Topics in AI (CPSC 532S): Multimodal Learning with Vision, Language and Sound

Lecture 16: Generative Models [part 2]
Logistics

Project Proposals due Monday 11:59pm

- They are graded for completeness / development of the proposal
- They are not graded for quality of the idea

I will provide feedback on the proposals, but don’t wait for it to work on the projects

I will hold additional office hours for feedback on Friday … tentatively 4:30-5:30pm
PixelRNN and PixelCNN
Explicit Density model

Use chain rule to decompose likelihood of an image $x$ into product of (many) 1-d distributions

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$

then maximize likelihood of training data

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
**Explicit Density model**

Use chain rule to decompose likelihood of an image $x$ into product of (many) 1-d distributions

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$

then maximize likelihood of training data

Complex distribution over pixel values, so lets model using **neural network**

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Optional subtitle
Explicit Density model

Use chain rule to decompose likelihood of an image $x$ into product of (many) 1-d distributions

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$$

then maximize likelihood of training data

Complex distribution over pixel values, so lets model using neural network

Also requires defining ordering of “previous pixels”
Generate image pixels starting from the corner

Dependency on previous pixels model using an RNN (LSTM)

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Generate image pixels starting from the corner

Dependency on previous pixels model using an RNN (LSTM)
Generate image pixels starting from the corner

Dependency on previous pixels model using an RNN (LSTM)
PixelRNN

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Generate image pixels starting from the corner

Dependency on previous pixels model using an RNN (LSTM)
Generate image pixels starting from the corner

Dependency on previous pixels model using an RNN (LSTM)

**Problem:** sequential generation is slow
Still generate image pixels starting from the corner

Dependency on previous pixels now modeled using a CNN over context region

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Still generate image pixels starting from the corner

Dependency on previous pixels now modeled using a CNN over context region

**Training:** maximize likelihood of training images

\[ p(x) = \prod_{i=1}^{n} p(x_i|x_1, \ldots, x_{i-1}) \]
PixelCNN

Still generate image pixels starting from the corner

Dependency on previous pixels now modeled using a CNN over context region

**Training:** maximize likelihood of training images

\[
p(x) = \prod_{i=1}^{n} p(x_i|x_1, \ldots, x_{i-1})
\]

Generation is still slow (sequential), but learning is faster

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Generated Samples

32x32 CIFAR-10

32x32 ImageNet

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford

[ van der Oord et al., 2016 ]
PixelRNN and PixelCNN

Pros:
— Can explicitly compute likelihood \( p(x) \)
— Explicit likelihood of training data gives good evaluation metric
— Good samples

Con:
— Sequential generation => slow

Improving PixelCNN performance
— Gated convolutional layers
— Short-cut connections
— Discretized logistic loss
— Multi-scale
— Training tricks
— Etc…

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Multi-scale PixelRNN

Take sub-sampled pixels as additional input pixels

Can capture better global information (more visually coherent)
Multi-scale PixelRNN

[ van der Oord et al., 2016 ]

* slide from Hsiao-Ching Chang, Ameya Patil, Anand Bhattad
Conditional Image Generation

Similar to PixelRNN/CNN but conditioned on a high-level image description vector $h$

\[
p(x) = p(x_1, x_2, \ldots, x_{n^2})
\]

\[
p(x|h) = p(x_1, x_2, \ldots, x_{n^2}|h)
\]

* slide from Hsiao-Ching Chang, Ameya Patil, Anand Bhattad
Conditional Image Generation

African elephant

Sandbar

* slide from Hsiao-Ching Chang, Ameya Patil, Anand Bhattad

[ van der Oord et al., 2016 ]
Attention RNN: Structured Spatial Attention Mechanism

Siddhesh Khandelwal
Leonid Sigal
Attention is widely used in vision: helps identify relevant regions of the image.

Q: What color is a hydrant?

A: It is red
AttentionRNN: Structured Spatial Attention

Novel **autoregressive attention mechanism** that can encode structural dependencies among attention values

- Inspired by diagonal Bi-LSTM architecture from PixelRNN
- Spatial attention values are generated sequentially
- Image is traversed diagonally from top-left to bottom-right
**AttentionRNN**: Structured Spatial Attention

Each **attention value** depends on

- Local image context
- Previously generated attention values

Novel **autoregressive attention mechanism** that can encode structural dependencies among attention values

- Inspired by diagonal Bi-LSTM architecture from PixelRNN
- Spatial attention values are generated sequentially
- Image is traversed diagonally from top-left to bottom-right
注意力 RNN（Attention RNN）: 结构化空间注意力

每个注意力值依赖于

- 局部图像上下文
- 前期生成的注意力值
AttentionRNN: Structured Spatial Attention

Each attention value depends on

- Local image context
- Previously generated attention values

![Image](image.png)

- **k x k convolution**

![Attention Mask](attention_mask.png)
AttentionRNN: Structured Spatial Attention

Each **attention value** depends on

- Local image context
- Previously generated attention values
AttentionRNN: Structured Spatial Attention

Each **attention value** depends on

- Local image context
- *Previously generated attention values*

![Image](image.png)

**k x k** convolution

---

**Image**

**Attention Mask**
AttentionRNN: Structured Spatial Attention

Each attention value depends on

- Local image context
- Previously generated attention values
AttentionRNN: Structured Spatial Attention

Each attention value depends on

- Local image context
- Previously generated attention values

**k x k** convolution

Image

LSTM with 2x1 kernel

Attention Mask
AttentionRNN: Structured Spatial Attention

Each **attention value** depends on

- Local image context
- Previously generated attention values
**Experiments: Visual Digit Prediction**

**Task:** Given an image, predict a digit number specified by a query color.
Variational Autoencoders (VAE)
So far …

PixelCNNs define tractable density function, optimize likelihood of training data:

\[ p(x) = \prod_{i=1}^{n} p(x_i|x_1, \ldots, x_{i-1}) \]
So far …

PixelCNNs define tractable density function, optimize likelihood of training data:

\[ p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1}) \]

VAEs define intractable density function with latent variables z (that we need to marginalize):

\[ p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz \]

cannot optimize directly, derive and optimize lower bound of likelihood instead

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Autoencoders Reminder …

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

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

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

\[ z \text{ usually smaller than } x \]

(dimensionality reduction)

Originally: Linear + nonlinearity (sigmoid)

Later: Deep, fully-connected

Later: ReLU CNN

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Autoencoders Reminder …

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data.

\[ z \text{ usually smaller than } x \] (dimensionality reduction)

Want features that capture meaningful factors of variation

\[ \text{Input data} \]

\[ \text{Features} \]

\[ \text{Encoder} \]

\[ \text{Output features} \]

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

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Autoencoders Reminder …

Train such that features can reconstruct original data best they can

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Autoencoders Reminder ...

Train such that features can reconstruct original data best they can.

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

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Autoencoders

Reminder ...

L2 Loss function:
\[ \| x - \hat{x} \|^2 \]

Reconstructed data

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

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Autoencoders Reminder ...

L2 Loss function:

$$\|x - \hat{x}\|^2$$

Doesn’t use labels!

Input data

Features

Encoder

Decoder

Reconstructed data

Reconstructed input data

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

Input data

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Autoencoders Reminder …

Loss function (e.g., softmax)

\( \hat{y} \) \( y \)

Classifier

Features

Encoder

Input data

\( x \)

Fine-tune encoder jointly with classifier

Train for final task (sometimes with small data)

bird  plane
dog  deer  truck

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoders

Probabilistic spin on autoencoder - will let us sample from the model to generate

Assume training data is generated from underlying unobserved (latent) representation $z$

- Sample from true **conditional** $p_{\theta^*}(x \mid z^{(i)})$
- Sample from true **prior** $p_{\theta^*}(z)$
Variational Autoencoders

Probabilistic spin on autoencoder - will let us sample from the model to generate

Assume training data is generated from underlying unobserved (latent) representation $z$

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

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

**Intuition**: $x$ is an image, $z$ is latent factors used to generate $x$ (e.g., attributes, orientation, etc.)

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford [ Kingma and Welling, 2014 ]
Variational Autoencoders

We want to **estimate the true parameters** $\theta^*$ of this generative model.

$$p_{\theta^*}(x \mid z^{(i)})$$

$$p_{\theta^*}(z)$$

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
We want to **estimate the true parameters** $\theta^*$ of this generative model.

How do we **represent** this model?

Sample from true **conditional**

$$p_{\theta^*}(x \mid z^{(i)})$$

Sample from true **prior**

$$p_{\theta^*}(z)$$
Variational Autoencoders

We want to **estimate the true parameters** $\theta^*$ of this generative model.

**How do we represent this model?**

Sample from true **conditional**

\[ p_{\theta^*}(x \mid z^{(i)}) \]

Choose prior $p(z)$ to be simple, e.g., Gaussian
Reasonable for latent attributes, e.g., pose, amount of smile

Sample from true **prior**

\[ p_{\theta^*}(z) \]
Variational Autoencoders

We want to **estimate the true parameters** $\theta^*$ of this generative model.

How do we **represent** this model?

Choose prior $p(z)$ to be simple, e.g., Gaussian. Reasonable for latent attributes, e.g., pose, amount of smile.

Conditional $p(x|z)$ is complex (generates an image). Represent with Neural Network.

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoders

We want to **estimate the true parameters** $\theta^*$ of this generative model.

How do we **train** this model?

Sample from true **conditional**

$$p_{\theta^*}(x \mid z^{(i)})$$

Sample from true **prior**

$$p_{\theta^*}(z)$$

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
We want to **estimate the true parameters** $\theta^*$ of this generative model.

How do we **train** this model?

Remember the strategy from earlier — learn model parameters to maximize likelihood of training data:

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

(now with latent $z$ that we need to marginalize)

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
We want to **estimate the true parameters** \( \theta^* \) of this generative model.

How do we **train** this model?

Remember the strategy from earlier — learn model parameters to maximize likelihood of training data:

\[
p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz
\]

(now with latent \( z \) that we need to marginalize)

---

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, [cs231n Stanford](https://cs231n.stanford.edu/)
Variational Autoencoders

We want to **estimate the true parameters** $\theta^*$ of this generative model.

**How do we train** this model?

Remember the strategy from earlier — learn model parameters to maximize likelihood of training data:

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

(now with latent $z$ that we need to marginalize)

---

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Intractability in Variational Autoencoder

Data likelihood: \[ p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz \]

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford

[ Kingma and Welling, 2014 ]
Intractability in Variational Autoencoder

Data likelihood: \( p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz \)

Simple Gaussian Prior

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Intractability in Variational Autoencoder

Data likelihood: $p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz$

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
**Intractability** in Variational Autoencoder

Data likelihood: \[ p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz \]

[Kingma and Welling, 2014]

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
**Intractability** in Variational Autoencoder

Data likelihood: \[ p_\theta(x) = \int p_\theta(z)p_\theta(x|z) dz \]

Posterior density is also intractable: \[ p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)} \]

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, [cs231n Stanford](#)
Intractability in Variational Autoencoder

Data likelihood: \( p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz \)

Posterior density is also intractable: \( p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)} \)

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
**Intractability in Variational Autoencoder**

Kingma and Welling, 2014

Data **likelihood**: \[ p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz \]

Simple **Gaussian** Prior

Posterior density is also intractable: \[ p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)} \]

Solution: In addition to decoder network modeling \( p_\theta(x|z) \), define additional encoder network \( q_\phi(z|x) \) that approximates \( p_\theta(z|x) \)

— Will see that this allows us to derive a lower bound on the data likelihood that is tractable, which we can optimize

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder

Since we are modeling probabilistic generation of data, encoder and decoder networks are probabilistic (they model distributions)

Encoder Network

$$\mathcal{q}_\phi(z|x)$$
(parameters $\phi$)

$$\mu_z|x$$

$$\Sigma_z|x$$

$$x$$

Decoder Network

$$\mathcal{p}_\theta(x|z)$$
(parameters $\theta$)

$$\mu_x|z$$

$$\Sigma_x|z$$

$$z$$

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder

Since we are modeling probabilistic generation of data, encoder and decoder networks are probabilistic (they model distributions)

Why?  
Mean and (diagonal) covariance of $z \mid x$

Encoder Network  
$q_\phi(z \mid x)$  
(parameters $\phi$)  
$x$

Decoder Network  
$p_\theta(x \mid z)$  
(parameters $\theta$)  
$z$

Mean and (diagonal) covariance of $x \mid z$
Variational Autoencoder

Since we are modeling probabilistic generation of data, encoder and decoder networks are probabilistic (they model distributions)

$$z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$$

$$x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$$

Encoder Network

$q_\phi(z|x)$

(parameters $\phi$)

Decoder Network

$p_\theta(x|z)$

(parameters $\theta$)

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder

Derivation of lower bound of the data likelihood

Now equipped with **encoder** and **decoder** networks, let’s see (log) data likelihood:

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

Taking expectation with respect to \(z\) (using encoder network) will come in handy later

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
**Variational Autoencoder**

Derivation of lower bound of the data likelihood

Now equipped with **encoder** and **decoder** networks, let’s see (log) data likelihood:

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

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

[Kingma and Welling, 2014]

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder

Derivation of lower bound of the data likelihood

Now equipped with encoder and decoder networks, let’s see (log) data likelihood:

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

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

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

Derivation of lower bound of the data likelihood

Now equipped with **encoder** and **decoder** networks, let's see (log) data likelihood:

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

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

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

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

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**
Derivation of lower bound of the data likelihood

Variational Autoencoder

Now equipped with encoder and decoder networks, let’s see (log) data likelihood:

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

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

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

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

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

Expectation with respect to z
(using encoder network) leads to nice KL terms

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder

Derivation of lower bound of the data likelihood

Now equipped with **encoder** and **decoder** networks, let's see (log) data likelihood:

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

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

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

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

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

Decoder network gives \( p_\theta(x|z) \), can compute estimate of this term through sampling. (Sampling differentiable through reparam. trick, see paper.)

This KL term (between Gaussians for encoder and z prior) has nice **closed-form solution**!

\( p_\theta(z|x) \) **intractable** (saw earlier), can’t compute this KL term :(.

But we know KL divergence always \( >= 0 \).

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**
Now equipped with **encoder** and **decoder** networks, let’s see (log) data likelihood:

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

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

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

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

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

**Tractable lower bound** which we can take gradient of and optimize! (\(p\theta(x|z)\) differentiable, KL term differentiable)

---

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder

Derivation of lower bound of the data likelihood

Now equipped with \textbf{encoder} and \textbf{decoder} networks, let’s see (log) data likelihood:

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

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

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

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

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

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

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

Variational lower bound ("ELBO")

Training: Maximize lower bound

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

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, \textbf{cs231n Stanford}
Variational Autoencoder

Derivation of lower bound of the data likelihood

Now equipped with **encoder** and **decoder** networks, let’s see (log) data likelihood:

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

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

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

Reconstruct Input Data
Make approximate posterior close to the prior

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

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

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

Variational lower bound (“ELBO”)

Training: Maximize lower bound

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

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder: Learning

Putting it all together:

maximizing the likelihood lower bound

\[ \mathbf{E}_z \left[ \log p_{\theta}(x^{(i)} | z) \right] - D_{KL}(q_{\phi}(z | x^{(i)}) \| p_{\theta}(z)) \]

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

Let's look at computing the bound (forward pass) for a given mini batch of input data

Input Data \(X\)

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Putting it all together:
maximizing the likelihood lower bound

\[
\mathbb{E}_z \left[ \log p_\theta(x^{(i)} | z) \right] - D_{KL}(q_\phi(z | x^{(i)}) \| p_\theta(z)) \\
\mathcal{L}(x^{(i)}, \theta, \phi)
\]

Encoder network

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

Input Data

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder: Learning

Putting it all together:
maximizing the likelihood lower bound

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

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

Make approximate posterior distribution close to prior

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder: Learning

Putting it all together:
maximizing the likelihood lower bound

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

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

Make approximate posterior distribution close to prior

Sample \( z \) from \( z | x \sim \mathcal{N}(\mu_z | x, \Sigma_z | x) \)

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder: Learning

Putting it all together:
maximizing the likelihood lower bound

\[
\mathbb{E}_{z} \left[ \log p_{\theta}(x^{(i)} | z) \right] - D_{KL}(q_{\phi}(z | x^{(i)}) \| p_{\theta}(z))
\]

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

Make approximate posterior distribution close to prior

Encoder network \( q_{\phi}(z | x) \)

Input Data \( X \)

Sample \( z \) from \( z | x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x}) \)

Decoder network \( p_{\theta}(x | z) \)

\( \mu_{x|z} \) \( \Sigma_{x|z} \)
Variational Autoencoder: Learning

Putting it all together:
maximizing the likelihood lower bound

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

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

Maximize likelihood of original input being reconstructed

Sample \( z \) from \( z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x}) \)

Decoder network

\( p_{\theta}(x | z) \)

Make approximate posterior distribution close to prior

Sample \( x|z \) from \( x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z}) \)

Encoder network

\( q_{\phi}(z | x) \)

Input Data

\[ \hat{x} \]

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder: Learning

Putting it all together: maximizing the likelihood lower bound

Maximize likelihood of original input being reconstructed

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

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

Make approximate posterior distribution close to prior

Decoder network

\[ p_{\theta}(x | z) \]

Sample \( x | z \) from \( x | z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z}) \)

Encoder network

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

Sample \( z | x \) from \( z | x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x}) \)

For every minibatch of input data: compute this forward pass, and then backprop!

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder: Learning

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

what can happen without regularisation \[\times\] what we want to obtain with regularisation

https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73
Variational Autoencoder: Generating Data

Use decoder network and sample $z$ from prior

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

Sample $x \mid z$ from $x \mid z \sim \mathcal{N}(\mu_x \mid z, \Sigma_x \mid z)$

Decoder network $p_\theta(x \mid z)$
Variational Autoencoder: Generating Data

https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73
Variational Autoencoder: Generating Data

Use decoder network and sample \( z \) from prior

Data manifold for 2-d \( z \)

Sample \( x | z \) from \( x | z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z}) \)

Decoder network

\( p_\theta(x | z) \)

Sample \( z \) from \( z \sim \mathcal{N}(0, I) \)

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder: Generating Data

Diagonal prior on $z \Rightarrow$ independent latent variables

Different dimensions of $z$ encode interpretable factors of variation

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder: Generating Data

Diagonal prior on $z \Rightarrow$ independent latent variables

Different dimensions of $z$ encode interpretable factors of variation

Also good feature representation that can be computed using $q_\phi(z|x)$!

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Variational Autoencoder: Generating Data

32x32 CIFAR-10

Labeled Faces in the Wild

* slide from Fei-Fei Li, Justin Johnson, Serena Yeung, cs231n Stanford