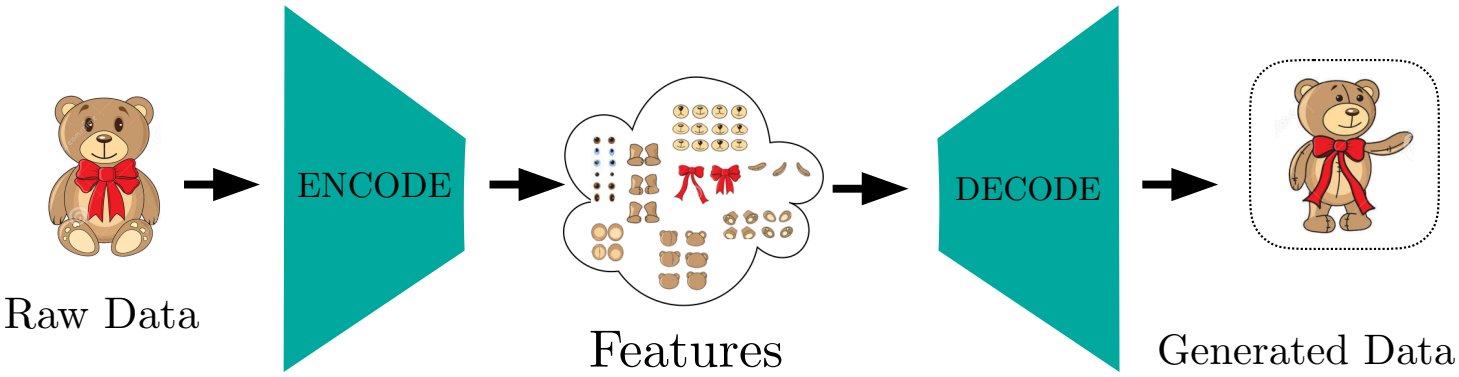
11-695: AI Engineering
Variational AutoEncoder

LTI/SCS

Spring 2020

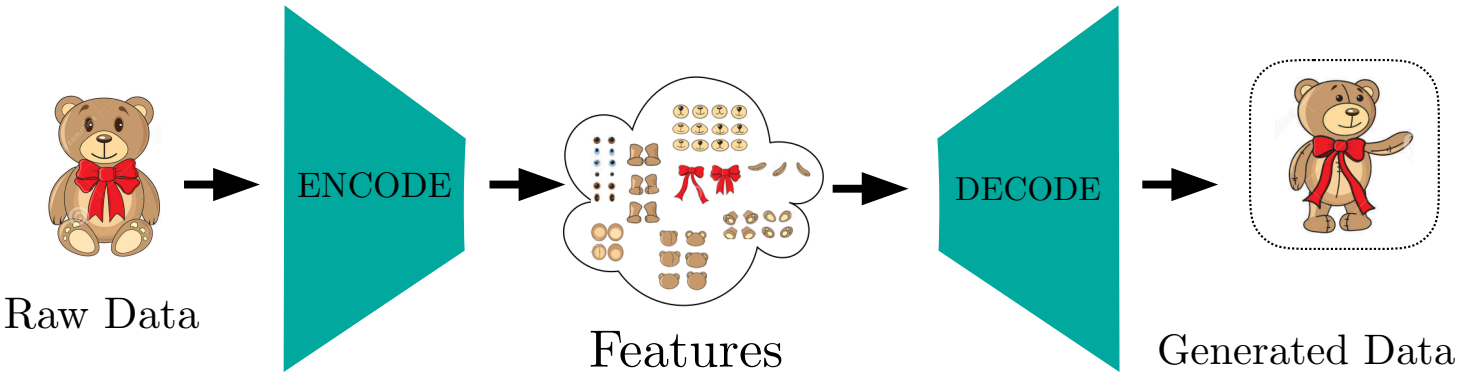
Outline

AutoEncoder



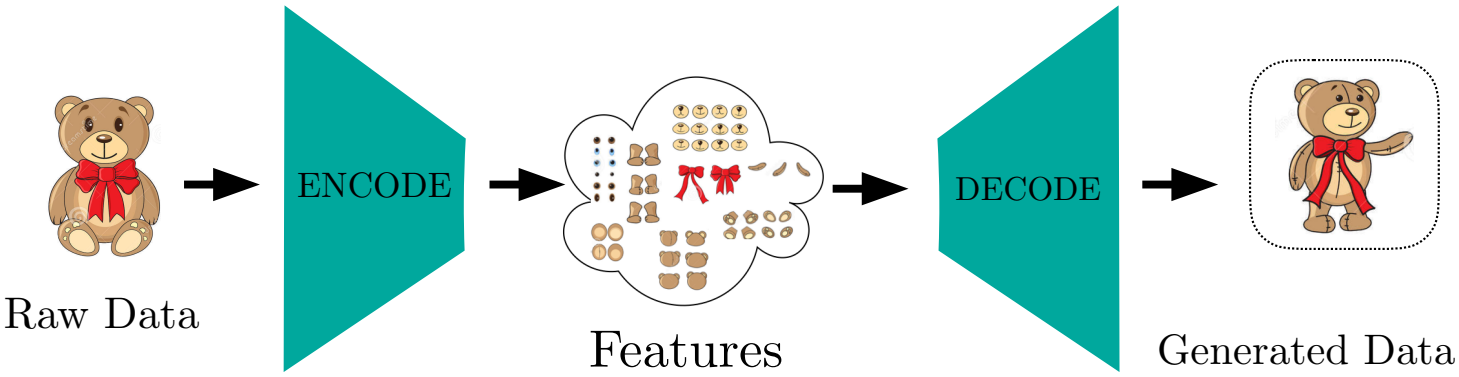
- Extract and Reconstruct *itself* → *Auto*-Encoder + Decoder
- Encoding is a compression process
- Compression: we need to learn prominent features only
  - Yet should be enough to reconstruct data

AutoEncoder



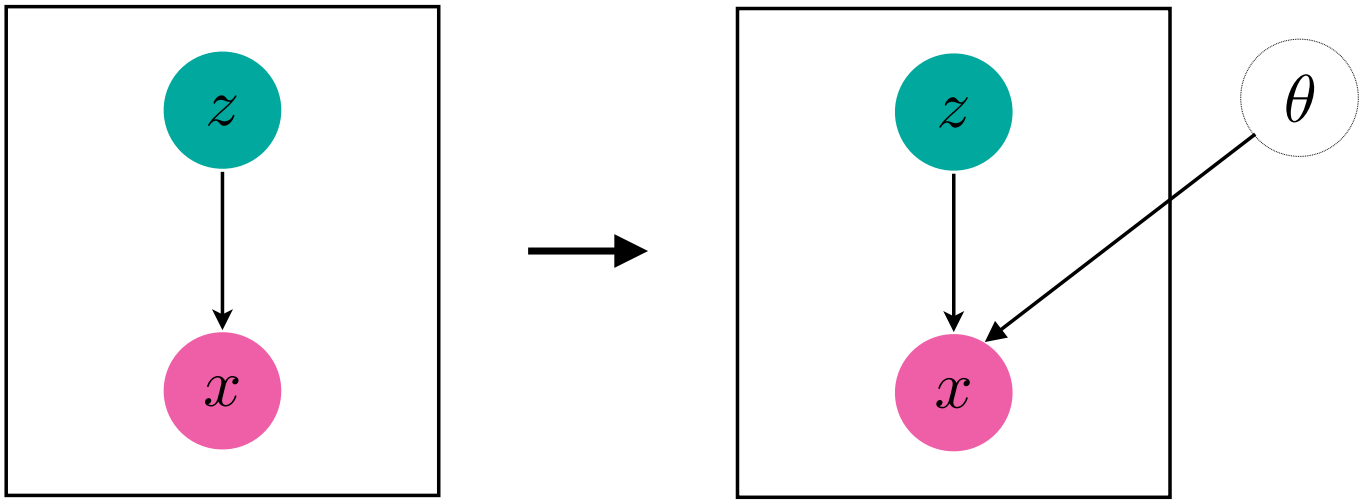
- Extract and Reconstruct *itself* → *Auto*-Encoder + Decoder
- Encoding is a compression process
- Compression: we need to learn prominent features only
  - Yet should be enough to reconstruct data
  - How? Unless for trivial cases, we need params θ

AutoEncoder



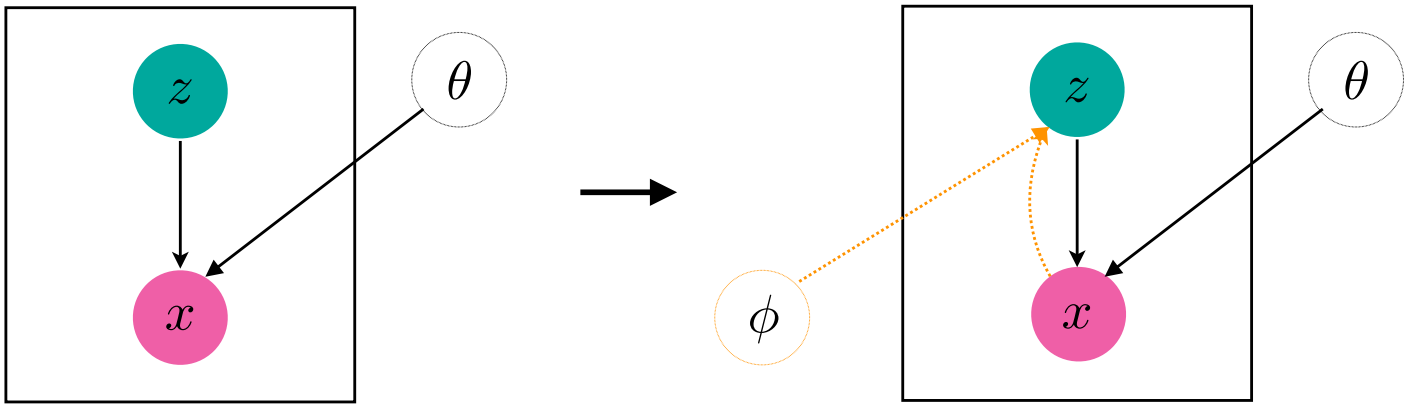
- Extract and Reconstruct *itself* $\rightarrow$ *Auto*-Encoder + Decoder
- Encoding is a compression process
- Compression: we need to learn prominent features only
  - Yet should be enough to reconstruct data
  - How? Unless for trivial cases, we need params θ
  - In general, when we need to learn (a set of) many params, just uses a deep NN to approximate!

AE - Probabilistic Graphical View



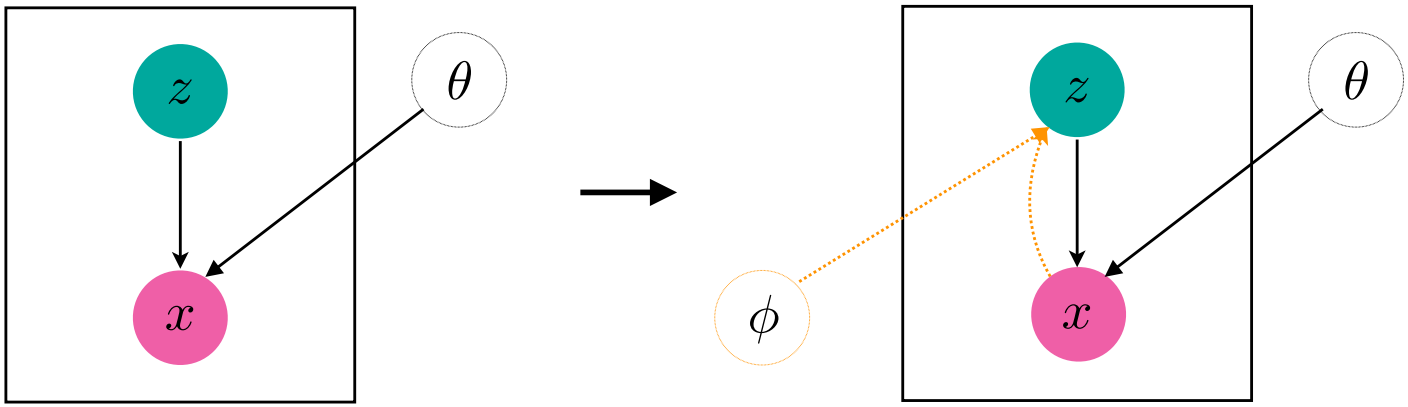
- Embedded Space is considered a Latent Space z
  - Input x, encoding proces: $z \sim p(z|x) = f_\theta(x)$
  - Decoding process: $\hat{x} \sim p(x|z) = f_\psi(x|z)$
- We can approximate f_θ and f_ψ by 2 neural networks

More constraint needed for z



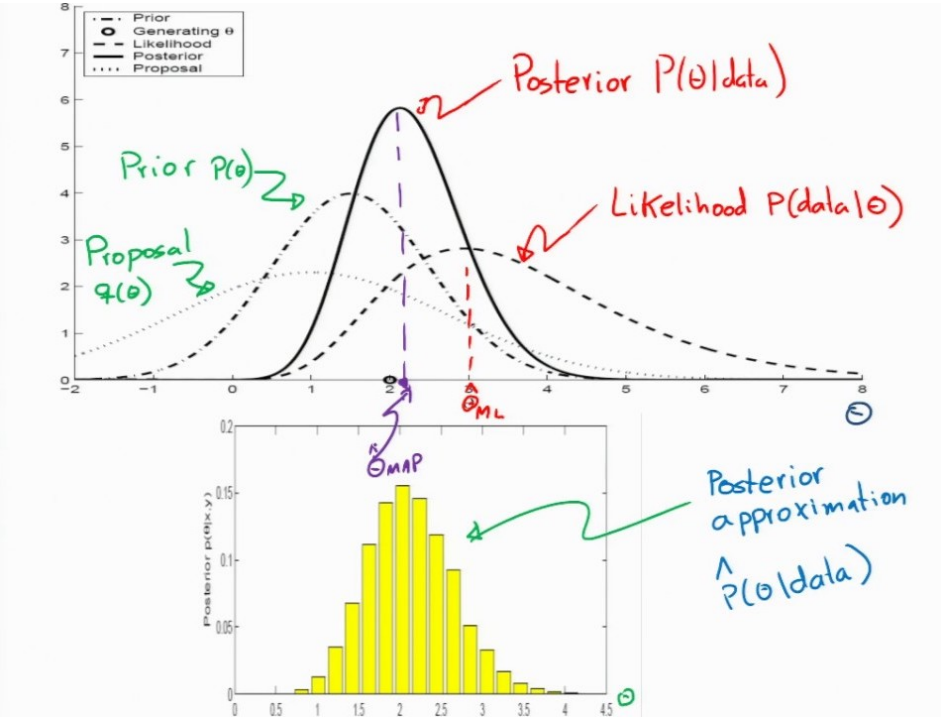
- But as we know, it's not good enough, *e.g.* vision.
- AE cannot *generate* new samples
- To actually form a generative model, we need a new randomness injected into the model
- Choose to inject into z (why?), and call it new params ϕ.

More constraint needed for z



- All's about *Approximation*: need another approximation method (besides f_θ to characterize ϕ)
- Choice: variational bayes approximation
- Note: modeling $p(z|x)$ is called the *inference* problem, *i.e.* to infer the latent representation
- So $p(z|x)$ is called posterior with likelihood $p(x|z)$ and prior $p(z)$

Some justifications by comparison?

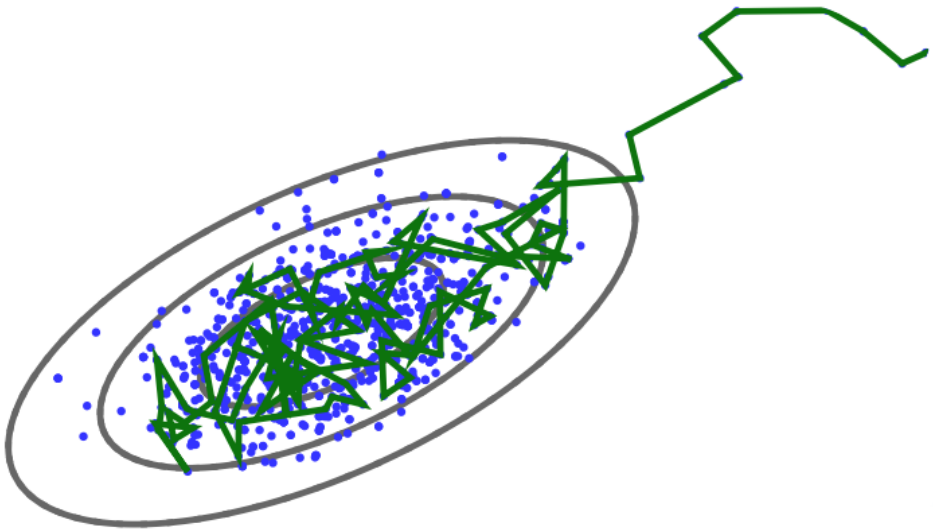


- Our job: model $p(z|x)$
- Proposed method: MAP with $p(z|x) = p(x|z)p(z)/p(x)$ with a prior e.g. $p(z) \sim \mathbf{N}(\mathbf{0}, \mathbf{I})$
  - Too much biased to prior.

Image credit: Nando De Freitas

Some justifications by comparison?

Carnegie Mellon

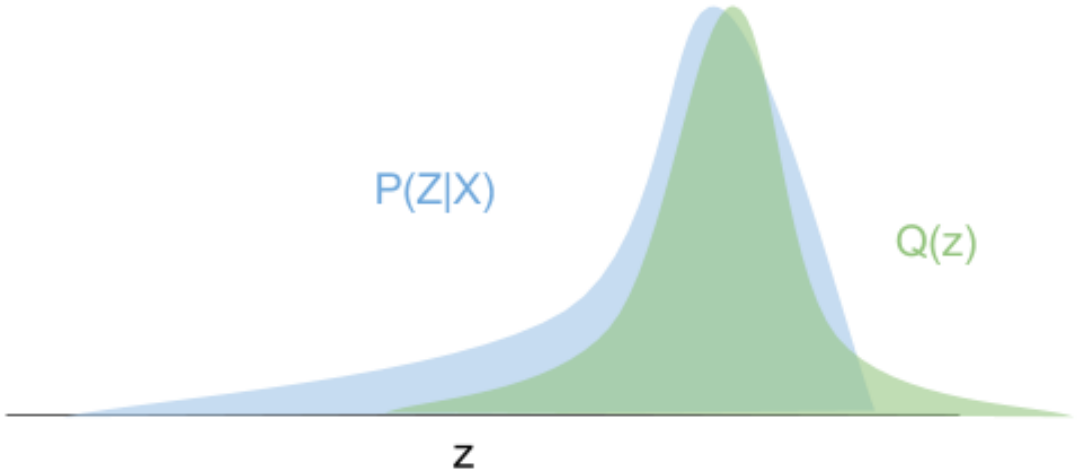


- Our job: model $p(z|x)$
- Proposed method: sampling to estimate, such as MCMC
  - MC chain is hard to control to convergence (mix), and it's slow.
- Resource: ▸ MCMC Visualization

Image credit: UBC Course CPSC522

Outline

Mean-Field Variational Method



- Problem: intractable distribution $p(z|x)$
- Pick a well known, simple distribution $q(z)$
- Push $q(z)$ as close to $p(z|x)$ as possible (tuning params of $q(z)$)
- $q(z)$ is the (variational) approximation of $p(z|x)$

Image credit: Eric Jang

Mean-Field Variational Method

Carnegie Mellon

- Problem: Generative model: $p(x)$

$$\log p(x) = \log \int p(x|z)p(z)dz \tag{1}$$

$$= \log \int p(x|z)\frac{p(z)}{q(z)}q(z)d(z) \tag{2}$$

$$\geq \int q(z) \log \left(p(x|z)\frac{p(z)}{q(z)} \right) dz \tag{3}$$

$$= \int q(z) \left(\log \left(p(x|z) + \log \frac{p(z)}{q(z)} \right) dz \right. \tag{4}$$

$$= \int q(z) \log p(x|z) - \int q(z) \log \frac{q(z)}{p(z)} dz \tag{5}$$

$$= \mathbb{E}_{q(z)}[\log p(x|z)] - \mathbf{D}_{KL}[q(z)\|p(z)] \tag{6}$$

$$= \text{Reconstruction Error} - \text{Deviation Penalty} \tag{7}$$

Deviation Penalty

- There is a new term $\mathbf{D}_{KL}$ which tunes $q(z)$ to go close to $p(z)$

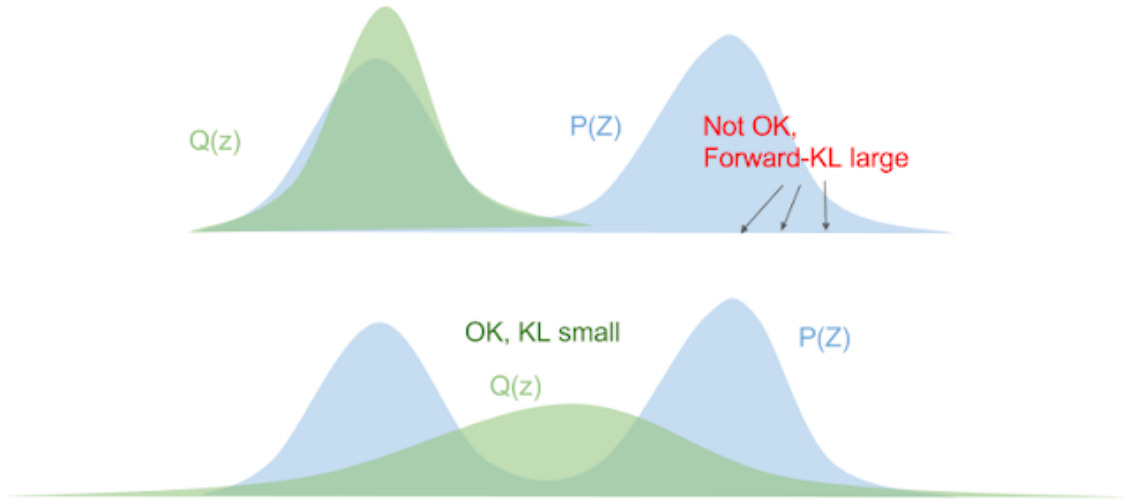
$$
\begin{aligned}
log\, p(x) &= \mathbb{E}_{q(z)}[\log p(x|z)] - \mathbf{D}_{KL}[q(z)\|p(z)] \\
&= \mathbb{E}_{q(z)}[log\, p(z,x)] - \mathbb{E}_{q(z)}[log\, p(z)] - \mathbf{D}_{KL}[q(z)\|p(z)]
\end{aligned}
$$

- If we call the reconstruction error ELBO:

$$
\begin{aligned}
\text{ELBO} &= \log\, p(x) - \mathbf{D}_{KL}(q(z)\|p(z|x)) && (8)\\
&= \mathbb{E}_{q(z)}[\log\, p(z,x)] - \mathbb{E}_{q(z)}[\log\, q(z)] && (9)\\
&= \mathbb{E}_{q(z)}[log\, p(x|z)] - \mathbf{D}_{KL}(q(z)\|p(z)) && (10)
\end{aligned}
$$

  - ELBO (Evidence Lower Bound) will be the lower-bound of $p(x)$
  - Minimizing $\mathbf{D}_{KL}(q(z)\|p(z)) \geq 0$ will make ELBO close to $p(x)$
  - Equivalent to maximizing ELBO.

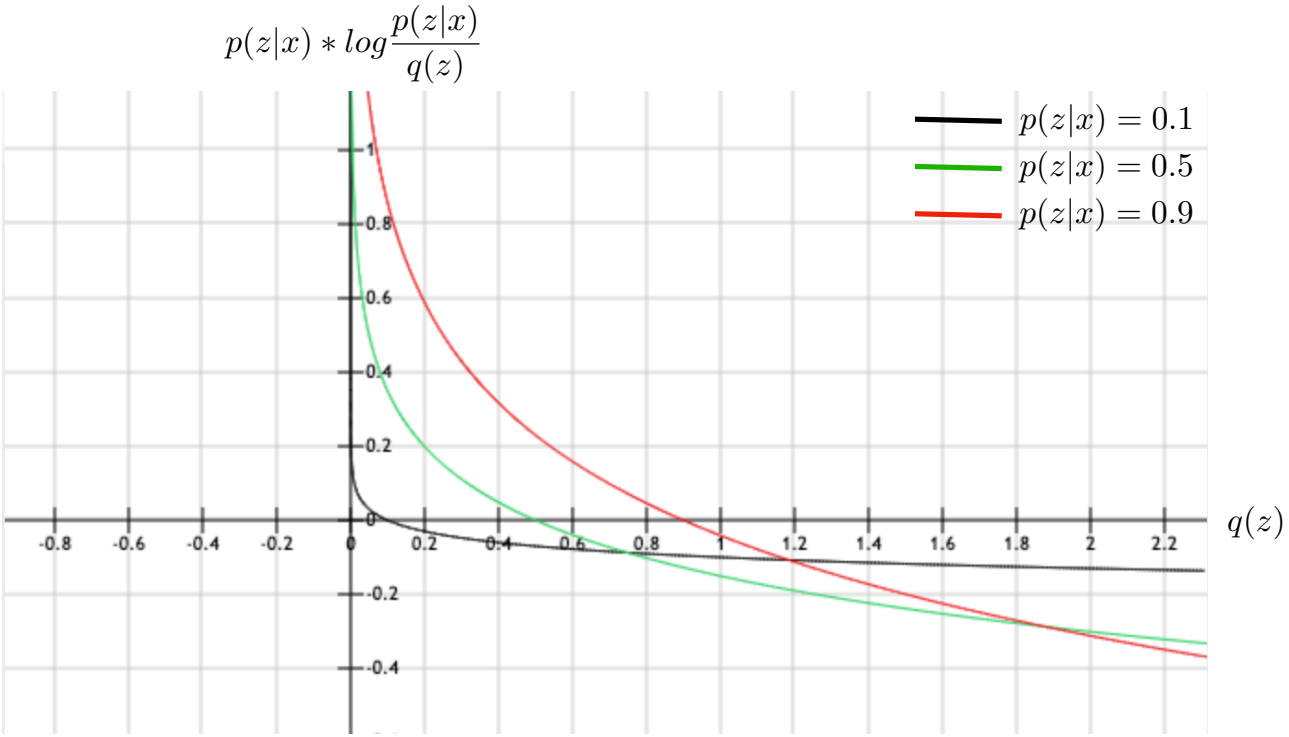
Problem with Forward KL



- Forward KL: $\mathbf{D}_{KL}(p(z|x)\|q(z)) = \int p(z|x) \log \frac{p(z|x)}{q(z)} dz$
- Problem:
  - We don't know $p(z|x)$
  - If $p(z|x) = 0$, KL is always zero $\rightarrow$ optimize nothing

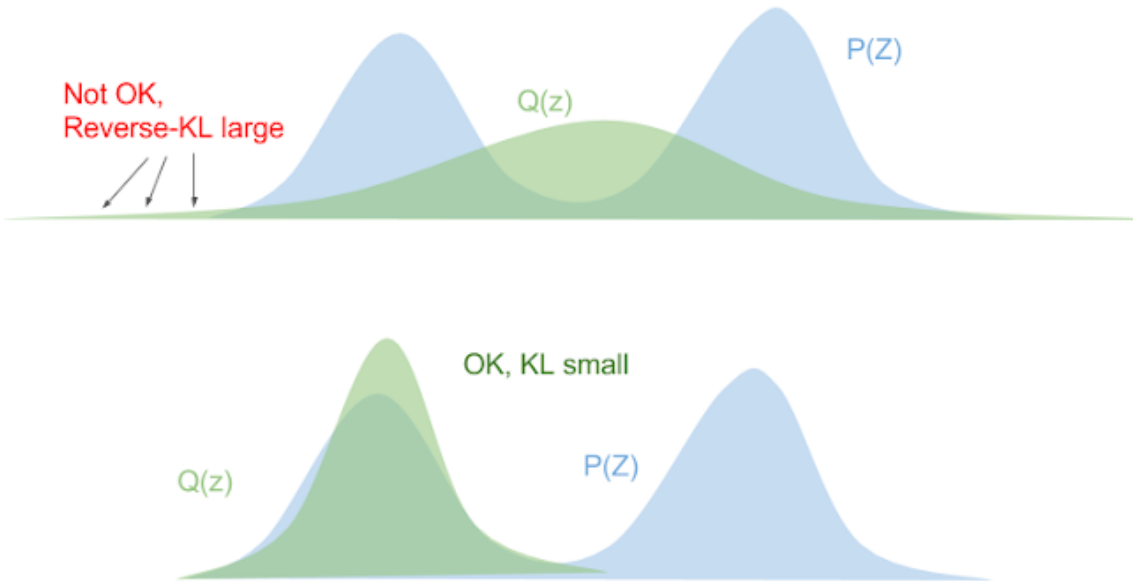
Image credit: Eric Jang

Problem with Forward KL



- Forward KL: $\mathbf{D}_{KL}(p(z|x)\|q(z)) = \int p(z|x) log\, \frac{p(z|x)}{q(z)}$
- Problem:
  - If $p(z|x) > 0$ learn to pick $q(z) > 0$ low as wanted ...

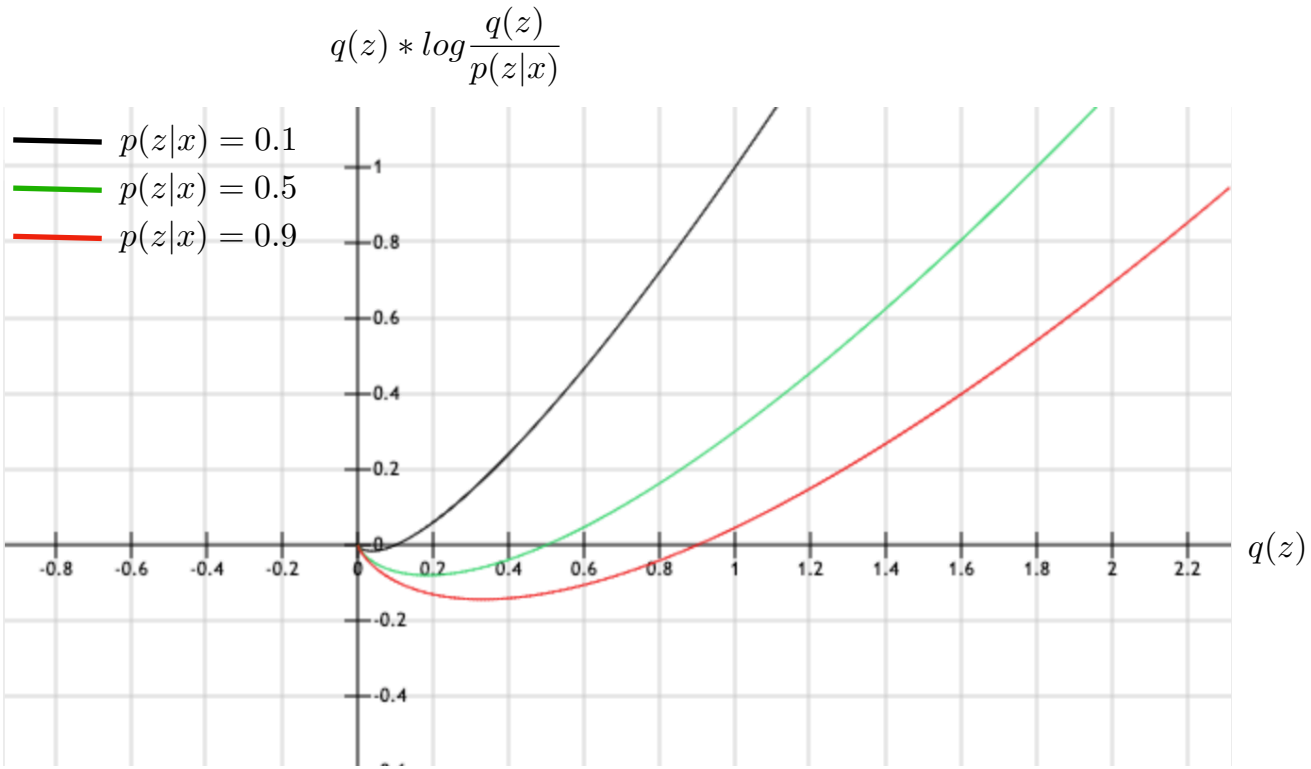
Why using Reverse KL



- Reverse KL: $\mathbf{D}_{KL}(q(z)\|p(z|x)) = \int q(z) \log \frac{q(z)}{p(z|x)} dz$
- Now:
  - We can control variational distribution $q(z)$
  - If $p(z|x) = 0$, learn to force $q(z)$ close to zero as well

Image credit: Eric Jang

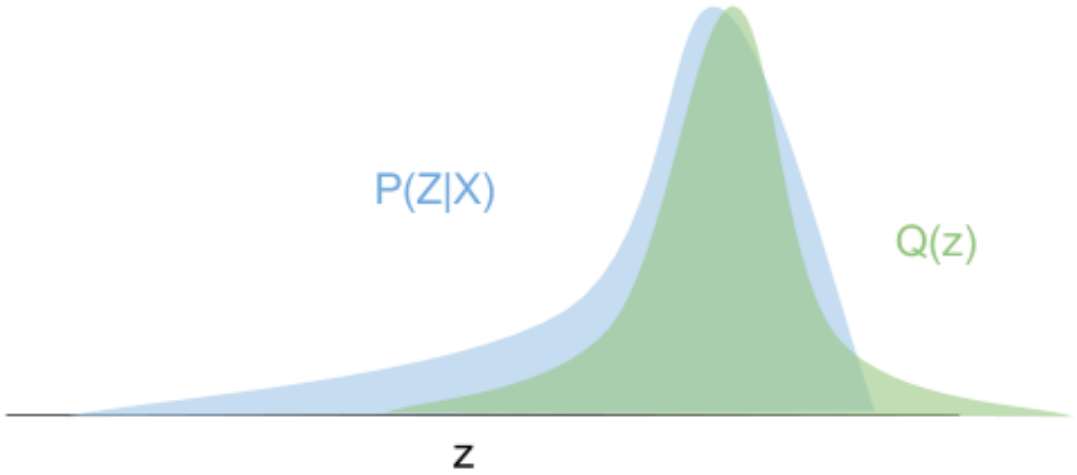
Why using Reverse KL



- Reverse KL: $\mathbf{D}_{KL}(q(z)\|p(z|x)) = \int q(z) \log \frac{q(z)}{p(z|x)} dz$
- Nice property:
  - If $p(z|x) > 0$ learn to grow $q(z)$ to "catch up with" $p(z|x)$

Outline

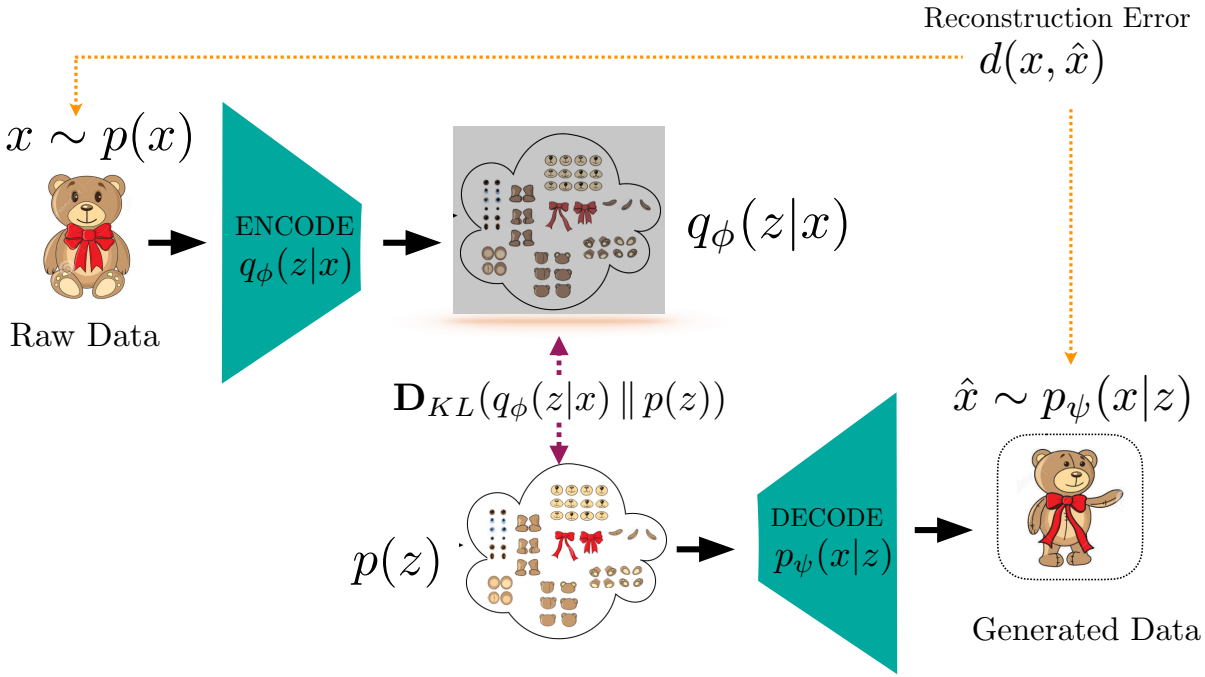
Variational Parameters



- So far, we use a variational distribution $q(z)$ to approximate $p(z|x)$
- In our case, we use variational distribution $q(z|x)$ to approximate the encoder (inference) with params ϕ
- And as usual, we aproximate ϕ by a neural network.

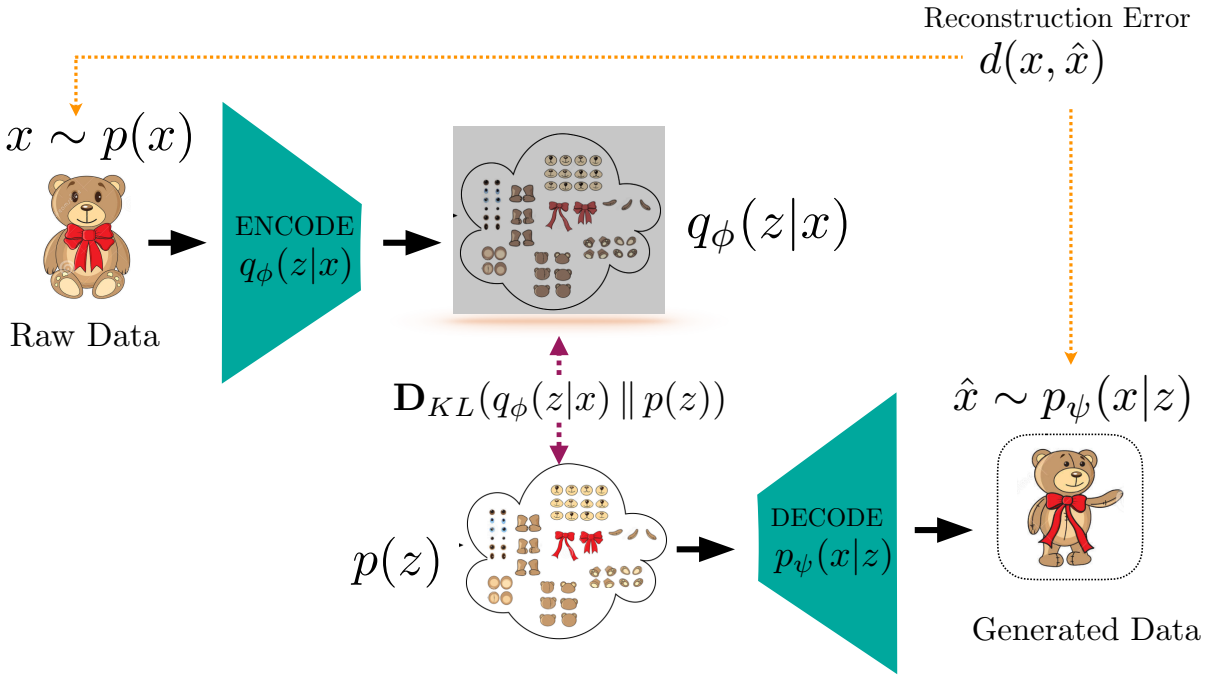
Image credit: Eric Jang

AE + Variational Approximation



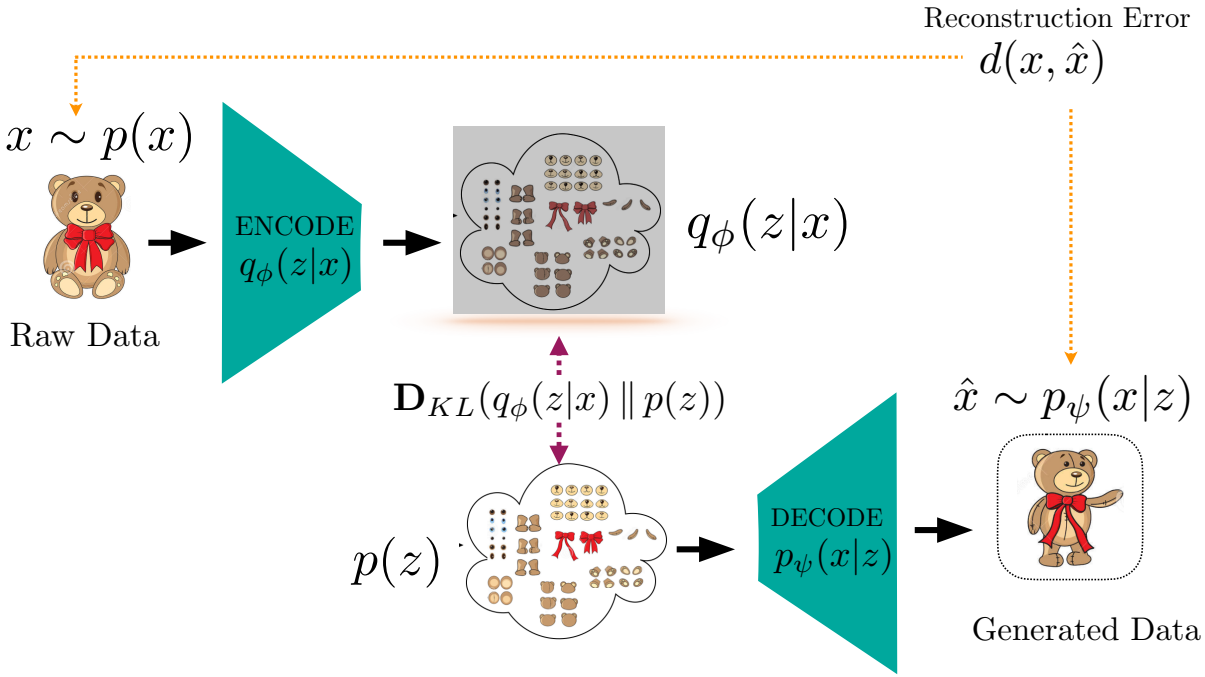
- Still AutoEncoder, with KL term for variational → VAE

AE + Variational Approximation



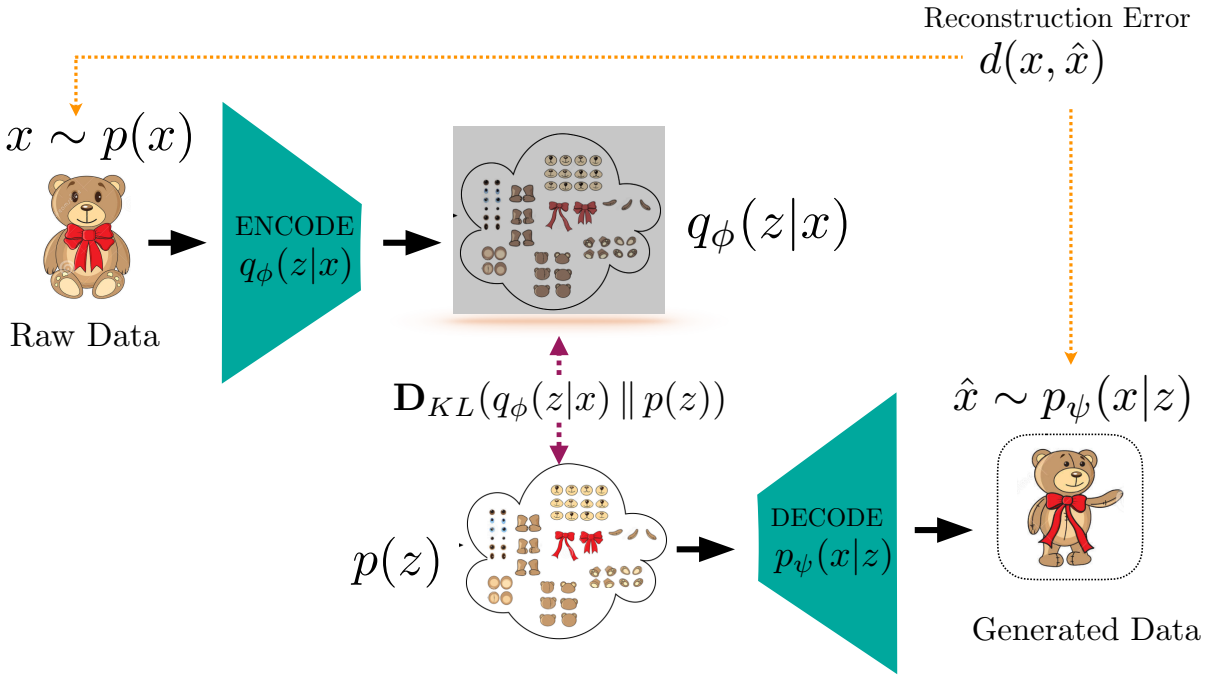
- Still AutoEncoder, with KL term for variational → VAE
- Note: Decoding phase is from $p(z)$, and not the auto-encoded probability $q(z|x)$

AE + Variational Approximation



- Still AutoEncoder, with KL term for variational $\rightarrow$ VAE
- Note: Decoding phase is from $p(z)$, and not the auto-encoded probability $q(z|x)$
- ELBO $= \mathbb{E}_{q_\phi(z|x))}[log\, p(x|z)] - \mathbf{D}_{KL}(q_\phi(z|x)\|p(z))$

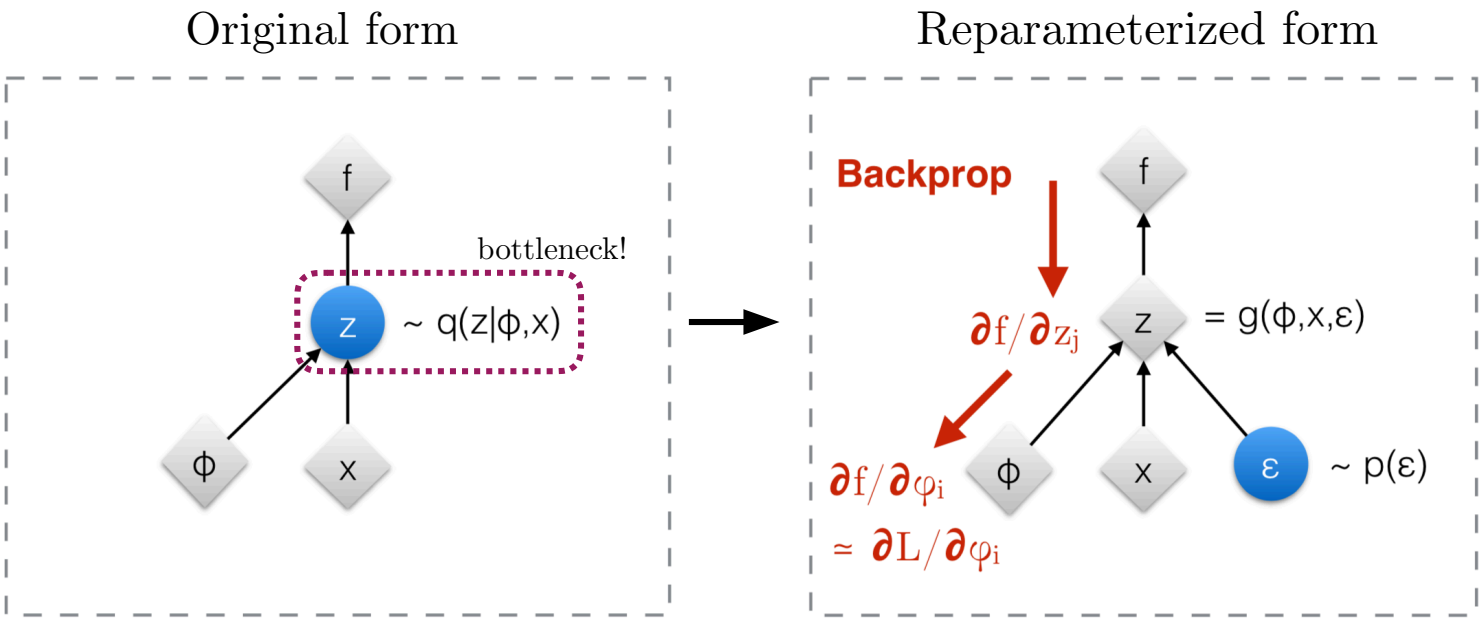
AE + Variational Approximation



- Still AutoEncoder, with KL term for variational → VAE
- Note: Decoding phase is from $p(z)$, and not the auto-encoded probability $q(z|x)$
- ELBO $= \mathbb{E}_{q_\phi(z|x))}[log\, p(x|z)] - \mathbf{D}_{KL}(q_\phi(z|x)\|p(z))$
- $p(z)$ is random, so now we can actually *generate* data

Outline

But, not yet fully differentiable



- Sampling $z \sim q_\phi(z|x) = q(z|x,\phi)$ is not continuous
- Trick: push the sampling to the input level of the graph
  - $\epsilon \sim p(\epsilon) = \mathbf{N}(\mathbf{0}, \mathbf{I})$
  - $z = g_\phi(z, \epsilon)$, a differentiable neural network now!

Image credit: Max Welling