Autoencoder and Variational Autoencoder

Group Meeting

Reporter: Liang Cao

2020.06.17

Introduction

- ✓ An autoencoder neural network is an unsupervised learning algorithm that applies back-propagation, setting the outputs to be equal to the inputs.

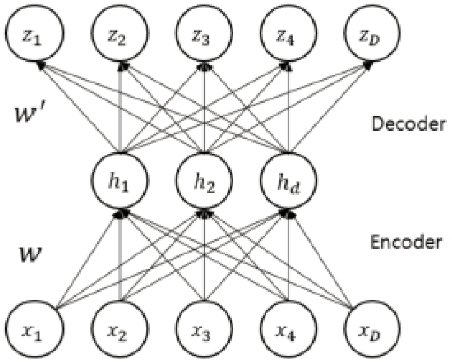
- ✓ Typically for the purpose of dimensionality reduction; was firstly introduced as a way of conducting pretraining in ANNs.

- ✓ The learned hidden layers are called the representation(encoding) of the input data.

Introduction

✓ Unsupervised learning: automatically extract meaningful features for you data; enhance the availability of unlabeled data.

- Autoencoder is a feedforward neural network trained to reproduce its input at the output layer: $\mathbf{z} \approx \mathbf{x}$



- Encoder:

$$\mathbf{h} = f_\theta(\mathbf{x}) = a(\mathbf{W}\mathbf{x} + \mathbf{b})$$

- Decoder:

$$\mathbf{z} = g_{\theta'}(\mathbf{h}) = a(\mathbf{W}'\mathbf{h} + \mathbf{b}')$$

$$\theta, \theta' : \{\mathbf{W}, \mathbf{b}\}, \{\mathbf{W}', \mathbf{b}'\}$$

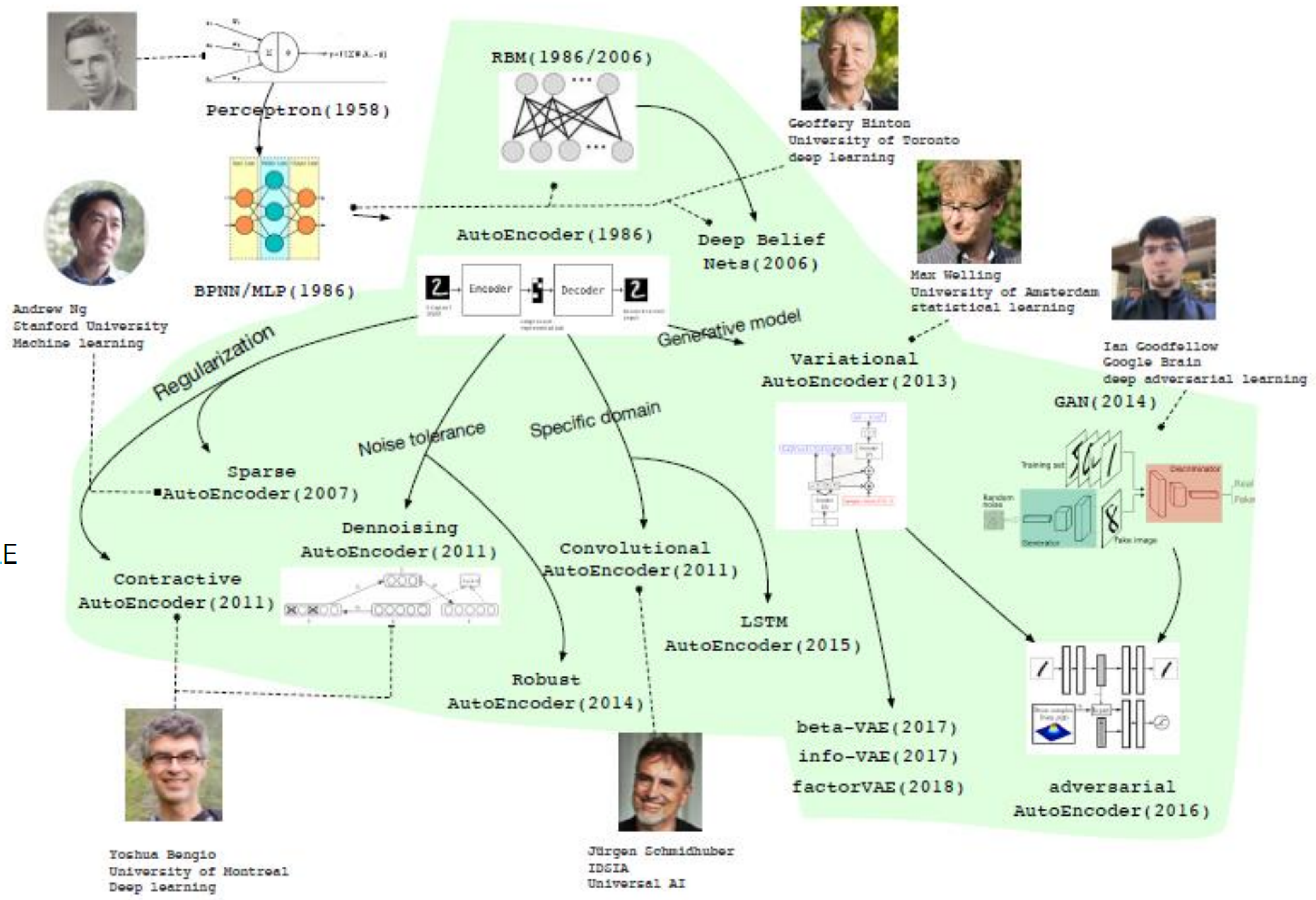
- Parameter estimation:

$$\theta^*, \theta'^* = \underset{\theta,\theta'}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L\left(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}\right)$$
$$= \underset{\theta,\theta'}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L\left(\mathbf{x}^{(i)}, g_{\theta'}(f_\theta(\mathbf{x}^{(i)}))\right)$$

where L is a loss function such as *squared error* $L(\mathbf{x}, \mathbf{z}) = ||\mathbf{x} - \mathbf{z}||^2$.

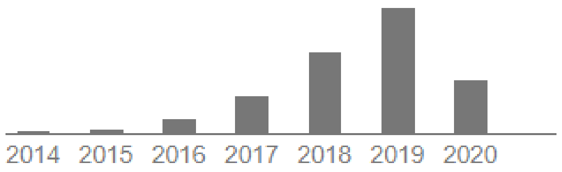
Introduction

- What is a good representation ?
  - Sparsity? ⇒ Sparse AE
  - Denoise? ⇒ Denoising AE
  - Robust to disturbance? ⇒ contractive AE
  - High level? ⇒ Stacked AE
  - Specific data (image) structure? Convolutional AE
  - Generate new data? ⇒ Variational AE



VAE

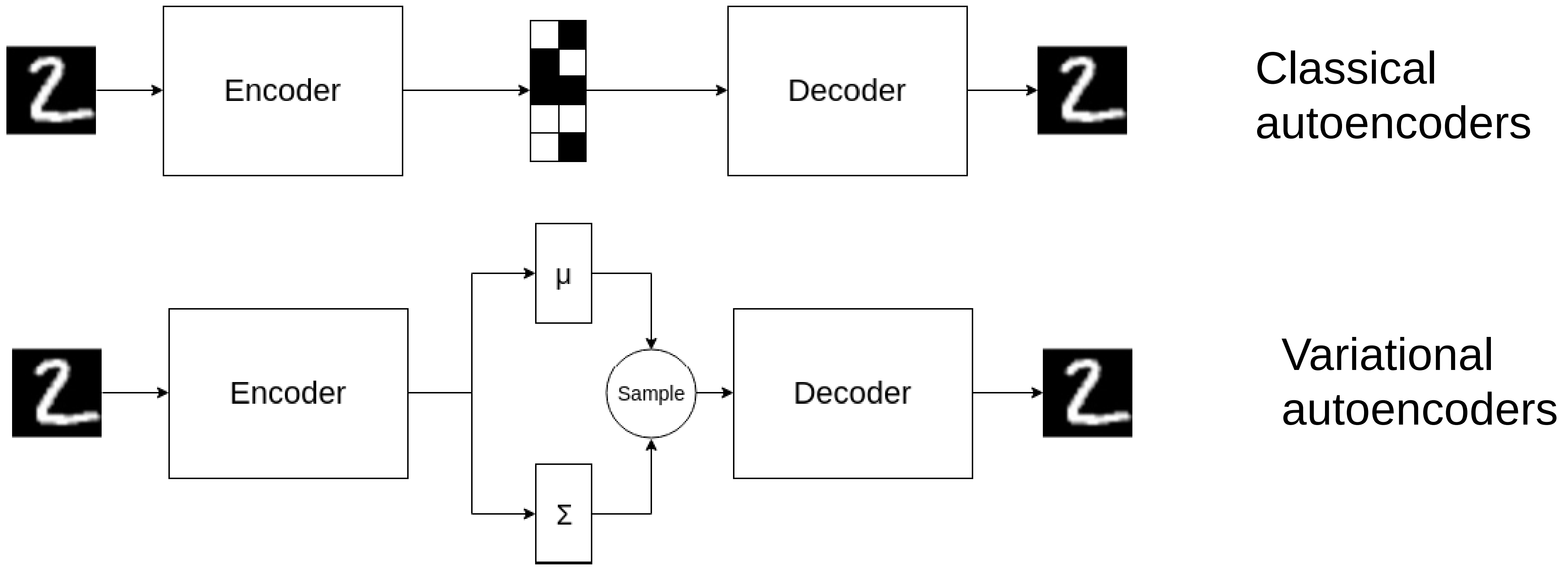
Diederik P Kingma, Max Welling. Universiteit van Amsterdam. **Auto-Encoding Variational Bayes**. December, 2013



Number of citations: 9201

- ✓ VAE is a generative model, we can use generative model to only learn the distribution of our data p(x).
- ✓ After training we can generate new data similar to x.
- ✓ Generated data instances should come from points with high probability in datasets distribution space.

VAE



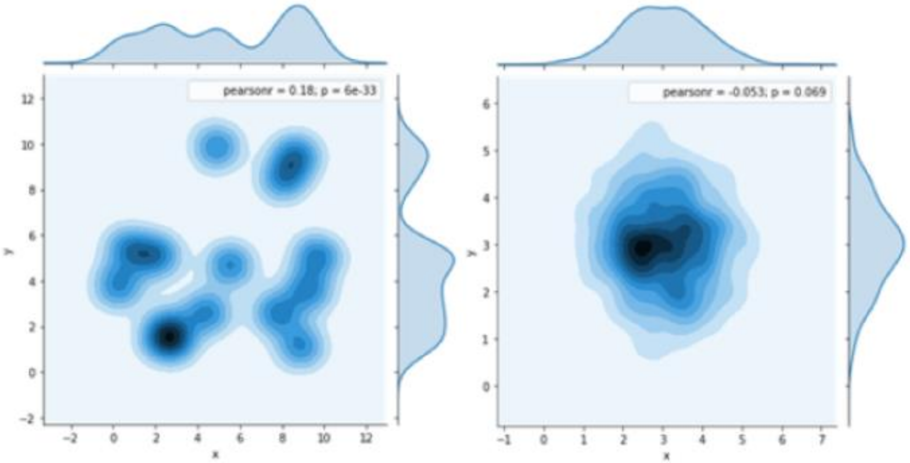
1. Classical autoencoders minimize a reconstruction loss $\|x-\hat{x}\|^2$, they are unregularized in latent space.

2. VAEs are one approach to regularizing (impose structure) the latent distribution.

VAE

Classical autoencoders minimize a reconstruction loss $\|x - \hat{x}\|^2$, they are unregularized in latent space.

- Cons
  - ✓ This yields an unstructured latent space.
  - ✓ Examples from the data distribution are mapped to codes scattered in the space.
  - ✓ No constraint that similar inputs are mapped to nearby points in the latent space
  - ✓ We cannot sample codes to generate novel examples.



On the left the Conventional AE latent space and on the right VAEs latent space

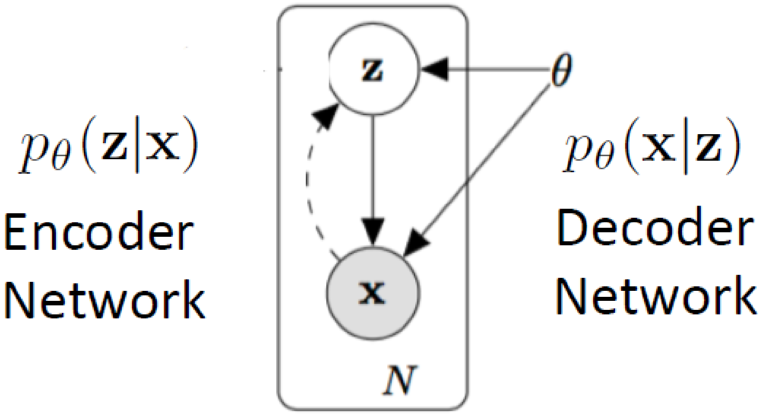
VAE

Problem definition

- ✓ Observable data: $X = \{x_1, x_2, x_3, \cdots, x_n\}$
- ✓ Hidden variable: $Z = \{z_1, z_2, z_3, \cdots, z_m\}$

In encoder network, we need to do inference (calculate the posterior):

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$



Need to calculate evidence: $p(x) = \int p(x|z)p(z)dz$

For higher dimensional latent variable: $p(x) = \int\int\int \dots \int p(x|z_i)p(z_i)dz_1dz_2\cdots dz_m$

$p(x)$ is intractable, which leads to $p(z|x)$ intractable.

VAE

$p(z|x)$ intractable.

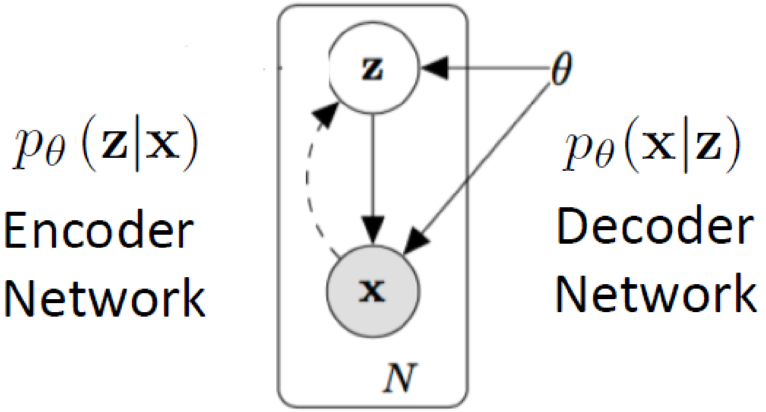
Solutions for intractable posterior: approximate inference

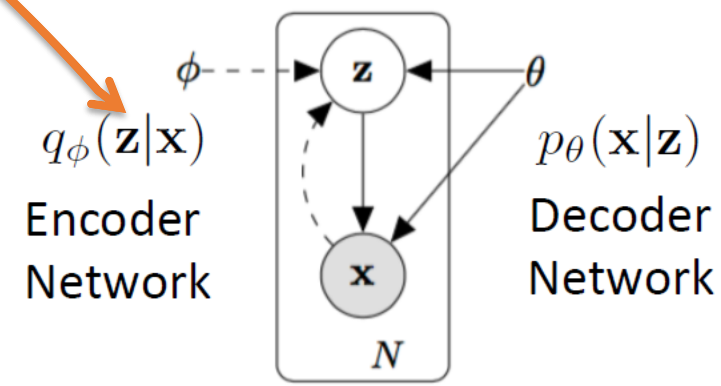
- Deterministic approximation (variational inference)
  - Approximate p with "closest" distribution q from a tractable family,

$$p(z|x) \approx q(z|x)$$

  - Turns inference into optimization (need to find best q).
    - Called variational Bayes.

- Stochastic approximation (Markov Chain Monte Carlo)
  - Approximate p with empirical distribution over samples,
  - Turns inference into sampling.





VAE

$p(z|x)$ intractable.

Solutions for intractable posterior: approximate inference

- Deterministic approximation (variational inference)
  - Approximate p with "closest" distribution q from a tractable family,

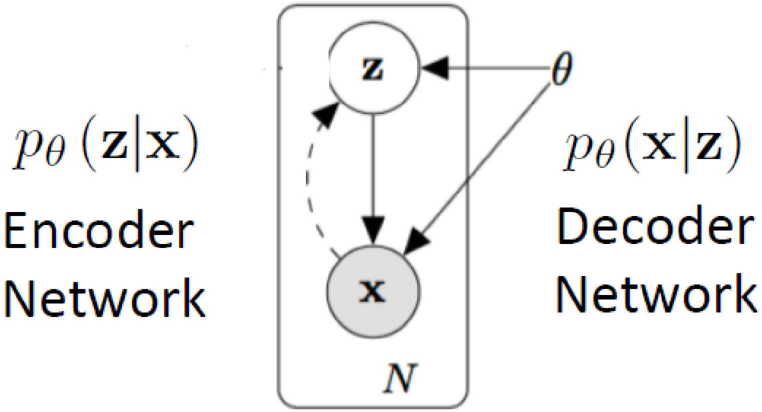
$$p(z|x) \approx q(z|x)$$

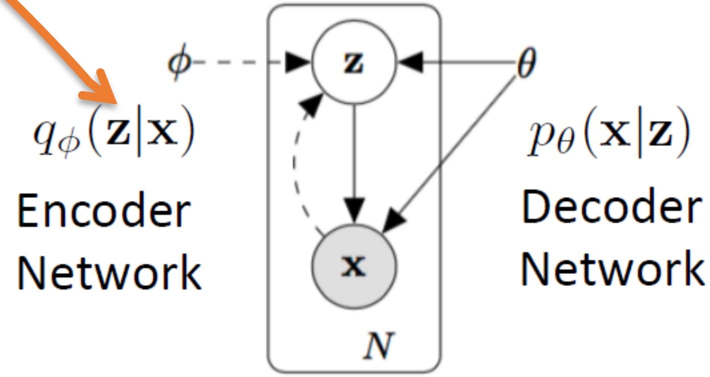
  - Turns inference into optimization (need to find best q).
    - Called variational Bayes.

Why not use MCMC?

- MCMC works asymptotically, but may take forever.
- Variational methods not consistent, but very fast.

(trade off accuracy vs. computation)

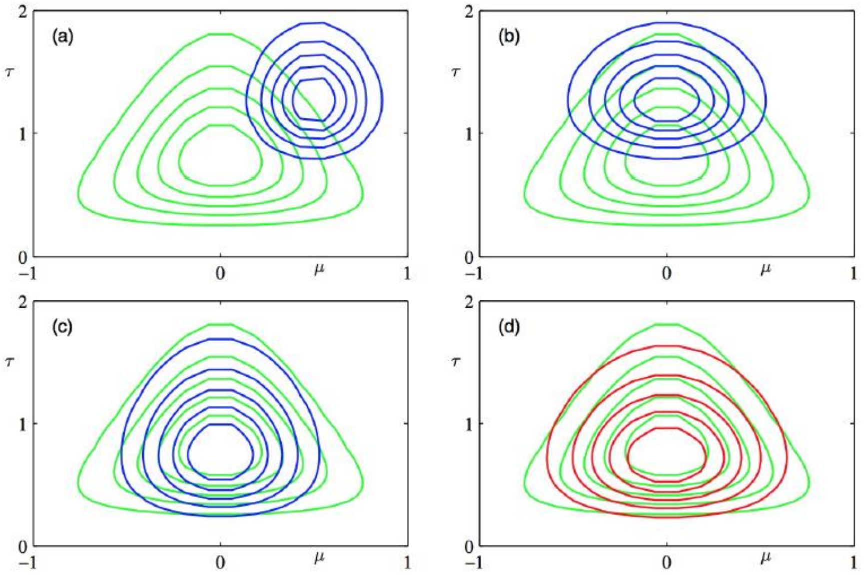




VAE

- Deterministic approximation (variational inference) $p(z|x) \approx q(z|x)$

  - ✓ The main idea behind variational inference is to, first pick a tractable family of distributions over the latent variables with initial variational parameters.
  - ✓ Then to find parameters that make it as close as possible to the true posterior.
  - ✓ KL divergence measures information lost when using $q_\phi(z|x)$ to approximate $p_\theta(z|x)$
  - ✓ We want to choose ϕ to minimize $D_{KL}\left(q_\phi(z|x) \| p_\theta(z|x)\right)$



VAE

- KL divergence

Used to measure similarity between two probability distributions(w.r.t. one of them)

Discrete and continuous form:

- $D_{KL}(p\|q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$
- $D_{KL}(p\|q) = \int_x p(x) \log \frac{p(x)}{q(x)}\, dx$

Properties:

- $D_{KL}(p\|q) \geq 0, \forall p, q$
- $D_{KL}(p\|q) = 0 \iff p = q$
- $D_{KL}(p\|q) \neq D_{KL}(q\|p)$ in general

VAE

- Variational Lower Bound

$$
\begin{aligned}
D_{KL}(q(z \mid x) \| p(z \mid x)) &= \int_z q(z \mid x) \log \frac{q(z \mid x)}{p(z \mid x)} \\
&= -\int_z q(z \mid x) \log \frac{p(z \mid x)}{q(z \mid x)} \\
&= -\left[\int_z q(z \mid x) \log \frac{p(x, z)}{q(z \mid x)} - \int_z q(z \mid x) \log p(x) \right] \\
&= -\int_z q(z \mid x) \log \frac{p(x, z)}{q(z \mid x)} + \log p(x) \int_z q(z \mid x) \\
&= -\mathcal{L} + \log p(x)
\end{aligned}
$$

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

- ✓ Still contains p(x) term! So cannot compute directly.
- ✓ But p(x) does not depend on parameter ϕ in $q_\phi(z|x)$ p(x) is a constant for different ϕ , so still hope.
- ✓ Define L as variational lower bound.

$$\log p(x) = \mathcal{L} + D_{KL}(q(z \mid x) \| p(z \mid x))$$

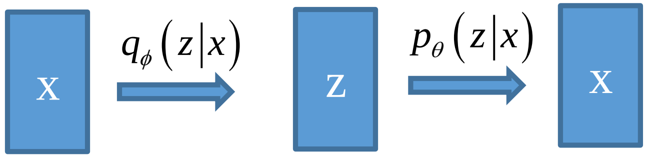
- ✓ Minimizing the KL divergence is equal to maximizing variational lower bound L.

VAE

- Variational Lower Bound

$$
\begin{aligned}
\mathcal{L} &= \int_z q(z \mid x) \log \frac{p(x,z)}{q(z \mid x)} \\
&= \int_z q(z \mid x) \log \frac{p(x \mid z)p(z)}{q(z \mid x)} \\
&= \int_z q(z \mid x) \log p(x \mid z) + \int_z q(z \mid x) \log \frac{p(z)}{q(z \mid x)}
\end{aligned}
$$

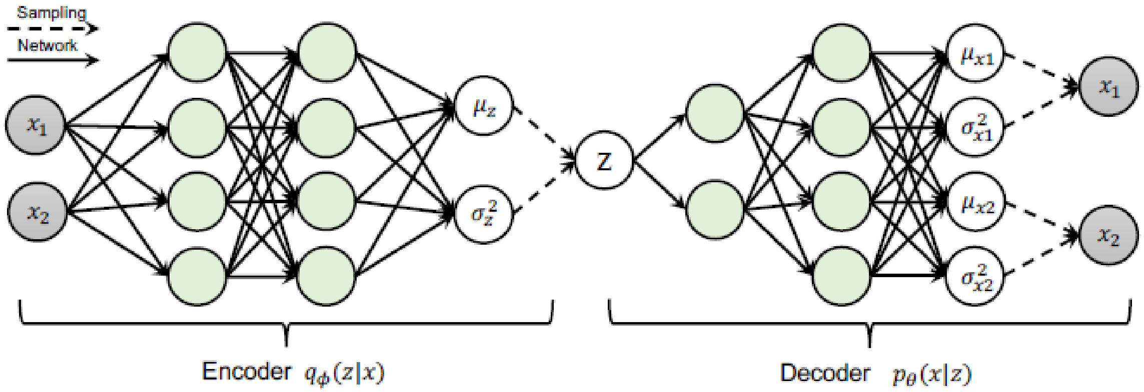
$$
\boldsymbol{\mathcal{L} = \mathbb{E}_{q(z|x)} \log p(x \mid z) - D_{KL}(q(z \parallel x) \| p(z))}
$$



The first term is conceptually the negative reconstruction error and the second is regularize, makes approximate inference term close to the prior p(z), the p(z) is usually chosen as standard normal distribution.

VAE

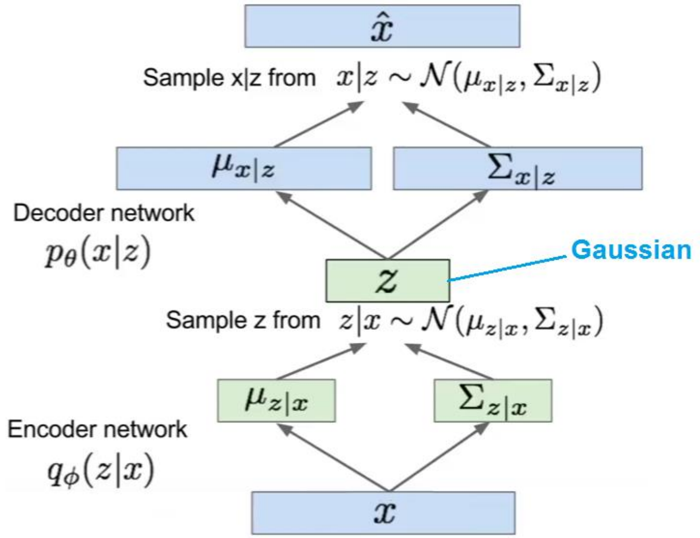
Deep Learning Perspective



- Encoder and decoder networks also called "recognition" / "inference" and "generation" networks
- We aim to learn the parameters ϕ and θ via backpropagation.

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

Probabilistic Model Perspective

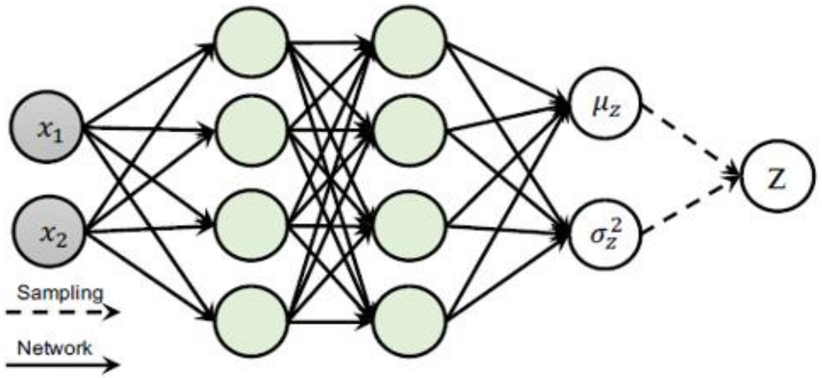


VAE

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

- Use $\mathcal{N}(0,1)$ as prior for z; $q_\phi(z|x)$ is Gaussian distribution $\mathcal{N}(\mu_z(x;\phi), \sigma_z^2(x;\phi))$ determined by NN.
  - The KL-divergence:

$$-D_{KL}(q_\phi(z|x)||p(z)) = \frac{1}{2}(1 + \log\sigma_z^2 - \mu_z^2 - \sigma_z^2)$$



Auto-Encoding Variational Bayes

Diederik P. Kingma
Machine Learning Group
Universiteit van Amsterdam
dpkingma@gmail.com

Max Welling
Machine Learning Group
Universiteit van Amsterdam
welling.max@gmail.com

The marginal likelihood is composed of a sum over the marginal likelihoods of individual datapoints $\log p_\theta(\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(N)}) = \sum_{i=1}^N \log p_\theta(\mathbf{x}^{(i)})$, which can each be rewritten as:

$$\log p_\theta(\mathbf{x}^{(i)}) = D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z}|\mathbf{x}^{(i)})) + \mathcal{L}(\theta, \phi; \mathbf{x}^{(i)}) \quad (1)$$

The first RHS term is the KL divergence of the approximate from the true posterior. Since this KL-divergence is non-negative, the second RHS term $\mathcal{L}(\theta, \phi; \mathbf{x}^{(i)})$ is called the (variational) *lower bound* on the marginal likelihood of datapoint i, and can be written as:

$$\log p_\theta(\mathbf{x}^{(i)}) \geq \mathcal{L}(\theta, \phi; \mathbf{x}^{(i)}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}, \mathbf{z})\right] \quad (2)$$

which can also be written as:

$$\mathcal{L}(\theta, \phi; \mathbf{x}^{(i)}) = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})\right] \quad (3)$$

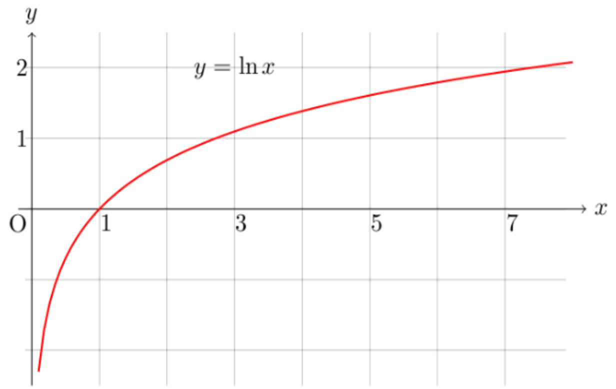
We want to differentiate and optimize the lower bound $\mathcal{L}(\theta, \phi; \mathbf{x}^{(i)})$ w.r.t. both the variational parameters ϕ and generative parameters θ. However, the gradient of the lower bound w.r.t. ϕ is a bit problematic. The usual (naïve) Monte Carlo gradient estimator for this type of problem is: $\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \mathbb{E}_{q_\phi(\mathbf{z})}\left[f(\mathbf{z})\nabla_{q_\phi(\mathbf{z})} \log q_\phi(\mathbf{z})\right] \simeq \frac{1}{L}\sum_{l=1}^L f(\mathbf{z})\nabla_{q_\phi(\mathbf{z}^{(l)})} \log q_\phi(\mathbf{z}^{(l)})$ where $\mathbf{z}^{(l)} \sim q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$. This gradient estimator exhibits exhibits very high variance (see e.g. [BJP12]) and is impractical for our purposes.

VAE

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

If we use Monte Carlo gradient estimator:

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \mathbb{E}_{q_\phi(\mathbf{z})}\left[f(\mathbf{z})\nabla_{q_\phi(\mathbf{z})} \log q_\phi(\mathbf{z})\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f(\mathbf{z})\nabla_{q_\phi(\mathbf{z}^{(l)})} \log q_\phi(\mathbf{z}^{(l)})$$

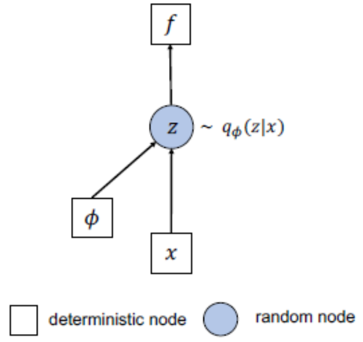


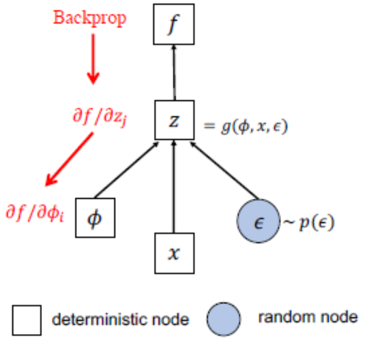
The gradient has log term, so it has high variance and needs lots of samples.

To solve this problem, we introduce **reparameterization trick**:

$\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})$ ⇨ auxiliary variable $\epsilon \sim p(\epsilon)$; deterministic variable $\mathbf{z} = g_\phi(\epsilon, \mathbf{x})$

Example:
$z \sim p(z|x) = \mathcal{N}(\mu, \sigma^2)$ ⇨ $\epsilon \sim \mathcal{N}(0, 1)$, $z = \mu + \sigma\epsilon$





$z^{(k)} \sim \mathcal{N}(\mu_z(x;\phi), \sigma_z^2(x;\phi))$

$\epsilon^{(k)} \sim \mathcal{N}(0, 1)$

$z^{(k)} = \mu_z(x, \phi) + \sigma_z(x, \phi) \cdot \epsilon^{(k)}$

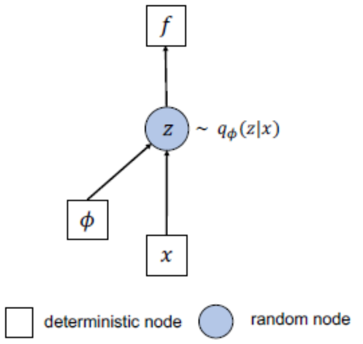
VAE

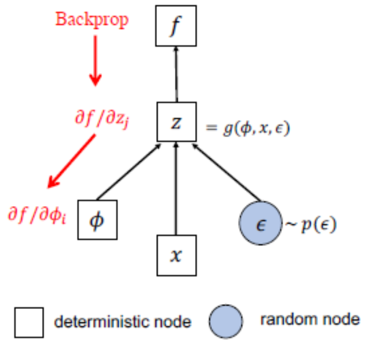
Loss function: $L(\theta,\phi,\mathbf{x}) = -D_{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})] + E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]$

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}[f(\mathbf{z})] = \mathbb{E}_{p(\epsilon)}\left[f\left(g_\phi\left(\boldsymbol{\epsilon},\mathbf{x}^{(i)}\right)\right)\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f\left(g_\phi\left(\boldsymbol{\epsilon}^{(l)},\mathbf{x}^{(i)}\right)\right) \quad \text{where} \quad \epsilon^{(l)} \sim p(\epsilon)$$

$$\boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon}), \quad l = 1,2,\ldots,L$$

$$\nabla_\phi \mathcal{L} \approx \frac{1}{L}\sum_{l=1}^{L}[(\frac{d}{dz}(\log p(x^{(i)},z) - \log q(z|\phi)))(\frac{d}{d\phi}g_\phi(\boldsymbol{\epsilon},\mathbf{x^{(i)}}))] \quad \text{where} \quad \mathbf{Z} = g_\phi(\boldsymbol{\epsilon},\mathbf{x})$$





$$z^{(k)} \sim \mathcal{N}(\mu_z(x;\phi), \sigma_z^2(x;\phi))$$

$$\epsilon^{(k)} \sim \mathcal{N}(0,1)$$

$$z^{(k)} = \mu_z(x,\phi) + \sigma_z(x,\phi) \cdot \epsilon^{(k)}$$

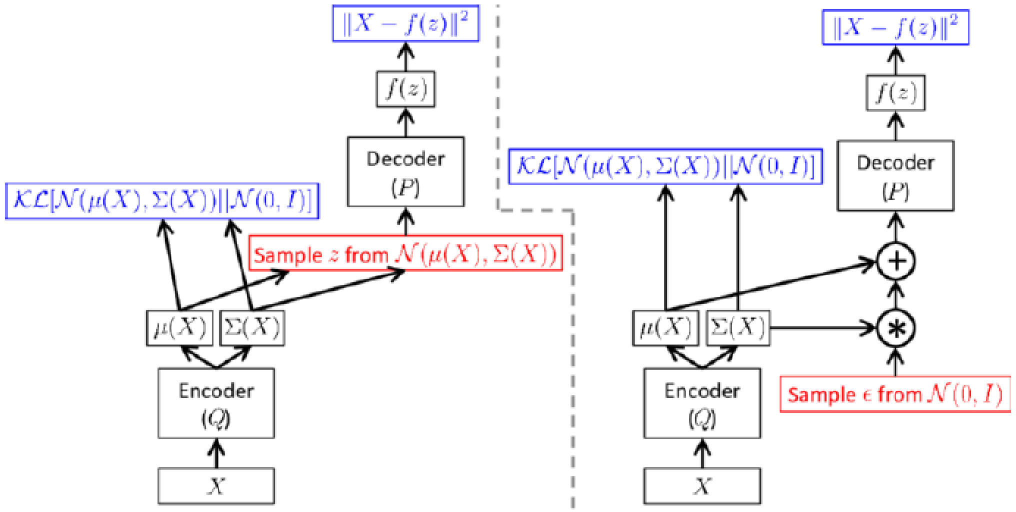
VAE

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

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}[f(\mathbf{z})] = \mathbb{E}_{p(\epsilon)}\left[f\left(g_\phi\left(\boldsymbol{\epsilon}, \mathbf{x}^{(i)}\right)\right)\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f\left(g_\phi\left(\boldsymbol{\epsilon}^{(l)}, \mathbf{x}^{(i)}\right)\right) \quad \text{where} \quad \boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})$$

$$\boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon}), \quad l = 1, 2, \ldots, L$$

$$\nabla_\phi \mathcal{L} \approx \frac{1}{L}\sum_{l=1}^{L}[(\frac{d}{dz}(\log p(x^{(i)}, z) - \log q(z|\phi)))(\frac{d}{d\phi}g_\phi(\boldsymbol{\epsilon}, \mathbf{x}^{(\mathbf{i})}))] \qquad \text{where} \quad \mathbf{Z} = g_\phi(\boldsymbol{\epsilon}, \mathbf{x})$$



- Loss function:

$$L(\theta, \phi, x) \approx \frac{1}{2}\sum_{j}^{J}(1 + \log \sigma_{zj}^2 - \mu_{zj}^2 - \sigma_{zj}^2) + \frac{1}{L}\sum_{k}^{L}\log(p_\theta(x|z^{(k)}))$$

where $z^{(k)} = \mu_z(x, \phi) + \sigma_z(x, \phi) \cdot \epsilon^{(k)}$ and $\epsilon^{(k)} \sim \mathcal{N}(0, 1)$.

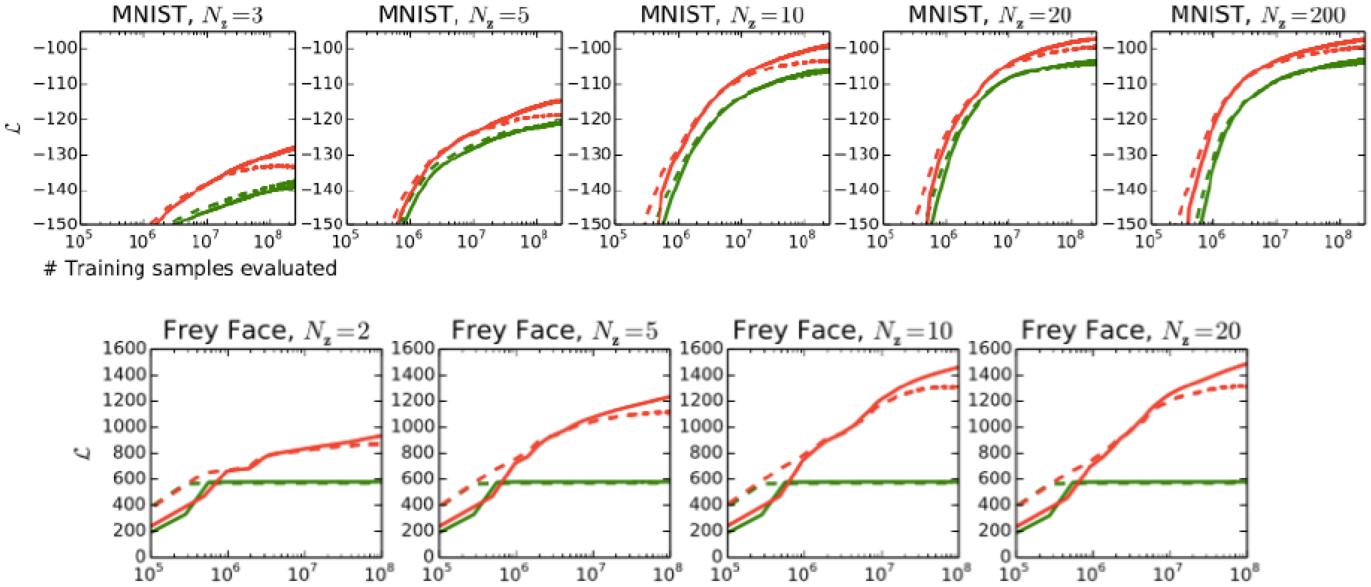
VAE

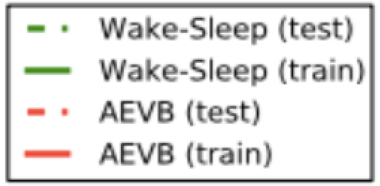
Training

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.

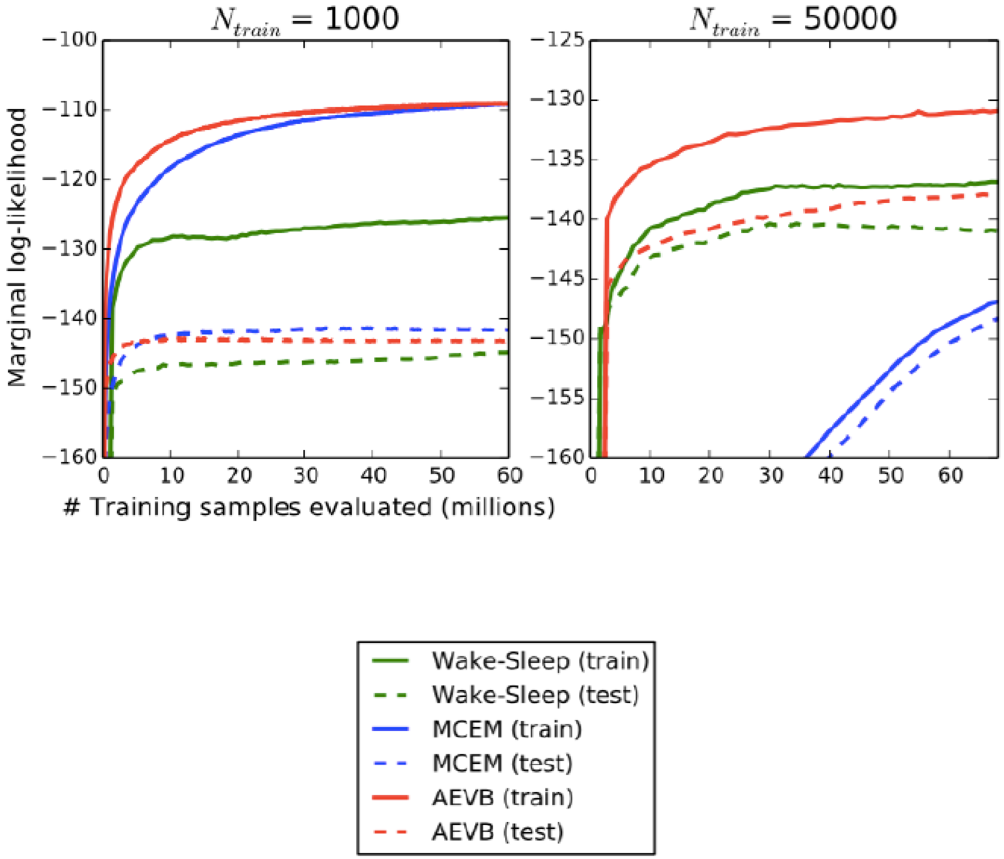
$\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Initialize parameters
repeat
  $\mathbf{X}^M \leftarrow$ Random minibatch of M datapoints (drawn from full dataset)
  $\boldsymbol{\epsilon} \leftarrow$ Random samples from noise distribution $p(\boldsymbol{\epsilon})$
  $\mathbf{g} \leftarrow \nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \widetilde{\mathcal{L}}^M(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{X}^M, \boldsymbol{\epsilon})$ (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 $\boldsymbol{\theta}, \boldsymbol{\phi}$

Results

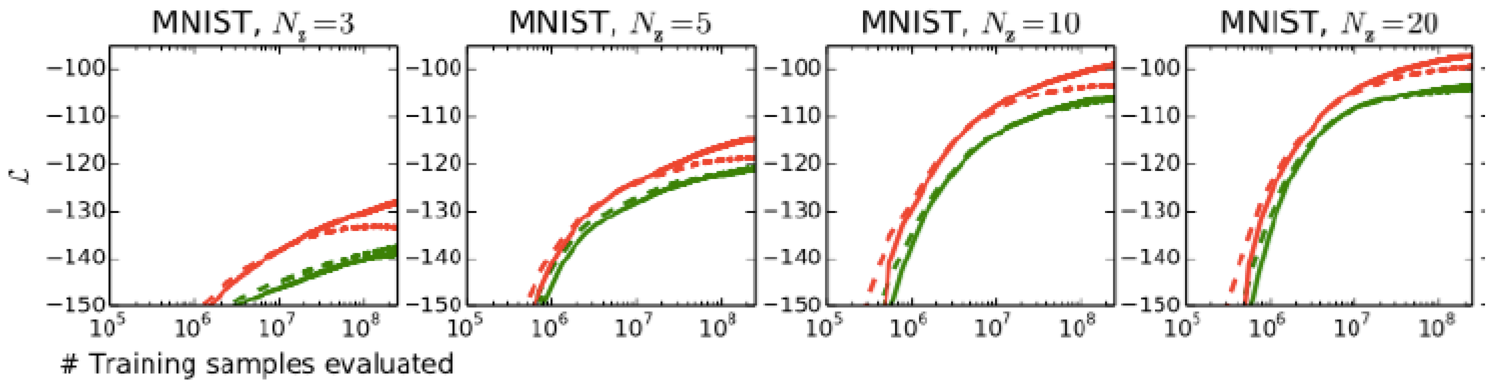


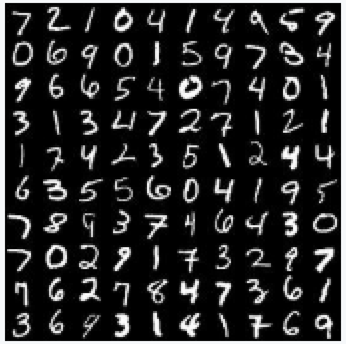
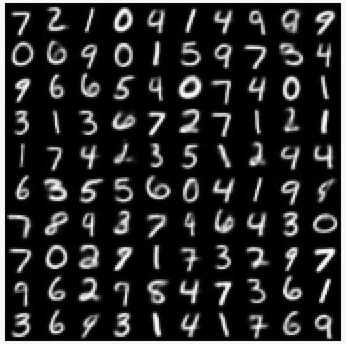
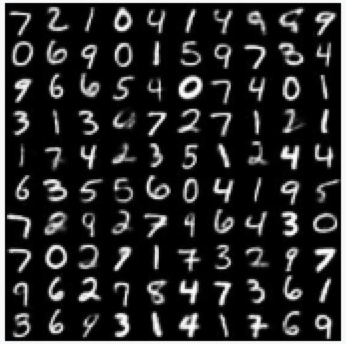
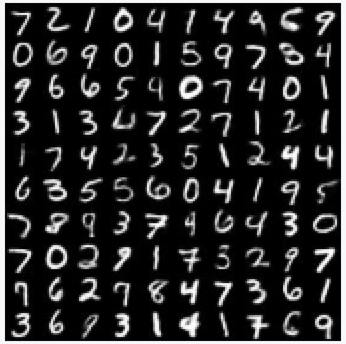


Horizontal axis: size of training data
Vertical axis: evidence Lower Bound
N_z : dimensions of hidden variables



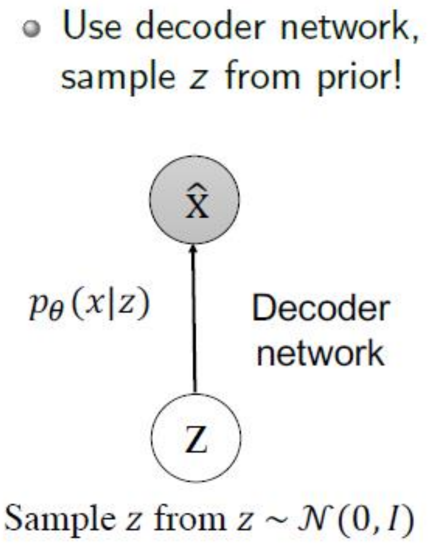
Results

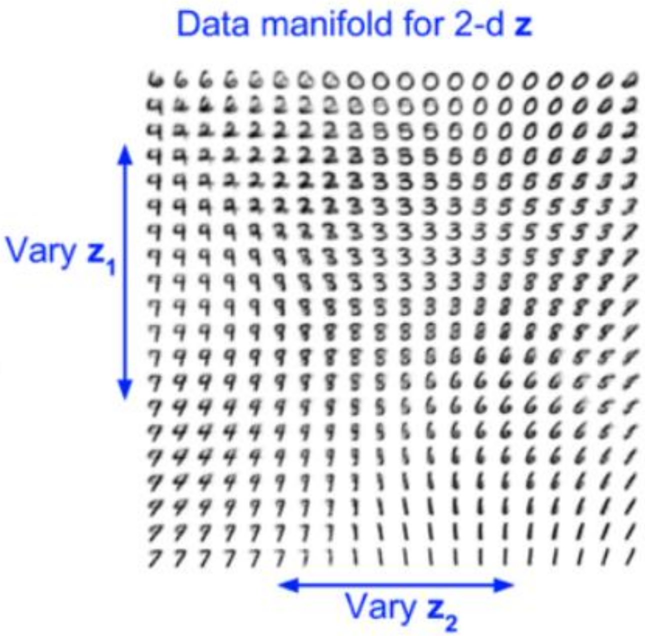


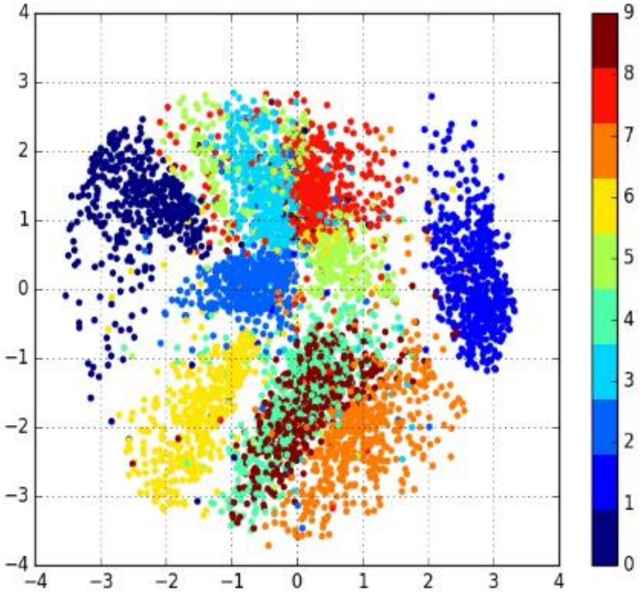
Input image	2-D latent space	5-D latent space	10-D latent space	20-D latent space
				

Well trained VAE must be able to reproduce input image, this figure shows reproduce performance of learned generative models for different dimensionalities.

Results: Generate new data







Visualizations of learned MNIST manifold for generative models with 2-dim and its latent space distribution.

https://github.com/hwalsuklee/tensorflow-mnist-VAE

Conclusion

1. Classical autoencoders minimize a reconstruction loss $\|x-\hat{x}\|^2$, they are unregularized in latent space.

2. VAEs are one approach to regularizing (impose structure) the latent distribution.

3. Probabilistic model: KL divergence, approximate inference, reparameterization trick.

Thanks!

Conditional VAE

By using the variational autoencoder, we do not have control over the data generation process.

E.g. we cannot generate only one specific digit from a model trained on MNIST dataset.

We want, for example, to input the character 9 to our model and get a generated image of a handwritten digit 9.

Conditional VAE

We will condition encoder and decoder on other inputs as well as the image, lets call those inputs c.

Encoder becomes: $q(z \mid x, c)$

Decoder becomes: $p(x \mid z, c)$

Now our variational lower bound objective becomes:

$$\mathcal{L} = \mathbb{E}[\log p(x \mid z, c)] - D_{KL}(q(z \mid x, c) \| p(z \mid c))$$

Conditional VAE

Learned MNIST manifold with a condition of label 2	Learned MNIST manifold with a condition of label 3	Learned MNIST manifold with a condition of label 4