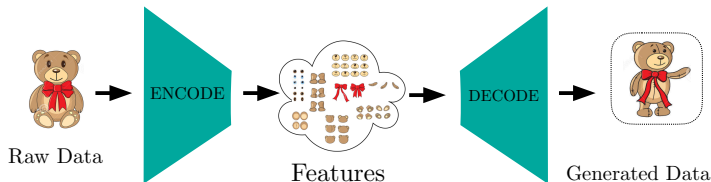


11-695: AI Engineering

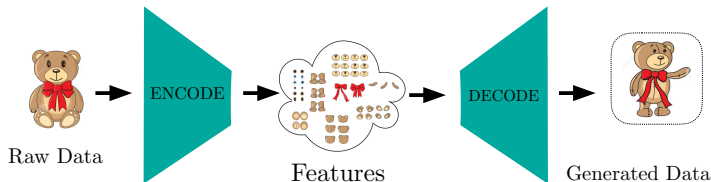
Variational AutoEncoder

Spring 2019

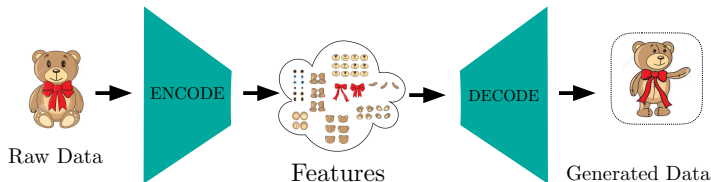
- 1 AutoEncoder Refresh
- 2 Variational Approximation
- 3 Variational AutoEncoder
- 4 Training with Reparametization



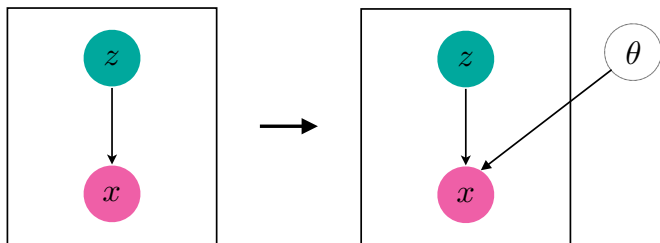
- Extract and Reconstruct *itself* $\rightarrow$ Auto-Encoder + Decoder
- Encoding is a compression process
- Compression: we need to learn prominent features only
 - Yet should be enough to reconstruct data



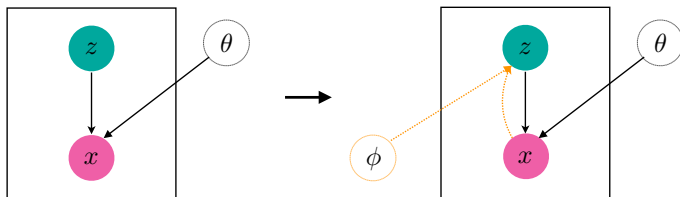
- Extract and Reconstruct *itself* $\rightarrow$ Auto-Encoder + Decoder
- Encoding is a compression process
- Compression: we need to learn prominent features only
 - Yet should be enough to reconstruct data
 - How? Unless for trivial cases, we need params θ



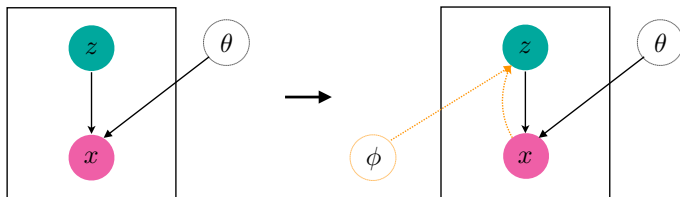
- Extract and Reconstruct *itself* $\rightarrow$ Auto-Encoder + Decoder
- Encoding is a compression process
- Compression: we need to learn prominent features only
 - Yet should be enough to reconstruct data
 - How? Unless for trivial cases, we need params θ
 - In general, when we need to learn (a set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



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

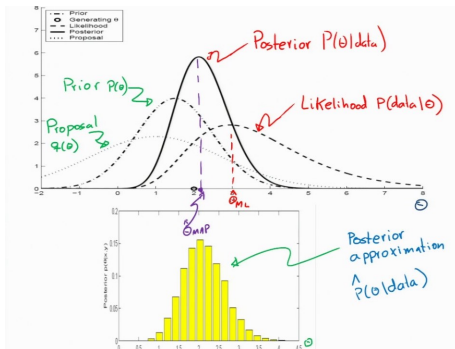


- But as we know, it's not good enough, *e.g.* for such complex problem as vision.
- Recall: AutoEncoder is a two-phase process, errors made in encoding to z will be *compounded* in the decoding to $\hat{x}$
- Let's call it a new set of parameter ϕ as a new constraint to refine the learning of latent space z from data

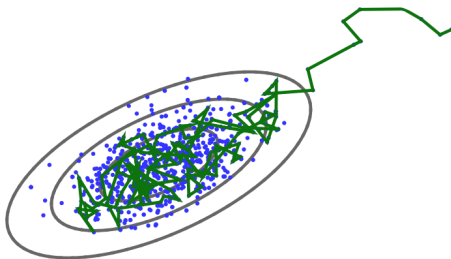


- All's about *Approximation*: need another approximation method (besides f_θ to characterize ϕ)
- Choice: variational approximation (probabistic method)

Some justifications by comparison?

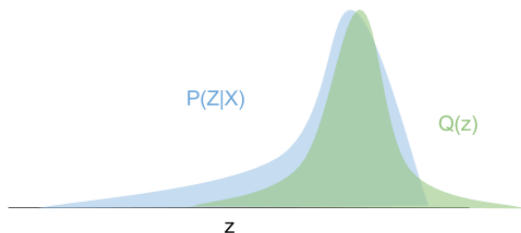


- Our job: model $p(z|x)$
- Proposed method: MAP with $p(z|x) = p(x|z)p(z)/p(x)$ with a prior e.g. $z \sim \mathbf{N}(\mathbf{0}, \mathbf{I})$
 - Too much biased to prior.



- 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](#)

- 1 AutoEncoder Refresh
- 2 Variational Approximation
- 3 Variational AutoEncoder
- 4 Training with Reparametization



- 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 $p(z|x)$

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

$$\log p(x) = \log \int p(x|z)p(z)dz \quad (1)$$

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

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

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

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

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

$$= \text{Reconstruction Error} - \text{Deviation Penalty} \quad (7)$$

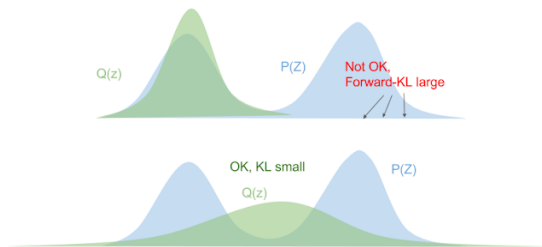
- There is a new term $\mathbf{D}_{KL}$:
 $\log p(x) = \mathbb{E}_{q(z)}[p(x|z)] - \mathbf{D}_{KL}[q(z)||p(z)]$
- This penalty will tune $q(z)$ to go close to $p(z|x)$
- Other derivation can lead to another theoretical result of ELBO:

$$\text{ELBO} = \mathbb{E}_{q(z|x)}[\log p(z, x)] - \mathbb{E}_{q(z|x)}[\log q(z)] \quad (8)$$

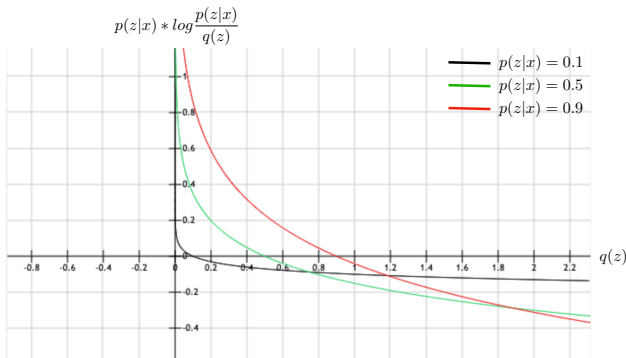
$$= \mathbb{E}_{q_\phi(z|x)}[f_\phi(x, z)] \quad (9)$$

$$= \log p(x) - \mathbf{D}_{KL}(q(z)||p(z|x)) \quad (10)$$

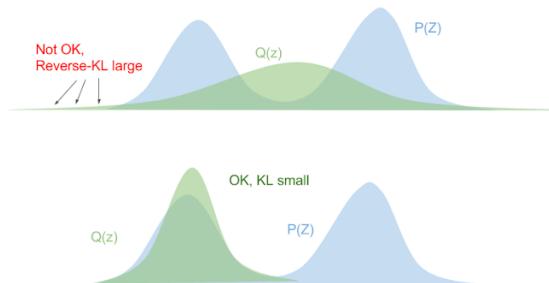
- ELBO (Evidence Lower Bound) will be the lower-bound of $p(x)$
- Minimizing $\mathbf{D}_{KL}(q(z)||p(z|x)) \geq 0$ will make ELBO close to $p(x)$
- Equivalent to maximizing ELBO.



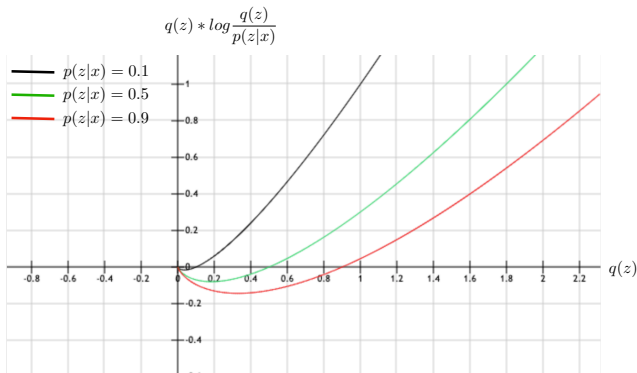
- Forward KL: $\mathbf{D}_{KL}(p(z|x) || q(z)) = \int p(z|x) \log \frac{p(z|x)}{q(z)}$
- Problem:
 - We don't know $p(z|x)$
 - If $p(z|x) = 0$, KL is always zero $\rightarrow$ optimize nothing



- 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 ...

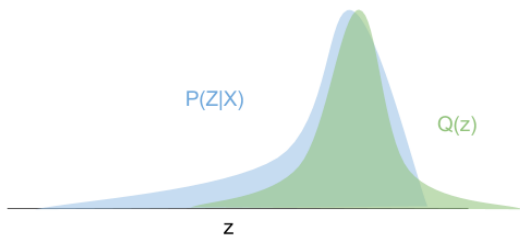


- Reverse KL: $\mathbf{D}_{KL}(q(z)||p(z|x)) = \int q(z) \log \frac{q(z)}{p(z|x)}$
- Problem:
 - We can control variational distribution $q(z)$
 - If $p(z|x) = 0$, learn to force $q(z)$ close to zero as well

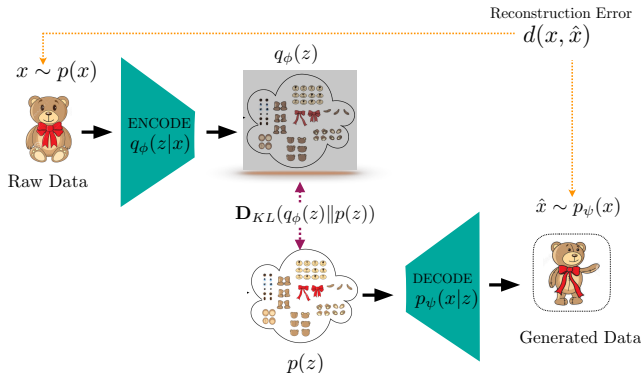


- Reverse KL: $\mathbf{D}_{KL}(q(z)||p(z|x)) = \int q(z) \log \frac{q(z)}{p(z|x)}$
- Problem:
 - If $p(z|x) > 0$ learn to grow $q(z)$ to “catch up with” $p(z|x)$

- ① AutoEncoder Refresh
- ② Variational Approximation
- ③ Variational AutoEncoder
- ④ Training with Reparametization



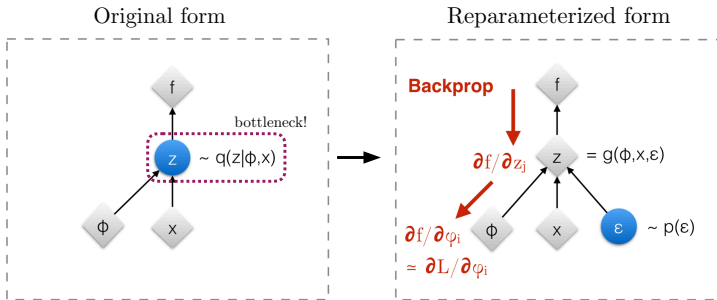
- So far, we use a variational distribution $q(z)$ to approximate $p(z|x)$
- Not yet mentioned: each variational distribution as a set of variational parameter ϕ
- And as usual, we approximate ϕ by a neural network (MLP).



- Still AutoEncoder, with KL term for variational vs. true distribution $\rightarrow$ Variational AutoEncoder

- 1 AutoEncoder Refresh
- 2 Variational Approximation
- 3 Variational AutoEncoder
- 4 Training with Reparametization

But, not yet fully differentiable



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