CS 559 Deep Learning

Unsupervised Learning

Gokberk Cinbis
Unsupervised Learning Overview

- Definitions
- Autoencoders
  - Vanilla
  - Variational
- Adversarial Networks
Supervised vs Unsupervised

Supervised Learning

**Data:** $(x, y)$

$x$ is data, $y$ is label

**Goal:** Learn a *function* to map $x \rightarrow y$

**Examples:** Classification, regression, object detection, semantic segmentation, image captioning, etc

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Supervised vs Unsupervised

**Supervised Learning**

**Data:** $(x, y)$

$x$ is data, $y$ is label

**Goal:** Learn a *function* to map $x \rightarrow y$

**Examples:** Classification, regression, object detection, semantic segmentation, image captioning, etc.

**Unsupervised Learning**

**Data:** $x$

Just data, no labels!

**Goal:** Learn some *structure* of the data

**Examples:** Clustering, dimensionality reduction, feature learning, generative models, etc.
Unsupervised Learning

- Autoencoders
  - Traditional: feature learning
  - Variational: generate samples
- Generative Adversarial Networks: Generate samples
Autoencoders

Input data $x$ → Encoder → Features $z$
Autoencoders

- **Originally**: Linear + nonlinearity (sigmoid)
- **Later**: Deep, fully-connected
- **Later**: ReLU CNN

![Diagram of Autoencoder](image)

Input data $x$ flows through the encoder to produce features $z$. The reason for these changes is typically to improve the model's ability to capture complex patterns in the data.
Autoencoders

\( z \) usually smaller than \( x \) (dimensionality reduction)

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN

Input data \( x \) ➔ Encoder ➔ Features \( z \) ➔ Input data

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Autoencoders

Reconstructed input data

xx

Decoder

Features

z

Encoder

Input data

x

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Autoencoders

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN (upconv)

Encoder: 4-layer conv
Decoder: 4-layer upconv

Input data $x$

Features $z$

Reconstructed input data $\hat{x}$

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Autoencoders

Encoder / decoder sometimes share weights

Example:
\( \text{dim}(x) = D \)
\( \text{dim}(z) = H \)
\( w_e : H \times D \)
\( w_d : D \times H = w_e^T \)

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN (upconv)

Train for reconstruction with no labels!
Autoencoders

Input data $x$ $\rightarrow$ Encoder $\rightarrow$ Features $z$ $\rightarrow$ Decoder $\rightarrow$ Reconstructed input data $x$

Loss function (Often L2)

Train for reconstruction with no labels!

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Autoencoders

After training, throw away decoder to obtain a pre-trained feature extractor.
Autoencoders

Use encoder to initialize a supervised model

Input data \( x \)

Features \( z \)

Predicted Label \( y \)

Classifier

Loss function (Softmax, etc)

Fine-tune encoder jointly with classifier

Train for final task (sometimes with small data)

bird  plane
dog  deer  truck

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Autoencoders: Greedy Training

In mid 2000s layer-wise pretraining with Restricted Boltzmann Machines (RBM) was common.

Training deep nets was hard in 2006!

It is difficult to optimize the weights in nonlinear autoencoders that have multiple hidden layers (2–4). With large initial weights, autoencoders typically find poor local minima; with small initial weights, the gradients in the early layers are tiny, making it infeasible to train autoencoders with many hidden layers. If

Hinton and Salakhutdinov, “Reducing the Dimensionality of Data with Neural Networks”, Science 2006

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Autoencoders: Greedy Training

In mid 2000s layer-wise pretraining with Restricted Boltzmann Machines (RBM) was common

Training deep nets was hard in 2006!

It is difficult to optimize the weights in nonlinear autoencoders that have multiple hidden layers (2–4). With large initial weights, autoencoders typically find poor local minima; with small initial weights, the gradients in the early layers are tiny, making it infeasible to train autoencoders with many hidden layers. If

Hinton and Salakhutdinov, “Reducing the Dimensionality of Data with Neural Networks”, Science 2006

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Autoencoders can reconstruct data, and can learn features to initialize a supervised model.

Can we generate images from an autoencoder?
Variational Autoencoder

A Bayesian spin on an autoencoder - lets us generate data!

Assume our data \( \{ x^{(i)} \}_{i=1}^{N} \) is generated like this:

Sample from \( p_{\theta^*}(z) \)

Sample from \( true \) conditional \( p_{\theta^*}(x \mid z^{(i)}) \)

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014
Variational Autoencoder

A Bayesian spin on an autoencoder!

Assume our data \( \{ x^{(i)} \}_{i=1}^{N} \) is generated like this:

```
Sample from \( \text{true prior} \) \( p_{\theta^*}(z) \)

Sample from \( \text{true conditional} \) \( p_{\theta^*}(x \mid z^{(i)}) \)
```

Intuition: \( x \) is an image, \( z \) gives class, orientation, attributes, etc

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014
Variational Autoencoder

A Bayesian spin on an autoencoder!

Assume our data \( \{ x^{(i)} \}_{i=1}^{N} \) is generated like this:

Sample from true prior \( p_{\theta^*}(z) \)

Sample from true conditional \( p_{\theta^*}(x \mid z^{(i)}) \)

Intuition: \( x \) is an image, \( z \) gives class, orientation, attributes, etc.

Problem: Estimate \( \theta \) without access to latent states \( z^{(i)} \)!

Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014
Variational Autoencoder

**Prior**: Assume $p_\theta(z)$ is a unit Gaussian

Kingma and Welling, ICLR 2014

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder

**Prior:** Assume $p_\theta(z)$ is a unit Gaussian.

**Conditional:** Assume $p_\theta(x \mid z)$ is a diagonal Gaussian, predict mean and variance with neural net.

Kingma and Welling, ICLR 2014

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder

**Prior**: Assume $p_\theta(z)$ is a unit Gaussian.

**Conditional**: Assume $p_\theta(x \mid z)$ is a diagonal Gaussian, predict mean and variance with neural net.

Mean and (diagonal) covariance of $p_\theta(x \mid z)$

- $\mu^x$
- $\Sigma^x$

Decoder network with parameters $\theta$

Latent state

Kingma and Welling, ICLR 2014

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder

**Prior:** Assume $p_\theta(z)$ is a unit Gaussian.

**Conditional:** Assume $p_\theta(x \mid z)$ is a diagonal Gaussian, predict mean and variance with neural net.

Kingma and Welling, ICLR 2014

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Encoder

By Bayes Rule the posterior is:

\[ p_\theta(z \mid x) = \frac{p_\theta(x \mid z) p_\theta(z)}{p_\theta(x)} \]

Kingma and Welling, ICLR 2014
Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Encoder

By Bayes Rule the posterior is:

\[ p_\theta(z \mid x) = \frac{p_\theta(x \mid z) p_\theta(z)}{p_\theta(x)} \]

Use decoder network =)
Gaussian =)
Intractible integral =(}

Kingma and Welling,
ICLR 2014
Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Encoder

By Bayes Rule the posterior is:

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

Use decoder network =)
Gaussian =)
Intractible integral =(

Kingma and Welling,
ICLR 2014
Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Encoder

By Bayes Rule the posterior is:

$$p(\mathbf{z} \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z})}{p(\mathbf{x})}$$

Use decoder network =)
Gaussian =)
Intractible integral =()

Approximate posterior with encoder network

$$q_{\phi}(\mathbf{z} \mid \mathbf{x})$$

Mean and (diagonal) covariance of

$$q_{\phi}(\mathbf{z} \mid \mathbf{x}) = \mu^z$$

Encoder network with parameters \( \phi \)

Data point

Kingma and Welling, ICLR 2014
Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Encoder

By Bayes Rule the posterior is:

\[
p_\theta(z \mid x) = \frac{p_\theta(x \mid z) p_\theta(z)}{p_\theta(x)}
\]

Use decoder network =)
Gaussian =)
Intractible integral =(}

Fully-connected
or convolutional

Mean and (diagonal) covariance of

\[
q_\phi(z \mid x) = \mu^z \Sigma^z
\]

Approximate posterior with encoder network

\[
q_\phi(z \mid x)
\]

Encoder network with parameters \( \phi \)

Data point

Kingma and Welling, ICLR 2014

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder

Data point $x$

Kingma and Welling, ICLR 2014
Variational Autoencoder

Encoder network

Data point

\( \mu^z \) \hspace{1cm} \( \Sigma^z \) \hspace{1cm} Mean and (diagonal) covariance of \( q_\phi(z \mid x) \)

Kingma and Welling, ICLR 2014
Variational Autoencoder

Sample from $q_\phi(z \mid x)$

Mean and (diagonal) covariance of $q_\phi(z \mid x)$

Encoder network

Data point

Kingma and Welling, ICLR 2014

\[ z \]

\[ \mu^z \]

\[ \Sigma^z \]
Variational Autoencoder

$\mu^x$ $\Sigma^x$

Decoder network

Mean and (diagonal) covariance of $p_\theta(x \mid z)$

$\mu^z$ $\Sigma^z$

Encoder network

Mean and (diagonal) covariance of $q_\phi(z \mid x)$

Sample from $q_\phi(z \mid x)$

$z$

$x$

Data point

Kingma and Welling, ICLR 2014

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder

Encoder network

Data point $x$

Sample from $q_\phi(z \mid x)$

$\mu^z$

$\Sigma^z$

Decoder network

Reconstructed $xx$

Sample from $p_\theta(x \mid z)$

$\mu^x$

$\Sigma^x$

Mean and (diagonal) covariance of $p_\theta(x \mid z)$

Mean and (diagonal) covariance of $q_\phi(z \mid x)$

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson

Kingma and Welling, ICLR 2014
**Variational Autoencoder**

Training like a normal autoencoder: **reconstruction loss** at the end, **regularization toward prior** in middle

Mean and (diagonal) covariance of $p_{\theta}(x \mid z)$

(should be close to data $x$)

Mean and (diagonal) covariance of $q_{\phi}(z \mid x)$

(should be close to prior $p_{\theta}(z)$)

Kingma and Welling, ICLR 2014

---

Decoder network

Sample from $p_{\theta}(x \mid z)$

$x$

Reconstructed $xx$

Encoder network

Sample from $q_{\phi}(z \mid x)$

Mean and (diagonal) covariance of $p_{\theta}(x \mid z)$

(should be close to prior $p_{\theta}(z)$)

Mean and (diagonal) covariance of $q_{\phi}(z \mid x)$

(should be close to prior $p_{\theta}(z)$)
Variational Autoencoder: Generate Data!

After network is trained:

\[ z \]

Sample from prior \( p_\theta(z) \)
Variational Autoencoder: Generate Data!

After network is trained:

\[
\mu^x \quad \Sigma^x \\
\downarrow \quad \downarrow \\
\mu \quad \Sigma \\
\leftarrow \quad \leftarrow \\
\text{Sample from prior } p_\theta(z)
\]
Variational Autoencoder: Generate Data!

After network is trained:

Generated: $xx$

Sample from $p_{\theta}(x | z)$

$\mu^x$  $\Sigma^x$

Decoder network

Sample from prior $p_{\theta}(z)$

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Generate Data!

After network is trained:

- Sample from $p_\theta(x \mid z)$
- Sample from prior $p_\theta(z)$

Generated $x$:
- Decoder network
- $\mu^x$, $\Sigma^x$
Variational Autoencoder: Generate Data!

After network is trained:

Generated $\mathbf{x}$

Sample from $p_\theta(x \mid z)$

$\mu^x$ $\Sigma^x$

Decoder network

Sample from prior $p_\theta(z)$

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Generate Data!

After network is trained:

Generated \( xx \)

Sample from \( p_\theta(x \mid z) \)

\( \mu^x \) \( \Sigma^x \)

Decoder network

Sample from prior \( p_\theta(z) \)

Diagonal prior on \( z \) => independent latent variables

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Math

Maximum Likelihood?

\[ \theta^* = \arg \max_{\theta} \prod_{i=1}^{N} p_{\theta}(x^{(i)}) \quad \text{Maximize likelihood of dataset } \left\{ x^{(i)} \right\}_{i=1}^{N} \]

Kingma and Welling, ICLR 2014
Variational Autoencoder: Math
Maximum Likelihood?

\[ \theta^* = \arg \max_\theta \prod_{i=1}^{N} p_\theta(x^{(i)}) \]  
Maximize likelihood of dataset \( \{x^{(i)}\}_{i=1}^{N} \)

\[ = \arg \max_\theta \sum_{i=1}^{N} \log p_\theta(x^{(i)}) \]  
Maximize log-likelihood instead because sums are nicer

Kingma and Welling, ICLR 2014
Variational Autoencoder: Math

Maximum Likelihood?

\[ \theta^* = \arg \max_\theta \prod_{i=1}^N p_\theta(x^{(i)}) \]  
Maximize likelihood of dataset \( \{x^{(i)}\}_{i=1}^N \)

\[ = \arg \max_\theta \sum_{i=1}^N \log p_\theta(x^{(i)}) \]  
Maximize log-likelihood instead because sums are nicer

\[ p_\theta(x^{(i)}) = \int p_\theta(x^{(i)}, z) dz \]  
Marginalize joint distribution

Kingma and Welling, ICLR 2014
Variational Autoencoder: Math

Maximum Likelihood?

\[
\theta^* = \arg \max_\theta \prod_{i=1}^N p_\theta(x^{(i)}) \quad \text{Maximize likelihood of dataset } \{x^{(i)}\}_{i=1}^N
\]

\[
= \arg \max_\theta \sum_{i=1}^N \log p_\theta(x^{(i)}) \quad \text{Maximize log-likelihood instead because sums are nicer}
\]

\[
p_\theta(x^{(i)}) = \int p_\theta(x^{(i)}, z) dz = \int p_\theta(x^{(i)} | z)p_\theta(z) dz \quad \text{Intractible integral } = (\]

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Math

$$\log p_\theta(x^{(i)})$$
Variational Autoencoder: Math

\[
\log p_\theta(x^{(i)}) = E_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)
\]
Variational Autoencoder: Math

\[
\log p_\theta(x^{(i)}) = E_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)
\]

\[
= E_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} \right] \quad \text{(Bayes’ Rule)}
\]
Variational Autoencoder: Math

$$\log p_\theta(x^{(i)}) = E_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= E_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} \right] \quad (\text{Bayes’ Rule})$$

$$= E_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z) q_\phi(z | x^{(i)})}{p_\theta(z | x^{(i)}) q_\phi(z | x^{(i)})} \right] \quad (\text{Multiply by constant})$$
Variational Autoencoder: Math

\[ \log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z) \]

\[ = \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} \right] \quad \text{(Bayes’ Rule)} \]

\[ = \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} \frac{q_\phi(z | x^{(i)})}{q_\phi(z | x^{(i)})} \right] \quad \text{(Multiply by constant)} \]

\[ = \mathbb{E}_z \left[ \log p_\theta(x^{(i)} | z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z | x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z | x^{(i)})}{p_\theta(z | x^{(i)})} \right] \quad \text{(Logarithms)} \]
Variational Autoencoder: Math

\[
\log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)
\]

\[
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} \right] \quad (\text{Bayes’ Rule})
\]

\[
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} q_\phi(z | x^{(i)}) \right] \quad (\text{Multiply by constant})
\]

\[
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)} | z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z | x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z | x^{(i)})}{p_\theta(z | x^{(i)})} \right] \quad (\text{Logarithms})
\]

\[
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)} | z) \right] - D_{KL}(q_\phi(z | x^{(i)}) || p_\theta(z)) + D_{KL}(q_\phi(z | x^{(i)}) || p_\theta(z | x^{(i)}))
\]
Variational Autoencoder: Math

\[ \log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z \mid x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad \text{\(p_\theta(x^{(i)})\) Does not depend on} \ z \]

\[ = \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})} \right] \quad \text{(Bayes’ Rule)} \]

\[ = \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})} q_\phi(z \mid x^{(i)}) \right] \quad \text{(Multiply by constant)} \]

\[ = \mathbb{E}_z \left[ \log p_\theta(x^{(i)} \mid z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})} \right] \quad \text{(Logarithms)} \]

\[ = \mathbb{E}_z \left[ \log p_\theta(x^{(i)} \mid z) \right] - D_{KL}(q_\phi(z \mid x^{(i)}) \parallel p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \parallel p_\theta(z \mid x^{(i)})) \]

\[ \mathcal{L}(x^{(i)}, \theta, \phi) \ \text{“ELBO”} \]
Variational Autoencoder: Math

\[ \log p_{\theta}(x^{(i)}) = E_{z \sim q_{\phi}(z|x^{(i)})} \left[ \log p_{\theta}(x^{(i)}) \right] \quad (p_{\theta}(x^{(i)}) \text{ Does not depend on } z) \]

\[ = E_{z} \left[ \log \frac{p_{\theta}(x^{(i)} | z)p_{\theta}(z)}{p_{\theta}(z | x^{(i)})} \right] \quad \text{(Bayes’ Rule)} \]

\[ = E_{z} \left[ \log \frac{p_{\theta}(x^{(i)} | z)p_{\theta}(z)}{p_{\theta}(z | x^{(i)})} \frac{q_{\phi}(z | x^{(i)})}{q_{\phi}(z | x^{(i)})} \right] \quad \text{(Multiply by constant)} \]

\[ = E_{z} \left[ \log p_{\theta}(x^{(i)} | z) \right] - E_{z} \left[ \log \frac{q_{\phi}(z | x^{(i)})}{p_{\theta}(z)} \right] + E_{z} \left[ \log \frac{q_{\phi}(z | x^{(i)})}{p_{\theta}(z | x^{(i)})} \right] \quad \text{(Logarithms)} \]

\[ = E_{z} \left[ \log p_{\theta}(x^{(i)} | z) \right] - D_{KL}(q_{\phi}(z | x^{(i)}) || p_{\theta}(z)) + D_{KL}(q_{\phi}(z | x^{(i)}) || p_{\theta}(z | x^{(i)})) \geq 0 \]

\[ \mathcal{L}(x^{(i)}, \theta, \phi) \quad \text{“ELBO”} \]
Variational Autoencoder: Math

\[
\log p_\theta(x^{(i)}) = E_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)
\]

\[
= E_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} \right] \quad \text{(Bayes’ Rule)}
\]

\[
= E_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} \frac{q_\phi(z | x^{(i)})}{q_\phi(z | x^{(i)})} \right] \quad \text{(Multiply by constant)}
\]

\[
= E_z \left[ \log p_\theta(x^{(i)} | z) \right] - E_z \left[ \log \frac{q_\phi(z | x^{(i)})}{p_\theta(z)} \right] + E_z \left[ \log \frac{q_\phi(z | x^{(i)})}{p_\theta(z | x^{(i)})} \right] \quad \text{(Logarithms)}
\]

\[
= E_z \left[ \log p_\theta(x^{(i)} | z) \right] - D_{KL}(q_\phi(z | x^{(i)}) || p_\theta(z)) + D_{KL}(q_\phi(z | x^{(i)}) || p_\theta(z | x^{(i)}))
\]

\[
\mathcal{L}(x^{(i)}, \theta, \phi) \quad \text{“ELBO”}
\]

\[
\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi)
\]

Variational lower bound (ELBO)

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Math

\[
\log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)
\]

\[
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})} \right] \quad \text{(Bayes’ Rule)}
\]

\[
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})} \frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})} \right] \quad \text{(Multiply by constant)}
\]

\[
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)} \mid z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})} \right] \quad \text{(Logarithms)}
\]

\[
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)} \mid z) \right] - D_{KL}(q_\phi(z \mid x^{(i)}) || p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) || p_\theta(z \mid x^{(i)})) \geq 0
\]

\[
\mathcal{L}(x^{(i)}, \theta, \phi) \quad \text{“ELBO”}
\]

\[
\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi)
\]

Variational lower bound (ELBO)

\[
\theta^*, \phi^* = \arg \max_{\theta, \phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)}, \theta, \phi)
\]

Training: Maximize lower bound

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Variational Autoencoder: Math

\[
\log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \\
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} \cdot \frac{q_\phi(z | x^{(i)})}{q_\phi(z | x^{(i)})} \right] \\
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)} | z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z | x^{(i)})}{p_\theta(z | x^{(i)})} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z | x^{(i)})}{p_\theta(z | x^{(i)})} \right] \\
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)} | z) \right] - D_{KL}(q_\phi(z | x^{(i)}) \| p_\theta(z)) + D_{KL}(q_\phi(z | x^{(i)}) \| p_\theta(z | x^{(i)})) \\
\geq 0
\]

\( \mathcal{L}(x^{(i)}, \theta, \phi) \) “ELBO”

\[
\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi)
\]

Variational lower bound (ELBO)

Training: Maximize lower bound

\( \theta^*, \phi^* = \arg \max_{\theta, \phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)}, \theta, \phi) \)
Variational Autoencoder: Math

\[ \log p_\theta(x^{(i)}) = E_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \]

(p_\theta(x^{(i)}) \text{ Does not depend on } z)

= \mathbb{E}_z \left[ \log \left( \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})} \right) \right] \quad \text{(Bayes’ Rule)}

= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})} \right] q_\phi(z \mid x^{(i)}) \quad \text{(Multiply by constant)}

= \mathbb{E}_z \left[ \log p_\theta(x^{(i)} \mid z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})} \right]

\[ \mathcal{L}(x^{(i)}, \theta, \phi) \quad \text{“ELBO”} \]

\[ \log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi) \]

Variational lower bound (ELBO)

\[ \theta^*, \phi^* = \arg \max_{\theta,\phi} \sum_{i=1}^N \mathcal{L}(x^{(i)}, \theta, \phi) \]

Training: Maximize lower bound

Latent states should follow the prior

Reconstruct the input data
Variational Autoencoder: Math

\[ \log p_\theta(x^{(i)}) = E_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \]  
\[ = E_z \left[ \log \frac{p_\theta(x^{(i)} | z)p_\theta(z)}{p_\theta(z | x^{(i)})} \right] \]  
\[ = E_z \left[ \log p_\theta(x^{(i)} | z) \right] - E_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})} \right] + E_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})} \right] \]  

\[ \mathcal{L}(x^{(i)}, \theta, \phi) \quad \text{“ELBO”} \]

\[ \log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi) \]

Variational lower bound (ELBO)

\[ \theta^*, \phi^* = \arg \max_{\theta, \phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)}, \theta, \phi) \]

Training: Maximize lower bound

Reconstruct the input data

Sampling with reparam. trick (see paper)

Latent states should follow the prior

Reconstruct the input data

Sampling with reparam. trick (see paper)
Variational Autoencoder: Math

\[ \log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z \mid x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \]

Reconstruct the input data

\[ \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)} \mid z) p_\theta(z)}{p_\theta(z \mid x^{(i)})} \right] \]

(Multiply by constant)

\[ = \mathbb{E}_z \left[ \log q_\phi(z \mid x^{(i)}) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})} \right] \]

\[ \mathcal{L}(x^{(i)}, \theta, \phi) \quad \text{“ELBO”} \]

Variational lower bound (ELBO)

\[ \log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi) \]

Training: Maximize lower bound

\[ \theta^*, \phi^* = \arg \max_{\theta, \phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)}, \theta, \phi) \]

Latent states should follow the prior

Everything is Gaussian, closed form solution!

Sampling with reparam. trick (see paper)
Autoencoder Overview

- Traditional Autoencoders
  - Try to reconstruct input
  - Used to learn features, initialize supervised model
  - Not used much anymore
- Variational Autoencoders
  - Bayesian meets deep learning
  - Sample from model to generate images
Generative Adversarial Nets

Can we generate images with less math?

Random noise $\mathbf{z}$
Generative Adversarial Nets

Can we generate images with less math?

Generative Adversarial Nets

Can we generate images with less math?

Real or fake? (Discriminator)

Fake image (Generator)

Random noise

\[ y \]

\[ x \]

\[ z \]


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets

Can we generate images with less math?

Real or fake?  

Discriminator

Fake image

Generator

Random noise

y  

\[ x \]

\[ x \]

\[ z \]

Real image

Fake examples: from generator

Real examples: from dataset

Generative Adversarial Nets

Can we generate images with less math?

Train generator and discriminator jointly
After training, easy to generate images

Real or fake?

Discriminator

Fake image

Generator

Random noise


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets

Generated samples

Nearest neighbor from training set


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets

Generated samples (CIFAR-10)

Nearest neighbor from training set


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Multiscale


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Multiscale


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Multiscale


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Multiscale


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Multiscale


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Multiscale

Generative Adversarial Nets: Multiscale


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Multiscale

Discriminators work at every scale!

Denton et al, NIPS 2015
Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Multiscale

Train separate model per-class on CIFAR-10

Denton et al, NIPS 2015
Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Simplifying

Generator is an upsampling network with fractionally-strided convolutions
Discriminator is a convolutional network

Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

Generative Adversarial Nets: Simplifying


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Simplifying

Samples from the model look amazing!

Radford et al, ICLR 2016

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Simplifying

Interpolating between random points in latent space

Radford et al, ICLR 2016

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Vector Math

Samples from the model

Smiling woman

Neutral woman

Neutral man

Radford et al, ICLR 2016
Generative Adversarial Nets: Vector Math

Samples from the model

Smiling woman
Neutral woman
Neutral man

Average Z vectors, do arithmetic

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson

Radford et al, ICLR 2016
Generative Adversarial Nets: Vector Math

Samples from the model

Average Z vectors, do arithmetic

Smiling woman  Neutral woman  Neutral man

Radford et al, ICLR 2016

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Vector Math

Glasses man  No glasses man  No glasses woman

Radford et al, ICLR 2016

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Generative Adversarial Nets: Vector Math

Glasses man  No glasses man  No glasses woman

Woman with glasses

Radford et al, ICLR 2016

Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Putting everything together

Variational Autoencoder

Pixel loss

Putting everything together

Real or Generated

Discriminator network

Pixel loss

Variational Autoencoder

Putting everything together

Real or Generated

Discriminator network

Pixel loss

Variational Autoencoder

Putting everything together

Real or Generated

Discriminator network

Pixel loss

Variational Autoencoder

Features of real image

Features of reconstructed image


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Putting everything together

Real or Generated

Discriminator network

Pixel loss

Variational Autoencoder

\[
\begin{align*}
\mathbf{y} & \quad \mu^x \\
x & \quad \Sigma^x \\
\mathbf{z} & \quad \mu^z \\
\mathbf{x}_f & \quad \Sigma^z \\
x & \quad \mathbf{x}_f
\end{align*}
\]

Pretrained AlexNet

Features of real image

Features of reconstructed image

L2 loss


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Putting everything together

Samples from the model, trained on ImageNet


Slide courtesy of Fei-Fei Li, Andrej Karpathy & Justin Johnson
Recap

● Unsupervised learning
  ○ Autoencoders: Traditional / variational
  ○ Generative Adversarial Networks

● Very active research area
  ○ Follow arxiv.org updates for "daily news"