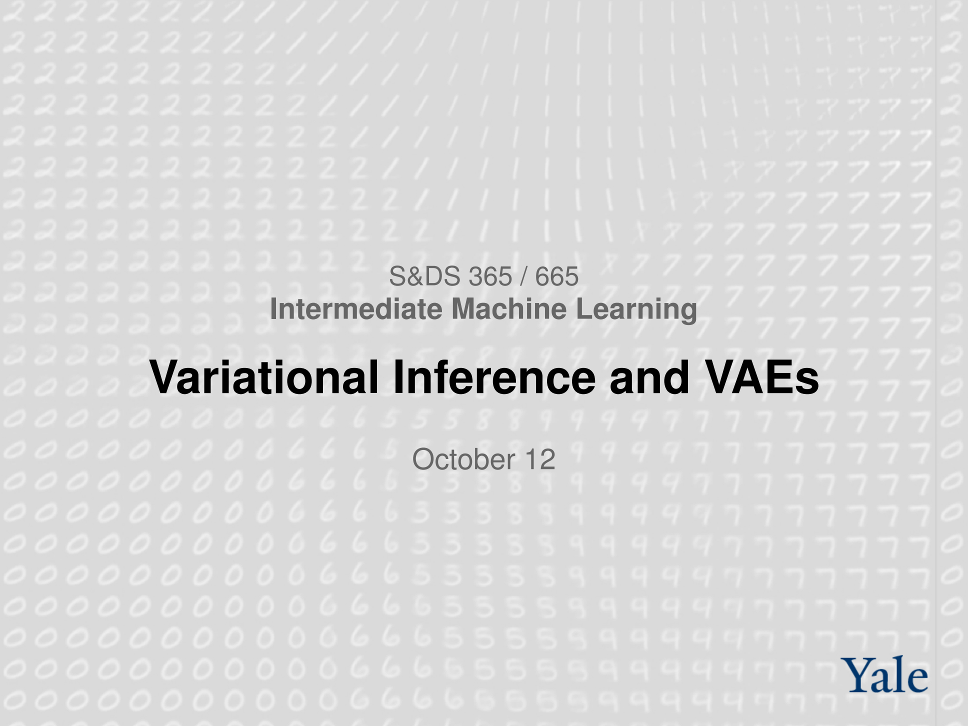


S&DS 365 / 665

**Intermediate Machine Learning**

# Variational Inference and VAEs

October 12

Yale

## Reminders

- Practice midterm and solutions posted
- Multiple review sessions
- Exam Monday: Cheat sheet, parallels practice

## For Today

- Variational inference: The ELBO (rehash)
- Variational autoencoders
- Demo notebook
- Questions for review

## Variational inference: Strategy

- We'd like to compute $p(\theta, z \mid x)$, but it's too complicated.
- Strategy: Approximate as $q(\theta, z)$ that has a "nice" form
- $q$ is a function of variational parameters, optimized for each $x$.
- Maximize a lower bound on $p(x)$.

## Variational inference: The ELBO

The ELBO is the following lower bound on $\log p(x)$:

$$\log\; p(x) = \int \sum_{z} q(z,\theta) \log p(x)\, d\theta$$

## Variational inference: The ELBO

The ELBO is the following lower bound on $\log p(x)$:

$$\log\ p(x) = \int \sum_z q(z,\theta) \log p(x)\, d\theta$$

$$= \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)\, q(z,\theta)}{p(z,\theta \mid x)\, q(z,\theta)} \right) d\theta$$

## Variational inference: The ELBO

The ELBO is the following lower bound on $\log p(x)$:

$$\log\ p(x) = \int \sum_z q(z,\theta) \log p(x)\, d\theta$$

$$= \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)\, q(z,\theta)}{p(z,\theta \mid x)\, q(z,\theta)} \right)\ d\theta$$

$$= \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)}{q(z,\theta)} \right) d\theta + \sum_z \int q(z,\theta) \log \left( \frac{q(z,\theta)}{p(z,\theta \mid x)} \right) d$$

## Variational inference: The ELBO

The ELBO is the following lower bound on $\log p(x)$:

$$\log\ p(x) = \int \sum_z q(z,\theta) \log p(x)\, d\theta$$

$$= \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)\, q(z,\theta)}{p(z,\theta \,|\, x)\, q(z,\theta)} \right) d\theta$$

$$= \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)}{q(z,\theta)} \right) d\theta + \sum_z \int q(z,\theta) \log \left( \frac{q(z,\theta)}{p(z,\theta \,|\, x)} \right) d$$

$$\geq \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)}{q(z,\theta)} \right) d\theta$$

## Variational inference: The ELBO

The ELBO is the following lower bound on $\log p(x)$:

$$\log\ p(x) = \int \sum_z q(z,\theta) \log p(x)\, d\theta$$

$$\begin{aligned}
&= \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)\, q(z,\theta)}{p(z,\theta \mid x)\, q(z,\theta)} \right)\ d\theta \\
&= \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)}{q(z,\theta)} \right) d\theta + \sum_z \int q(z,\theta) \log \left( \frac{q(z,\theta)}{p(z,\theta \mid x)} \right) d \\
&\geq \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)}{q(z,\theta)} \right) d\theta \\
&= H(q) + \mathbb{E}_q\big(\log p(x,z,\theta)\big)
\end{aligned}$$

## Variational inference: The ELBO

The ELBO is the following lower bound on $\log p(x)$:

$$
\begin{aligned}
\log\ p(x) &= \int \sum_z q(z,\theta) \log p(x)\, d\theta \\
&= \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)\, q(z,\theta)}{p(z,\theta \mid x)\, q(z,\theta)} \right) d\theta \\
&= \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)}{q(z,\theta)} \right) d\theta + \sum_z \int q(z,\theta) \log \left( \frac{q(z,\theta)}{p(z,\theta \mid x)} \right) d \\
&\geq \sum_z \int q(z,\theta) \log \left( \frac{p(x,z,\theta)}{q(z,\theta)} \right) d\theta \\
&= H(q) + \mathbb{E}_q\big(\log p(x,z,\theta)\big)
\end{aligned}
$$

We maximize this over the parameters of $q$

## Variational inference: The ELBO

The inequality above uses concavity of the logarithm:

$$\log\left(\sum_{\alpha} w_\alpha x_\alpha\right) \geq \sum_{\alpha} w_\alpha \log x_\alpha$$

So, if $q_\alpha \geq 0$ and $p_\alpha \geq 0$ sum (or integrate) to one, then

$$0 = \log\left(\sum_{\alpha} p_\alpha\right) = \log\left(\sum_{\alpha} q_\alpha \frac{p_\alpha}{q_\alpha}\right) \geq \sum_{\alpha} q_\alpha \log\left(\frac{p_\alpha}{q_\alpha}\right)$$

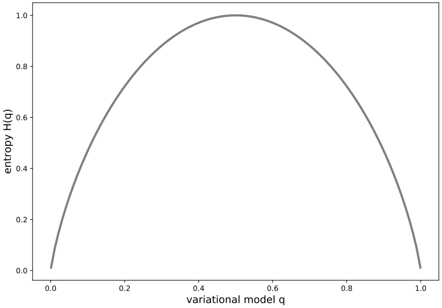
Therefore

$$D_{KL}(q \parallel p) \equiv \sum_{\alpha} q_\alpha \log\left(\frac{q_\alpha}{p_\alpha}\right) \geq 0$$

## Variational inference: The ELBO

The ELBO is $H(q) + \mathbb{E}_q(\log p)$

The entropy term $H(q)$ encourages $q$ to be spread out:



The cross-entropy $\mathbb{E}_q \log p$ tries to match $q$ to $p$

# Variational inference: The ELBO

Since $\log p(x, z, \theta) = \log p(z, \theta) + \log p(x \mid z, \theta)$, the ELBO can be written as

$$\text{ELBO} = \mathbb{E}_q(\log p(x \mid Z, \theta)) - D_{KL}(q(Z, \theta) \parallel p(Z, \theta))$$

- The Kullback-Leibler divergence term acts as a regularizer, encouraging the variational distribution to be close to the prior

## Example 2: A finite mixture model

Fix two distributions $F_0$ and $F_1$, with densities $f_0(x)$ and $f_1(x)$, and form the mixture model

$$\begin{aligned} \theta &\sim \text{Beta}(\alpha, \beta) \\ X \mid \theta &\sim \theta F_1 + (1-\theta) F_0. \end{aligned}$$

The likelihood for data $x_1, \ldots, x_n$ is

$$p(x_{1:n}) \; = \; \int_0^1 \text{Beta}(\theta \mid \alpha, \beta) \prod_{i=1}^n \left(\theta f_1(x_i) + (1-\theta) f_0(x_i)\right) \, d\theta.$$

Our goal is to approximate the posterior $p(\theta \mid x_{1:n})$

## Variational approximation

Our variational approximation is

$$q(z, \theta) = q(\theta \mid \gamma_1, \gamma_2) \prod_{i=1}^{n} q_i^{z_i} (1 - q_i)^{(1 - z_i)}$$

where $q(\theta \mid \gamma_1, \gamma_2)$ is a Beta$(\gamma_1, \gamma_2)$ distribution, and $0 \leq q_i \leq 1$ are $n$ free parameters.

Need to maximize ELBO $H(q) + \mathbb{E}_q \log p$

## Variational algorithm for mixture

**Variational inference**

Iterate the following steps for variational parameters $q_{1:n}$ and $(\gamma_1, \gamma_2)$:

1. Holding $q_i$ fixed, set $\gamma = (\gamma_1, \gamma_2)$ to

$$\gamma_1 = \alpha + \sum_{i=1}^{n} q_i \qquad \gamma_2 = \beta + n - \sum_{i=1}^{n} q_i$$

2. Holding $\gamma_1$ and $\gamma_2$ fixed, set $q_i$ to

$$q_i = \frac{f_1(x_i) \exp \psi(\gamma_1)}{f_1(x_i) \exp \psi(\gamma_1) + f_0(x_i) \exp \psi(\gamma_2)}$$

After convergence, approximate posterior distribution over $\theta$ is

$$\widehat{p}(\theta \,|\, x_{1:n}) = \mathrm{Beta}(\theta \,|\, \gamma_1, \gamma_2)$$

## Deterministic approximation

- Convergence is numerical, not stochastic
- Posterior is approximated as a *single* Beta
- Very similar algorithm is used for topic models

## Some sample questions

## Variational inference

Q: What is the best $q$ we could use?

## Variational inference

Q: What is the best $q$ we could use?

A: The true posterior $q(z, \theta \mid x) = p(z, \theta \mid x)$

## Variational inference

Q: What is the best $q$ we could use?

A: The true posterior $q(z, \theta \mid x) = p(z, \theta \mid x)$

Why? Because this maximizes the ELBO. Mathematically,

$$\sum_z \int q(z, \theta) \log\left(\frac{q(z, \theta)}{p(z, \theta \mid x)}\right) d\theta = 0$$

in this case, so the ELBO inequality is an equality.

## Variational inference

Q: How does the ELBO regularize?

## Variational inference

Q: How does the ELBO regularize?

A: The entropy term favors distributions that are “spread out”

## Variational inference

Q: How does the ELBO regularize?

A: The entropy term favors distributions that are “spread out”

Why? For discrete distributions the maximum entropy distribution is uniform. For Gaussian, the entropy is $\log \sigma^2$ which favors $\sigma^2$ large.

## Variational inference

Q: Is the ELBO easy to maximize?

## Variational inference

Q: Is the ELBO easy to maximize?

A: No.

## Variational inference

Q: Is the ELBO easy to maximize?

A: No.

Why? In general it is non-convex, and the solution depends on where we start an iterative algorithm. This is unlike the Gibbs sampler, which converges to the right thing if we wait long enough.

## Variational autoencoders

- Variational autoencoders are generative models that are trained using variational inference
- The "decoder" is a neural net that generates from a latent variable
- The "encoder" approximates the posterior distribution with another neural network trained using variational inference

## Variational autoencoders

Start with a generative model

$$z \sim N(0, I_k)$$
$$x \mid z \sim N(G(z), I_d)$$

$G(z)$ is the *generator network* or *decoder*. The latent $z$ is $k$-dimensional and the output $x$ is $d$-dimensional.

For example, use a 2-layer network

$$G(z) = A_2 \mathrm{ReLU}(A_1 z + b_1) + b_2$$

## Posterior inference

How do we train the generative network?

$$\begin{aligned} Z &\sim N(0, I_k) \\ X \mid z &\sim N(G(z), I_d) \end{aligned}$$

We want the posterior distribution $p(z \mid x)$

But this is intractable, because $G(\cdot)$ is nonlinear

Approach: Use variational inference

## Using variational inference

For variational inference we take

$$q(z \mid x) = N(\mu_x, \sigma_x^2 I_k)$$

where now $\mu_x$ and $\sigma_x^2$ are the *variational parameters*

## Using neural networks

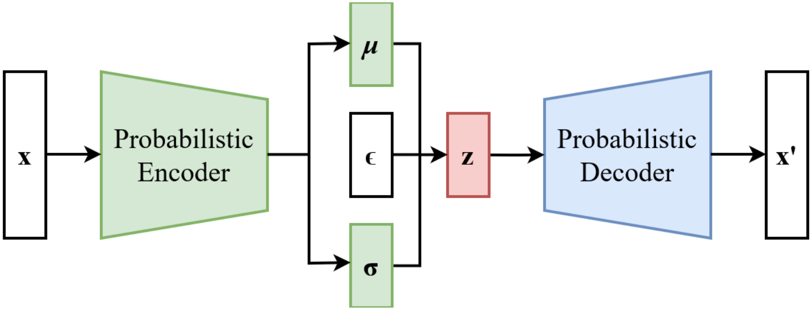
Rather than estimate $\mu_x$ for each $x$, we build an encoder neural network that outputs the mean and variance.

For example:

$$\mu(x) = B_2 \,\mathrm{ReLU}(B_1 x + d_1) + d_2$$

and similarly for $\log \sigma^2(x)$.

## VAE architecture



## Training the neural networks

Decoder network: $z \mapsto x$, weights $A, b$

Encoder network: $x \mapsto \mu(x),\ \log \sigma^2(x)$, weights $B, d$

Train both networks simultaneously, to maximize the ELBO

- Decoder trained with $\mathbb{E}_q \log p(x, Z_s)$ over weights $A, b$
- Encoder trained with $H(q) + \mathbb{E}_q \log p(x, Z_s)$ over weights $B, d$

The entropy of a multivariate Gaussian with covariance $\Sigma$ is $\frac{1}{2} \log |\Sigma| +$ constant

The entropy term here is $\log \sigma^2(x) +$ constant, which is taken to be an output of the encoder network.

## Using neural networks

Now, approximate $\mathbb{E}_q(\log p(x, Z))$ by sampling (weak law of large numbers)

$$\mathbb{E}_q(\log p(x, Z)) \approx \frac{1}{N} \sum_{s=1}^{N} \log p(x, Z_s)$$

Problem: The parameters of the recognition network have disappeared!

Solution: Reparameterize the samples by $Z_i = \mu(x) + \sigma(x)\epsilon_i$ where $\epsilon_i \sim N(0, I_k)$.

This is called "the reparameterization trick." D. Kingma and M. Welling, "Autoencoding Variational Bayes," `https://arxiv.org/abs/1312.6114`.

## Simple example

Suppose $x \mid z \sim N(G(z), I)$ where generator network is

$$G(z) = \mathrm{ReLU}(Az + b).$$

Then $-\log p(x \mid z)$ is

$$\frac{1}{2}\|x - \mathrm{ReLU}(Az + b)\|^2 + \text{constant}$$

## Simple example

Suppose $x \mid z \sim N(G(z), I)$ where generator network is

$$G(z) = \mathrm{ReLU}(Az + b).$$

Then $-\log p(x \mid z)$ is

$$\frac{1}{2}\|x - \mathrm{ReLU}(Az + b)\|^2 + \text{constant}$$

And $-\log p(z)$ is

$$\frac{1}{2}\|z\|^2 + \text{constant}$$

## Simple example (continued)

Next, suppose the approximate posterior is

$$q(z \mid x) = N(\mu(x), \sigma^2(x) I_k)$$

where encoder network is $\mu(x) = \mathrm{ReLU}(Bx + d)$. Then

$$-\mathbb{E}_q \log p(x \mid Z) \approx \frac{1}{N} \sum_{s=1}^{N} \frac{1}{2} \|x - \mathrm{ReLU}(AZ_s + b)\|^2$$

$$\overset{d}{=} \frac{1}{N} \sum_{s=1}^{N} \frac{1}{2} \|x - \mathrm{ReLU}(A(\mathrm{ReLU}(Bx + d) + \epsilon_s \sigma(x)) + b)\|^2$$

Here $\overset{d}{=}$ means equal in distribution.

## Simple example (continued)

Next, suppose the approximate posterior is

$$q(z \mid x) = N(\mu(x), \sigma^2(x) I_k)$$

where encoder network is $\mu(x) = \mathrm{ReLU}(Bx + d)$. Then

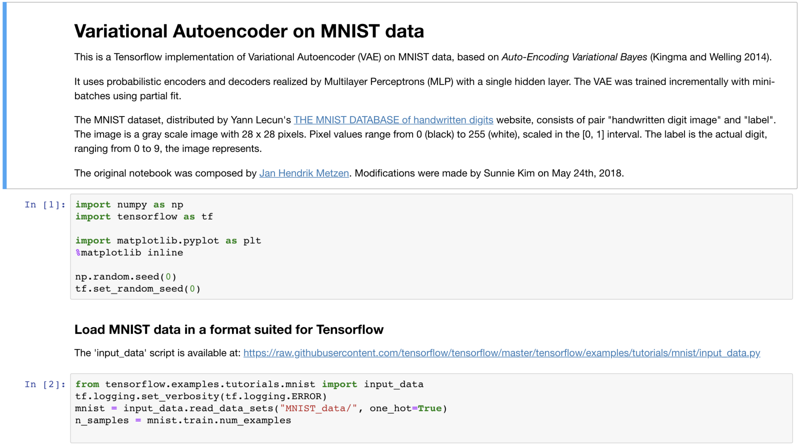
$$\begin{aligned} -\mathbb{E}_q \log p(Z) &= \frac{1}{2}\mathbb{E}_q \|Z\|^2 \\ &= \frac{1}{2}\|\mu(x)\|^2 + \frac{k}{2}\sigma^2(x) \end{aligned}$$

Here $\overset{d}{=}$ means equal in distribution.

## Resulting training objective

- This gives all of the pieces for the ELBO objective
- No need to work out the detailed calculations — TensorFlow does this for you
- The ELBO is optimized by alternating stochastic gradient steps on the weights in the decoder and encoder networks

# Demo



## Variational Autoencoder on MNIST data

This is a Tensorflow implementation of Variational Autoencoder (VAE) on MNIST data, based on *Auto-Encoding Variational Bayes* (Kingma and Welling 2014).

It uses probabilistic encoders and decoders realized by Multilayer Perceptrons (MLP) with a single hidden layer. The VAE was trained incrementally with mini-batches using partial fit.

The MNIST dataset, distributed by Yann Lecun's THE MNIST DATABASE of handwritten digits website, consists of pair "handwritten digit image" and "label". The image is a gray scale image with 28 x 28 pixels. Pixel values range from 0 (black) to 255 (white), scaled in the [0, 1] interval. The label is the actual digit, ranging from 0 to 9, the image represents.

The original notebook was composed by Jan Hendrik Metzen. Modifications were made by Sunnie Kim on May 24th, 2018.

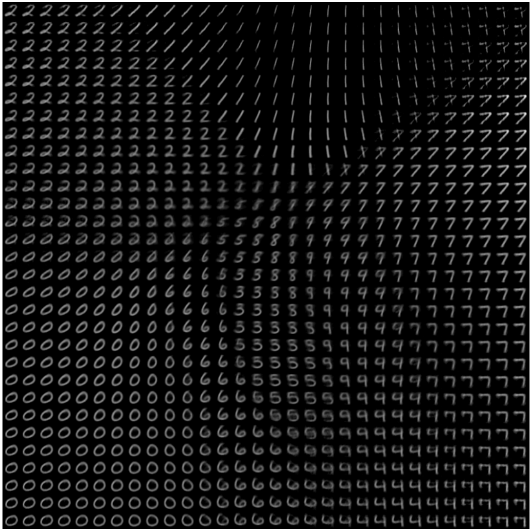
In [1]:
```
import numpy as np
import tensorflow as tf

import matplotlib.pyplot as plt
%matplotlib inline

np.random.seed(0)
tf.set_random_seed(0)
```

### Load MNIST data in a format suited for Tensorflow

The 'input_data' script is available at: https://raw.githubusercontent.com/tensorflow/tensorflow/master/tensorflow/examples/tutorials/mnist/input_data.py

In [2]:
```
from tensorflow.examples.tutorials.mnist import input_data
tf.logging.set_verbosity(tf.logging.ERROR)
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)
n_samples = mnist.train.num_examples
```

# Visualizing a 2-dim latent space



## Deconvolutional neural networks

- A convolutional network goes from a high dimensional input to a lower dimensional output
- The decoder / generator network goes from a low dimensional latent vector to a high dimensional output
- “Deconvolutional” or transposed convolutional networks are the “opposite” of CNNs
- Useful for this type of problem when data are images
- We won’t dive into the details here

Many tutorials can be found, for example https://towardsdatascience.com/transposed-convolution-demystified-84ca81b4baba

## Summary

- Variational methods make deterministic approximations
- General recipe: Maximize ELBO over variational parameters
- VAEs: Generator / decoder is a neural network
- Variational mean is output of a second neural network
- The two networks are trained together to maximize the ELBO