

VAE

Outline

- Variational Inference
- Variational Autoencoder
- Experiments

VARIATIONAL INFERENCE

Variational Inference

- Problem Definition

- Observable Data: $\mathbf{x} = \{x_1, x_2, \dots, x_m\}$

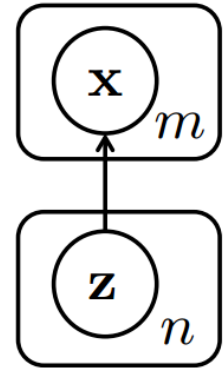
- Hidden Variable: $\mathbf{z} = \{z_1, z_2, \dots, z_n\}$

- Posterior Distribution of hidden variable given some data:

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{z}, \mathbf{x})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{\int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}}$$



Intractable to compute



Variational Inference

- Approximate $p(\mathbf{z}|\mathbf{x})$ by $q(\mathbf{z})$
- Minimize the KL Divergence:

$$D_{KL}[q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})] = \int q(\mathbf{z}) \log \frac{q(\mathbf{z})}{p(\mathbf{z}|\mathbf{x})} d\mathbf{z}$$

Evidence Lower Bound

$$\begin{aligned}D_{KL}[q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})] &= \int q(\mathbf{z})\log\frac{q(\mathbf{z})}{p(\mathbf{z}|\mathbf{x})}d\mathbf{z} \\&= \int q(\mathbf{z})\log\frac{q(\mathbf{z})p(\mathbf{x})}{p(\mathbf{z},\mathbf{x})}d\mathbf{z} \\&= \int q(\mathbf{z})\log\frac{q(\mathbf{z})}{p(\mathbf{z},\mathbf{x})}d\mathbf{z} + \int q(\mathbf{z})\log p(\mathbf{x})d\mathbf{z} \\&= \int q(\mathbf{z})(\log q(\mathbf{z}) - \log p(\mathbf{z},\mathbf{x}))d\mathbf{z} + \log p(\mathbf{x}) \\&= \underbrace{-(E_{q(\mathbf{z})}[\log p(\mathbf{z},\mathbf{x})] - E_{q(\mathbf{z})}[\log q(\mathbf{z})])}_{L[q(\mathbf{z})]} + \log p(\mathbf{x})\end{aligned}$$

Evidence Lower Bound

$$D_{KL}[q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})] = -L[q(\mathbf{z})] + \log p(\mathbf{x})$$

$$\log p(\mathbf{x}) = D_{KL}[q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})] + L[q(\mathbf{z})]$$

Minimize $D_{KL}[q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})]$

is equal to Maximize $L[q(\mathbf{z})]$

VARIATIONAL AUTOENCODER

Introduction

- Generative modeling
 - Producing more examples that are *like* those already in a database, e.g.
 - Take in a database of 3D models of something like plants and produce more of them to fill a forest in a video game
 - Take handwritten text and try to produce more handwritten text
 - Get examples X distributed according to some unknown distribution $P_{gt}(X)$, and our goal is to learn a model P which we can sample from, such that P is as similar as possible to P_{gt}

Variational Autoencoder

- Observable Data: $\mathbf{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}\}$
- Assume that the data are generated by some random process, involving an unobserved continuous random variable \mathbf{z}
- The process consists of two steps:
 - 1. $\mathbf{z} \sim p_{\theta}(\mathbf{z})$
 - 2. $\mathbf{x} | \mathbf{z} \sim p_{\theta}(\mathbf{x} | \mathbf{z})$
- Introduce a recognition model $q_{\phi}(\mathbf{z} | \mathbf{x})$: an approximation to the intractable true posterior $p_{\theta}(\mathbf{z} | \mathbf{x})$

Variational Autoencoder

Marginal Likelihood: $\log p(\mathbf{x}) = D_{KL}[q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})] + L[q(\mathbf{z})]$

$$\log p_{\theta}(\mathbf{x}) = D_{KL}[q_{\phi}(\mathbf{z}|\mathbf{x})||p_{\theta}(\mathbf{z}|\mathbf{x})] + \mathcal{L}(\theta, \phi, \mathbf{x})$$

Variational Lower Bound:

$$L(\theta, \phi, \mathbf{x}) = E_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log p_{\theta}(\mathbf{x}, \mathbf{z}) - \log q_{\phi}(\mathbf{z}|\mathbf{x})]$$

$$= E_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log p_{\theta}(\mathbf{z}) + \log p_{\theta}(\mathbf{x}|\mathbf{z}) - \log q_{\phi}(\mathbf{z}|\mathbf{x})]$$

$$= E_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log \frac{p_{\theta}(\mathbf{z})}{q_{\phi}(\mathbf{z}|\mathbf{x})} + \log p_{\theta}(\mathbf{x}|\mathbf{z})]$$

$$= -D_{KL}[q_{\phi}(\mathbf{z}|\mathbf{x})||p_{\theta}(\mathbf{z})] + E_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log p_{\theta}(\mathbf{x}|\mathbf{z})]$$

Monte Carlo Gradient Estimator

Gradient of $\mathcal{L}(\theta, \phi, \mathbf{x})$ contains $\nabla_{\phi} E_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log p_{\theta}(\mathbf{x}|\mathbf{z})]$
which is Intractable

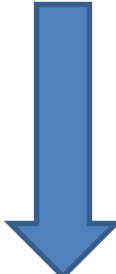
Use **Monte Carlo Gradient Estimator** :

$$\begin{aligned}\nabla_{\phi} E_{q_{\phi}(\mathbf{z})}[f(\mathbf{z})] &= \nabla_{\phi} \int q_{\phi}(\mathbf{z}) f(\mathbf{z}) d\mathbf{z} \\&= \int q_{\phi}(\mathbf{z}) f(\mathbf{z}) \frac{\nabla_{\phi} q_{\phi}(\mathbf{z})}{q_{\phi}(\mathbf{z})} d\mathbf{z} = \int q_{\phi}(\mathbf{z}) f(\mathbf{z}) \nabla_{\phi} \log q_{\phi}(\mathbf{z}) d\mathbf{z} \\&= E_{q_{\phi}(\mathbf{z})}[f(\mathbf{z}) \nabla_{\phi} \log q_{\phi}(\mathbf{z})] \\&\approx \frac{1}{L} \sum_{l=1}^L f(\mathbf{z}) \nabla_{\phi} \log q_{\phi}(\mathbf{z}^{(l)}) \quad \text{where } \mathbf{z}^{(l)} \sim q_{\phi}(\mathbf{z})\end{aligned}$$

Objective Function

$$L(\theta, \phi, \mathbf{x}^{(i)}) = -D_{KL}[q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z})] + E_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})]$$

Monte Carlo Gradient Estimator


$$\tilde{L}(\theta, \phi, \mathbf{x}^{(i)}) = -D_{KL}[q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z})] + \frac{1}{L} \sum_{l=1}^L \log p_\theta(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)})$$

Given multiple datapoints from a dataset \mathbf{X} with N datapoints, we can construct an estimator of the marginal likelihood lower bound of the full dataset, based on minibatches:

$$\mathcal{L}(\theta, \phi; \mathbf{X}) \simeq \tilde{\mathcal{L}}^M(\theta, \phi; \mathbf{X}^M) = \frac{N}{M} \sum_{i=1}^M \tilde{\mathcal{L}}(\theta, \phi; \mathbf{x}^{(i)})$$

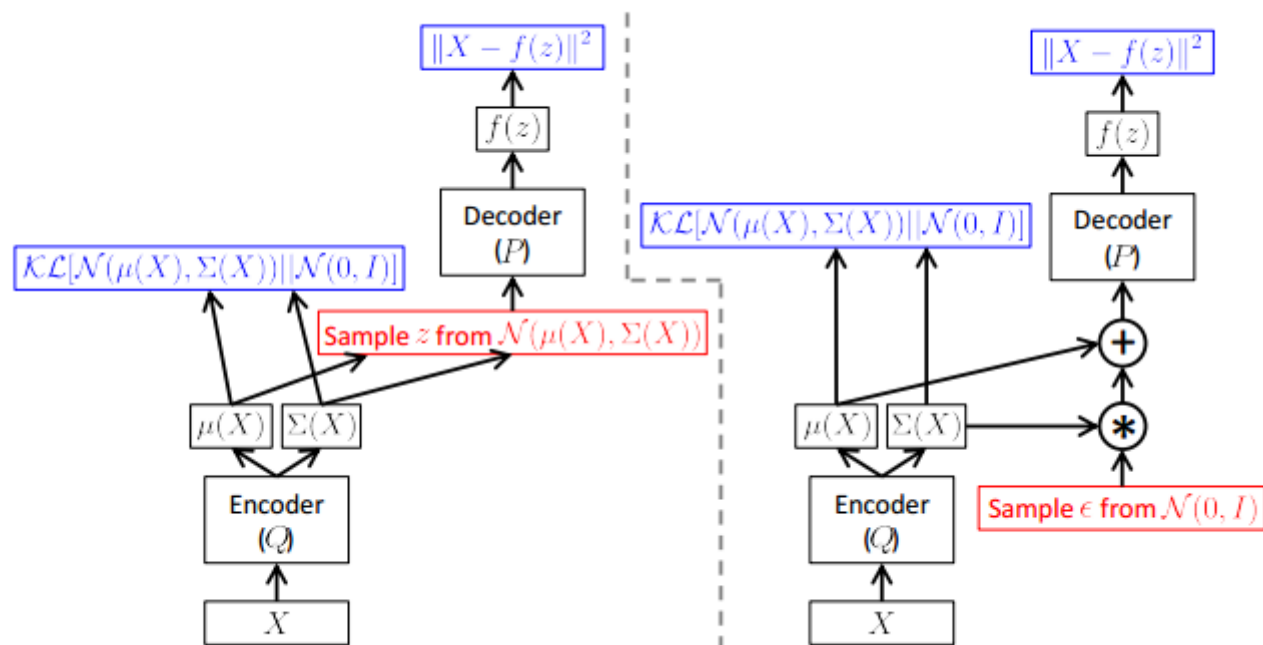
Reparameterization Trick

$$\mathbf{z} \sim q_{\phi}(\mathbf{z}|\mathbf{x}) \quad \Rightarrow \quad \begin{array}{l} \text{auxiliary variable} \\ \epsilon \sim p(\epsilon) \\ \text{deterministic variable} \\ \mathbf{z} = g_{\phi}(\epsilon, \mathbf{x}) \end{array}$$

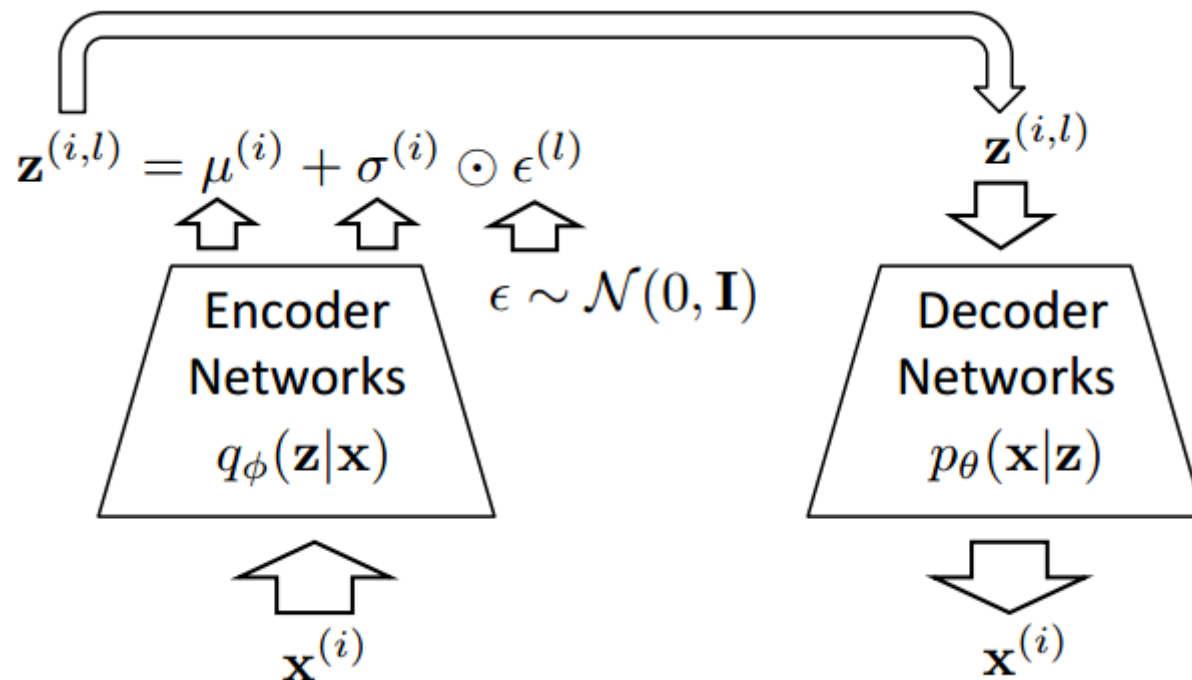
Example:

$$z \sim p(z|x) = \mathcal{N}(\mu, \sigma^2) \quad \Rightarrow \quad \begin{array}{l} \epsilon \sim \mathcal{N}(0, 1) \\ z = \mu + \sigma\epsilon \end{array}$$

Reparameterization Trick



Reparameterization Trick

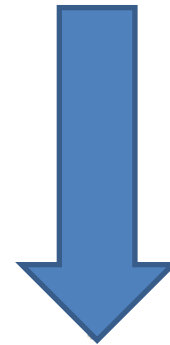


VAE

$$\tilde{L}(\theta, \phi, \mathbf{x}^{(i)}) = -D_{KL}[q_{\phi}(\mathbf{z}|\mathbf{x}^{(i)})||p_{\theta}(\mathbf{z})] + \frac{1}{L} \sum_{l=1}^L \log p_{\theta}(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)})$$

$$q_{\phi}(\mathbf{z}|\mathbf{x}^{(i)}) = \mathcal{N}(\mathbf{z}, \mu^{(i)}, \sigma^{2(i)}\mathbf{I})$$

$$p_{\theta}(\mathbf{z}) = \mathcal{N}(\mathbf{z}, 0, \mathbf{I})$$



$$\tilde{L}(\theta, \phi, \mathbf{x}^{(i)}) = \frac{1}{2} \sum_{j=1}^J (1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2) + \frac{1}{L} \sum_{l=1}^L (\log p_{\theta}(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}))$$

Regularization

Reconstruction
Error

Let J be the dimensionality of \mathbf{z}

Training

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.

$\theta, \phi \leftarrow$ Initialize parameters

repeat

$\mathbf{X}^M \leftarrow$ Random minibatch of M datapoints (drawn from full dataset)

$\epsilon \leftarrow$ Random samples from noise distribution $p(\epsilon)$

$\mathbf{g} \leftarrow \nabla_{\theta, \phi} \tilde{\mathcal{L}}^M(\theta, \phi; \mathbf{X}^M, \epsilon)$ (Gradients of minibatch estimator (8))

$\theta, \phi \leftarrow$ Update parameters using gradients \mathbf{g} (e.g. SGD or Adagrad [DHS10])

until convergence of parameters (θ, ϕ)

return θ, ϕ

EXPERIMENTS

Experiments

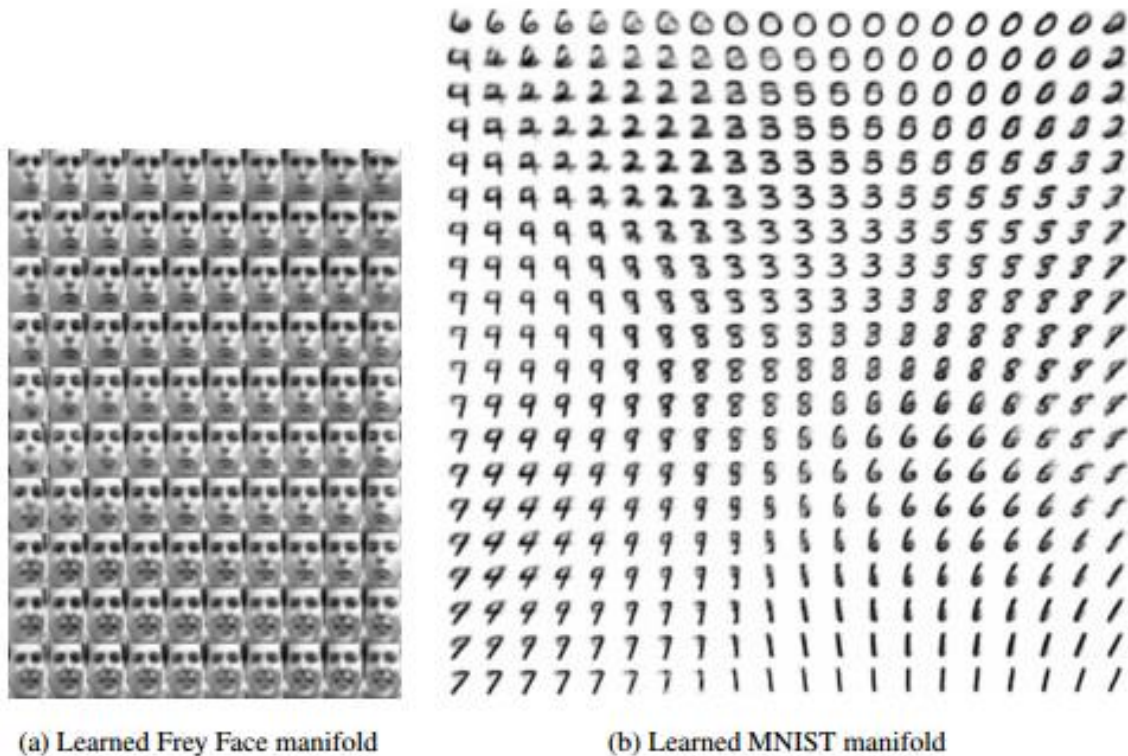


Figure 4: Visualisations of learned data manifold for generative models with two-dimensional latent space, learned with AEVB. Since the prior of the latent space is Gaussian, linearly spaced coordinates on the unit square were transformed through the inverse CDF of the Gaussian to produce values of the latent variables \mathbf{z} . For each of these values \mathbf{z} , we plotted the corresponding generative $p_{\theta}(\mathbf{x}|\mathbf{z})$ with the learned parameters θ .

Experiments

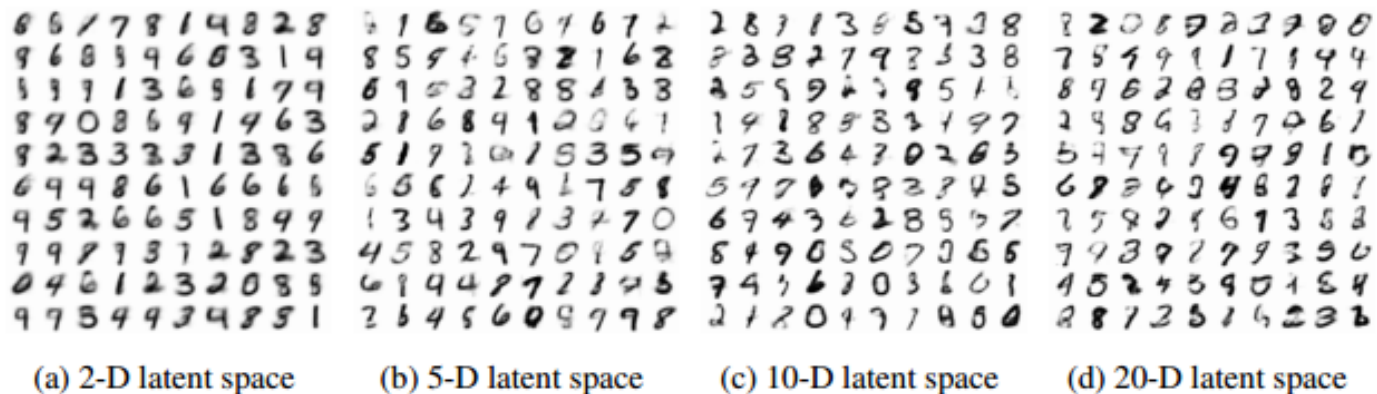


Figure 5: Random samples from learned generative models of MNIST for different dimensionalities of latent space.