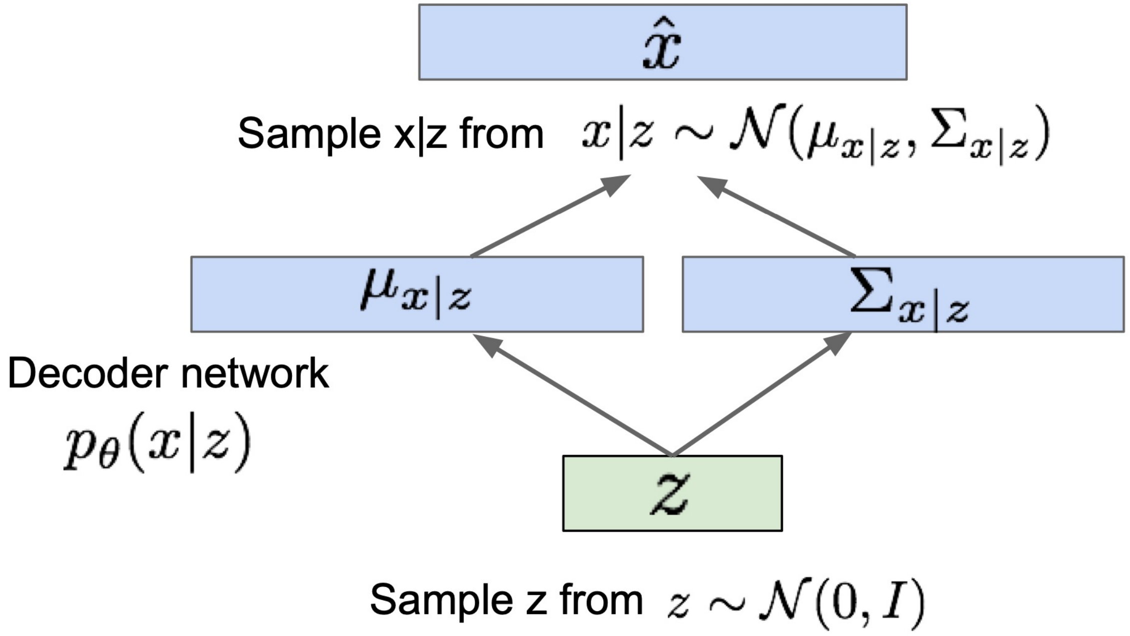


[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

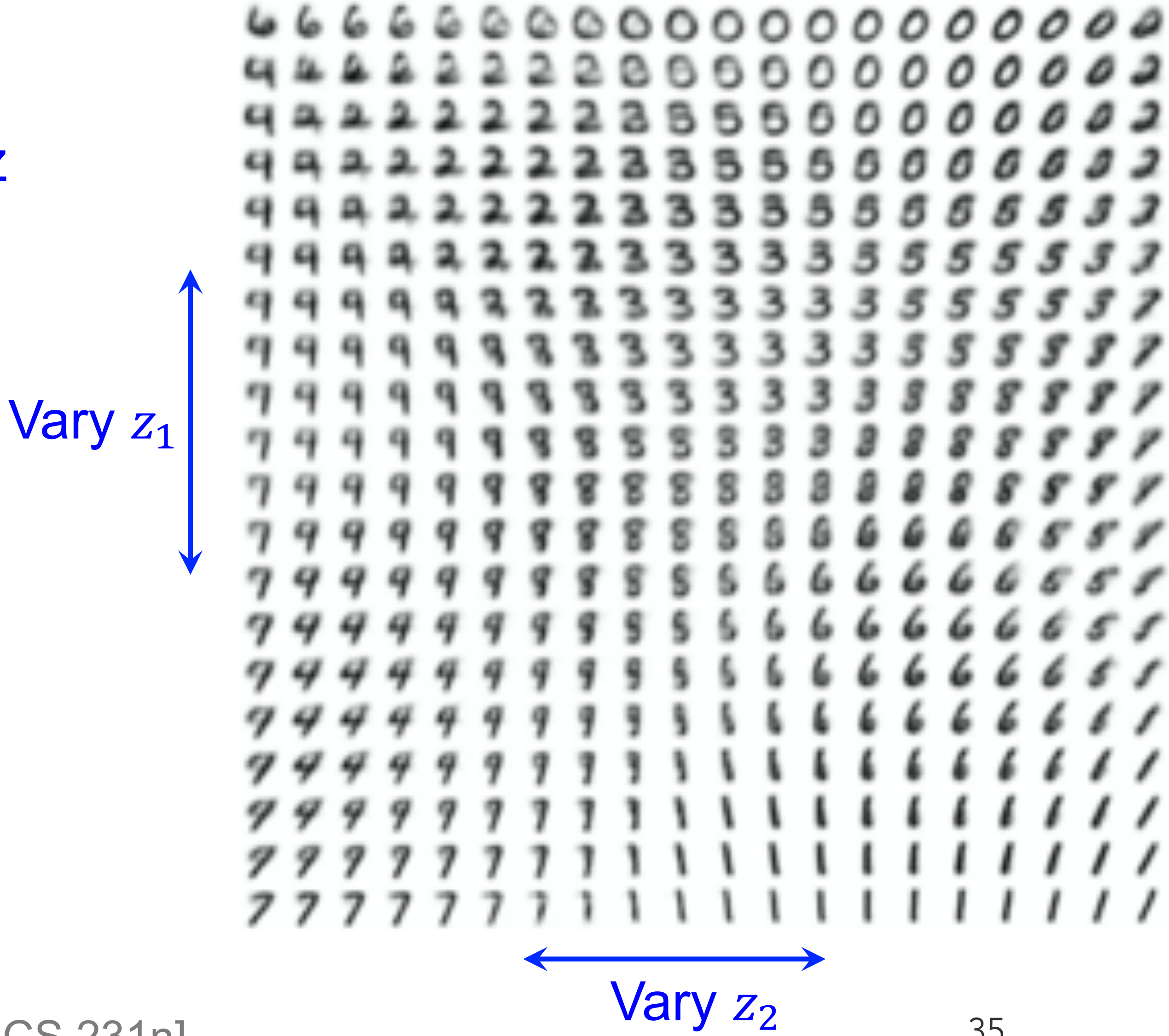
Example: VAEs for images

Generating samples:

- Use decoder network. Now sample z from prior!



Data manifold for 2-d **z**

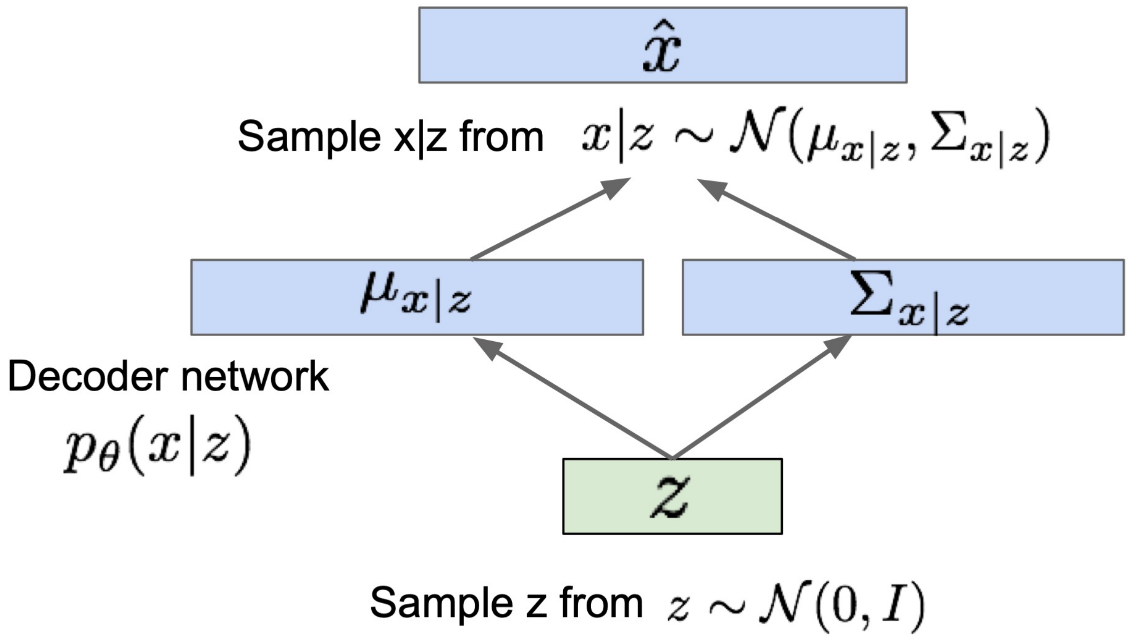


[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

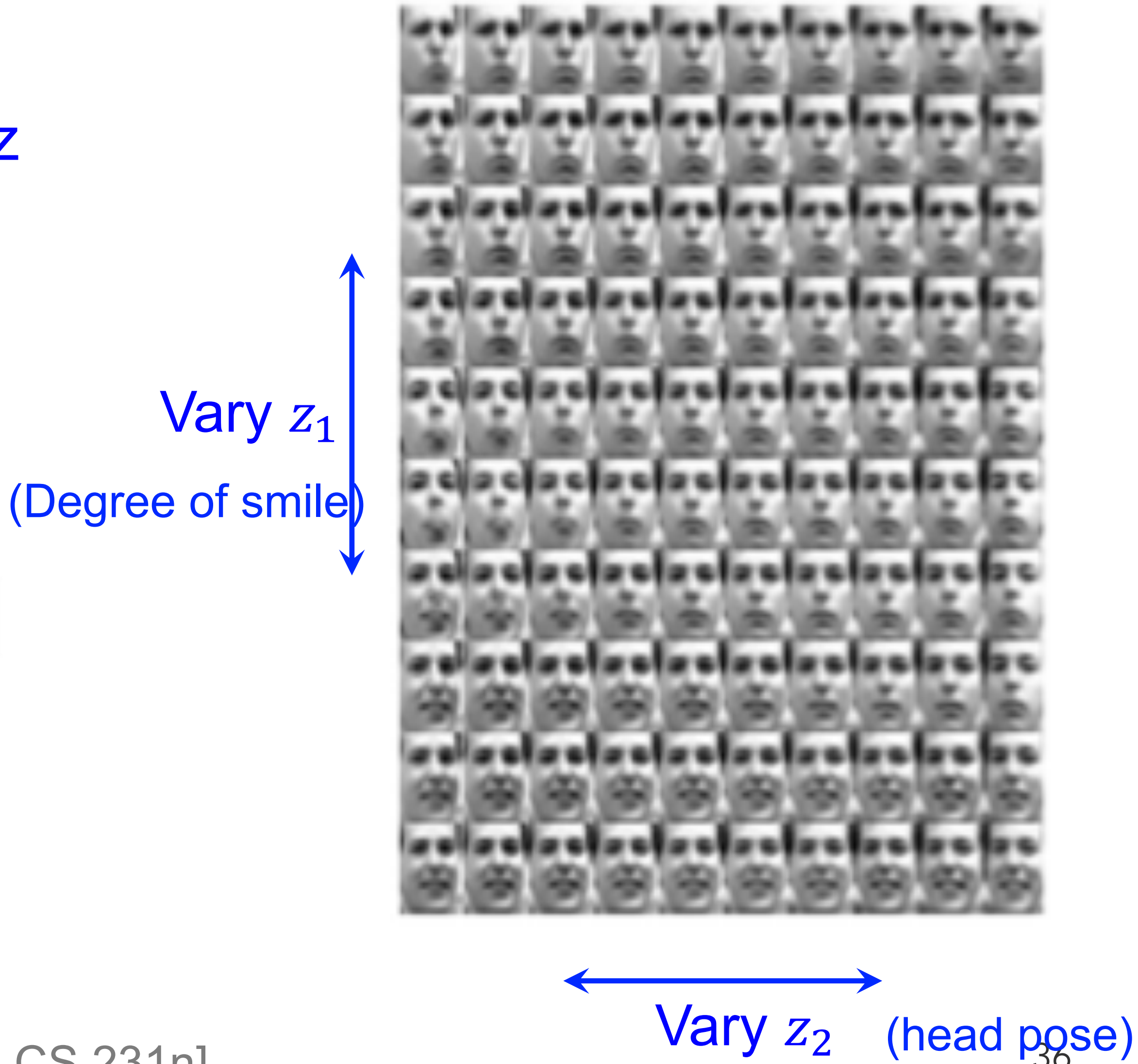
Example: VAEs for images

Generating samples:

- Use decoder network. Now sample z from prior!



Data manifold for 2-d **z**



[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

Example: VAEs for text

- Latent code interpolation and sentences generation from VAEs [Bowman et al., 2015].

“ i want to talk to you . ”
“i want to be with you . ”
“i do n’t want to be with you . ”
i do n’t want to be with you .
she did n’t want to be with him .

Variational Auto-Encoders (VAEs)

Algorithm 1 Minibatch version of the Auto-Encoding VB (AEVB) algorithm. Either of the two SGVB estimators in section 2.3 can be used. We use settings $M = 100$ and $L = 1$ in experiments.

$\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Initialize parameters
repeat
  $\mathbf{X}^M \leftarrow$ Random minibatch of M datapoints (drawn from full dataset)
  $\boldsymbol{\epsilon} \leftarrow$ Random samples from noise distribution $p(\boldsymbol{\epsilon})$
  $\mathbf{g} \leftarrow \nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \widetilde{\mathcal{L}}^M(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{X}^M, \boldsymbol{\epsilon})$ (Gradients of minibatch estimator (8))
  $\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Update parameters using gradients $\mathbf{g}$ (e.g. SGD or Adagrad [DHS10])
until convergence of parameters $(\boldsymbol{\theta}, \boldsymbol{\phi})$
return $\boldsymbol{\theta}, \boldsymbol{\phi}$

[Kingma & Welling, 2014]

Note: Amortized Variational Inference

- Variational distribution as an inference model $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ with parameters $\boldsymbol{\phi}$ (which was traditionally factored over samples)
- Amortize the cost of inference by learning a **single** data-dependent inference model
- The trained inference model can be used for quick inference on new data

Variational Auto-encoders: Summary

- A combination of the following ideas:
  - Variational Inference: ELBO
  - Variational distribution parametrized as neural networks
  - Reparameterization trick

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = [\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z}))$$

Reconstruction — Divergence from prior

(Razavi et al., 2019)

- Pros:
  - Principled approach to generative models
  - Allows inference of $q(z|x)$, can be useful feature representation for other tasks
- Cons:
  - Samples blurrier and lower quality compared to GANs
  - Tend to collapse on text data

Key Takeaways

- Stochastic VI
- Computing Gradients of Expectations $\mathcal{L} = \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{z})}[f_\lambda(\boldsymbol{z})]$
  - **Score gradient**

$$\nabla_\lambda \mathcal{L} = \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{z})}[f_\lambda(\boldsymbol{z}) \nabla_\lambda \log q_\theta(\boldsymbol{z}) + \nabla_\lambda f_\lambda(\boldsymbol{z})]$$

  - **Reparameterization gradient**

$$\nabla_\lambda \mathcal{L} = \mathbb{E}_{\boldsymbol{\epsilon} \sim \boldsymbol{s}(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{z}} f_\lambda(\boldsymbol{z})\, \nabla_\lambda t(\epsilon, \lambda) + \nabla_{\boldsymbol{\lambda}} f_\lambda(\boldsymbol{z})]$$

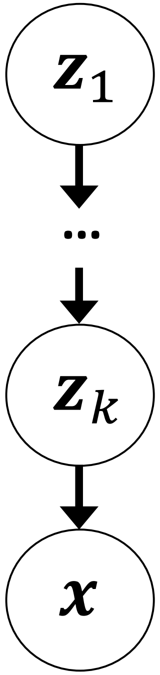
- Black-box VI
- Variational autoencoders (VAEs)

Questions?

Backups

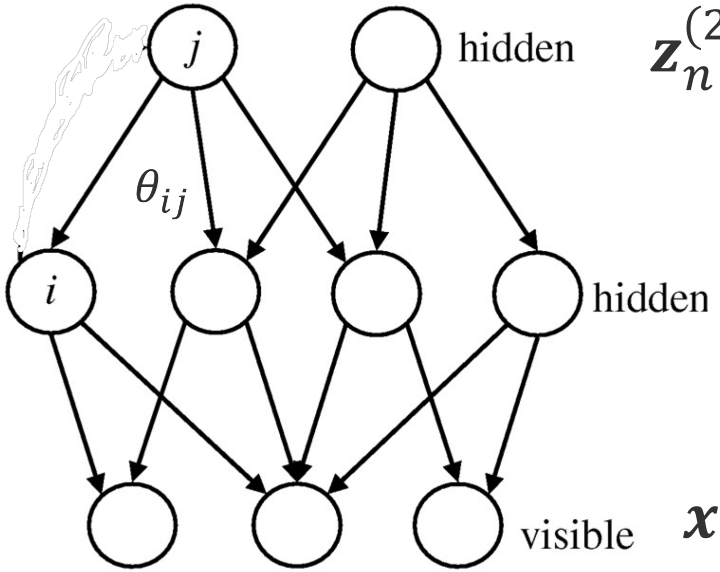
Deep generative models

- Define probabilistic distributions over a set of variables
- "Deep" means multiple layers of hidden variables!



Early forms of deep generative models

- Hierarchical Bayesian models
  - Sigmoid brief nets [Neal 1992]



$\boldsymbol{z}_n^{(2)} = \{0,1\}^J$

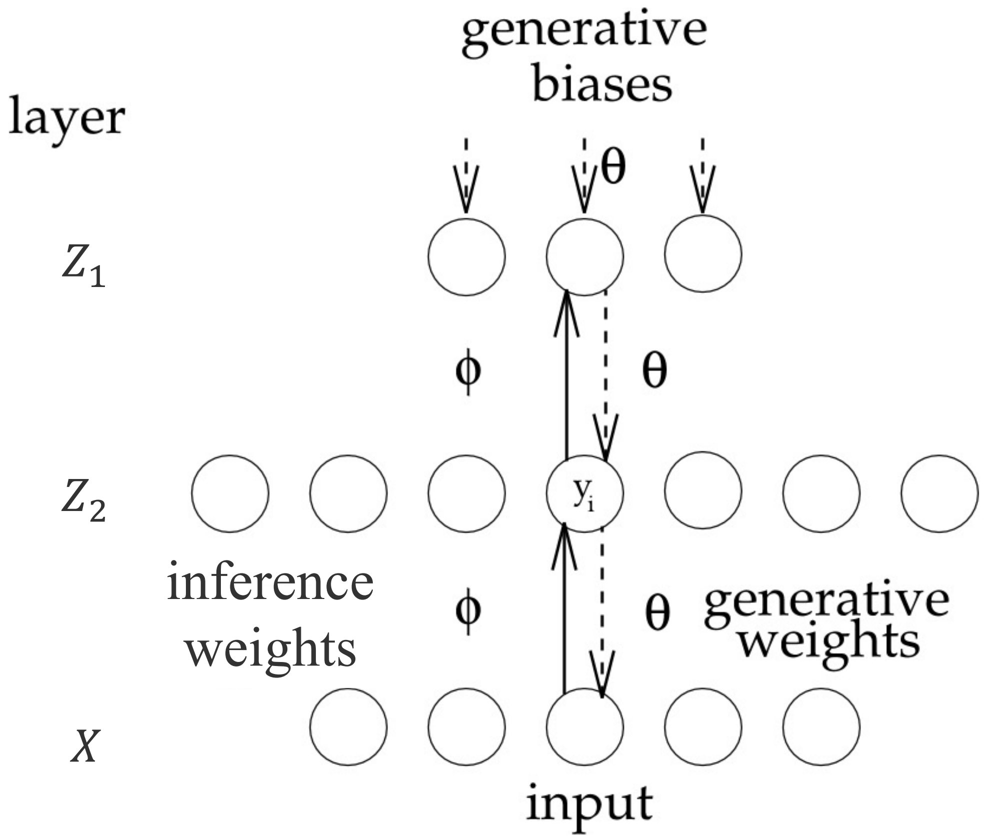
$\boldsymbol{z}_n^{(1)} = \{0,1\}^I$

$\boldsymbol{x}_n = \{0,1\}^K$

$$p\left(x_{kn} = 1 \middle| \boldsymbol{\theta}_k, \boldsymbol{z}_n^{(1)}\right) = \sigma\left(\boldsymbol{\theta}_k^T \boldsymbol{z}_n^{(1)}\right)$$
$$p\left(z_{in}^{(1)} = 1 \middle| \boldsymbol{\theta}_i, \boldsymbol{z}_n^{(2)}\right) = \sigma\left(\boldsymbol{\theta}_i^T \boldsymbol{z}_n^{(2)}\right)$$

Early forms of deep generative models

- Hierarchical Bayesian models
  - Sigmoid brief nets [Neal 1992]
- Neural network models
  - Helmholtz machines [Dayan et al.,1995]



[Dayan et al. 1995]

Early forms of deep generative models

- Hierarchical Bayesian models
  - Sigmoid brief nets [Neal 1992]

- Neural network models
  - Helmholtz machines [Dayan et al.,1995]
  - Predictability minimization [Schmidhuber 1995]

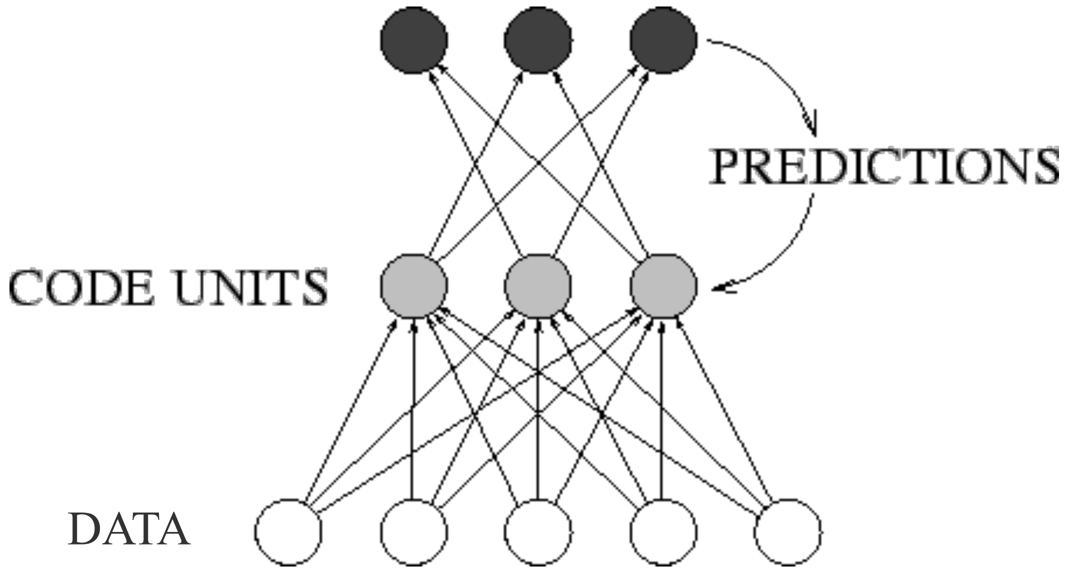


Figure courtesy: Schmidhuber 1996

Early forms of deep generative models

- Training of DGMs via an EM style framework
  - Sampling / data augmentation

$$\boldsymbol{z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2\}$$
$$\boldsymbol{z}_1^{new} \sim p(\boldsymbol{z}_1 | \boldsymbol{z}_2, \boldsymbol{x})$$
$$\boldsymbol{z}_2^{new} \sim p(\boldsymbol{z}_2 | \boldsymbol{z}_1^{new}, \boldsymbol{x})$$

  - Variational inference

$$\log p(\boldsymbol{x}) \geq \mathrm{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}, \boldsymbol{z})] - \mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z})) := \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x})$$

$$\max_{\boldsymbol{\theta}, \boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x})$$

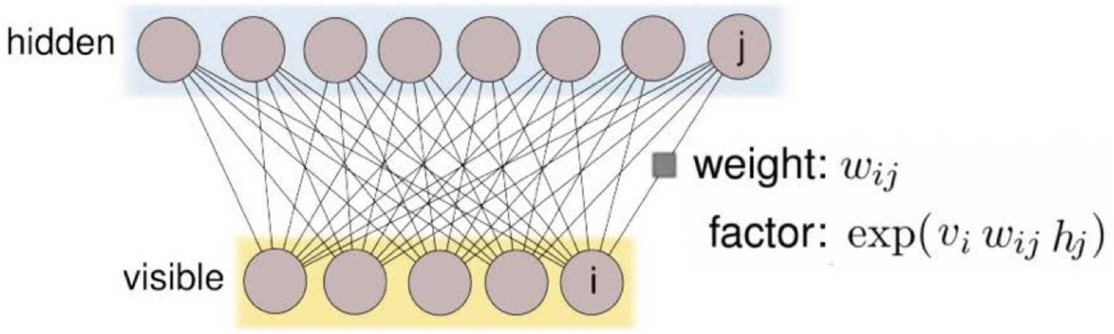
  - Wake sleep

Wake: $\min_\theta \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$

Sleep: $\min_\phi \mathbb{E}_{p_\theta(x|z)}[\log q_\phi(z|x)]$

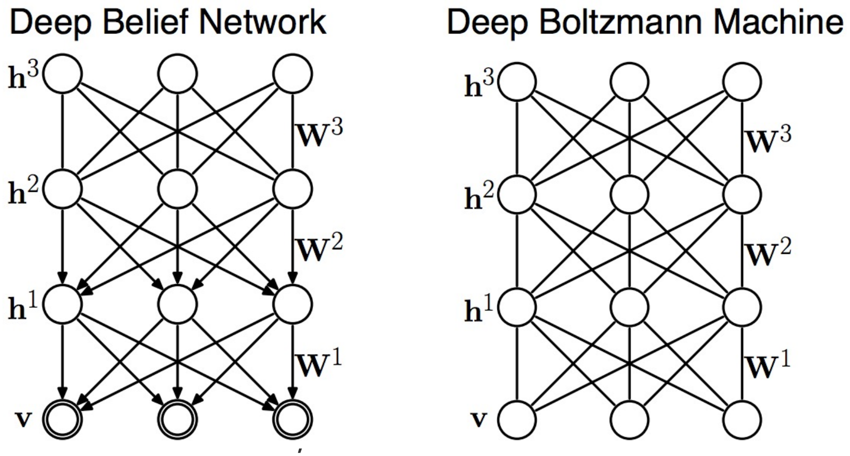
Resurgence of deep generative models

- Restricted Boltzmann machines (RBMs) [Smolensky, 1986]
  - Building blocks of deep probabilistic models



Resurgence of deep generative models

- Restricted Boltzmann machines (RBMs) [Smolensky, 1986]
  - Building blocks of deep probabilistic models
- Deep belief networks (DBNs) [Hinton et al., 2006]
  - Hybrid graphical model
  - Inference in DBNs is problematic due to explaining away
- Deep Boltzmann Machines (DBMs) [Salakhutdinov & Hinton, 2009]
  - Undirected model

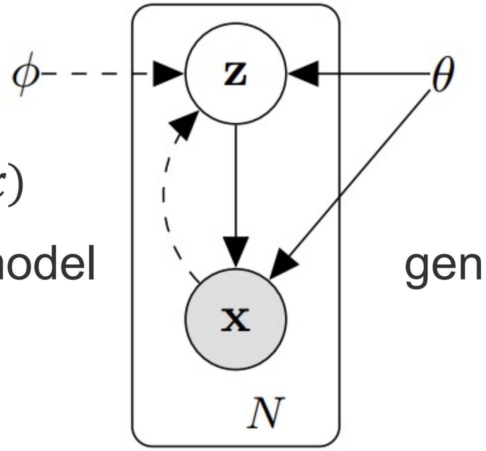


Resurgence of deep generative models

- Variational autoencoders (VAEs) [Kingma & Welling, 2014]
  / Neural Variational Inference and Learning (NVIL) [Mnih & Gregor, 2014]

$q_{\phi}(\boldsymbol{z}|\boldsymbol{x})$

inference model



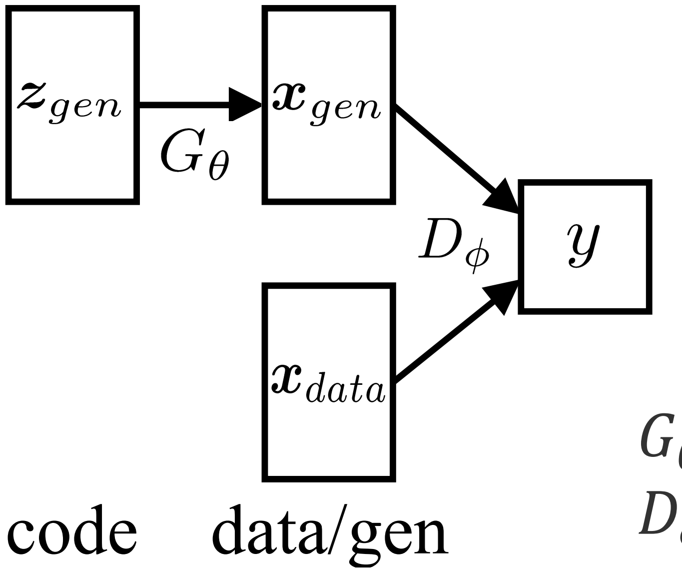
$p_{\theta}(\boldsymbol{x}|\boldsymbol{z})$

generative model

Figure courtesy: Kingma & Welling, 2014

Resurgence of deep generative models

- Variational autoencoders (VAEs) [Kingma & Welling, 2014]
  / Neural Variational Inference and Learning (NVIL) [Mnih & Gregor, 2014]
- Generative adversarial networks (GANs) [Goodfellow et al,. 2014]



G_θ: generative model ?

D_ϕ: discriminator

Resurgence of deep generative models

- Variational autoencoders (VAEs) [Kingma & Welling, 2014]
  / Neural Variational Inference and Learning (NVIL) [Mnih & Gregor, 2014]
- Generative adversarial networks (GANs) [Goodfellow et al,. 2014]
- Generative moment matching networks (GMMNs) [Li et al., 2015; Dziugaite et al., 2015]

Resurgence of deep generative models

- Variational autoencoders (VAEs) [Kingma & Welling, 2014]
  / Neural Variational Inference and Learning (NVIL) [Mnih & Gregor, 2014]
- Generative adversarial networks (GANs) [Goodfellow et al,. 2014]
- Generative moment matching networks (GMMNs) [Li et al., 2015; Dziugaite et al., 2015]
- Autoregressive neural networks