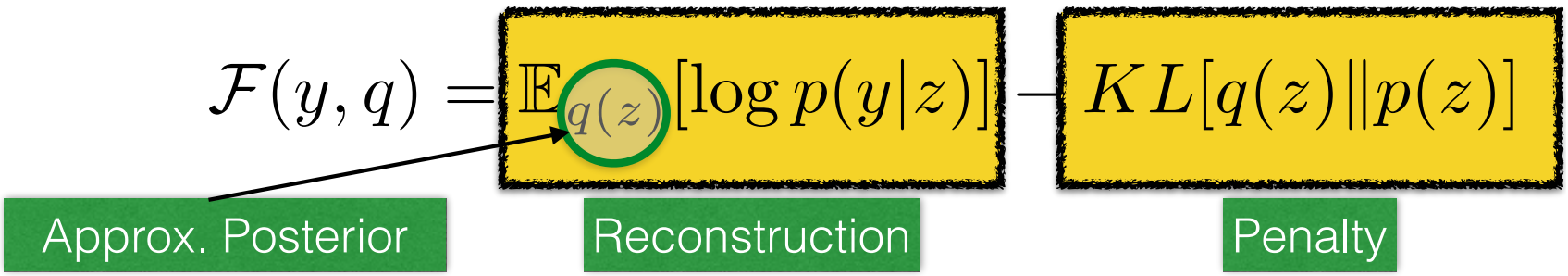
Deep Generative Models

Changyou Chen

Department of Computer Science and Engineering
Universitpy at Buffalo, SUNY
changyou@buffalo.edu

April 4, 2019

Variational Inference



Some comments on q

- Integration is now optimization: optimize for $q(z)$ directly:

  z typically depends on data y, but write $q(z)$ for notation simplicity. Easy convergence assessment.

- Variational parameters: parameters of $q(z)$:

  e.g., if a Gaussian, the parameters are mean and variance. Optimization allows us to tighten the bound and get as close as possible to the true marginal likelihood.

Free-form and Fixed-form Solutions

Free-form

- Solves for the exact distribution $q(z)$ by setting the functional derivative to zero via calculus of variations:

$$\frac{\delta \mathcal{F}(y, q)}{\delta q(z)} = 0, \quad s.t. \quad \int q(z) \mathrm{d}z = 1$$
$$\Rightarrow q(z) \propto p(z)p(y|z, \theta)$$

- The optimal solution is the true posterior distribution, but solving for the normalization is our original problem.

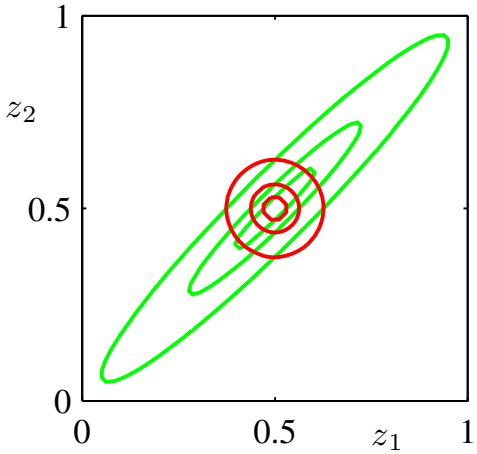
Fixed-form

- Specify an explicit form of $q(z)$, *e.g.*, a normal distribution.
- Parameter in q is called the variational parameter.

Mean-field Variational Inference

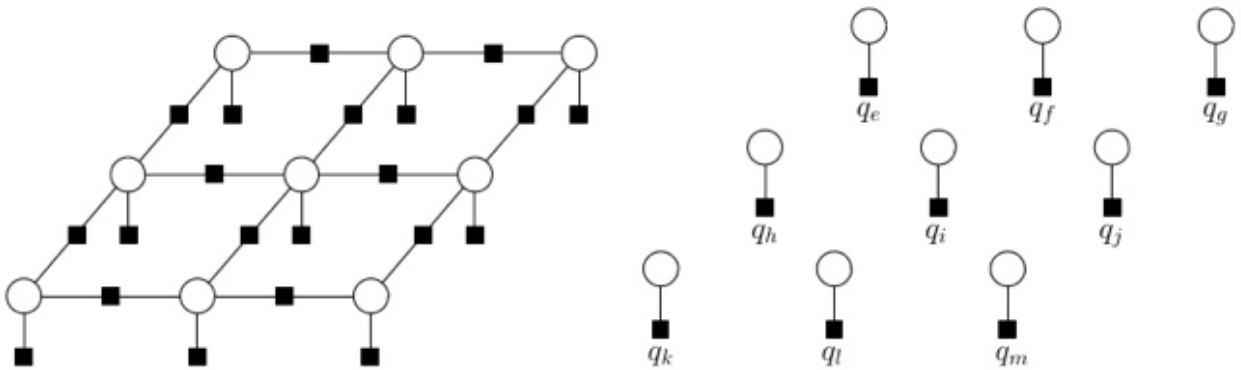
Mean-field methods assume that the distribution is factorised.

$$q(z) = \prod_i q_i(z_i)$$



$$q(z) = \prod_i \mathcal{N}(z_i|\mu_i, \sigma_i^2)$$

Restricted class of approximations: every dimension (or subset of dimensions) of the posterior is independent.



Example: Mean-field for Latent Gaussian Models

Generative model:

$$\mathbf{z} \sim \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I}), \quad \mathbf{y} \sim p(\mathbf{y} \,|\, f_\theta(\mathbf{z}))$$

Variational distribution:

$$q(\mathbf{z}) = \prod_i \mathcal{N}(z_i; \mu_i, \sigma_i^2)$$

Example: Mean-field for Latent Gaussian Models

$$\begin{aligned}
\mathcal{F}(\mathbf{y}, q) &= \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{y} \mid \mathbf{z})\right] - KL\left[q(\mathbf{z}) \| p(\mathbf{z})\right] \\
&= \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{y} \mid \mathbf{z})\right] - \sum_i KL\left[q(z_i) \| p(z_i)\right] \\
&= \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{y} \mid \mathbf{z})\right] - \sum_i KL\left[\mathcal{N}(z_i; \mu_i, \sigma_i^2) \| \mathcal{N}(z_i; 0, 1)\right] \\
&= \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{y} \mid \mathbf{z})\right] - \frac{1}{2}\sum_i \left(\sigma_i^2 + \mu_i^2 - 1 - \log \sigma_i^2\right)
\end{aligned}$$

KL between two Gaussians

Let $p(x) = \mathcal{N}(x; \mu_1, \sigma_1^2)$, $q(x) = \mathcal{N}(x; \mu_2, \sigma_2^2)$, then

$$KL(p(x) \| q(x)) = \log \frac{\sigma_2^2}{\sigma_1^2} + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}$$

Example: Mean-field for Latent Gaussian Models

$$
\begin{aligned}
\mathcal{F}(\mathbf{y}, q) &= \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{y} \mid \mathbf{z})\right] - KL\left[q(\mathbf{z}) \| p(\mathbf{z})\right] \\
&= \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{y} \mid \mathbf{z})\right] - \sum_i KL\left[q(z_i) \| p(z_i)\right] \\
&= \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{y} \mid \mathbf{z})\right] - \sum_i KL\left[\mathcal{N}(z_i; \mu_i, \sigma_i^2) \| \mathcal{N}(z_i; 0, 1)\right] \\
&= \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{y} \mid \mathbf{z})\right] - \frac{1}{2}\sum_i \left(\sigma_i^2 + \mu_i^2 - 1 - \log \sigma_i^2\right)
\end{aligned}
$$

Optimize parameters of $q(z)$ by gradient descent.

Optimization for the Variational Bound

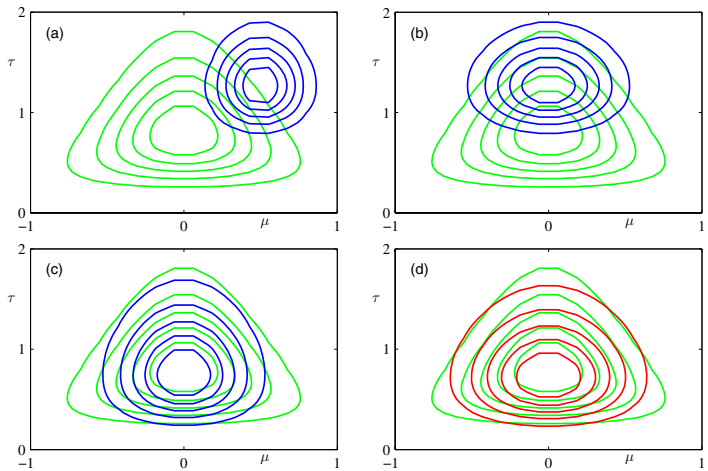
$$\max_{q,\theta} \mathcal{F}(y, q) = \mathbb{E}_{q(z)}[\log p(y|z, \theta)] - KL[q(z)||p(z)]$$

Approx. Posterior

Reconstruction

Penalty

- *Variational EM*
- *Stochastic Variational Inference*
- *Doubly Stochastic Variational Inference*
- *Amortised Inference*



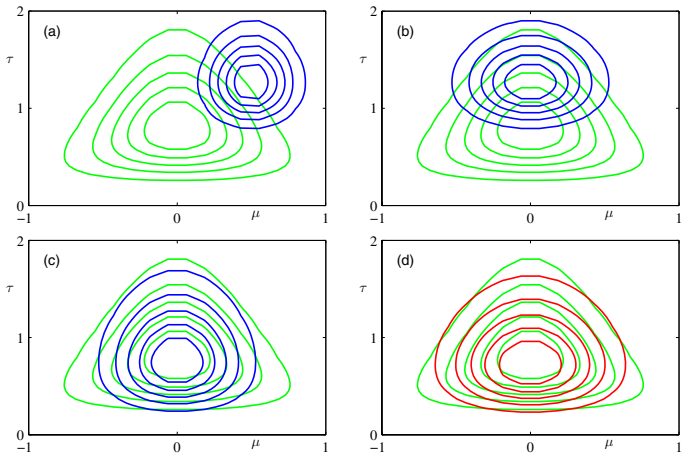
Optimization for the Variational Bound

$$\max_{q,\theta} \mathcal{F}(y, q) = \mathbb{E}_{q(z)}[\log p(y|z, \theta)] - KL[q(z)\|p(z)]$$

Approx. Posterior | Reconstruction | Penalty

- *Variational EM*

- *Stochastic Variational Inference*
- *Doubly Stochastic Variational Inference*
- *Amortised Inference*



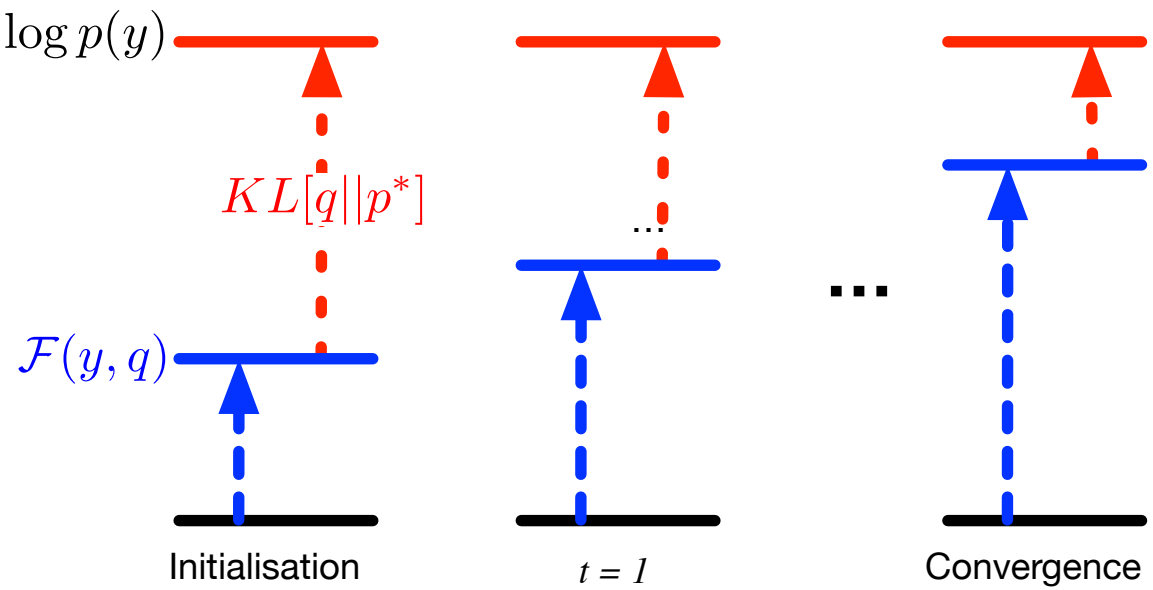
Dealt with big data and complicated $q(z)$

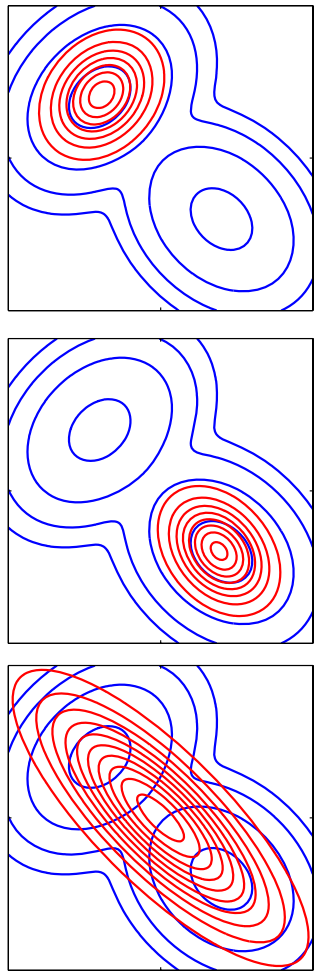
Variational Expectation Maximization

Alternating optimization for the variational parameters and then model parameters.

Repeat:

E-step	$\phi \propto \nabla_\phi \mathcal{F}(y, q)$	*Var. params*
M-step	$\theta \propto \nabla_\theta \mathcal{F}(y, q)$	*Model params*





Variational Expectation Maximization

Alternating optimization for the variational parameters and then model parameters.

Repeat:

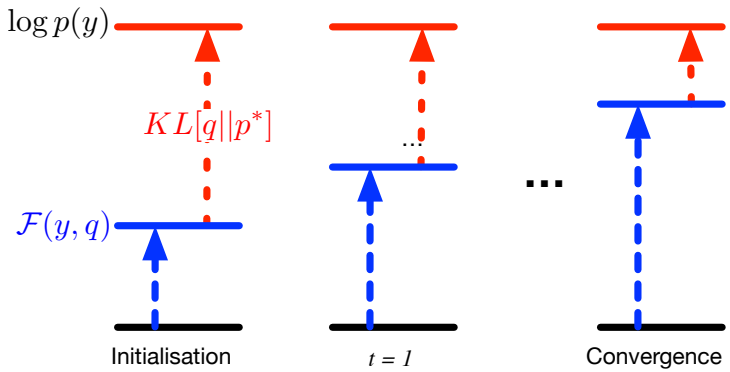
E-step *(Inference)*

For i = 1, ... N

$$\phi_n \propto \nabla_\phi \mathbb{E}_{q_\phi(z)}[\log p_\theta(y_n|z_n)] - \nabla_\phi KL[q(z_n)||p(z_n)]$$

M-step *(Parameter Learning)*

$$\theta \propto \frac{1}{N}\sum_n \mathbb{E}_{q_\phi(z)}[\nabla_\theta \log p_\theta(y_n|z_n)]$$

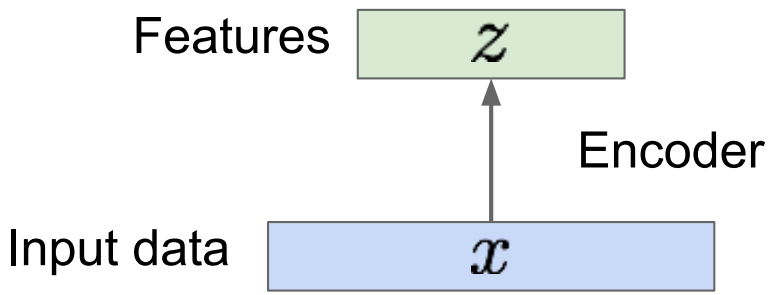


Autoencoder[1]

[1] Partially adapted from http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture13.pdf

Background: Autoencoder

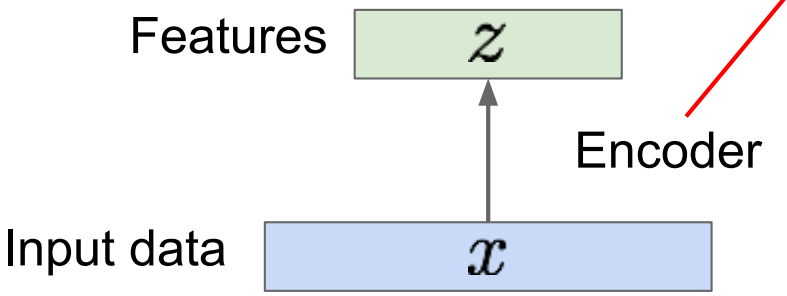
Learning a lower-dimensional feature representation from unlabeled training data



Background: Autoencoder

Learning a lower-dimensional feature representation from unlabeled training data

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



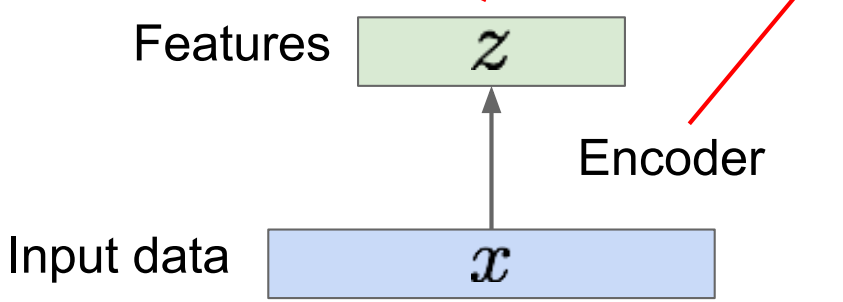
Background: Autoencoder

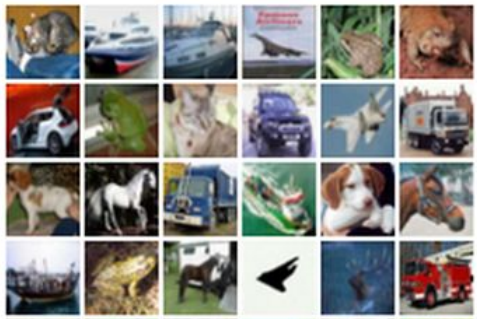
Learning a lower-dimensional feature representation from unlabeled training data

z usually smaller than **x** (dimensionality reduction)

Q: Why dimensionality reduction?

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





Background: Autoencoder

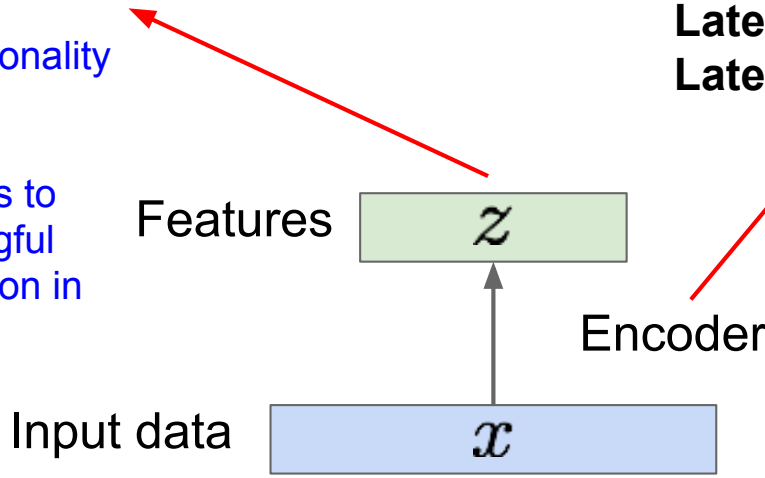
Learning a lower-dimensional feature representation from unlabeled training data

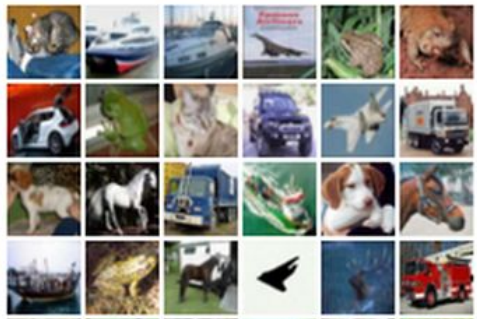
z usually smaller than **x** (dimensionality reduction)

Q: Why dimensionality reduction?

A: Want features to capture meaningful factors of variation in data

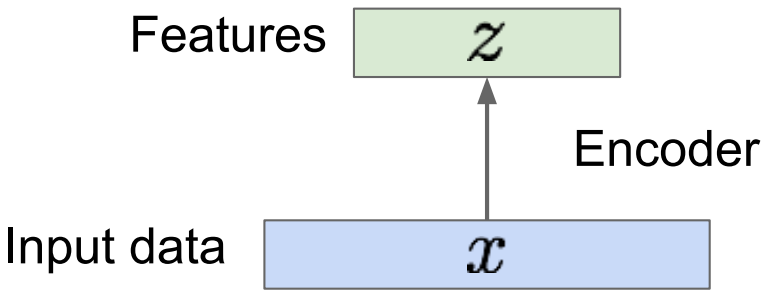
Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN

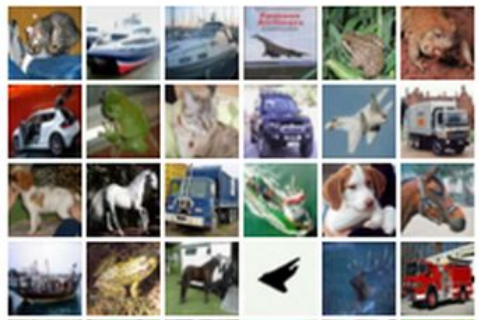




Background: Autoencoder

How to learning this feature representation?



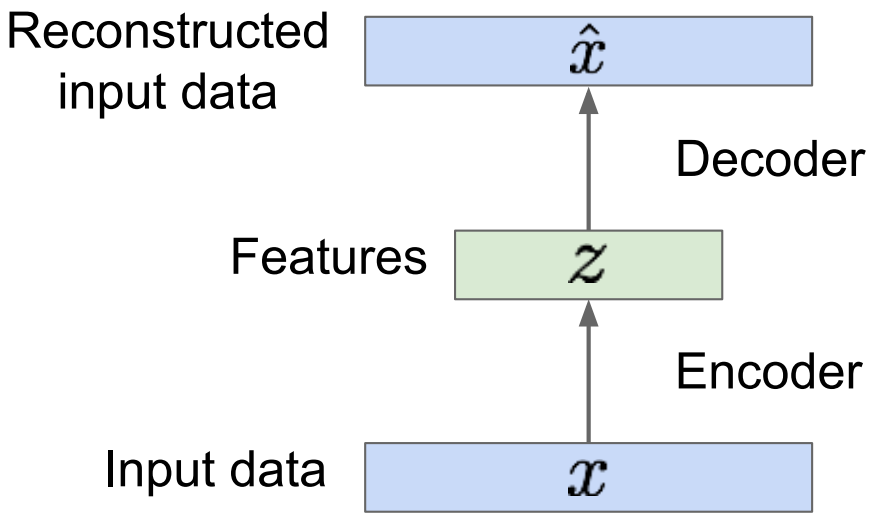


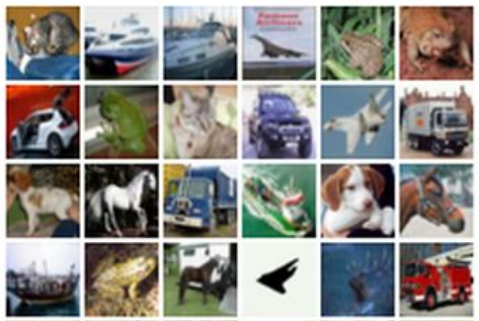
Background: Autoencoder

How to learning this feature representation?

We want that feature can be used to reconstruct itself:

- "Autoencoding" – encoding itself.



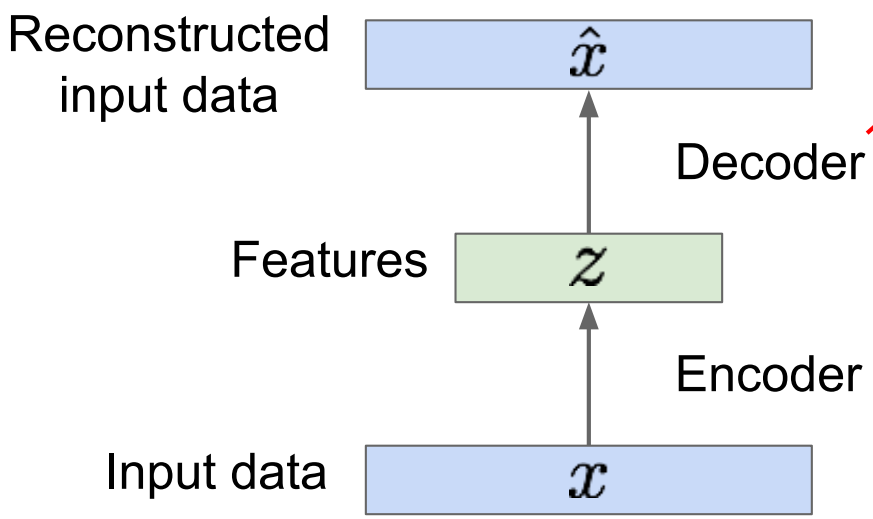


Background: Autoencoder

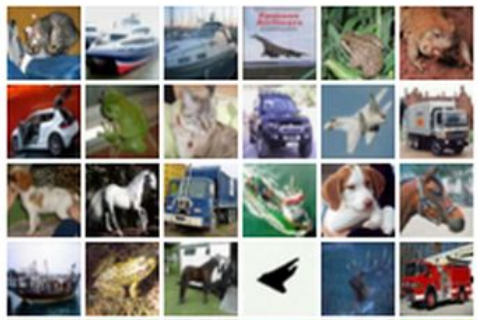
How to learning this feature representation?

We want that feature can be used to reconstruct itself:

- "Autoencoding" – encoding itself.

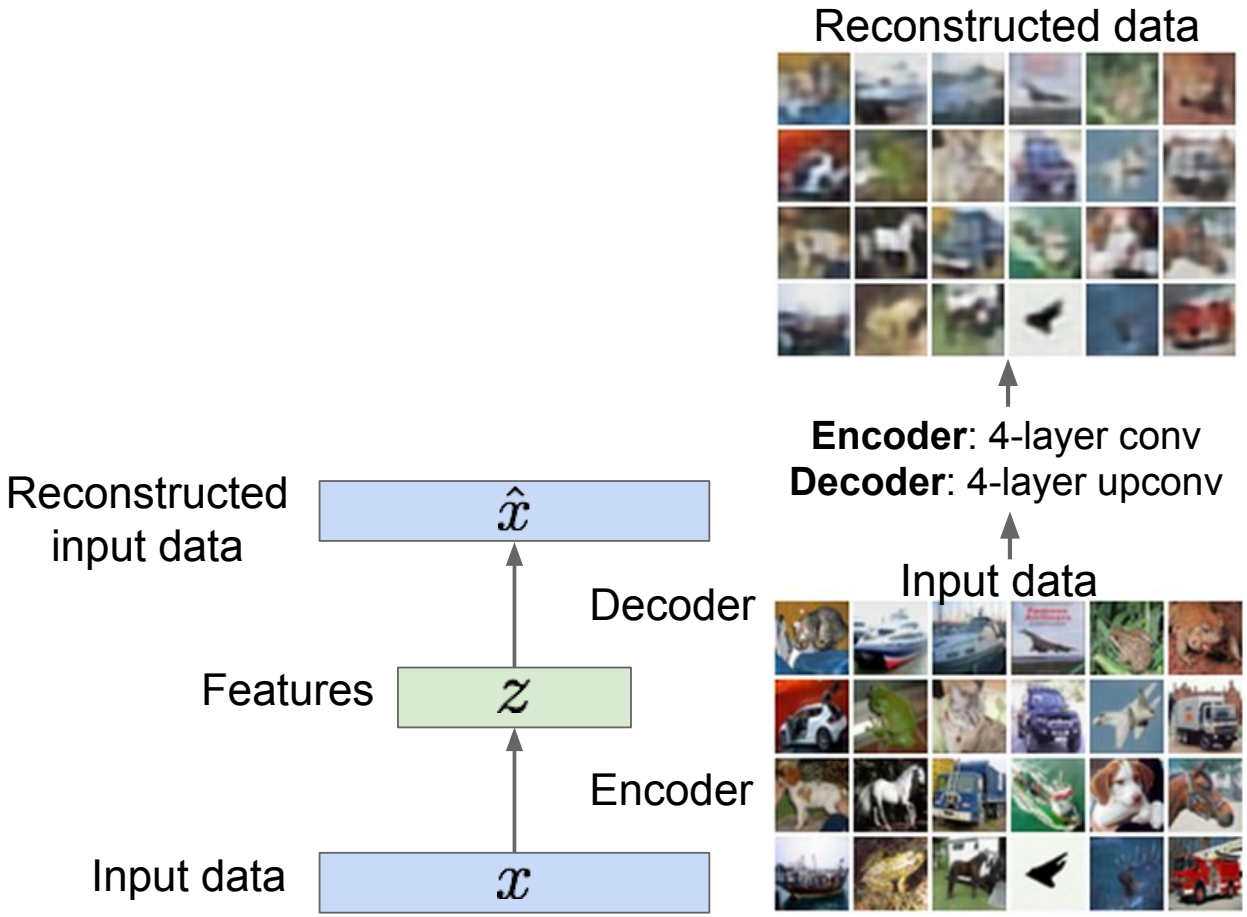


Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN (upconv)



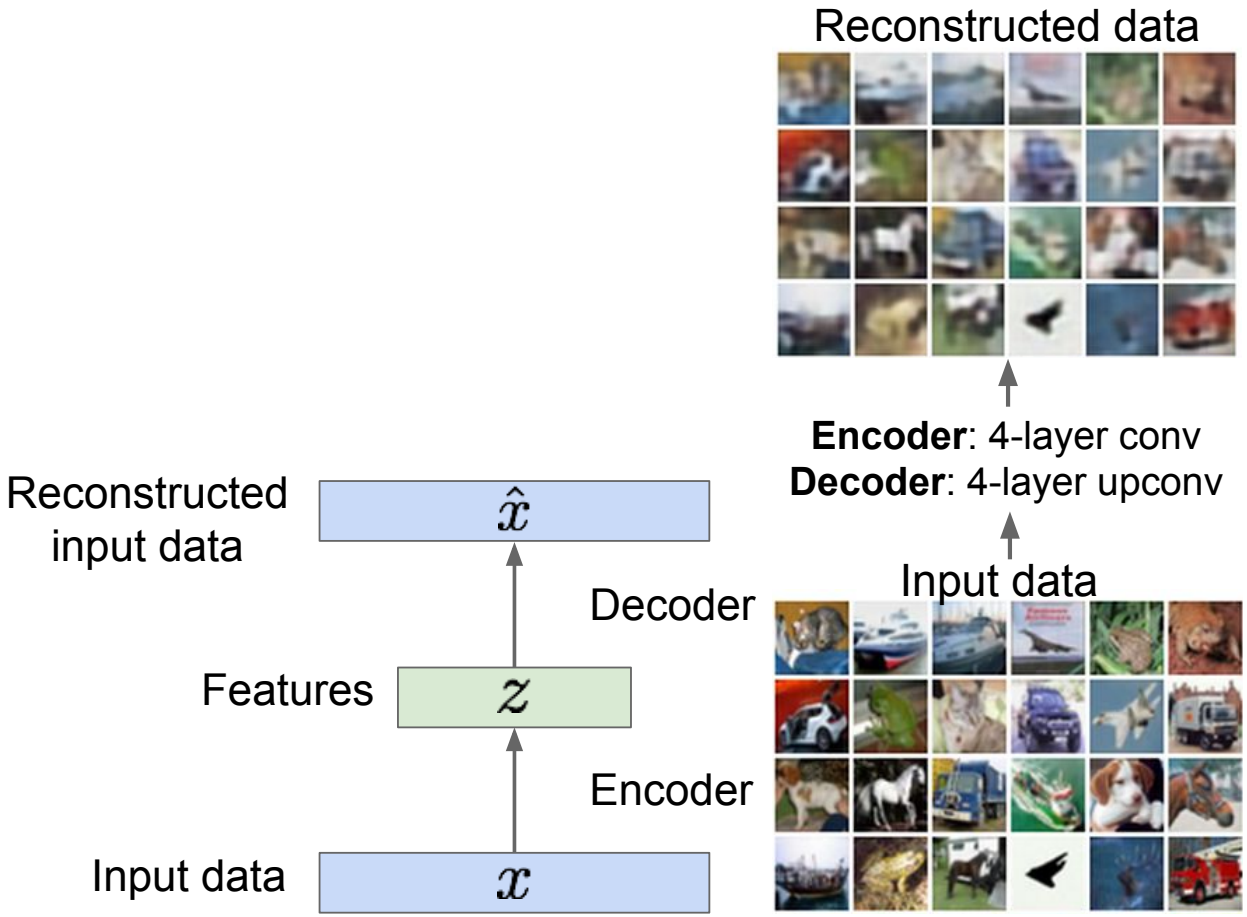
Background: Autoencoder

- We want that feature can be used to reconstruct itself
- Use L^2 loss: $\|\mathbf{x} - \hat{\mathbf{x}}\|^2$
- Doesn't use labels



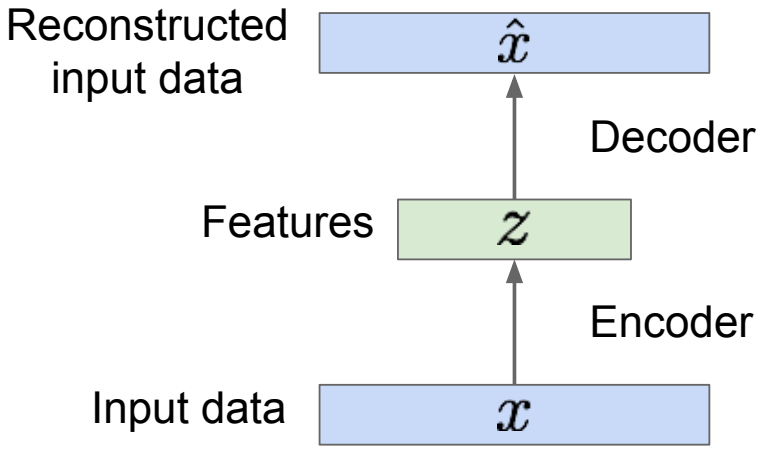
Background: Autoencoder

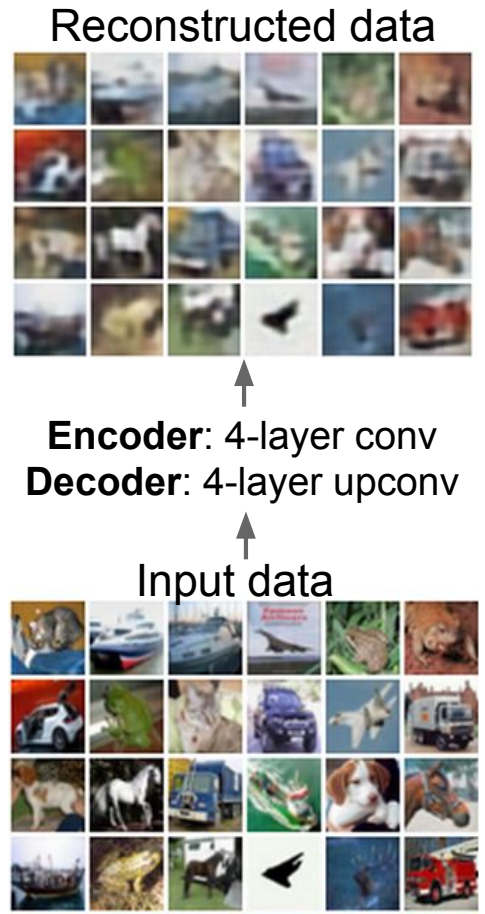
- We want that feature can be used to reconstruct itself
- Use L^2 loss: $\| \mathbf{x} - \hat{\mathbf{x}} \|^2$
- Doesn't use labels



Background: Autoencoder

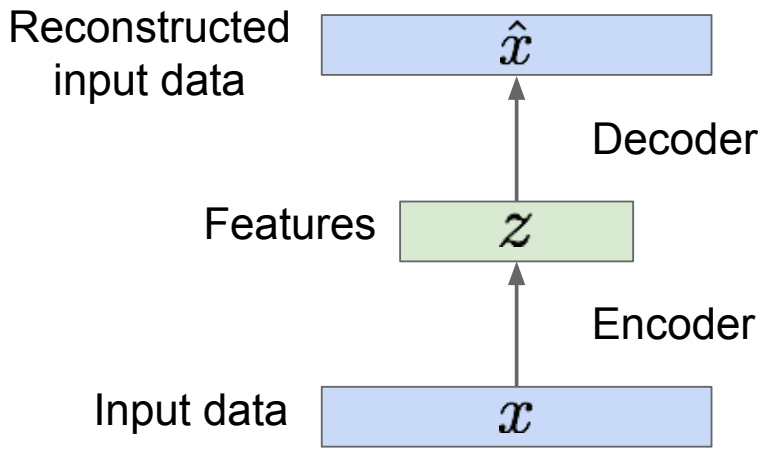
- We want that feature can be used to reconstruct itself
- Use L^2 loss: $\| \mathbf{x} - \hat{\mathbf{x}} \|^2$
- Doesn't use labels





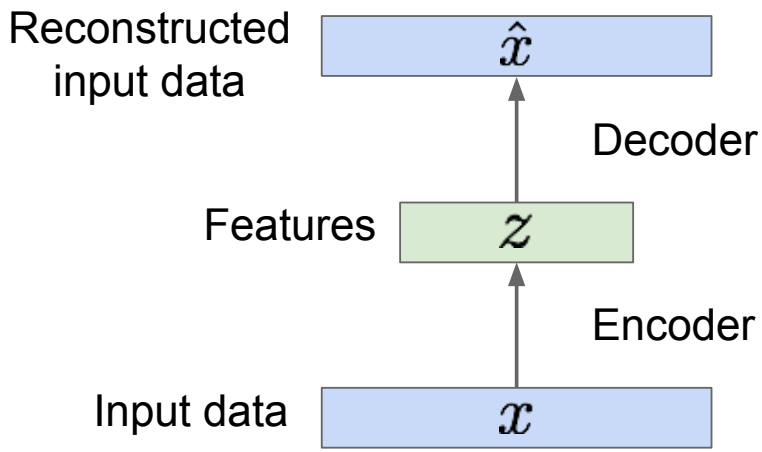
Background: Autoencoder

- What can an autoencoder do?
  - Reconstruct data.
  - Pretrain for supervised learning.



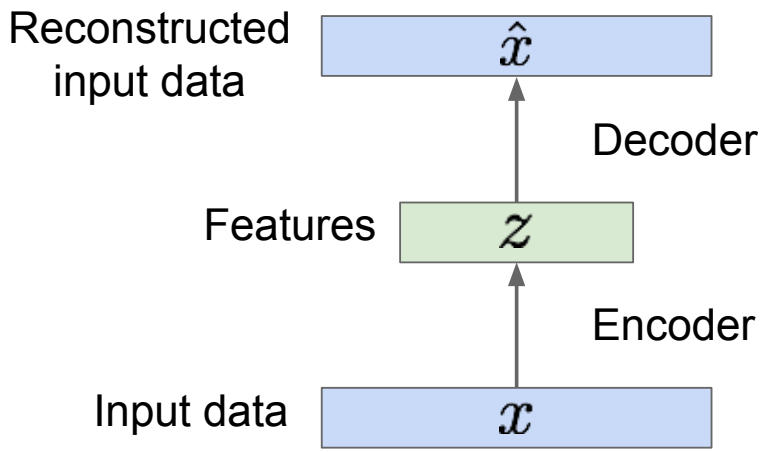
Background: Autoencoder

- What can an autoencoder do?
  - Reconstruct data.
  - Pretrain for supervised learning.



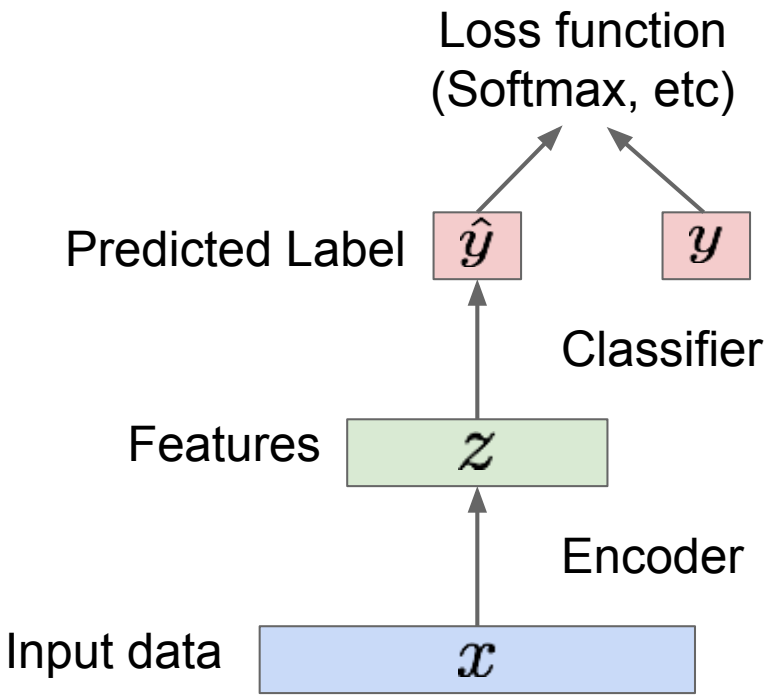
Background: Autoencoder

- What can an autoencoder do?
  - Reconstruct data.
  - Pretrain for supervised learning.



Background: Autoencoder

After training, throw away decoder.



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

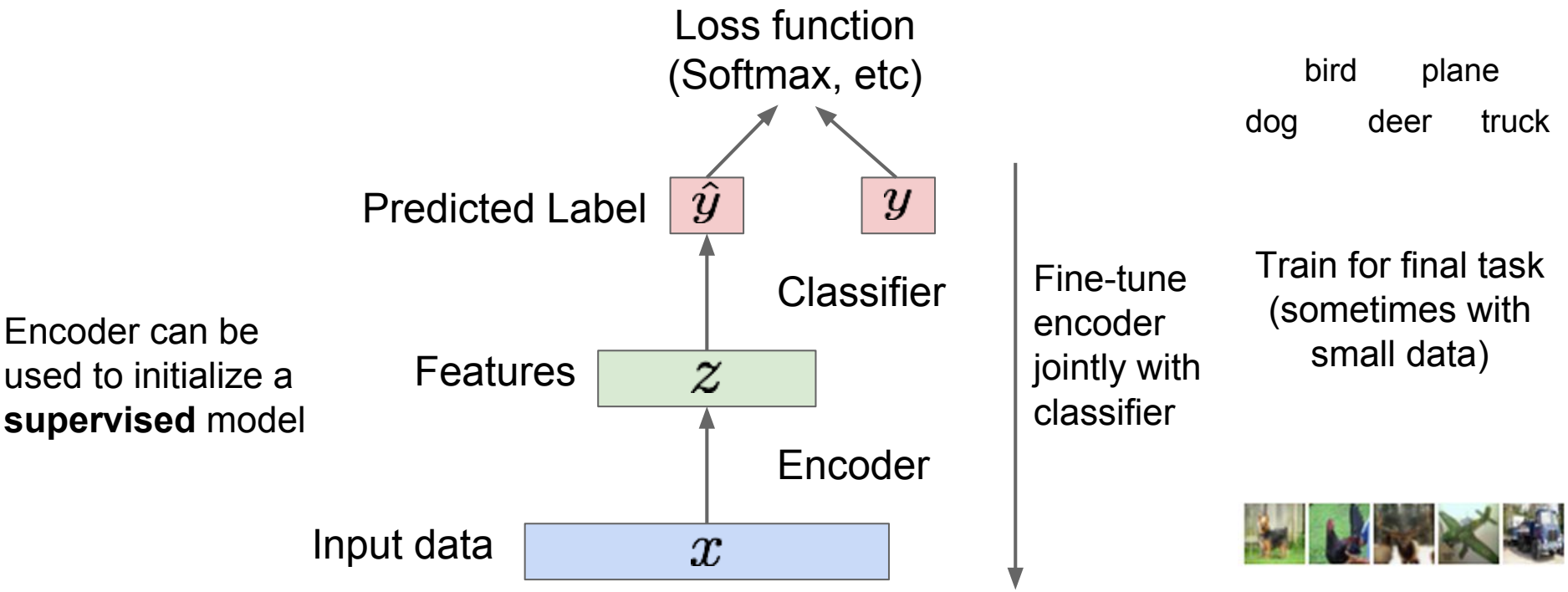
Fine-tune encoder jointly with classifier

bird plane
dog deer truck

Train for final task (sometimes with small data)

Background: Autoencoder

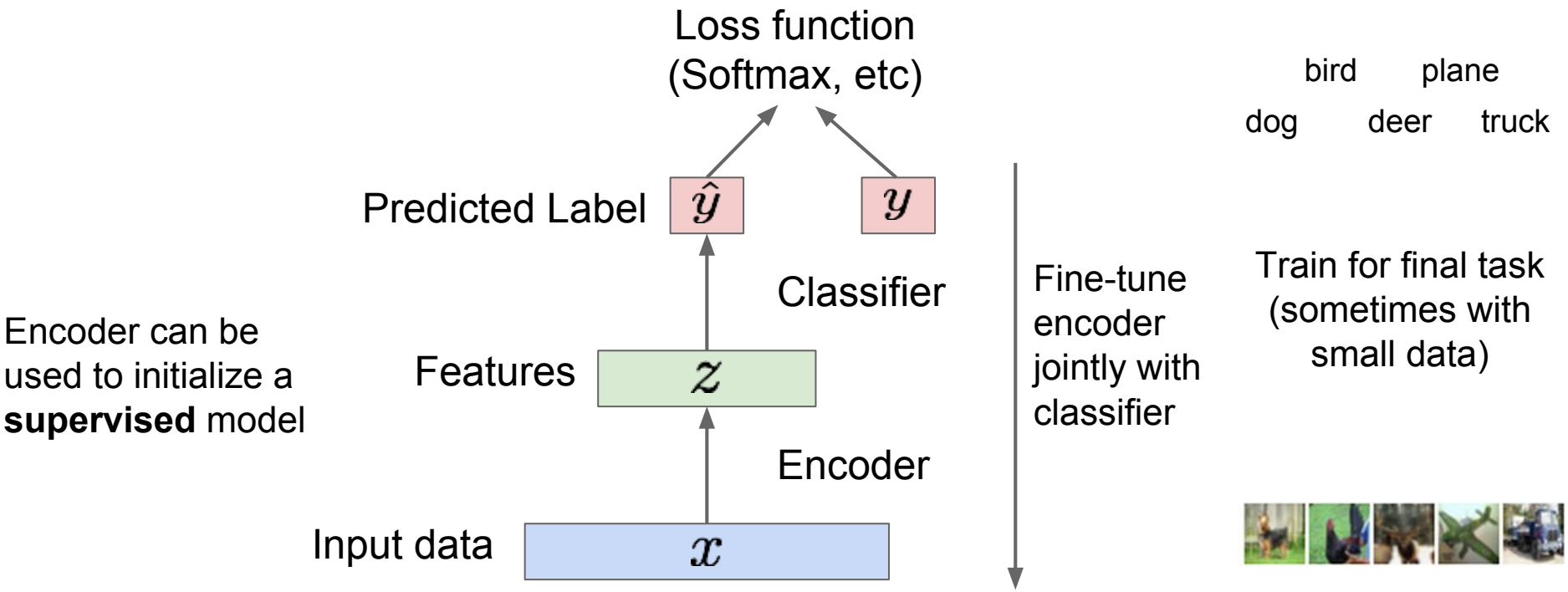
After training, throw away decoder.



- Autoencoders can reconstruct data, and can learn features to initialize a supervised model.
- However, can we generate new images from it?
  - NO

Background: Autoencoder

After training, throw away decoder.



- Autoencoders can reconstruct data, and can learn features to initialize a supervised model.
- However, can we generate new images from it?
  - NO

Variational Autoencoder

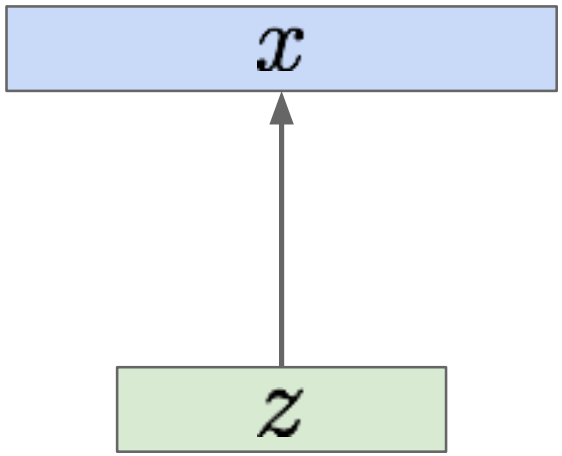
Variational Autoencoder $\Rightarrow$ Probabilistic Autoencoder

1. Define a generative probability model for the deconder.
2. Assume training data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation $\mathbf{z}$.

Kingma & Welling, ICLR 2014

Variational Autoencoder $\Rightarrow$ Probabilistic Autoencoder

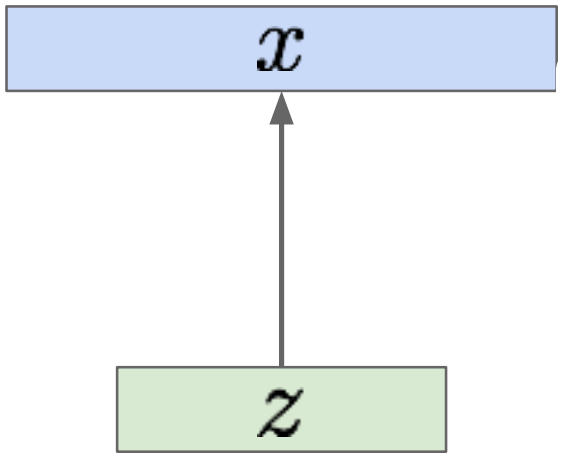
1. Define a generative probability model for the deconder.
2. Assume training data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation $\mathbf{z}$.
   - Sample latent $\mathbf{z}$ from true prior $p_{\theta^*}(\mathbf{z})$
   - Sample data $\mathbf{x}$ from true conditional $p_{\theta^*}(\mathbf{x} \mid \mathbf{z})$



Kingma & Welling, ICLR 2014

Variational Autoencoder $\Rightarrow$ Probabilistic Autoencoder

1. Define a generative probability model for the deconder.
2. Assume training data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation $\mathbf{z}$.
   - Sample latent $\mathbf{z}$ from true prior $p_{\theta^*}(\mathbf{z})$
   - Sample data $\mathbf{x}$ from true conditional $p_{\theta^*}(\mathbf{x} \mid \mathbf{z})$



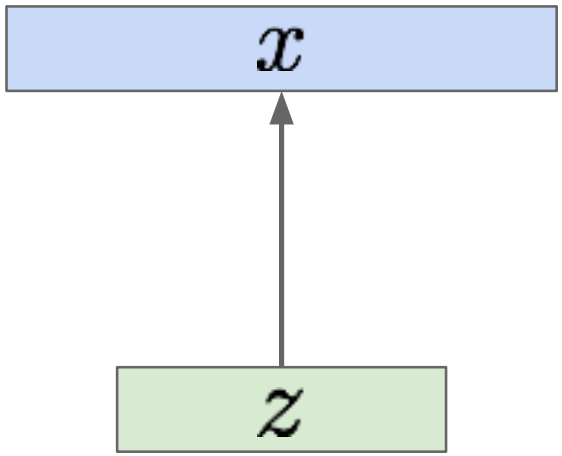
Intuition

$\mathbf{x}$ is an image, $\mathbf{z}$ is latent factors used to generate $\mathbf{x}$, *e.g.*, attributes, orientation, *etc*

Kingma & Welling, ICLR 2014

Variational Autoencoder $\Rightarrow$ Probabilistic Autoencoder

1. Define a generative probability model for the deconder.
2. Assume training data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation $\mathbf{z}$.
   - Sample latent $\mathbf{z}$ from true prior $p_{\theta^*}(\mathbf{z})$
   - Sample data $\mathbf{x}$ from true conditional $p_{\theta^*}(\mathbf{x} \mid \mathbf{z})$

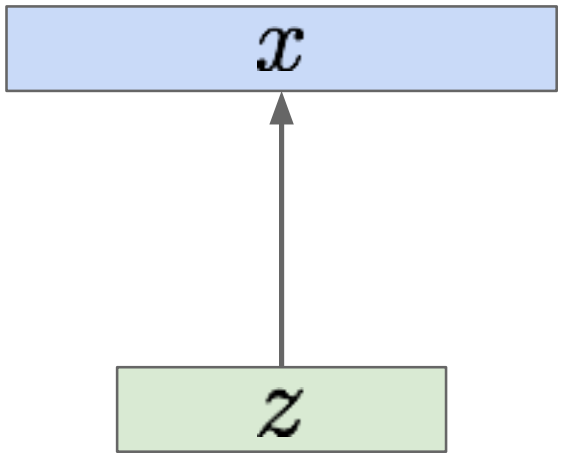


We want to estimate the true parameters θ^* of this generative model.

Kingma & Welling, ICLR 2014

Variational Autoencoder $\Rightarrow$ Probabilistic Autoencoder

1. Define a generative probability model for the deconder.
2. Assume training data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation $\mathbf{z}$.
   - Sample latent $\mathbf{z}$ from true prior $p_{\theta^*}(\mathbf{z})$
   - Sample data $\mathbf{x}$ from true conditional $p_{\theta^*}(\mathbf{x} \mid \mathbf{z})$

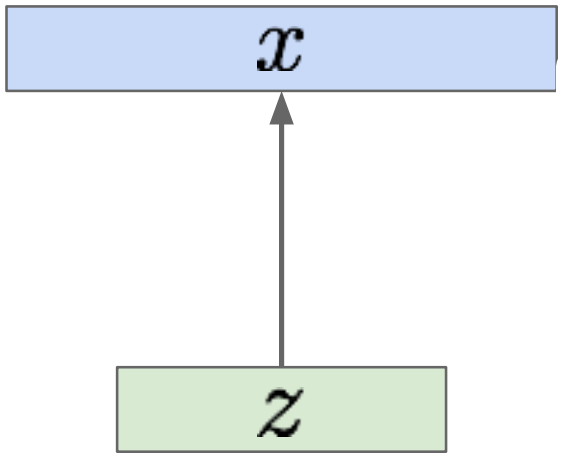


How should we represent this model?

Kingma & Welling, ICLR 2014

Variational Autoencoder $\Rightarrow$ Probabilistic Autoencoder

1. Define a generative probability model for the deconder.
2. Assume training data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation $\mathbf{z}$.
   - Sample latent $\mathbf{z}$ from true prior $p_{\theta^*}(\mathbf{z})$
   - Sample data $\mathbf{x}$ from true conditional $p_{\theta^*}(\mathbf{x} \mid \mathbf{z})$



How should we represent this model?

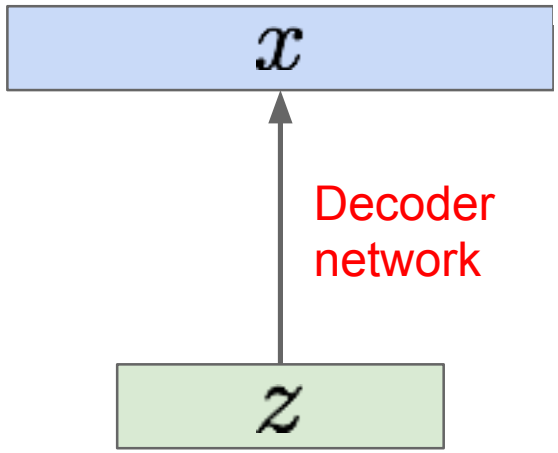
- Choose prior p(z) to be simple, *e.g.* Gaussian.

Kingma & Welling, ICLR 2014

Variational Autoencoder $\Rightarrow$ Probabilistic Autoencoder

1. Define a generative probability model for the deconder.
2. Assume training data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation $\mathbf{z}$.

- Sample latent $\mathbf{z}$ from true prior $p_{\theta^*}(\mathbf{z})$
- Sample data $\mathbf{x}$ from true conditional $p_{\theta^*}(\mathbf{x} \mid \mathbf{z})$



How should we represent this model?

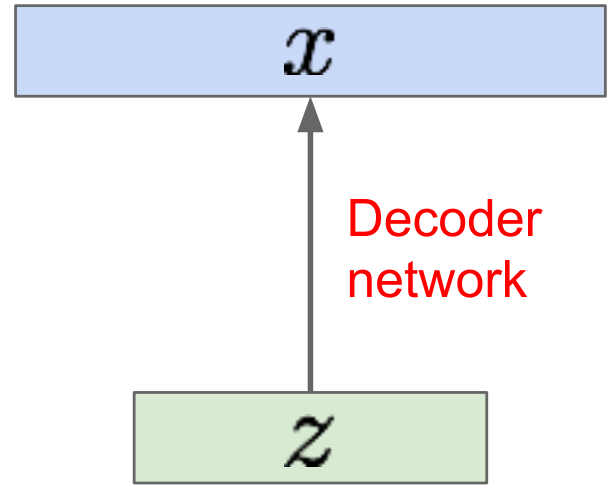
- Choose prior p(z) to be simple, *e.g.* Gaussian.
- Chose $p(\mathbf{x} \mid \mathbf{z})$ to be complex (generates image) $\rightarrow$ represented with a neural network.

Kingma & Welling, ICLR 2014

Variational Autoencoder $\Rightarrow$ Probabilistic Autoencoder

1. Define a generative probability model for the deconder.
2. Assume training data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation $\mathbf{z}$.

- Sample latent $\mathbf{z}$ from true prior $p_{\theta^*}(\mathbf{z})$
- Sample data $\mathbf{x}$ from true conditional $p_{\theta^*}(\mathbf{x} \mid \mathbf{z})$



How to train the model?

- Maximum likelihood?

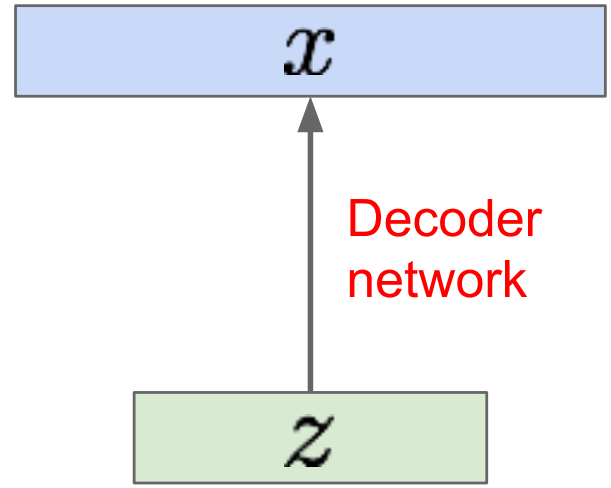
$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x} \mid \mathbf{z}) p_\theta(\mathbf{z}) \mathrm{d}\,\mathbf{z}$$

Kingma & Welling, ICLR 2014

Variational Autoencoder $\Rightarrow$ Probabilistic Autoencoder

1. Define a generative probability model for the deconder.
2. Assume training data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation $\mathbf{z}$.

- Sample latent $\mathbf{z}$ from true prior $p_{\theta^*}(\mathbf{z})$
- Sample data $\mathbf{x}$ from true conditional $p_{\theta^*}(\mathbf{x} \mid \mathbf{z})$



How to train the model?

- Maximum likelihood?

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x} \mid \mathbf{z}) p_\theta(\mathbf{z}) \mathrm{d}\,\mathbf{z}$$

Intractable $\Rightarrow$ Variational Inference! Kingma & Welling, ICLR 2014

Variational Autoencoder: Intractability

- Data likelihood: $p_{\theta}(\mathbf{x}) = \int p_{\theta}(\mathbf{x} \mid \mathbf{z}) p_{\theta}(\mathbf{z}) \mathrm{d}\,\mathbf{z}$

Variational Autoencoder: Intractability

- Data likelihood: $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x} \mid \mathbf{z}) \underbrace{p_\theta(\mathbf{z})}_{\text{simple Gaussian prior } \checkmark} \mathrm{d}\mathbf{z}$

Variational Autoencoder: Intractability

- Data likelihood: $p_\theta(\mathbf{x}) = \int \underbrace{p_\theta(\mathbf{x} \mid \mathbf{z})}_{\text{decoder neural network } \checkmark} p_\theta(\mathbf{z}) \mathrm{d}\mathbf{z}$
  - typically model as:

$$p_\theta(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}\left(\mathbf{x}; \mu_\theta(\mathbf{z}), \Sigma_\theta(\mathbf{z})\right) \ ,$$

where μ_θ and Σ_θ are two neural networks parameterized by θ.

Variational Autoencoder: Intractability

- Data likelihood: $p_\theta(\mathbf{x}) = \underbrace{\int}_{\text{intractible } \times} p_\theta(\mathbf{x} \mid \mathbf{z}) p_\theta(\mathbf{z}) \mathrm{d}\,\mathbf{z}$

Variational Autoencoder: Intractability

- Data likelihood: $p_\theta(\mathbf{x}) = \underbrace{\int}_{\text{intractible } \times} p_\theta(\mathbf{x} \mid \mathbf{z}) p_\theta(\mathbf{z}) \mathrm{d}\,\mathbf{z}$
- Posterior also intractable: $p_\theta(\mathbf{z} \mid \mathbf{x}) = \underbrace{p_\theta(\mathbf{x} \mid \mathbf{z})}_{\checkmark} \underbrace{p_\theta(\mathbf{z})}_{\checkmark} / \underbrace{p_\theta(\mathbf{x})}_{\times}$

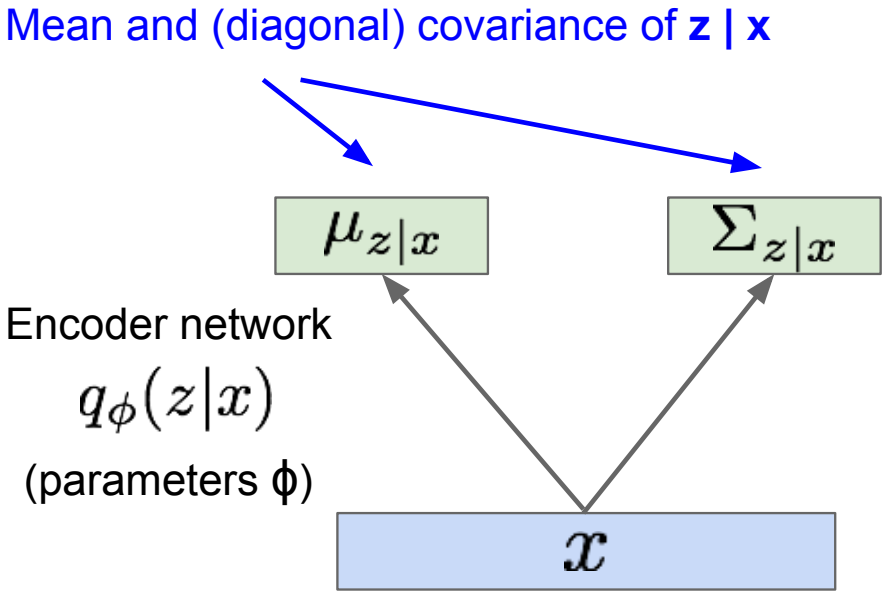
Variational Autoencoder: Intractability

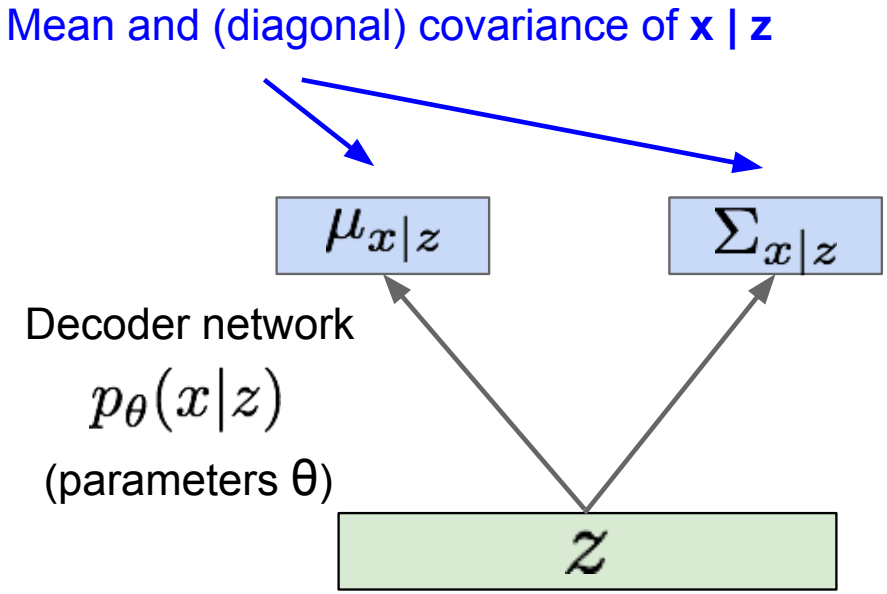
- Data likelihood: $p_\theta(\mathbf{x}) = \underbrace{\int}_{\text{intractible } \times} p_\theta(\mathbf{x} \mid \mathbf{z}) p_\theta(\mathbf{z}) \mathrm{d}\mathbf{z}$
- Posterior also intractable: $p_\theta(\mathbf{z} \mid \mathbf{x}) = \underbrace{p_\theta(\mathbf{x} \mid \mathbf{z})}_{\checkmark} \underbrace{p_\theta(\mathbf{z})}_{\checkmark} / \underbrace{p_\theta(\mathbf{x})}_{\times}$
- Solution:
  - In additional to define the decoder network $p_\theta(\mathbf{x} \mid \mathbf{z})$, define an additional *encoder network* $q_\phi(\mathbf{z} \mid \mathbf{x})$ that approximates the posterior $p_\theta(\mathbf{z} \mid \mathbf{x})$ $\Rightarrow$ variational inference with variational distribution $q_\phi(\mathbf{z} \mid \mathbf{x})$!.
  - We will see this allows us to derive a lower bound on the data likelihood, which can be optimized tractably.

Variational Autoencoder

Encoder and decoder networks are probabilistic

- Encoder also called inference/recognition network.
- Decoder also called generation network.

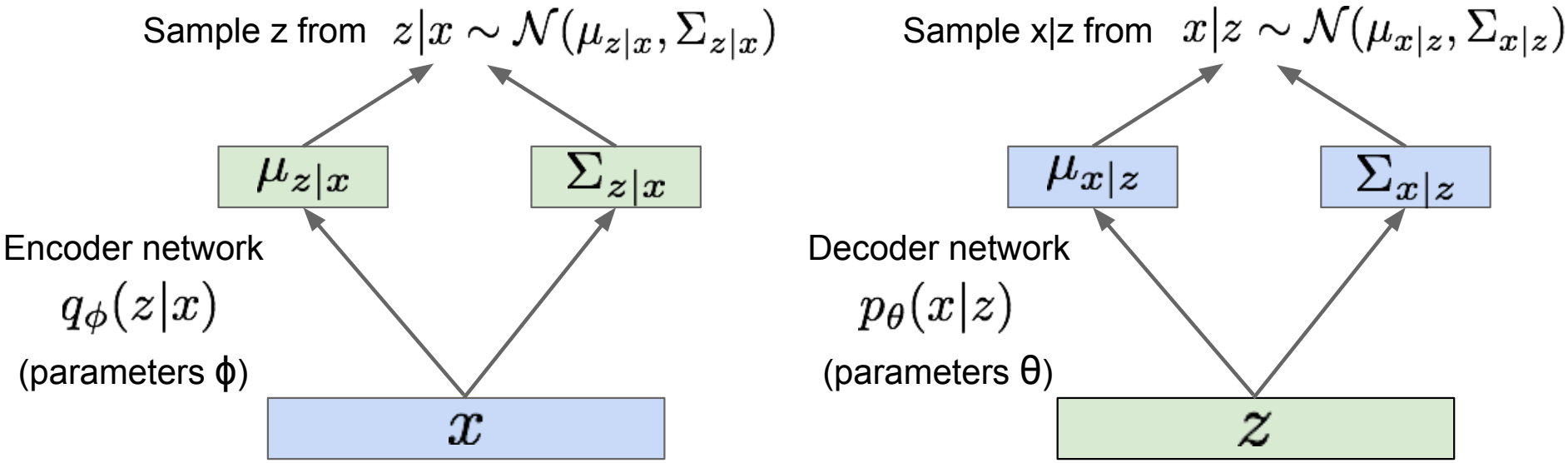




Variational Autoencoder

Encoder and decoder networks are probabilistic

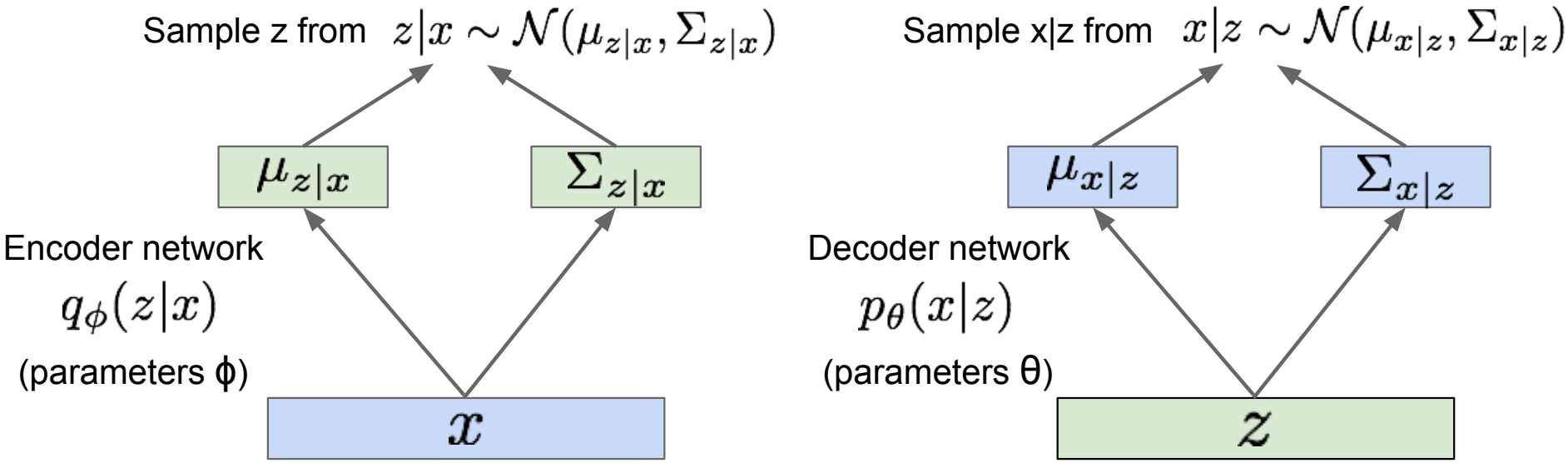
- Encoder also called inference/recognition network.
- Decoder also called generation network.



Variational Autoencoder

Encoder and decoder networks are probabilistic

- Encoder also called inference/recognition network.
- Decoder also called generation network.



Want to match $q_\phi(z|x)$ and $p(z|x) \propto p(z)p(x|z)$.

Variational Autoencoder: Log Data Likelihood

Derive from a different but equivalent way as in variational inference:

- Don't need to apply Jensen's inequality, or equivalently, prove Jensen's inequality.

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

Variational Autoencoder: Log Data Likelihood

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

Taking expectation wrt. z (using encoder network) will come in handy later

Variational Autoencoder: Log Data Likelihood

$$\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && (\text{Bayes' Rule})
\end{aligned}$$

Variational Autoencoder: Log Data Likelihood

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && \text{(Bayes' Rule)} \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && \text{(Multiply by constant)}
\end{aligned}
$$

Variational Autoencoder: Log Data Likelihood

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && \text{(Bayes' Rule)}\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && \text{(Multiply by constant)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] && \text{(Logarithms)}
\end{aligned}
$$

Variational Autoencoder: Log Data Likelihood

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && \text{(Bayes' Rule)}\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && \text{(Multiply by constant)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] && \text{(Logarithms)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}
$$

Variational Autoencoder: Log Data Likelihood

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && \text{(Bayes' Rule)}\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && \text{(Multiply by constant)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] && \text{(Logarithms)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}
$$

The expectation wrt. z (using encoder network) let us write nice KL terms

Variational Autoencoder: Log Data Likelihood

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

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

Decoder network gives $p_\theta(x|z)$, can compute estimate of this term through sampling. (Sampling differentiable through reparam. trick, see paper.)

This KL term (between Gaussians for encoder and z prior) has nice closed-form solution!

$p_\theta(z|x)$ intractable (saw earlier), can't compute this KL term :(But we know KL divergence always >= 0.

Variational Autoencoder: Log Data Likelihood

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && \text{(Bayes' Rule)}\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && \text{(Multiply by constant)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] && \text{(Logarithms)}\\
&= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}
\end{aligned}
$$

Tractable lower bound which we can take gradient of and optimize! ($p_\theta(x|z)$ differentiable, KL term differentiable)

Variational Autoencoder: Log Data Likelihood

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{>0}$$

$$\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)},\theta,\phi)$$

Variational lower bound ("ELBO")

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)},\theta,\phi)$$

Training: Maximize lower bound

Variational Autoencoder: Log Data Likelihood

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)},\theta,\phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{>0}$$

Reconstruct the input data

Make approximate posterior distribution close to prior

$$\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)},\theta,\phi)$$

Variational lower bound ("ELBO")

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)},\theta,\phi)$$

Training: Maximize lower bound

Variational Autoencoder: Log Data Likelihood

Putting it all together: maximizing the likelihood lower bound

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

Variational Autoencoder: Log Data Likelihood

Putting it all together: maximizing the likelihood lower bound

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

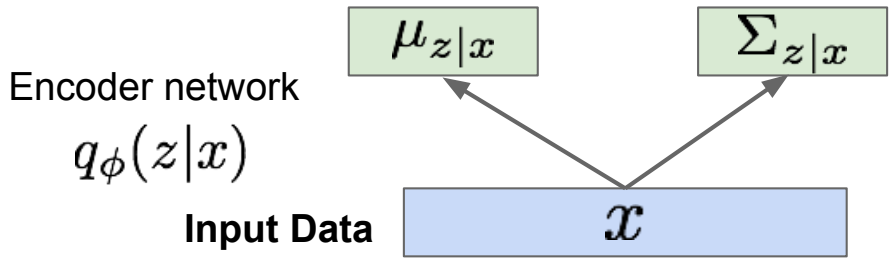
Let's look at computing the bound (forward pass) for a given minibatch of input data

Input Data x

Variational Autoencoder: Log Data Likelihood

Putting it all together: maximizing the likelihood lower bound

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

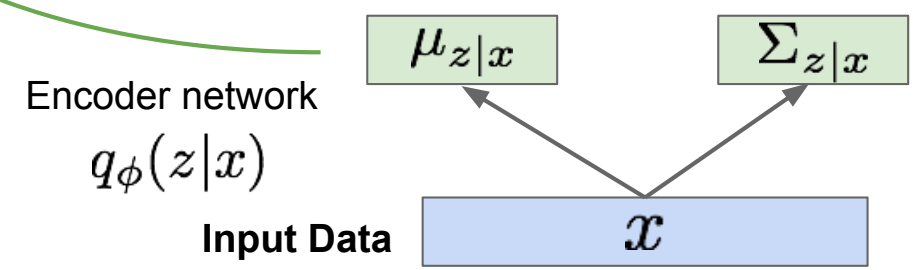


Variational Autoencoder: Log Data Likelihood

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

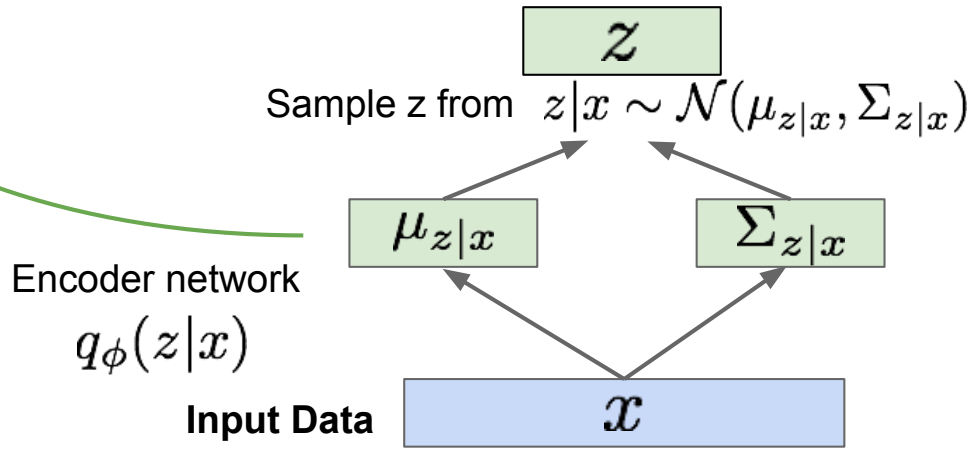


Variational Autoencoder: Log Data Likelihood

Putting it all together: maximizing the likelihood lower bound

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

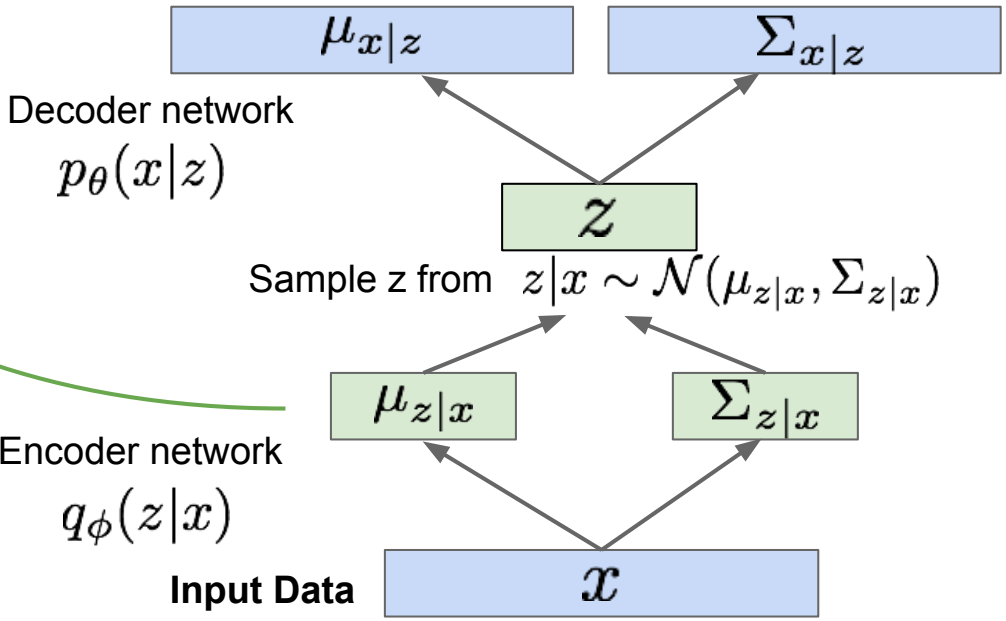
Make approximate posterior distribution close to prior



Variational Autoencoder: Log Data Likelihood

Putting it all together: maximizing the likelihood lower bound

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

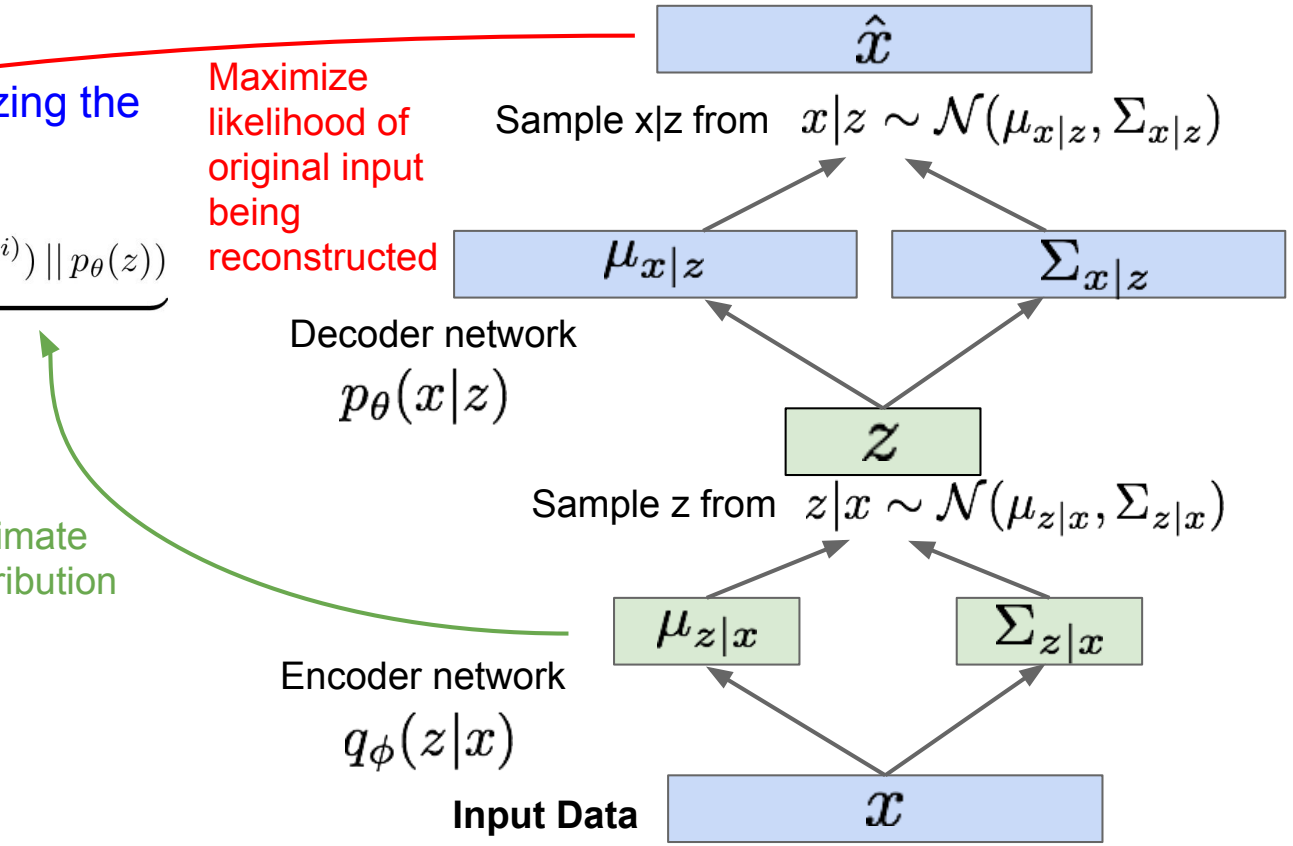


Variational Autoencoder: Log Data Likelihood

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



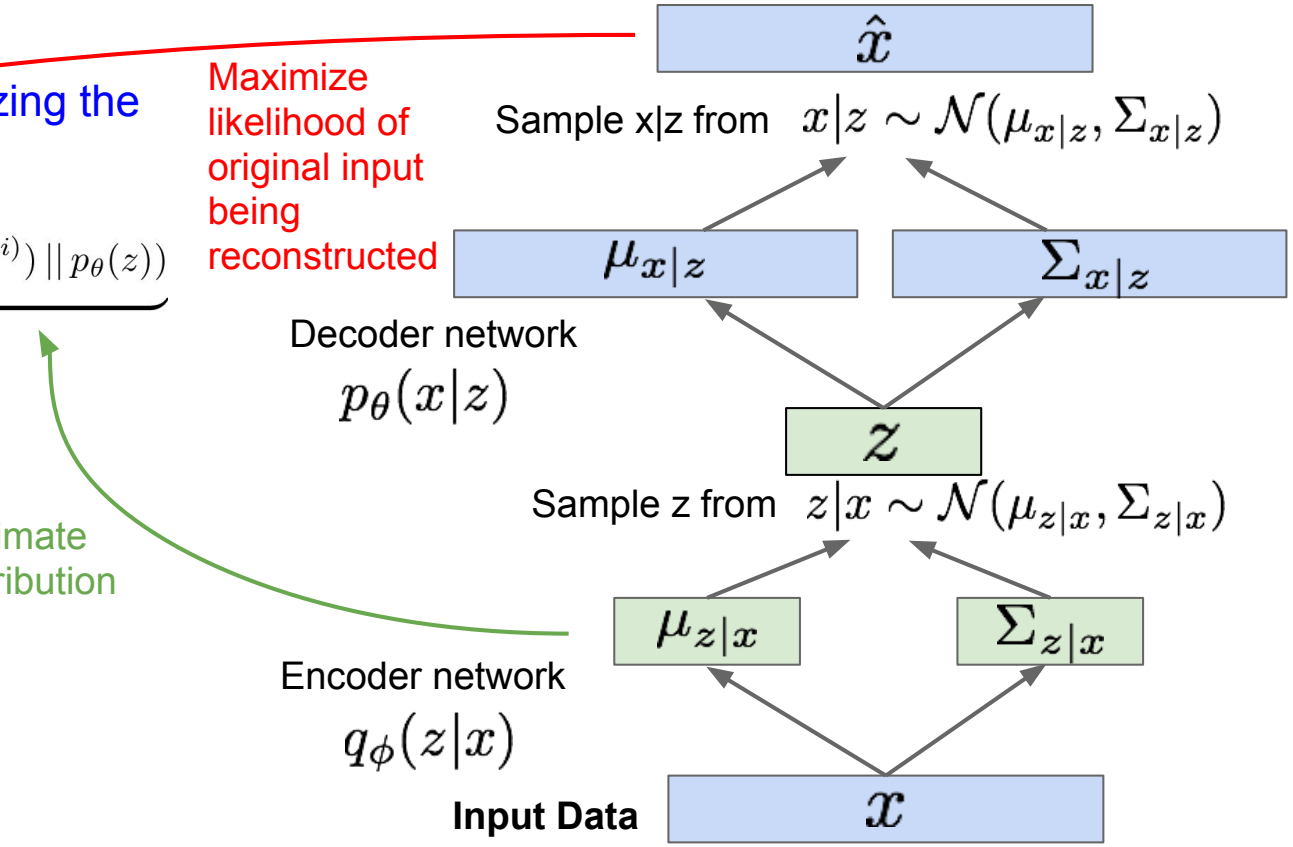
Make approximate posterior distribution close to prior

Variational Autoencoder: Log Data Likelihood

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

For every minibatch of input data: compute this forward pass, and then backprop!