DSC190: Machine Learning with Few Labels

Unsupervised Learning

Zhiting Hu

Lecture 16, November 4, 2024

UC San Diego
HALICIOĞLU DATA SCIENCE INSTITUTE

Outline

Unsupervised learning: Variational Inference

Presentations

- **Peiyuan Sun**: Reasoning with Language Model is Planning with World Model
- **Mingyang Yao**: Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions
- **Zhaoxiang Feng**: Learning Equilibria in Matching Markets from Bandit Feedback
- **Bella Wang**: Language Models Are Realistic Tabular Data Generators

Recall that in EM, we assume $q(z|x)$ can be any distribution. E-step shows the optimal $q(z|x)$ is the posterior distribution.

Recap: Variational Inference

The main idea behind variational inference:

- Choose a family of distributions over the latent variables $z_{1:m}$ with its own set of variational parameters ν , i.e.

$$q(z_{1:m}|\nu)$$

- Then, we find the setting of the parameters that makes our approximation q closest to the posterior distribution.
  - This is where optimization algorithms come in.

- Then we can use q with the fitted parameters in place of the posterior.
  - E.g. to form predictions about future data, or to investigate the posterior distribution over the hidden variables, find modes, etc.

Recap: Variational Inference

- We want to minimize the KL divergence between our approximation $q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})$ and our posterior $p(\boldsymbol{z}|\boldsymbol{x})$

$$\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu}) \,||\, p(\boldsymbol{z}|\boldsymbol{x}))$$

  - But we can't actually minimize this quantity w.r.t q because $p(\boldsymbol{z}|\boldsymbol{x})$ is unknown

- **Question:** how can we minimize the KL divergence?

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

Evidence Lower Bound (ELBO)

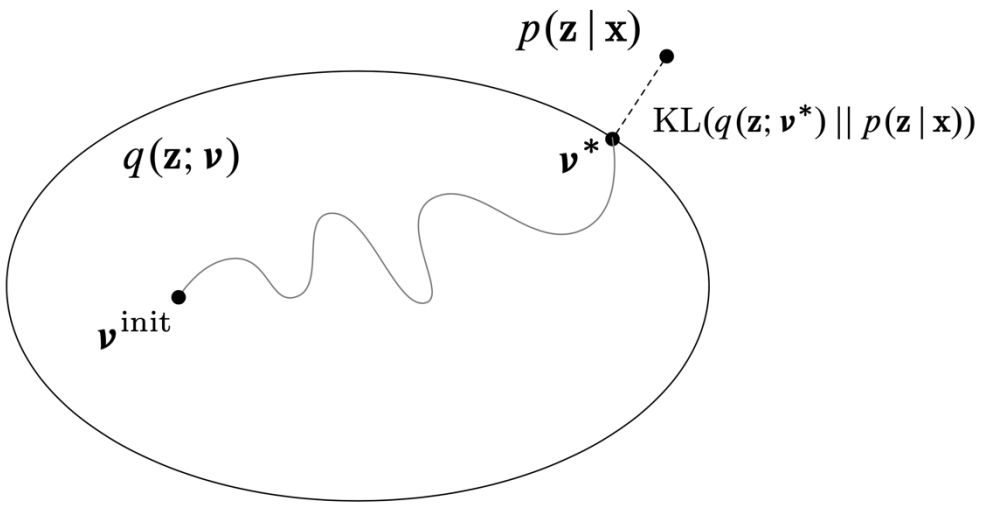
- The ELBO is equal to the negative KL divergence up to a constant $\ell(\theta;\boldsymbol{x})$
- We maximize the ELBO over q to find an "optimal approximation" to $p(\boldsymbol{z}|\boldsymbol{x})$

Variational Inference

- Choose a family of distributions over the latent variables $\boldsymbol{z}$ with its own set of variational parameters ν , i.e. $q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})$
- We maximize the ELBO over q to find an "optimal approximation" to $p(\boldsymbol{z}|\boldsymbol{x})$

$$\text{argmax}_{\nu}\ \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z}|\theta)}{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}\right]$$

$$= \text{argmax}_{\nu}\ \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}[\log p(\boldsymbol{x},\boldsymbol{z}|\theta)] - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}[\log q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})]$$

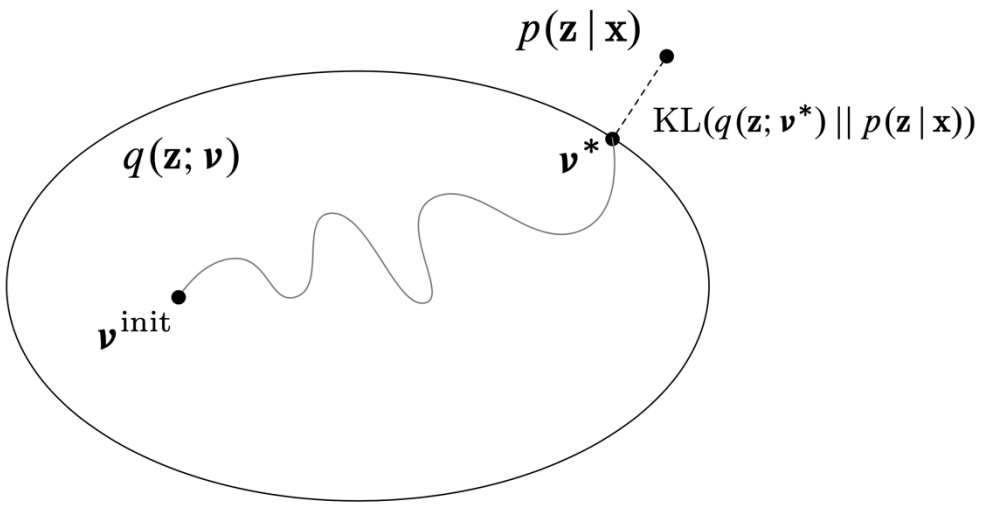


Variational Inference

- Choose a family of distributions over the latent variables $\boldsymbol{z}$ with its own set of variational parameters ν , i.e. $q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})$
- We maximize the ELBO over q to find an "optimal approximation" to $p(\boldsymbol{z}|\boldsymbol{x})$

$$\text{argmax}_{\nu}\ \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z}|\theta)}{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}\right]$$

$$= \text{argmax}_{\nu}\ \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}[\log p(\boldsymbol{x},\boldsymbol{z}|\theta)] - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}[\log q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})]$$



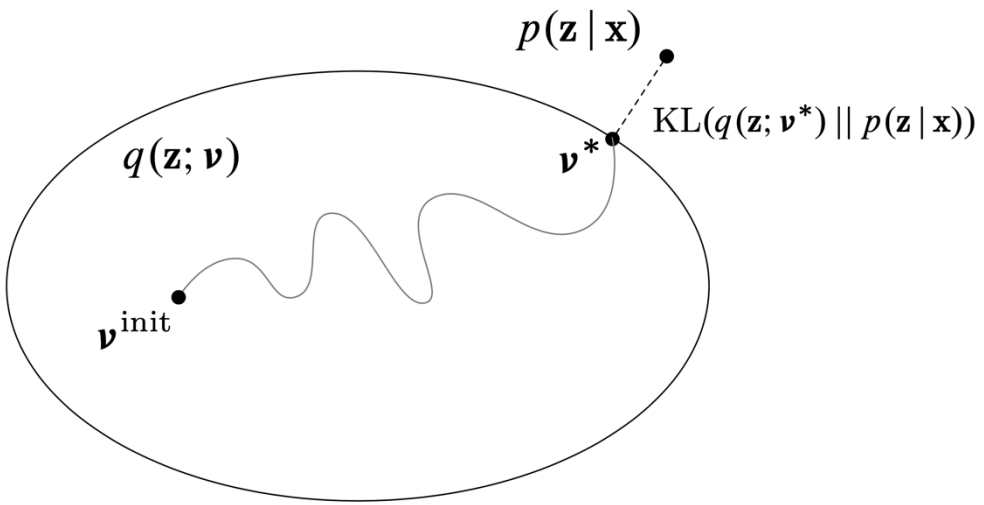
Question: How do we choose the variational family $q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})$?

Variational Inference

- Choose a family of distributions over the latent variables $\boldsymbol{z}$ with its own set of variational parameters ν , i.e. $q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})$
- We maximize the ELBO over q to find an "optimal approximation" to $p(\boldsymbol{z}|\boldsymbol{x})$

$$\text{argmax}_{\nu}\ \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z}|\theta)}{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}\right]$$

$$= \text{argmax}_{\nu}\ \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}[\log p(\boldsymbol{x},\boldsymbol{z}|\theta)] - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})}[\log q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})]$$



Question: How do we choose the variational family $q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\nu})$?

- Factorized distribution -> mean field VI
- Mixture of Gaussian distribution -> black-box VI
- Neural-based distribution -> Variational Autoencoders (VAEs)

Example: Mean Field Variational Inference

- A popular family of variational approximations

- In this type of variational inference, we assume the variational distribution over the latent variables factorizes as

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

  - (where we omit variational parameters for ease of notation)
  - We refer to $q(z_j)$, the variational approximation for a single latent variable, as a "local variational approximation"

- In the above expression, the variational approximation $q(z_j)$ over each latent variable z_j is independent

Mean Field Variational Inference with Coordinate Ascent

Recap: Bayesian mixture of Gaussians

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

- Initialize the global variational distributions $q(\mu_k)$ and parameters $\{\tau^2, \sigma^2, \pi\}$
- **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 $q(\mu_k)$
  - Update the parameters $\{\tau^2, \sigma^2, \pi\}$
- **Until** ELBO converges

(Variational) E-step

(Variational) M-step

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

Stochastic VI

Recap: Bayesian mixture of Gaussians

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

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

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

Black-box Variational Inference

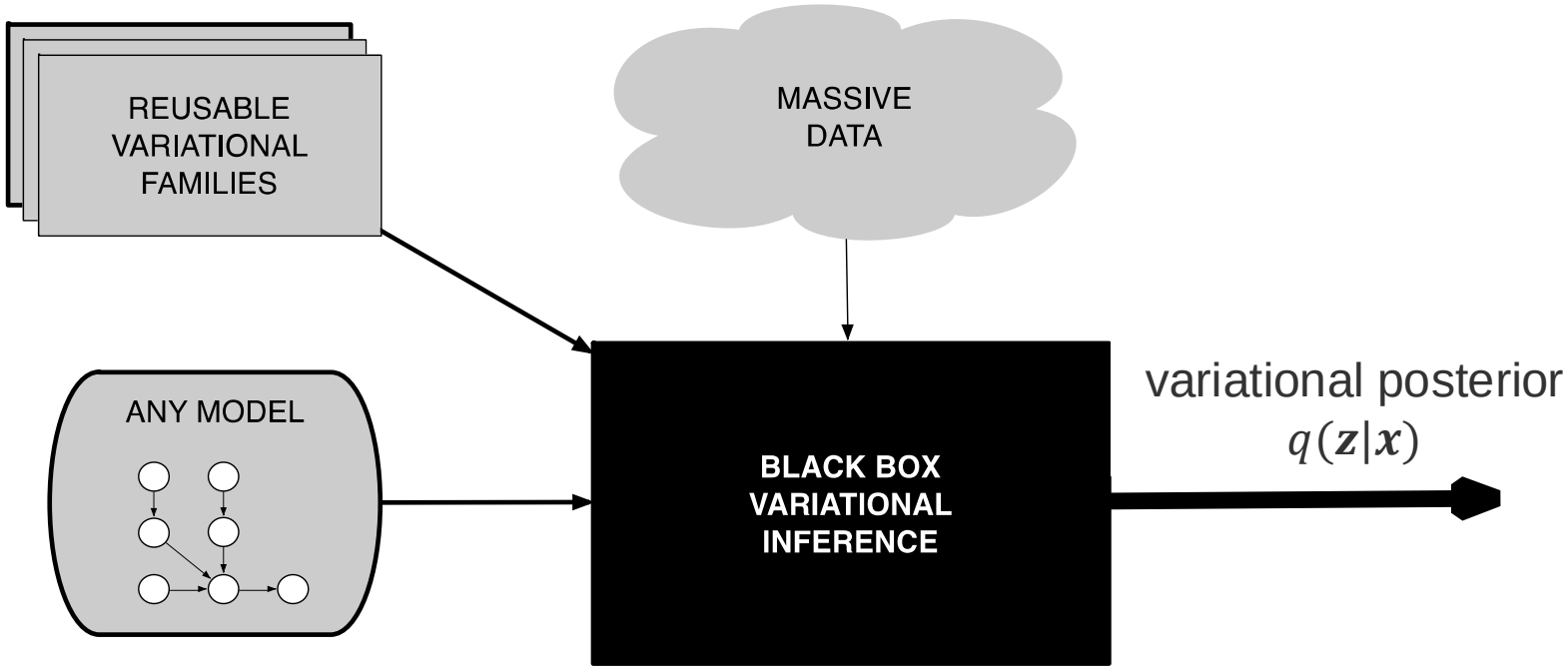
Black-box Variational Inference (BBVI)

- 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?

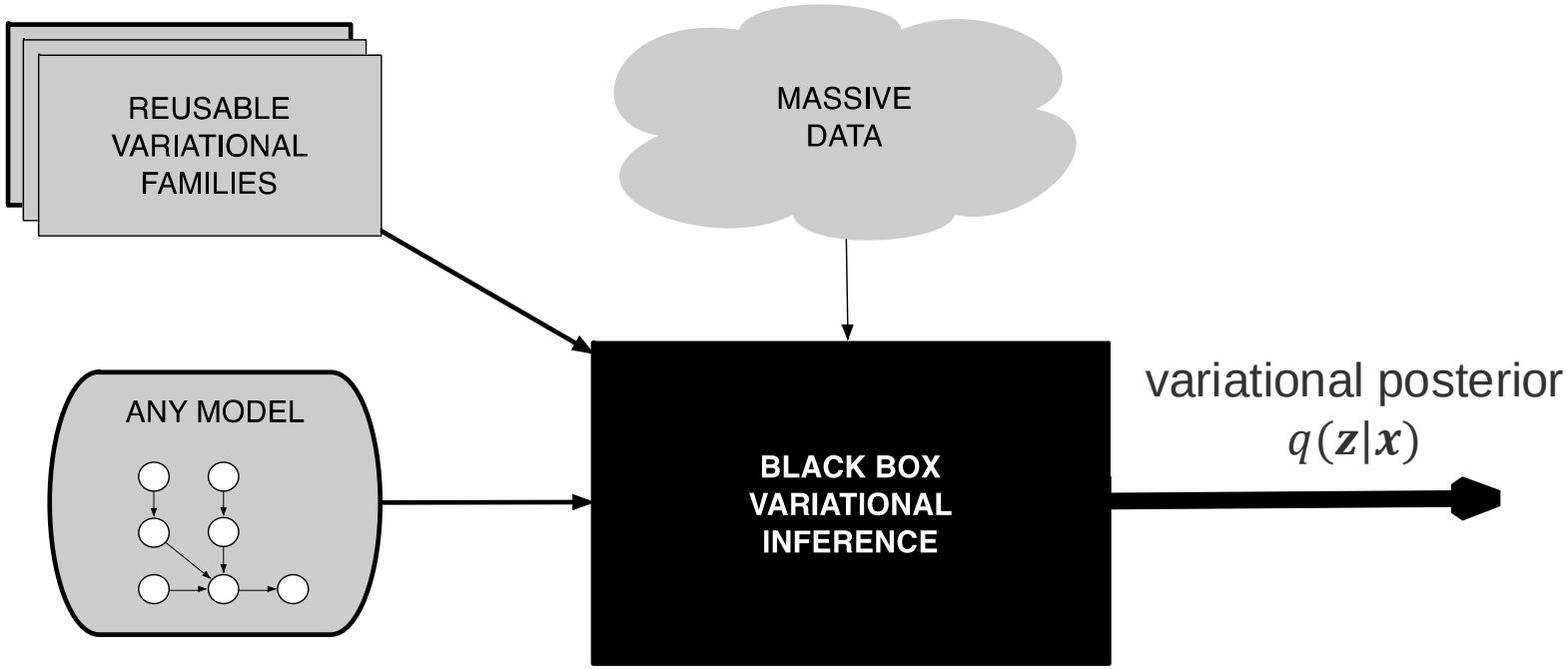
Black-box Variational Inference (BBVI)



- 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 $q_\lambda(\boldsymbol{z})$

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

- Computing exact gradients w.r.t. the parameters λ is often infeasible
- 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: $\mathcal{L} = \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{z})}[f_\lambda(\boldsymbol{z})]$
- Log-derivative trick: $\nabla_\lambda q_\lambda = q_\lambda \nabla_\lambda \log q_\lambda$
- **Question:** show that the gradient of $\mathcal{L}$ w.r.t. λ is:

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

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 of $\mathcal{L}$ 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
- Monte Carlo estimation of the expectation:
  - Compute noisy unbiased gradients with Monte Carlo samples from q_λ

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

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 of $\mathcal{L}$ 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
- Monte Carlo estimation of the expectation:
  - Compute noisy unbiased gradients with Monte Carlo samples from q_λ

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

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

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:

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

  - **Question:** what's the gradient of $\mathcal{L}$ w.r.t. λ ?

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:

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

  - **Question:** what's the gradient of $\mathcal{L}$ w.r.t. λ ?

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

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

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

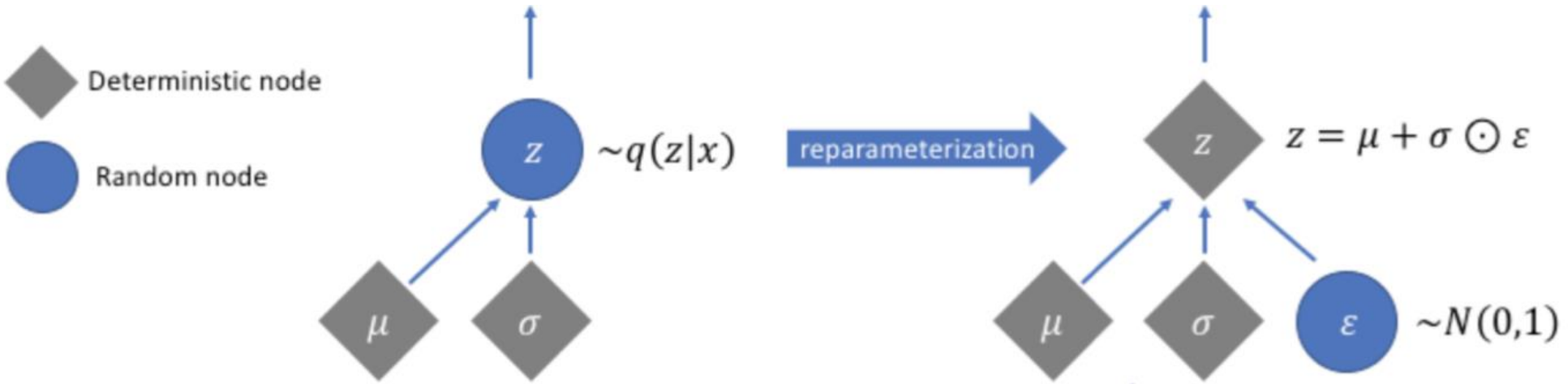
$$\nabla_\lambda \mathcal{L} = \mathbb{E}_{\boldsymbol{\epsilon}\sim \boldsymbol{s}(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{z}} f_\lambda(\boldsymbol{z})\, \nabla_\lambda t(\epsilon, \lambda)]$$

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

Reparameterization trick

- Reparametrizing Gaussian distribution

$$\epsilon \sim Normal(0,1) \\ z = \epsilon\sigma + \mu \Leftrightarrow \quad z \sim Normal(\mu, \sigma^2)$$



[Courtesy: Tansey, 2016]

Reparameterization trick

- Reparametrizing Gaussian distribution

$$\epsilon \sim Normal(0,1) \\ z = \epsilon\sigma + \mu \Leftrightarrow \quad z \sim Normal(\mu, \sigma^2)$$

- Other reparameterizable distributions: $\epsilon \sim Uniform(\epsilon) \\ z = F^{-1}(\epsilon) \Leftrightarrow \quad z \sim q(z)$
  - 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: $\mathcal{L} = \mathbb{E}_{q_{\boldsymbol{\lambda}}(\boldsymbol{z})}[f_\lambda(\boldsymbol{z})]$

- **Score gradient**

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

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

- **Reparameterization gradient**

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

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

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
- ELBO to be maximized:

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

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

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

[Ranganath et al.,14]

BBVI with the score gradient

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

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

- ELBO:

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

- **Question:** what's the score gradient w.r.t. λ ?

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

BBVI with the score gradient

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

- ELBO:

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

- **Question:** what's the score gradient w.r.t. λ ?

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

- 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

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

- ELBO:

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

- **Question:** what's the reparamerization 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 Autoencoders (VAEs)

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

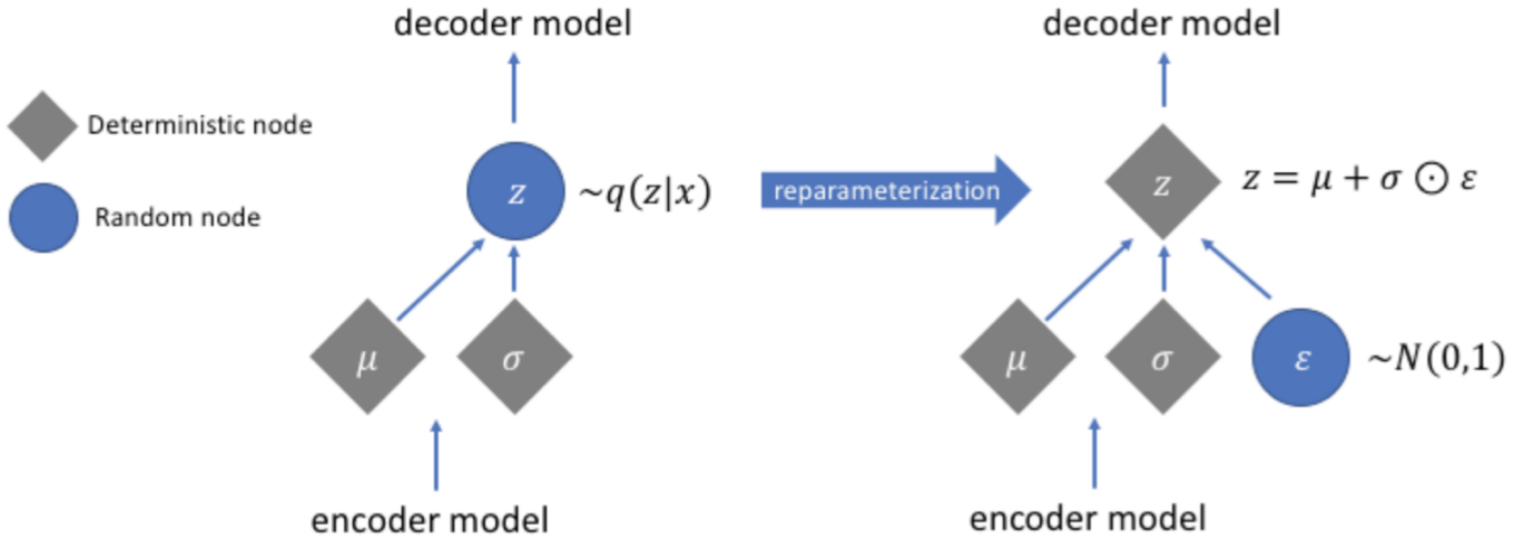
Divergence from prior
(KL divergence between two Guassians has an analytic form)

Variational Auto-Encoders (VAEs)

- ELBO:

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

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



Variational Auto-Encoders (VAEs)

- ELBO:

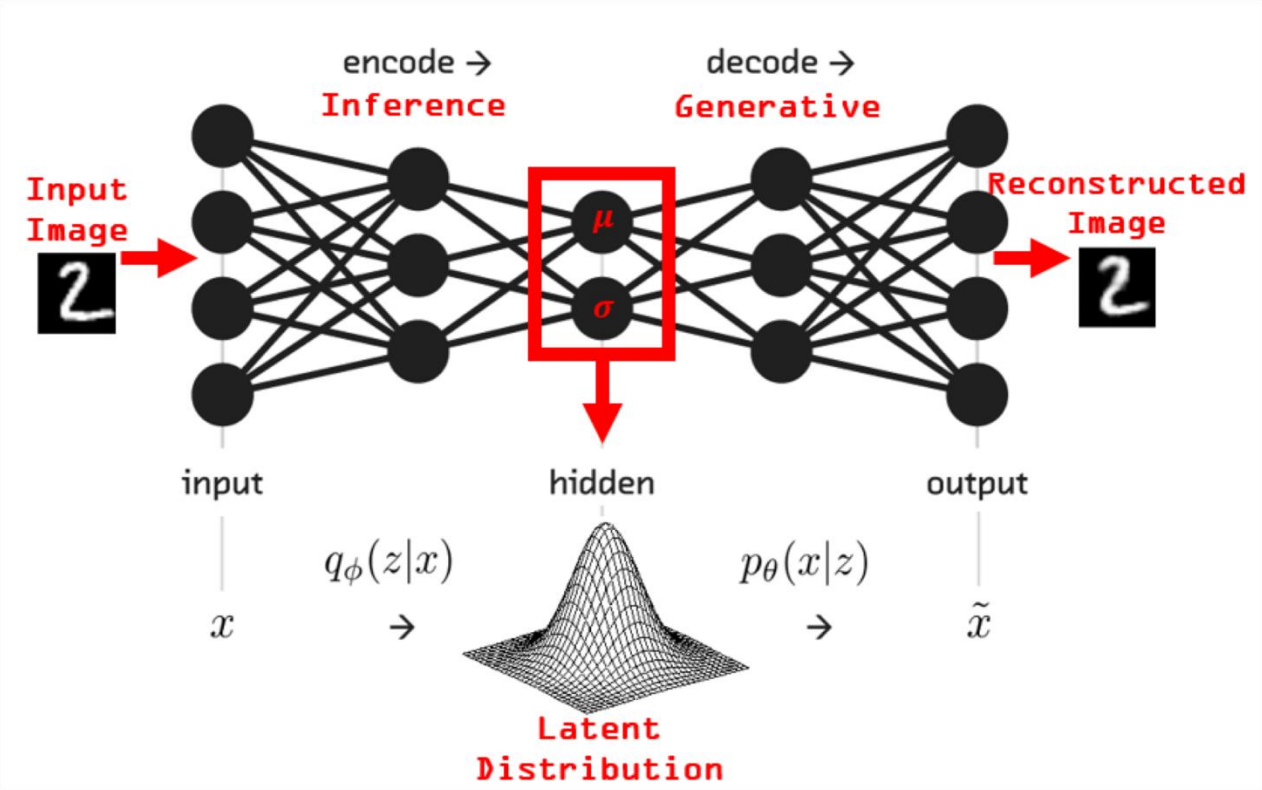
$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = \mathrm{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}, \boldsymbol{z})] + \mathrm{H}(q_\phi(\boldsymbol{z}|\boldsymbol{x}))$$
$$= \mathrm{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right]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

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

$$\nabla_{\boldsymbol{\phi}} \mathcal{L} = \mathrm{E}_{\epsilon \sim N(\boldsymbol{0},\boldsymbol{1})}\left[\nabla_{\boldsymbol{z}}\left[\log p_\theta(\boldsymbol{x}, \boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x})\right] \nabla_\phi z(\epsilon, \boldsymbol{\phi})\right]$$

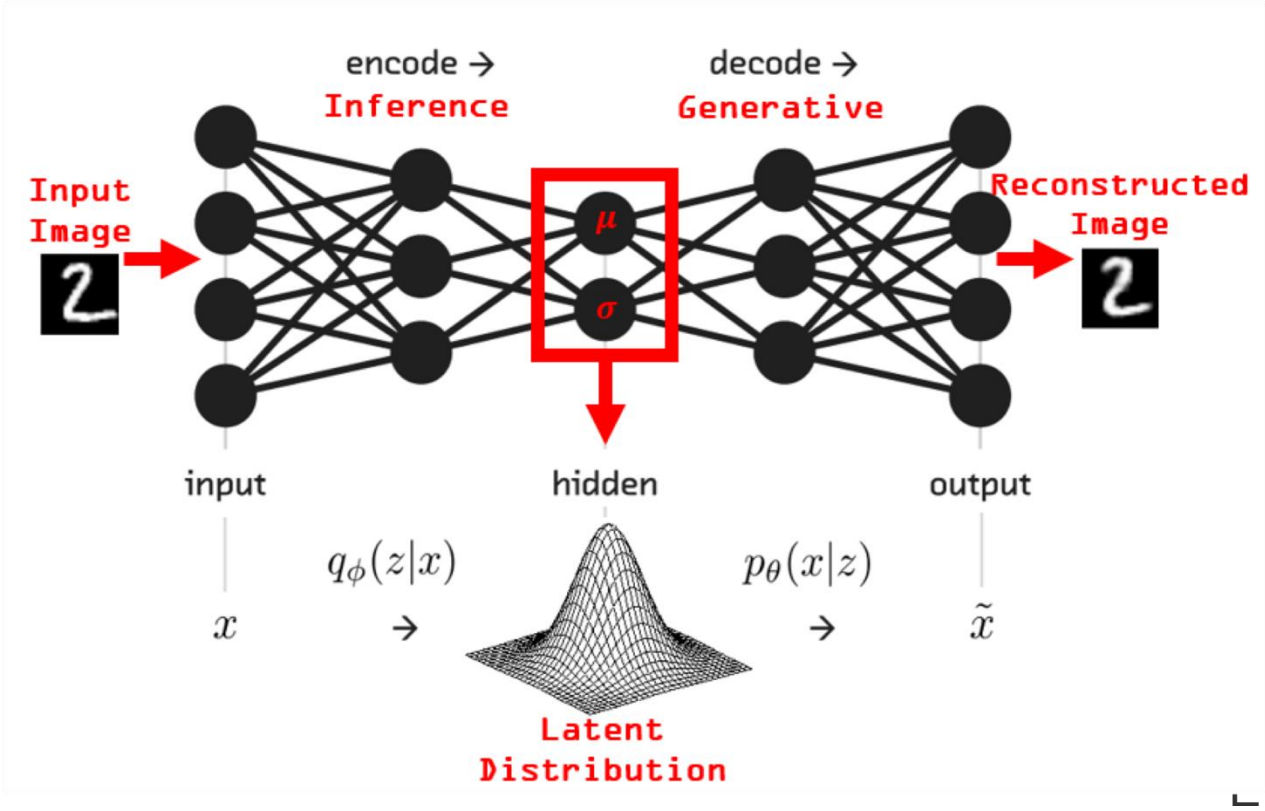
$$\nabla_\theta \mathcal{L} = \mathrm{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\nabla_\theta \log p_\theta(\boldsymbol{x}, \boldsymbol{z})]$$

Example: VAEs for images



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

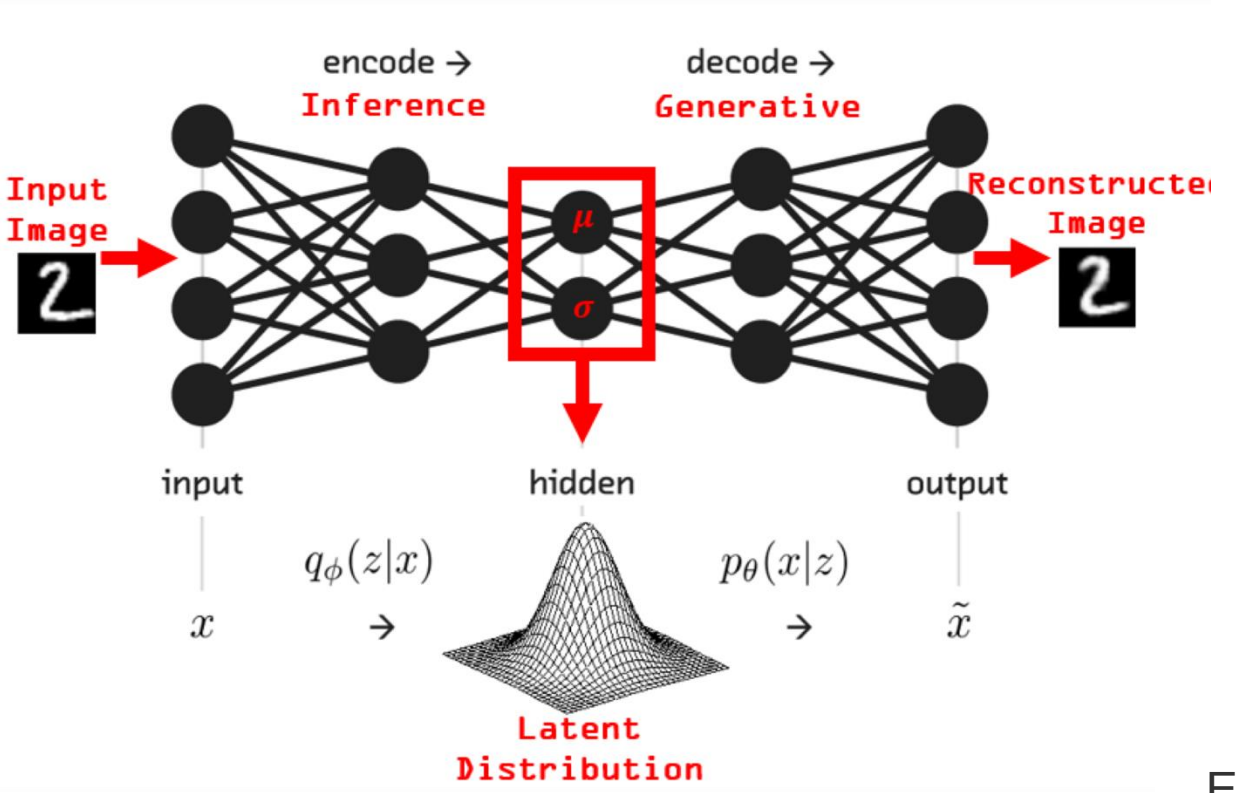
Example: VAEs for images



Input Data x

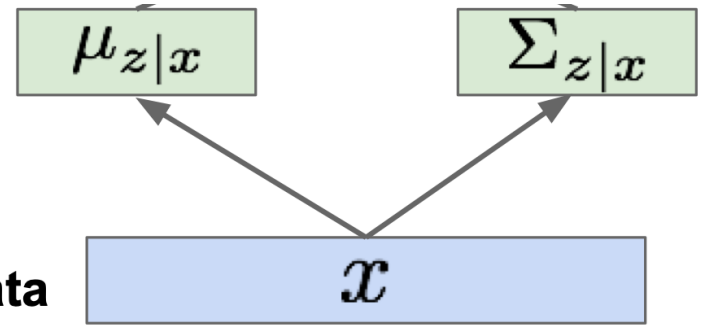
[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

Example: VAEs for images



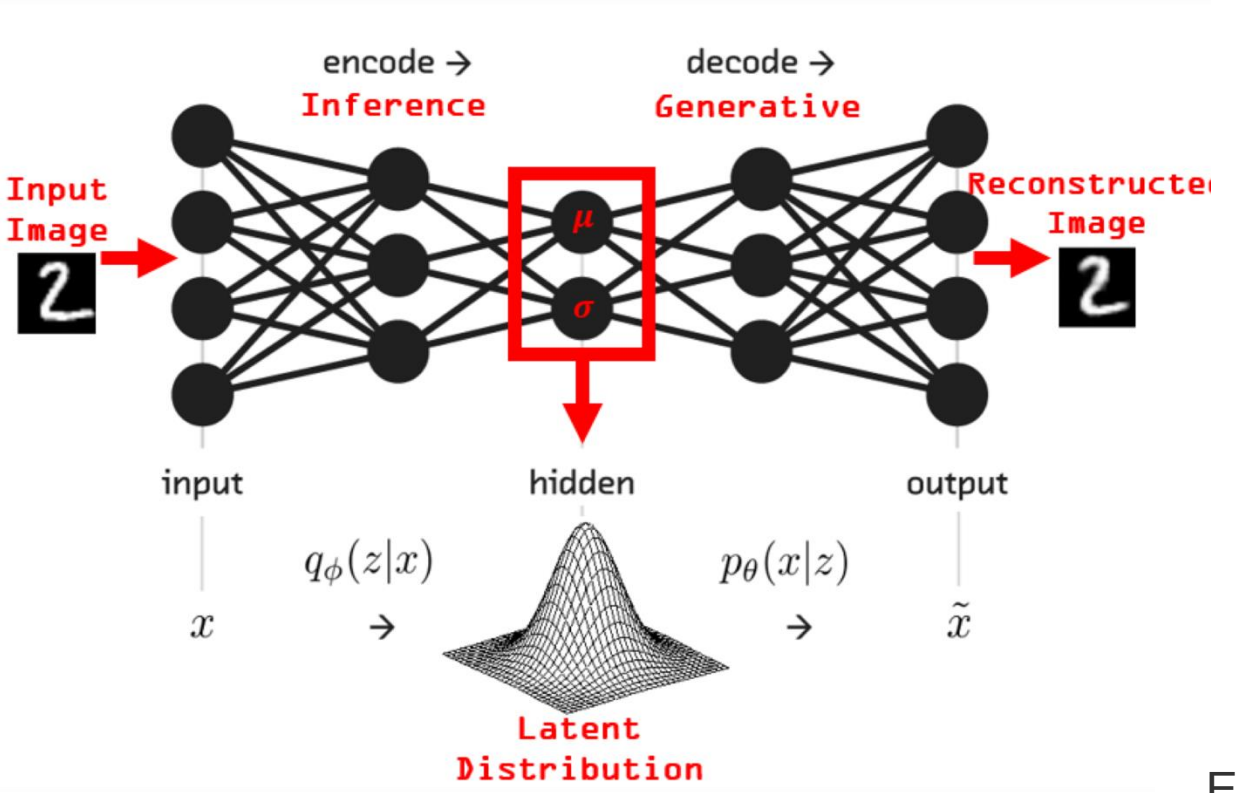
Encoder network

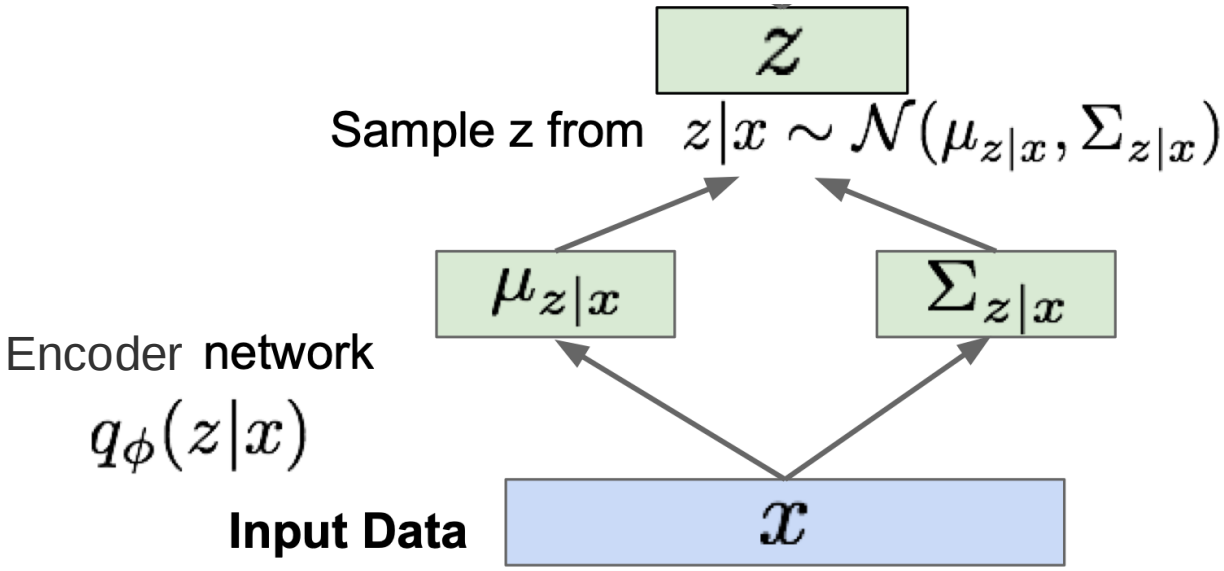
$q_\phi(z|x)$



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

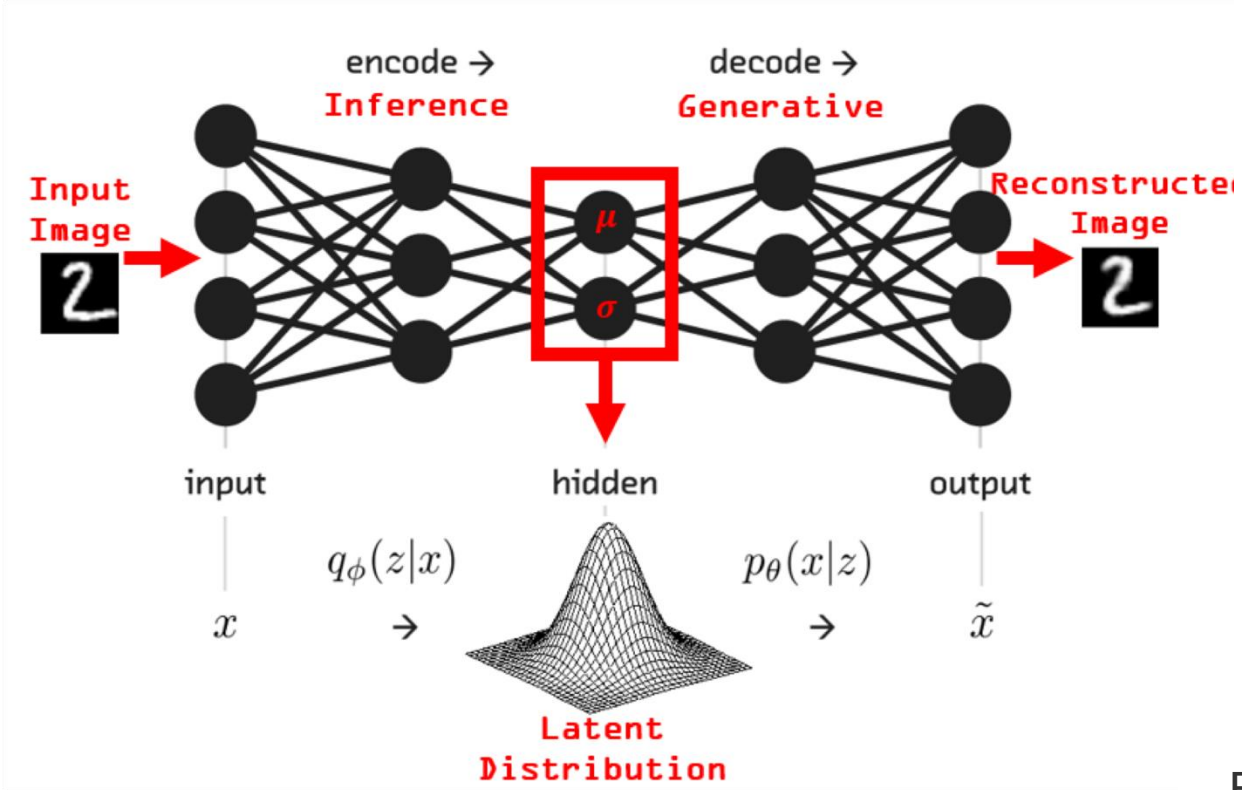
Example: VAEs for images

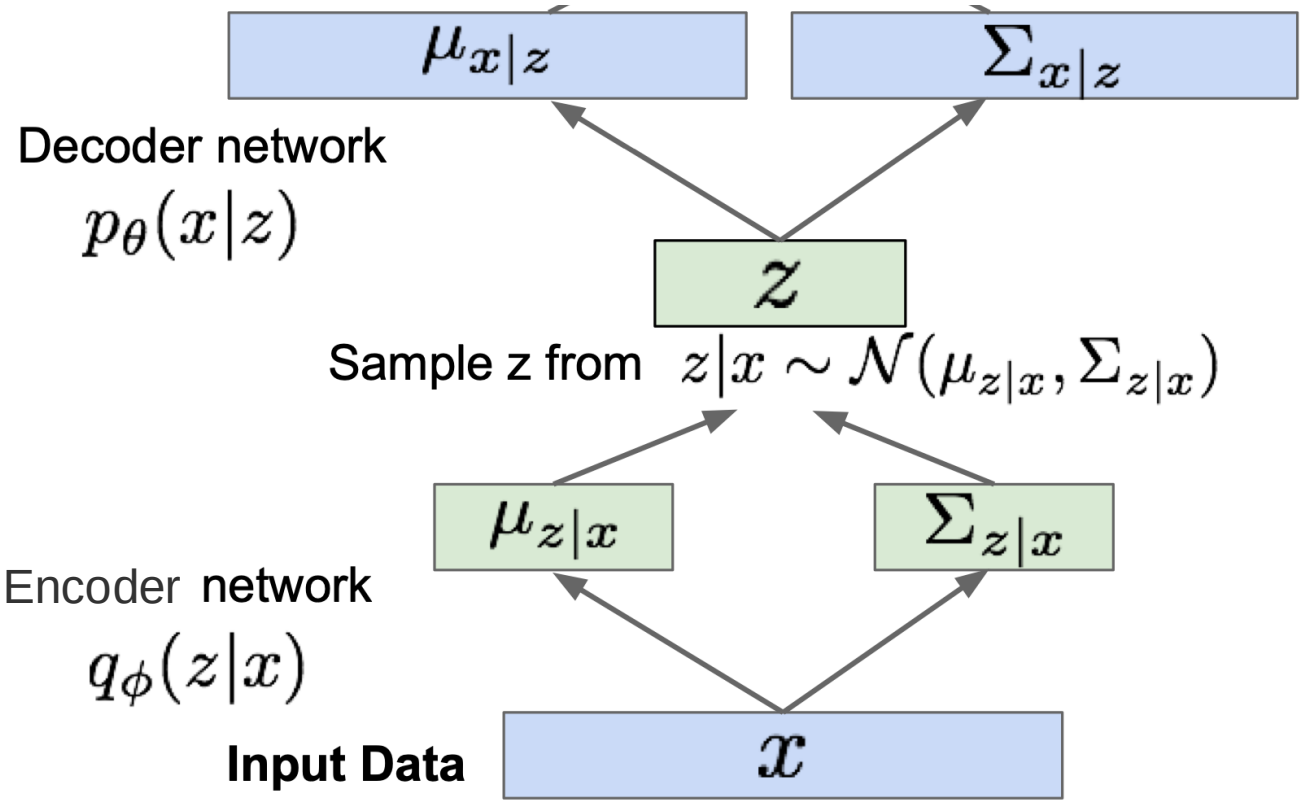




[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

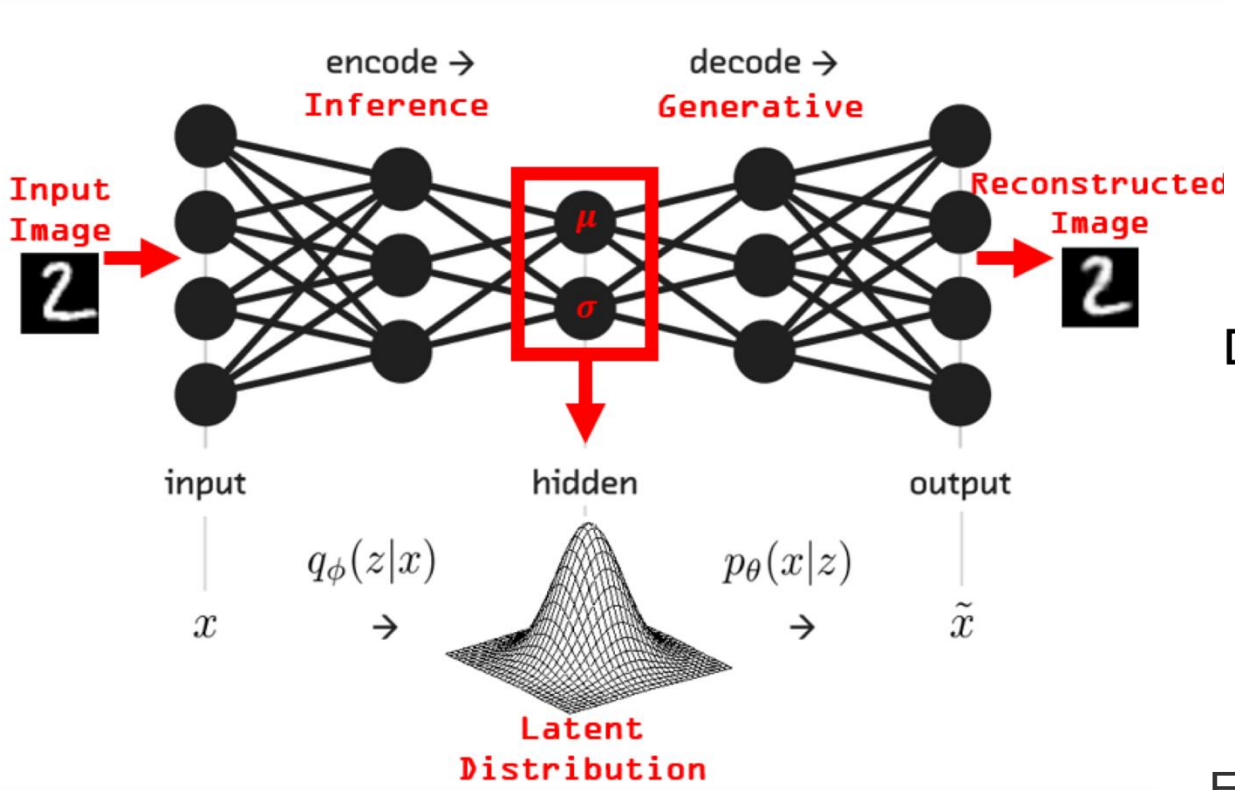
Example: VAEs for images

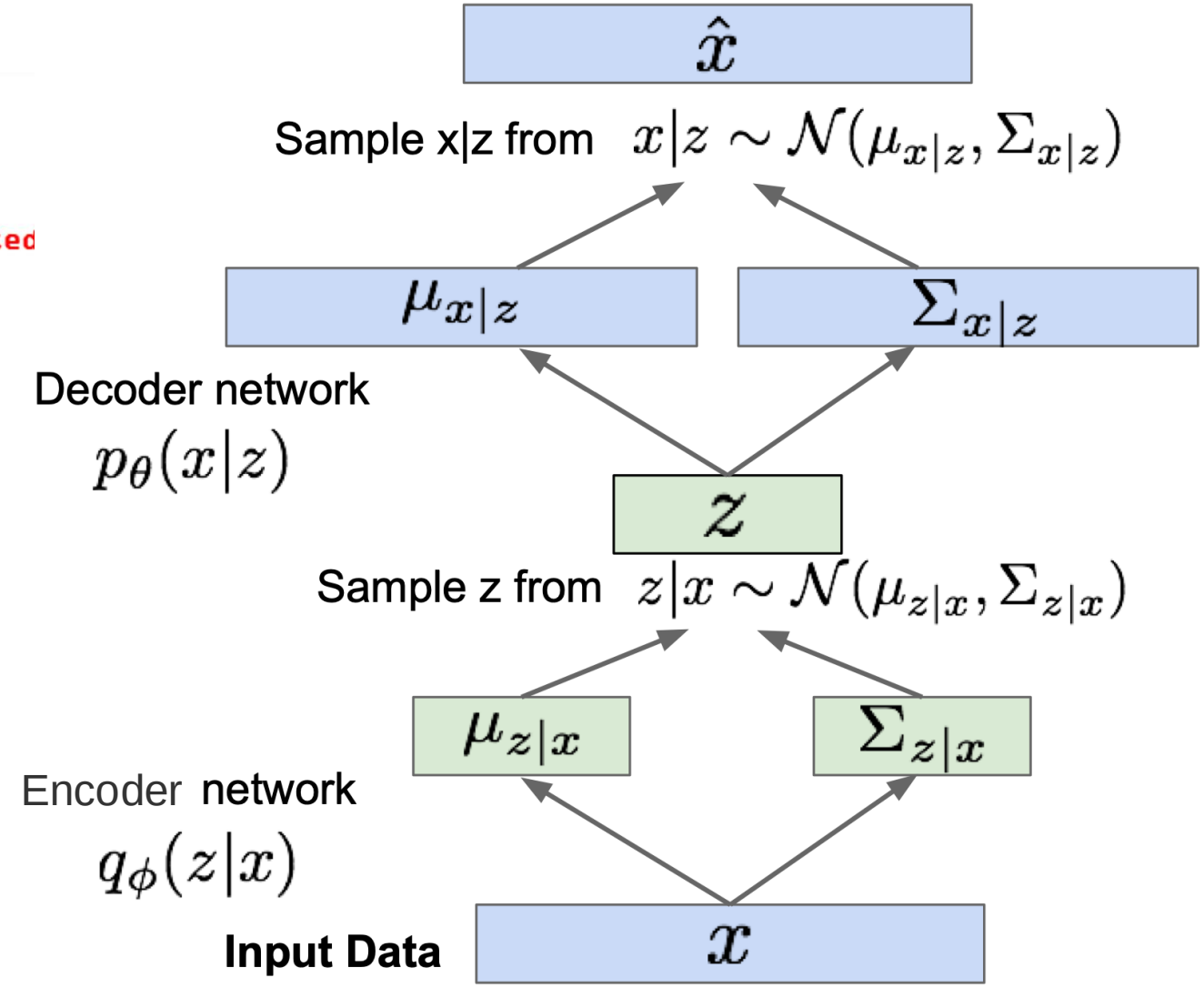




[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

Example: VAEs for images



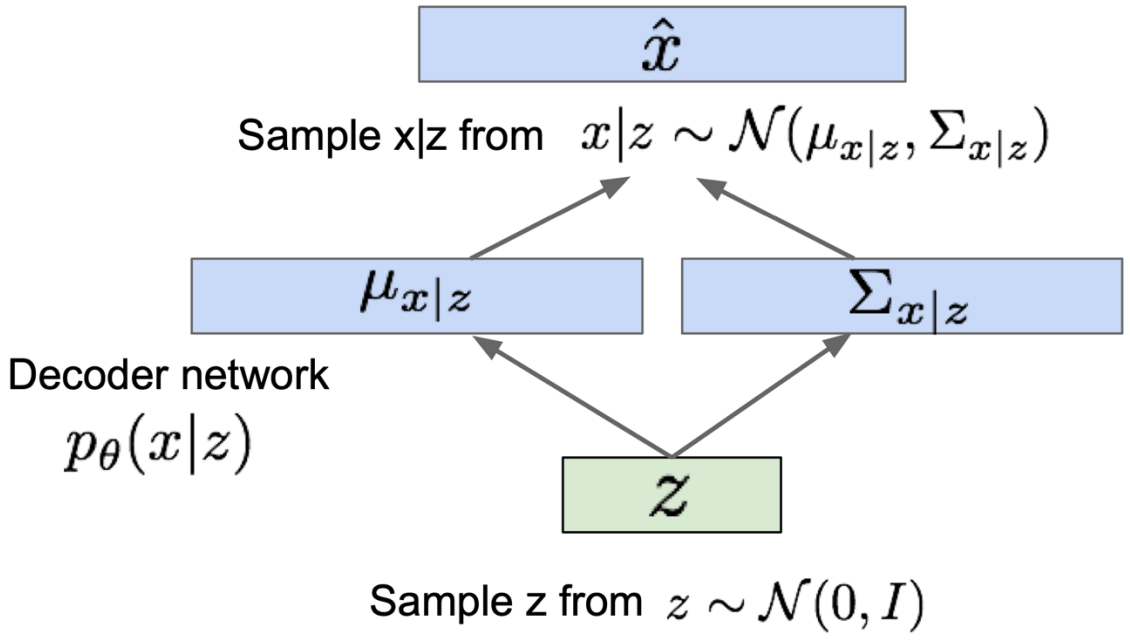


[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

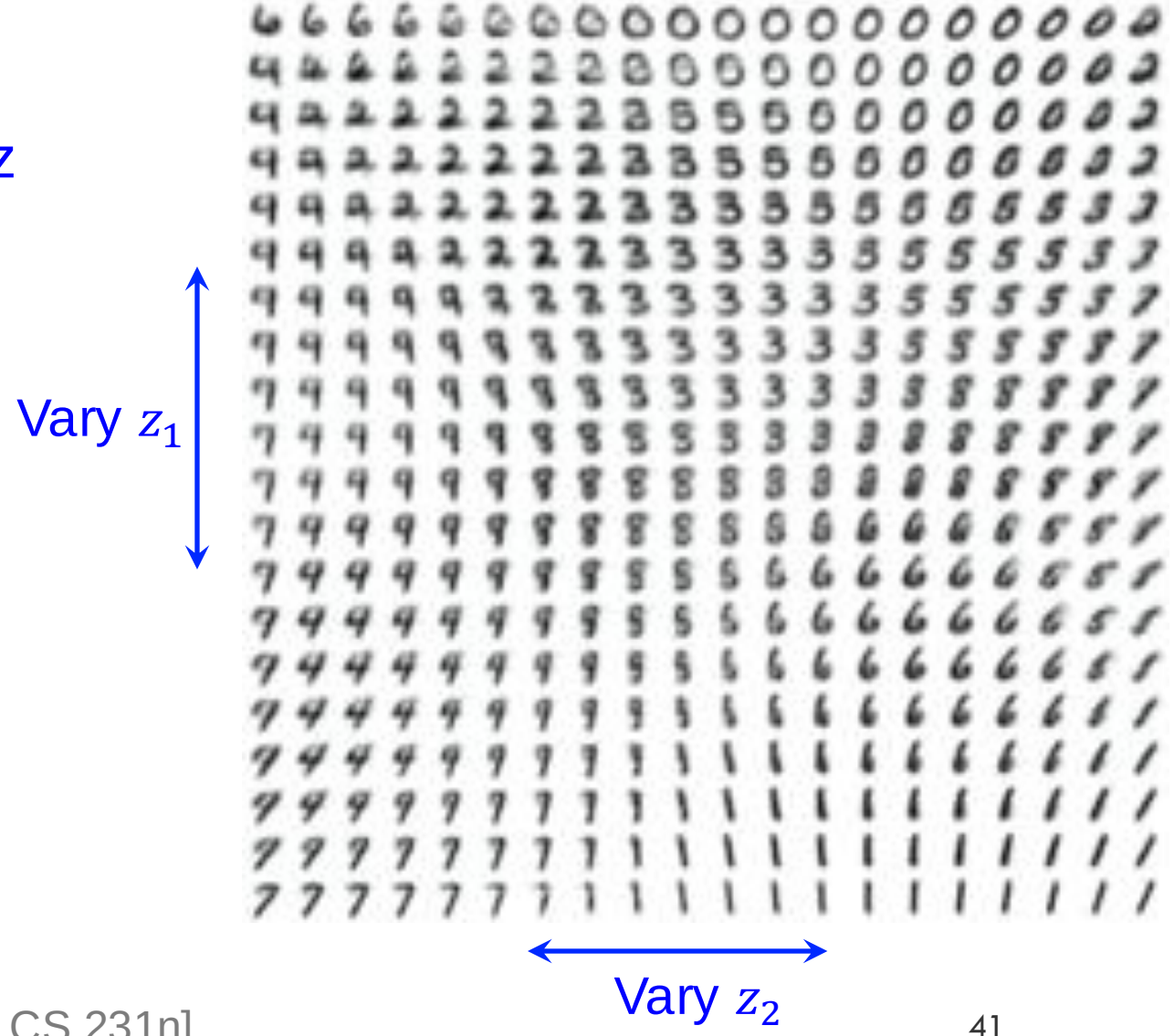
Example: VAEs for images

Generating samples:

- Use decoder network. Now sample z from prior!



Data manifold for 2-d **z**

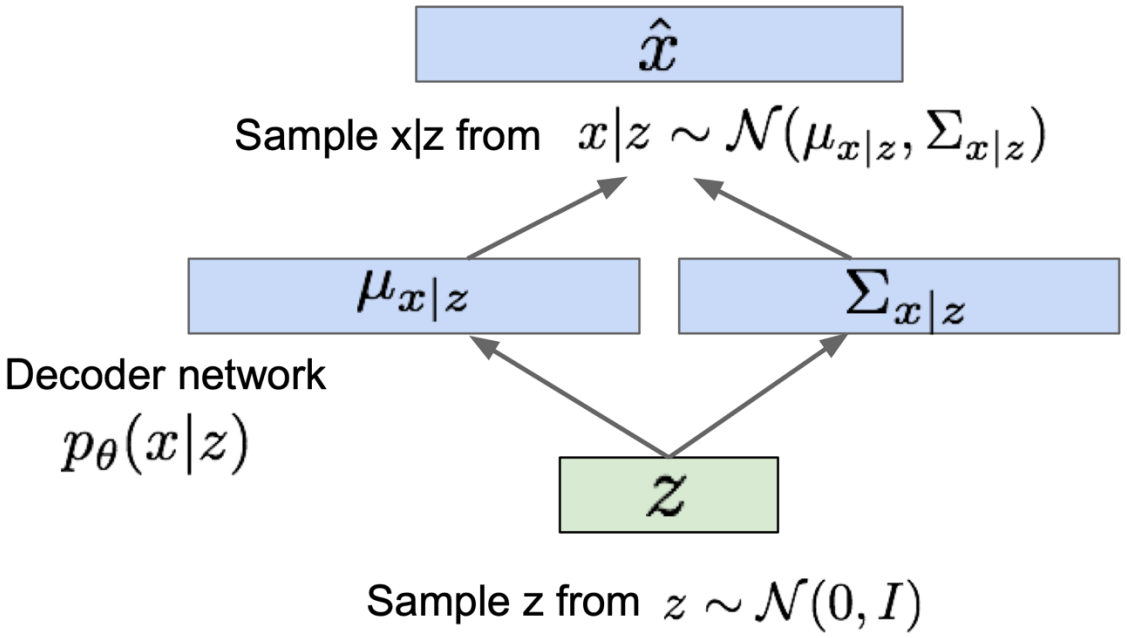


[Courtesy: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n]

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 .

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

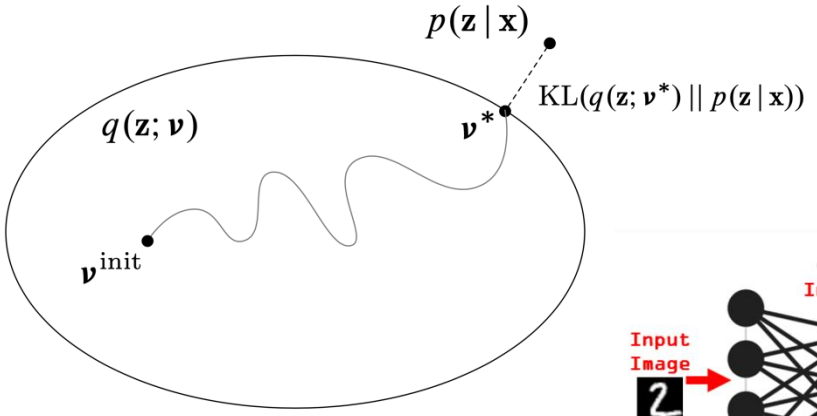
(Razavi et al., 2019)

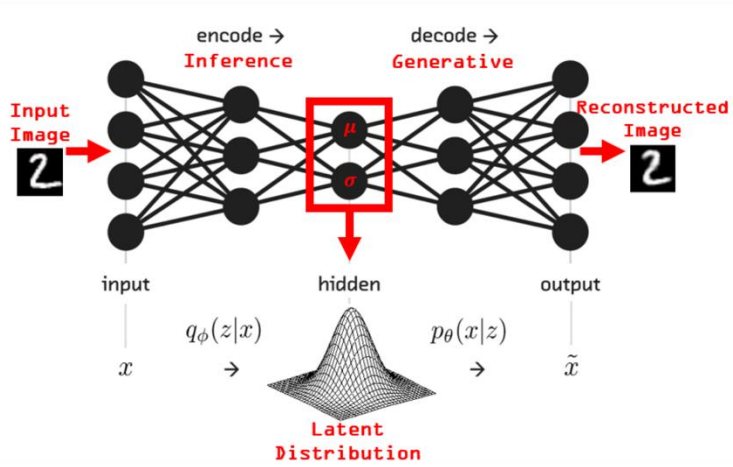
- Pros:
  - Principled approach to generative models
  - Allows inference of $q(z|x)$, can be useful feature representation for other tasks
- Cons:
  - Samples blurrier and lower quality compared to GANs
  - Tend to collapse on text data

Summary: Supervised / Unsupervised Learning

- Supervised Learning
  - Maximum likelihood estimation (MLE)

- Unsupervised learning
  - Maximum likelihood estimation (MLE) with latent variables
    - Marginal log-likelihood
  - EM algorithm for MLE
    - ELBO / Variational free energy
  - Variational Inference
    - ELBO / Variational free energy
    - Variational distributions
      - Factorized (mean-field VI)
      - Mixture of Gaussians (Black-box VI)
      - Neural-based (VAEs)





Presentations

Questions?