CS 8501 Advanced Topics in Machine Learning

Lecture 11: Variational Autoencoder

Yangfeng Ji
Information and Language Processing Lab
Department of Computer Science
University of Virginia
https://yangfengji.net/

A Quick Review

Generative Modeling

This lecture focuses on the discussion in the following form

- prior: $z \sim p_\theta(z)$
- generation model: $x|z \sim \mathrm{Expfam}(x|d_\theta(z))$

where $d_\theta(z)$ is a deep neural network and $\mathrm{Expfam}(x|\eta)$ is an exponential family with parameter η.

- For example, Gaussian distribution, with $\eta = \{\mu, \sigma^2\}$

Posterior Inference

Given x, infer the posterior distribution of z

$$p_\theta(z|x) = \frac{p_\theta(z)p_\theta(x|z)}{p_\theta(x)}$$

with

$$p(x) = \int p_\theta(x|z)p_\theta(z)dz$$

In practice, we often use **amortized inference**, which use a variational distribution $q_\phi(z|x)$ to approximate $p_\theta(z|x)$

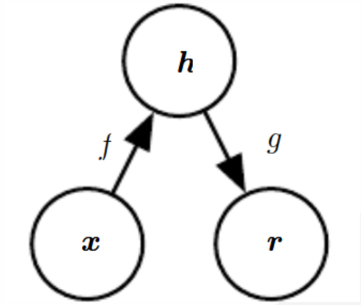
When q_ϕ is defined on a neural network, it is also called *inference network* or *recognition network*.

Autoencoder

Reference

- Goodfellow et al. Deep Learning. 2016

Autoencoder



- Encoder $f : x \rightarrow h$: mapping input x to a latent representation h
- Decoder $g : h \rightarrow r$: mapping latent representation h back to the input space as $\hat{x}$
- Training an auto-encoder by optimize the objective function defined on x and r, such as

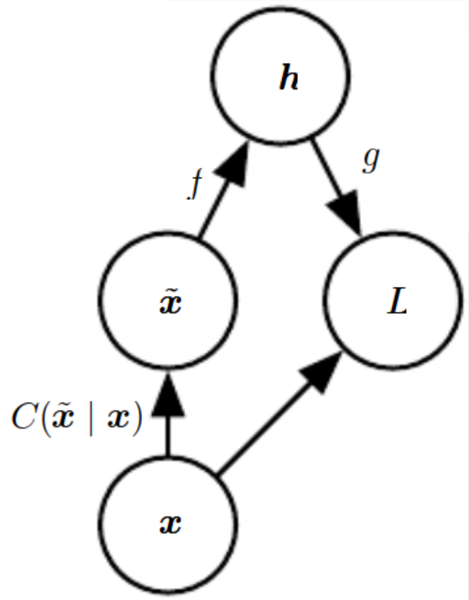
$$L(x, g(f(x))) = \|x - g(f(x))\|_2^2$$

Denoising Autoencoder

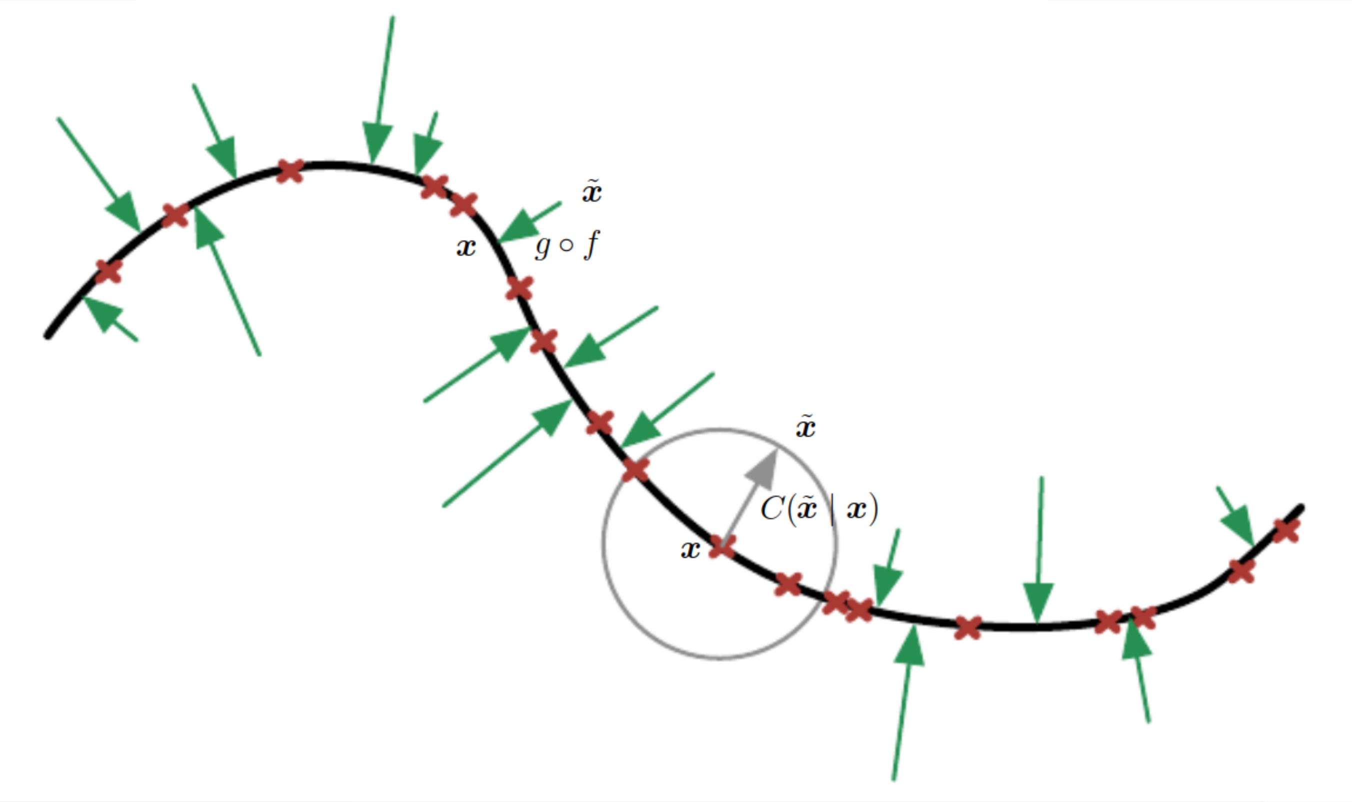
- Improve the generalization power of autoencoders by adding noise to inputs

$$L(x, g(f(\tilde{x})))$$

- $\tilde{x}$ is a copy of x that has been corrupted by some form of noise



Learning Denoising Autoencoder



It improve the encoder's representation power, but cannot do generation

VAE Basics

Generative Models

A VAE defines a generative model

$$p_\theta(z, x) = p_\theta(z)p_\theta(x|z)$$

The generation procedure can be formulated as

- Sample a latent variable $z \sim p_\theta(z)$
- Generate an observation based on z, $x \sim p_\theta(x|z)$

Example

Consider a binary image

$$p_\theta(x|z) = \prod_{d=1}^{D} \mathrm{Ber}(x_d|\sigma(d_\theta(z)))$$

where

- $d_\theta(\cdot)$ is a neural network model
- $\sigma(\cdot)$ is a Sigmoid function
- $\mathrm{Ber}(x_d|\sigma(d_\theta(z)))$ is a Bernoulli distribution with parameter $\sigma(d_\theta(z))$

Recognition Network

- In practice, instead of sampling from a prior distribution $p(z)$, we prefer to sample from $p_\theta(z|x)$ if possible, because it offers a reasonable starting point.
- Amortized inference offers us an approximation of $p_\theta(z|x)$

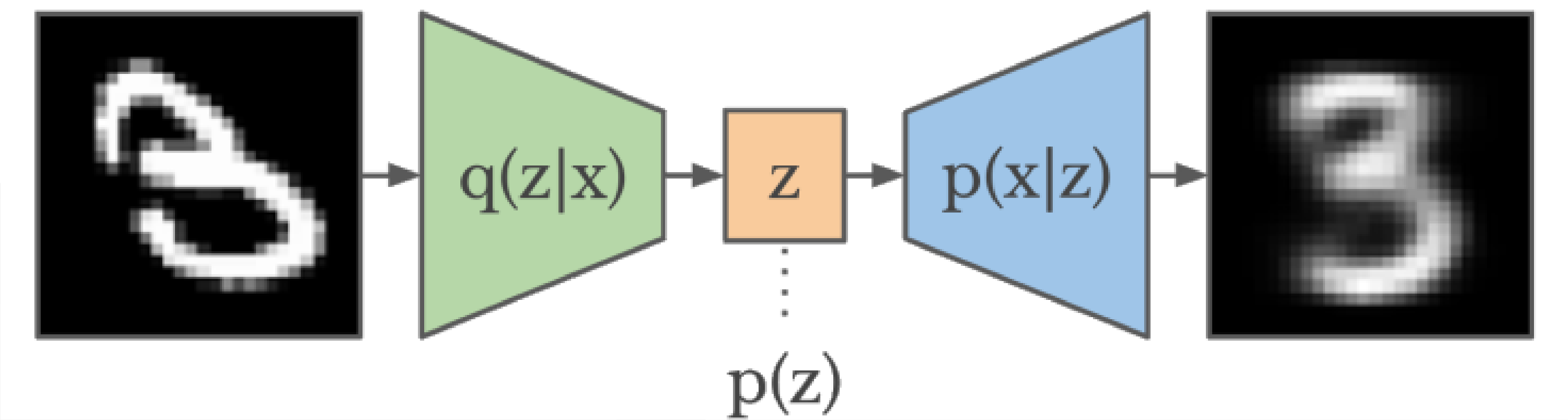
$$q_\phi(z|x) = \mathcal{N}(z; \mu, \mathrm{diag}(\exp(\ell)))$$

with an encoder network

$$(\mu, \ell) = e_\phi(x)$$

Illustration

The illustration of a VAE



Evidence Lower Bound

Starting from the evidence $\log p_\theta(x)$

$$\log p_\theta(x) = \log\{\int p_\theta(x,z)dz\} = \log\{\int q_\phi(z|x)\frac{p_\theta(x,z)}{q_\phi(z|x)}dz\}$$

With the Jensen's inequality, we have

$$\log p_\theta(x) \geq \int q_\phi(z|x)\log\frac{p_\theta(x,z)}{q_\phi(z|x)}dz = \int q_\phi(z|x)\log\frac{p_\theta(x|z)p_\theta(z)}{q_\phi(z|x)}dz$$

Therefore,

$$\log p_\theta(x) \geq E_q[\log p_\theta(x|z)] - \mathrm{KL}[q_\phi(z|x)\|p_\theta(z)]$$

Evidence Lower Bound (II)

Given x

$$\log p_\theta(x) \geq E_q[\log p_\theta(x|z)] - \mathrm{KL}[q_\phi(z|x) \| p_\theta(z)]$$

- $E_q[\log p_\theta(x|z)]$: reconstruction loss
- $\mathrm{KL}[q_\phi(z|x) \| p_\theta(z)]$: similarity between the variational distribution and the prior

Evaluating the ELBo

If both $q_\phi(z|x)$ and $p_\theta(z)$ are Gaussian distributions

- There is a closed-form solution for $\mathrm{KL}[q_\phi(z|x) \| p_\theta(z)]$
- $E_q[\log p_\theta(x|z)]$ is intractable, and can only be approximated with Monte Carlo methods

$$E_q[\log p_\theta(x, z)] \approx \frac{1}{S} \sum_{s=1}^{S} \log p_\theta(x|z_s)$$

where $z_s \sim q_\phi(z|x)$

Learning VAE (Conceptually)

Conceptually, learning VAE is basically a variational EM algorithm, iterating between θ and ϕ with the following objective

$$E_q[\log p_\theta(x|z)] - \mathrm{KL}[q_\phi(z|x) \| p_\theta(z)]$$

- Update θ: update the decoder to have a better generation model
- Update ϕ: update the encoder to have an informative latent space

Learning VAE (In Practice)

The reparameterization trick: for a Gaussian random variable z, we can reformulate the sampling

$$z \sim q_\phi(z|x) = \mathcal{N}(z; \mu(x), \sigma^2(x))$$

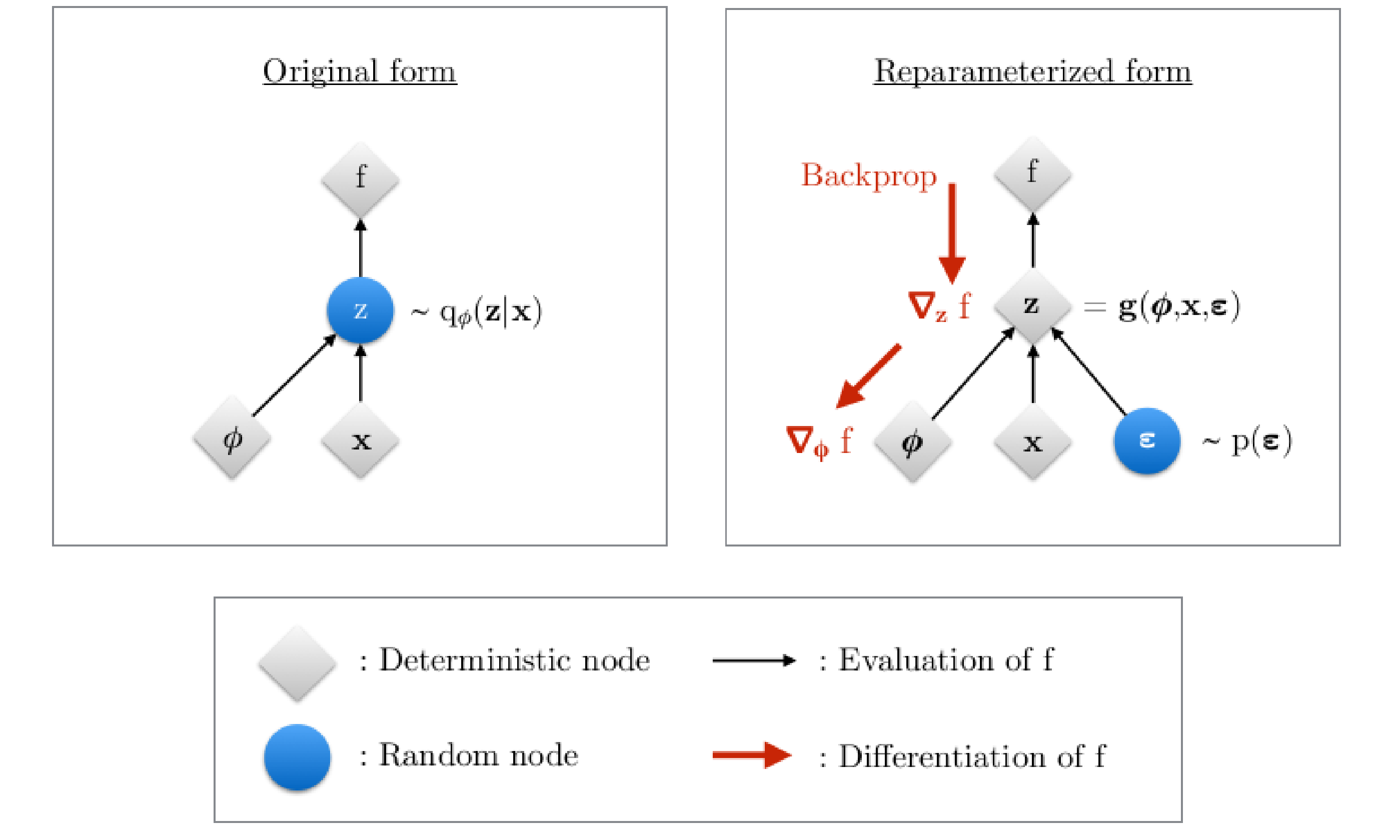
as

$$z = \mu(x) + \sigma(x) \cdot \varepsilon$$

where $\varepsilon \sim \mathcal{N}(0, I)$

Reparameterization Trick

It reduce the randomness in the back-propagation algorithm



Training VAE with Mini-batches

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

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

[Kingma and Welling, 2014]

Comparison: Reconstruction

Figure 21.4: Illustration of image reconstruction using (V)AEs trained and applied to CelebA. Row 1: Original images. Row 2: Deterministic autoencoder. Row 3: β-VAE with $\beta = 0.5$. Row 4: VAE (with $\beta = 1$). Generated by celeba_vae_ae_comparison.ipynb.

Comparison: Generation

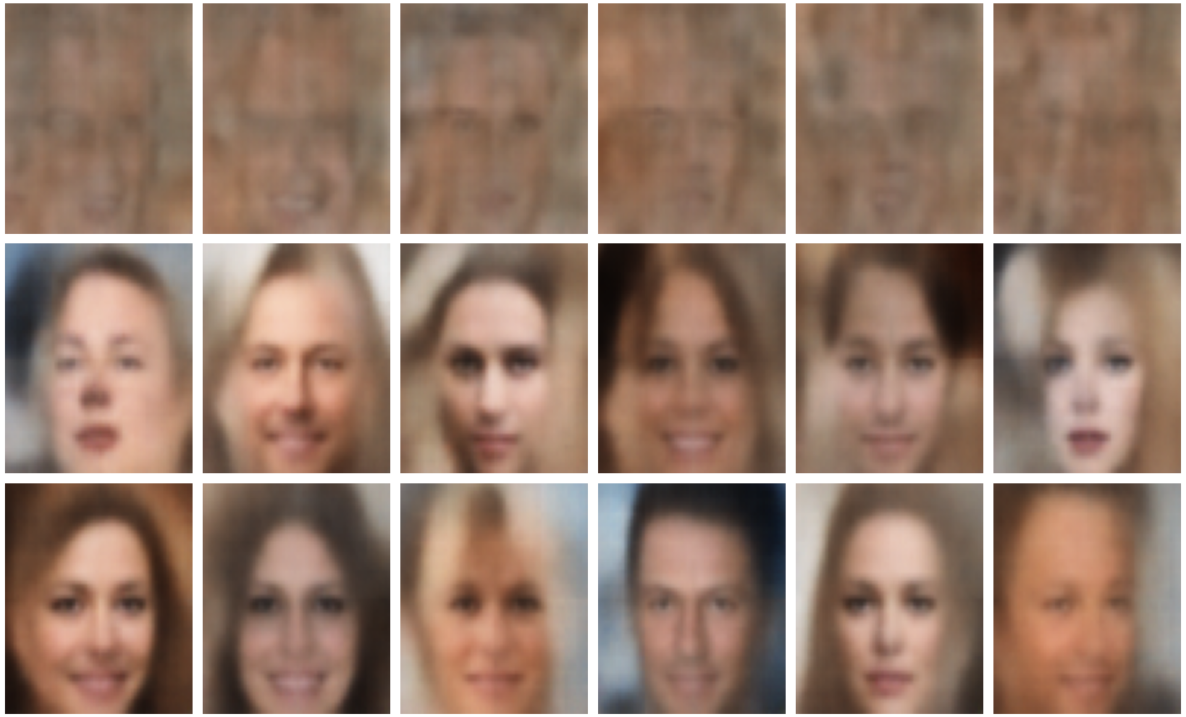


Figure 21.3: Illustration of unconditional image generation using (V)AEs trained on CelebA. Row 1: Deterministic autoencoder. Row 2: β-VAE with $\beta = 0.5$. Row 3: VAE (with $\beta = 1$). Generated by celeba_vae_ae_comparison.ipynb.

Theoretical and Empirical Analysis

β-VAE

By relaxing the original objective function, we can get a generalized version of VAE called β-VAE

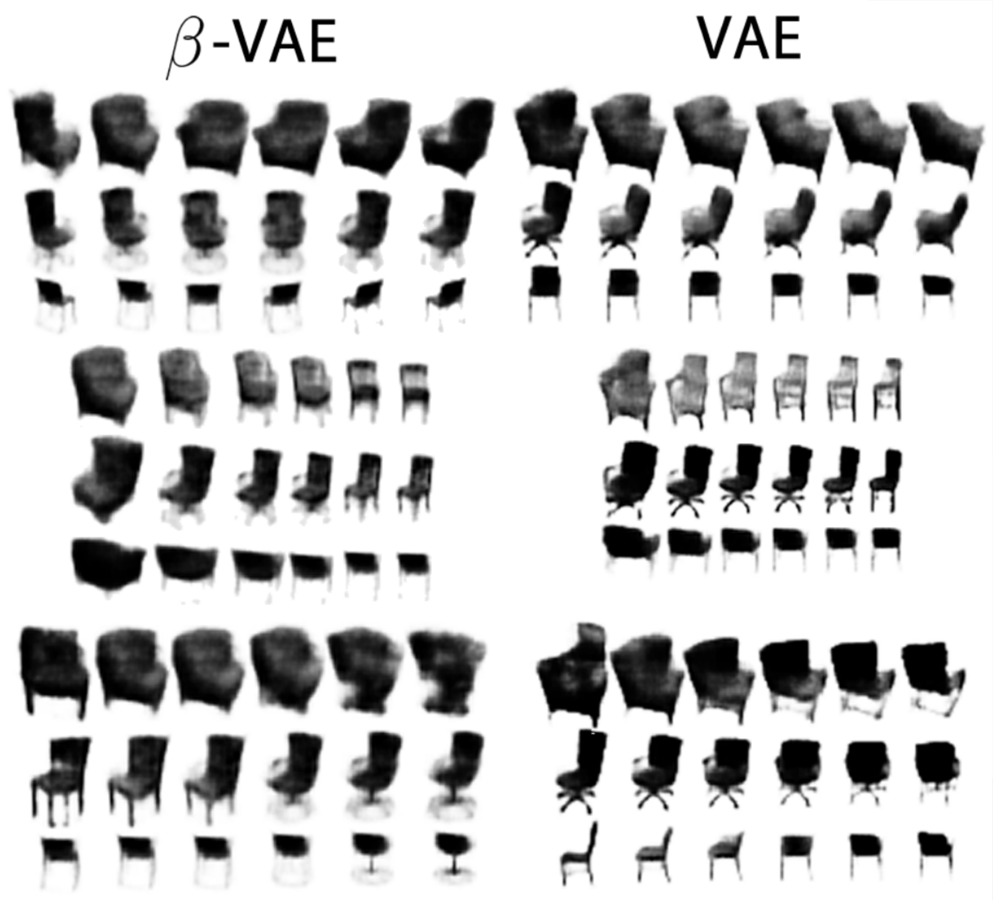
$$E_q[\log p_\theta(x|z)] - \beta \cdot \mathrm{KL}[q_\phi(z|x) \| p_\theta(z)]$$

- $\beta = 1.0$: standard VAE
- $\beta \geq 1.0$: forcing each $q_\phi(z|x)$ to be similar to $p_\theta(z)$
  - Furthermore, defining

$$q_\phi(z|x) = \prod_{k=1}^{K} q_\phi(z_k|x)$$

then minimizing the KL term will lead to *disentangled* representations

Examples



[Higgins et al., 2017]

Conceptual Framework

Consider the following lower bound

$$\log p_\theta(x) \geq E_q[\log p_\theta(x|z)] - \beta \mathrm{KL}[q_\phi(z|x) \| p_\theta(z)]$$

Calculating the integral of x on both side, we have

$$-\int p_\theta(x) \log p_\theta(x) dx \leq -\int p_\theta(x) E_q[\log p_\theta(x|z)] dx + \beta \int \mathrm{KL}[q_\phi(z|x) \| p_\theta(z)] dx$$

Rewrite it as

$$H \leq D + R$$

Conceptual Framework (II)

- Data entropy: the intrinsic data uncertainty

$$H = -\int p_\theta(x) \log p_\theta(x) dx$$

- Distortion: the reconstruction loss by using the approximation encoder $q_\phi(z|x)$

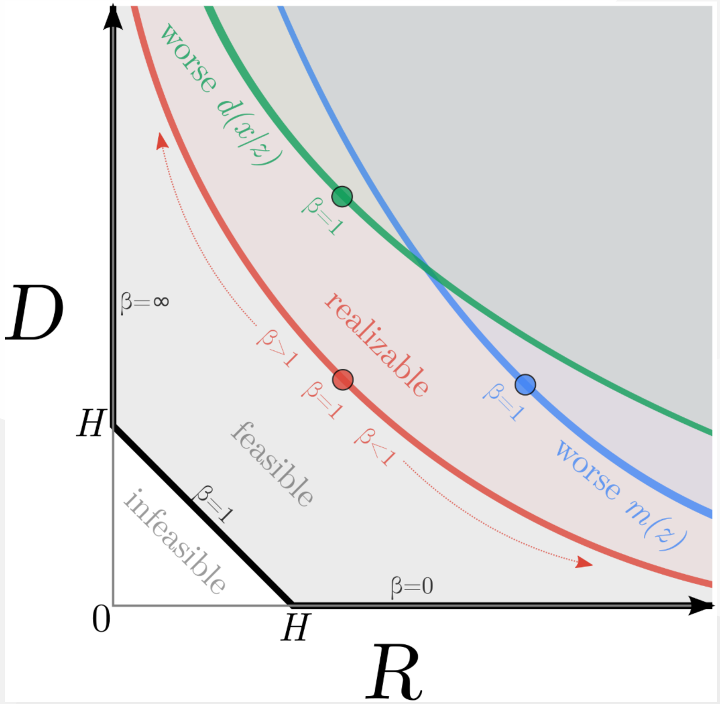
$$D = -\int p_\theta(x) E_q[\log p_\theta(x|z)] dx$$

- Rate: the average KL divergence

$$R = \int \mathrm{KL}[q_\phi(z|x) \| p_\theta(z)] dx$$

The RD Plane

In the following figure, consider $m(z)$ as $p_\theta(z)$ and $d(x|z)$ as $p_\theta(x|z)$



Different distributions can give the same lower bound

Case Study: About Disentangled Representations

Theorem 1. *For $d > 1$, let $\mathbf{z} \sim P$ denote any distribution which admits a density $p(\mathbf{z}) = \prod_{i=1}^{d} p(\mathrm{z}_i)$. Then, there exists an infinite family of bijective functions $f : \mathrm{supp}(\mathbf{z}) \to \mathrm{supp}(\mathbf{z})$ such that $\frac{\partial f_i(\boldsymbol{u})}{\partial u_j} \neq 0$ almost everywhere for all i and j (i.e., $\mathbf{z}$ and $f(\mathbf{z})$ are completely entangled) and $P(\mathbf{z} \leq \boldsymbol{u}) = P(f(\mathbf{z}) \leq \boldsymbol{u})$ for all $\boldsymbol{u} \in \mathrm{supp}(\mathbf{z})$ (i.e., they have the same marginal distribution).*

[Locatello et al., 2019]

Thank You!