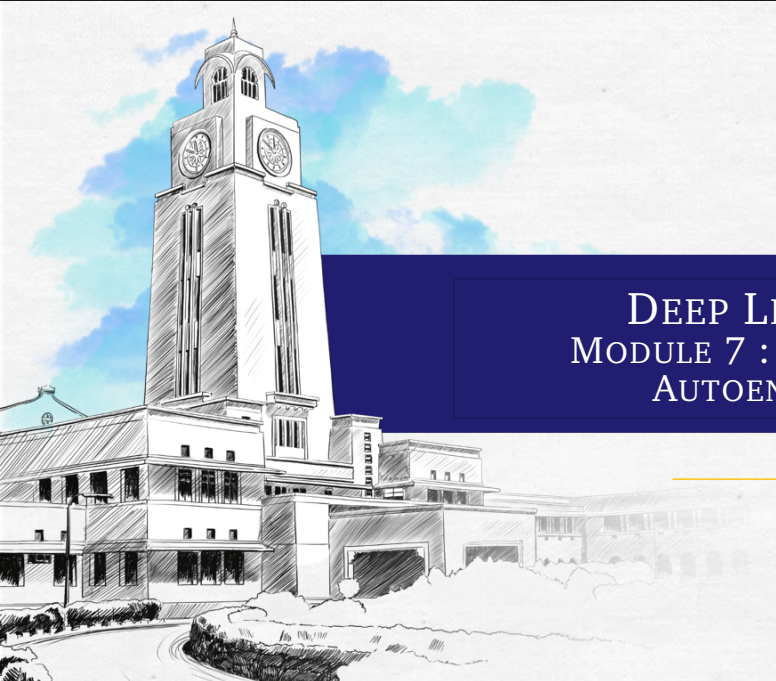


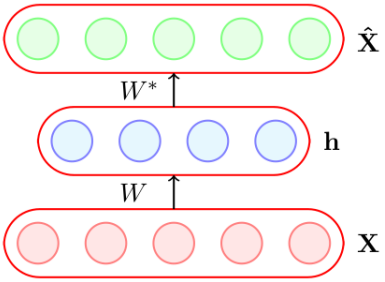
BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Deep Learning
Module 7 : Variational Autoencoders

Seetha Parameswaran
Asst Prof, BITS Pilani

The instructor is gratefully acknowledging the authors who made their course materials freely available online.

Autoencoder – Summary



$$\mathbf{h} = g(W\mathbf{X} + \mathbf{b})$$

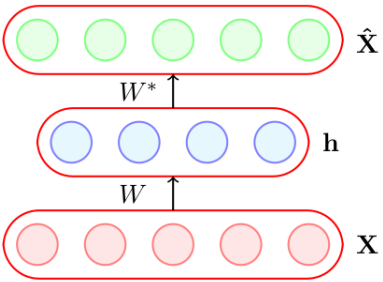
$$\hat{\mathbf{X}} = f(W^*\mathbf{h} + \mathbf{c})$$

- Before we start talking about VAEs, let us quickly revisit autoencoders
- An autoencoder contains an encoder which takes the input X and maps it to a hidden representation
- The decoder then takes this hidden representation and tries to reconstruct the input from it as $\hat{X}$
- The training happens using the following objective function

$$\min_{W,W^*,\mathbf{c},\mathbf{b}} \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{n} (\hat{x}_{ij} - x_{ij})^2$$

- where m is the number of training instances, $\{x_i\}_{i=1}^m$ and each $x_i \in R^n$ (x_{ij} is thus the j-th dimension of the i-th training instance)

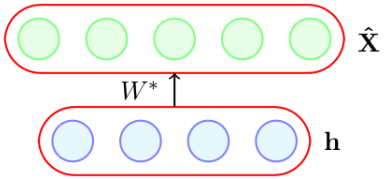
Autoencoder – Summary



$$\mathbf{h} = g(W\mathbf{X} + \mathbf{b})$$
$$\hat{\mathbf{X}} = f(W^*\mathbf{h} + \mathbf{c})$$

- Can we do generation with autoencoders ?
- In other words, once the autoencoder is trained can I remove the encoder, feed a hidden representation h to the decoder and decode a $\hat{X}$ from it ?
- In principle, yes! But in practice there is a problem with this approach
- h is a very high dimensional vector and only a few vectors in this space would actually correspond to meaningful latent representations of our input

Autoencoder – Summary



$$\hat{\mathbf{X}} = f(W^*\mathbf{h} + \mathbf{c})$$

- Ideally, we should only feed those values of h which are highly *likely*
- In other words, we are interested in sampling from $P(h|X)$ so that we pick only those h's which have a high probability
- But unlike RBMs, autoencoders do not have such a probabilistic interpretation
- They learn a hidden representation h but not a distribution $P(h|X)$
- Similarly the decoder is also deterministic and does not learn a distribution over X (given a h we can get a X but not $P(X|h)$)

Discriminative Models vs Generative Model

Discriminative Models

- Set of features X and Set of categories Y
- Learn mapping from X to Y
- eg: if features are wet nose and tongue out and no purr; then the per is a dog.
- Model $P(Y \mid X)$

Generative Models

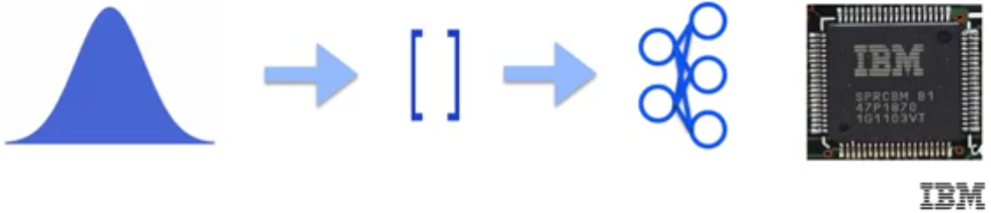
- Learn realistic representation of a class
- Learn mapping from Y to X
- eg: if we have to generate a dog image, then it should have wet nose and tongue out.
- Model $P(X \mid Y)$

Variational Autoencoder

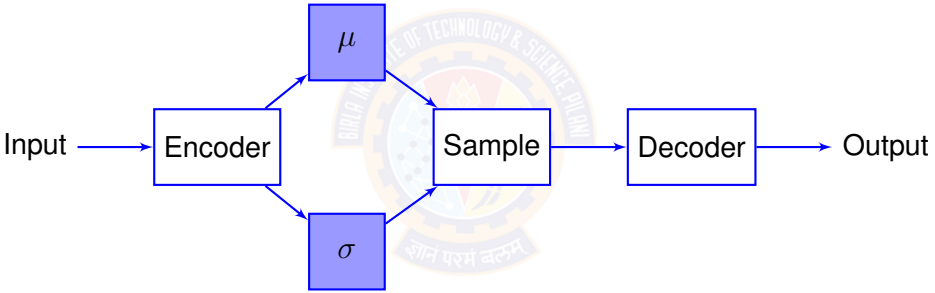
- Generative model
- Variational autoencoder generates a latent representation and then uses this representation to generate new images.
- Data are assumed to be represented by a set of normally-distributed latent factors.
- Encoder generates parameters of these parameters namely μ and σ.
- Images are generated by sampling from these distributions.

Variational Autoencoder

- Primary Goal: Generate images using decoder.
- Latent vector: each element drawn from a normal distribution.
- Parameters are learned by the encoder.
- Secondary Goal: Similar images be close together in latent space.

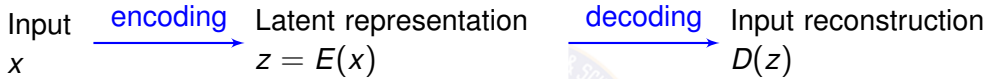


Variational Autoencoder

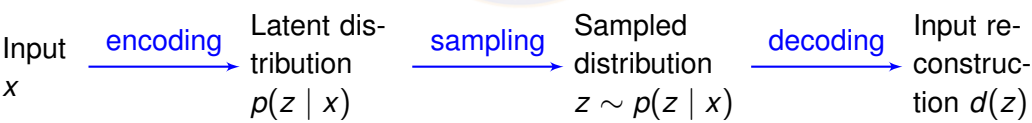


Autoencoder vs Variational Autoencoder

Autoencoder



Variational Autoencoder



Variational Autoencoder

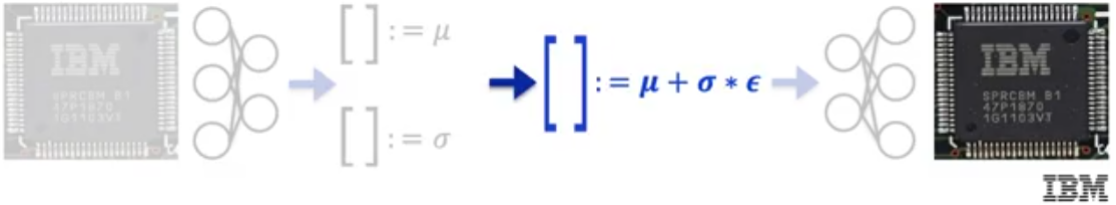
- Step 1 : Image is fed through the encoder network.

Variational Autoencoder

- Step 2 : The encoder network learns the parameters μ and σ.

Variational Autoencoder

- Step 3 : Combine the parameters μ and σ and normally distributed random noise $\epsilon \sim \mathcal{N}(0, 1)$.



Variational Autoencoder example

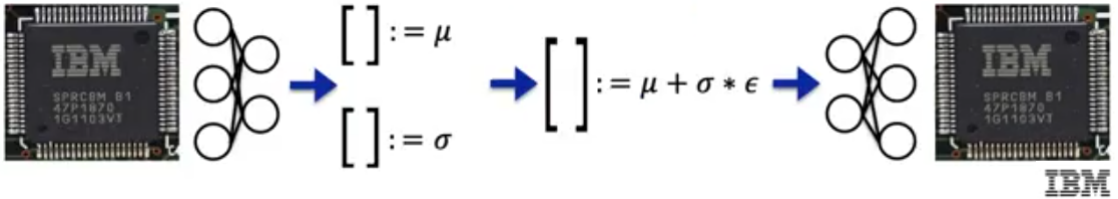
$$
\begin{aligned}
\text{Let} \quad \mu &= \begin{bmatrix} 0.7 \\ -0.6 \end{bmatrix} \\
\sigma &= \begin{bmatrix} 1.0 \\ 0.6 \end{bmatrix} \\
\epsilon &= \begin{bmatrix} 1.4 \\ 1.9 \end{bmatrix} \\
\textit{Decoder}_{in} &= \mu + \sigma * \epsilon \\
&= \begin{bmatrix} 0.7 \\ -0.6 \end{bmatrix} + \begin{bmatrix} 1.0 \\ 0.6 \end{bmatrix} * \begin{bmatrix} 1.4 \\ 1.9 \end{bmatrix} \\
&= \begin{bmatrix} 2.10 \\ 0.54 \end{bmatrix}
\end{aligned}
$$

Variational Autoencoder

- Step 4 : Feed the vector through the decoder.

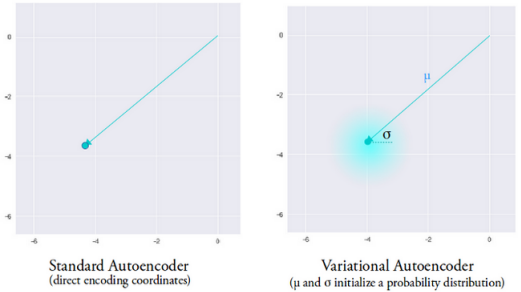
Variational Autoencoder

- Step 5 : The reconstructed image is generated using the decoder.



Variational Autoencoder

- Encoder learns two encodings or outputs two vectors of size n
  - a vector of means μ – controls where the encoding of an input should be centered around.
  - a vector of standard deviations σ – controls (area) how much from the mean the encoding can vary.
- Decoder learns that a single point in latent space refers to a sample of that class. It also learns that all nearby points refer to the same class as well.



Standard Autoencoder
(direct encoding coordinates)

Variational Autoencoder
(μ and σ initialize a probability distribution)

Variational Autoencoder – Loss Function

- VAE should learn to reconstruct the original images.
- The loss function will have 2 components.
  1. A penalty for NOT reconstructing the image correctly.
     - ⋆ Pixel-wise difference between the reconstructed image and the original image.
     - ⋆ Use Mean squared error (MSE)
     - ⋆ Use binary cross entropy
  2. A penalty for generating parameter vectors μ and σ that are different than 0 and 1, respectively.
     - ⋆ The difference between vectors produced by the encoder and the standard normal distribution parameters $\mathcal{N}(0, 1)$.
     - ⋆ Use Kullback–Leibler divergence (KL divergence).

Kullback–Leibler divergence

- The KL divergence between two probability distributions measures how much they diverge from each other.
- Minimizing the KL divergence means optimizing the probability distribution parameters (μ and σ) to closely resemble that of the target distribution.
- KL loss is equivalent to the sum of all the KL divergences between the component $X_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ in X, and the standard normal $\mathcal{N}(\mu = 0, \sigma = 1)$

$$\textit{KL loss} = \sum_{i=1}^{n} \sigma_i^2 + \mu_i^2 - \log(\sigma_i) - 1$$

KL Divergence example

$$\text{Compare} \quad \mu = \begin{bmatrix} 0.7 \\ -0.6 \end{bmatrix} \longleftrightarrow \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

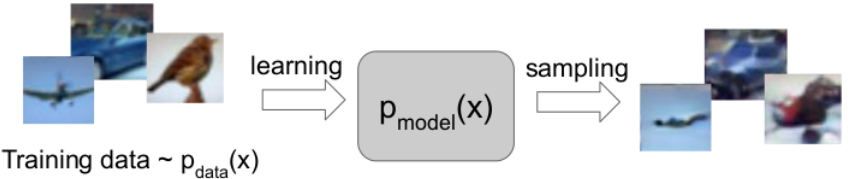
$$\text{Compare} \quad \log(\sigma) = \begin{bmatrix} 1.0 \\ 0.6 \end{bmatrix} \longleftrightarrow \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$\log(\sigma)$ compared to 0 since $\log(1) = 0$.
σ is interpreted as $\log(\sigma)$ since variance cannot be negative.

$$\text{KL Loss} \quad = \frac{1}{2} * \left(e^{\log(\sigma)} - \log(\sigma) - 1 + \mu^2\right)$$

Generative Model

Given training data, generate new samples from same distribution

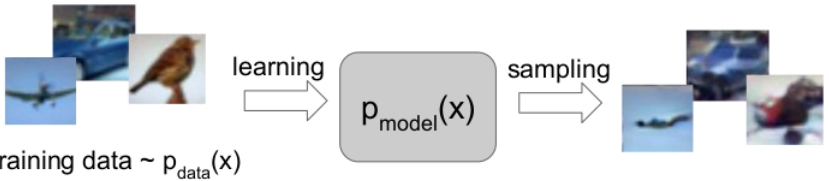


Objectives:

1. Learn $p_{model}(x)$ that approximates $p_{data}(x)$
2. **Sampling new x from $p_{model}(x)$**

Generative Model

Given training data, generate new samples from same distribution



Formulate as density estimation problems:

- **Explicit density estimation**: explicitly define and solve for $p_{model}(x)$
- **Implicit density estimation**: learn model that can sample from $p_{model}(x)$ **without explicitly defining it.**

Generative Model

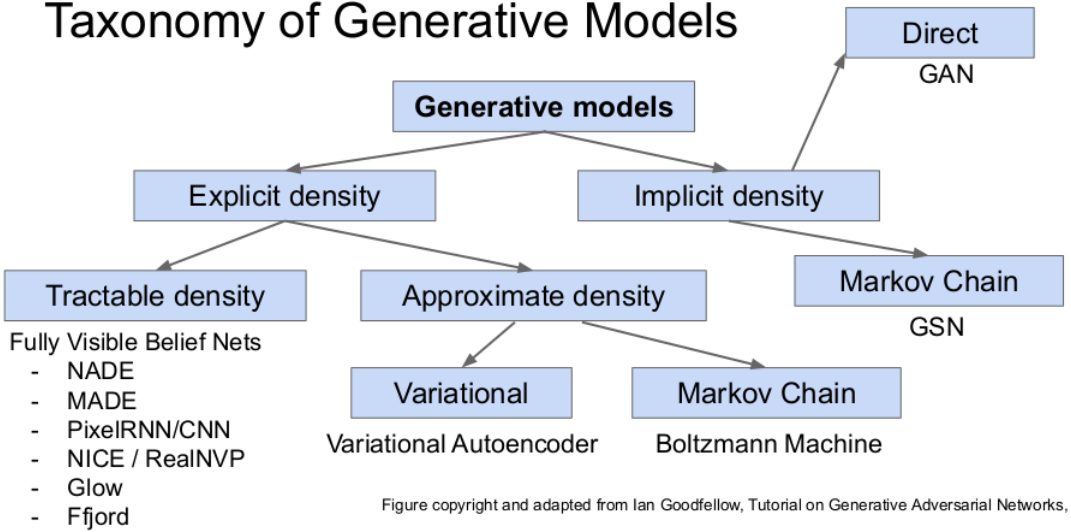


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

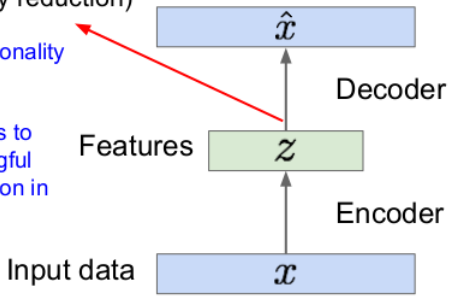
Autoencoders – Recap

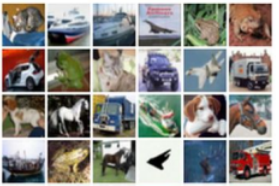
Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

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

Q: Why dimensionality reduction?

A: Want features to capture meaningful factors of variation in data

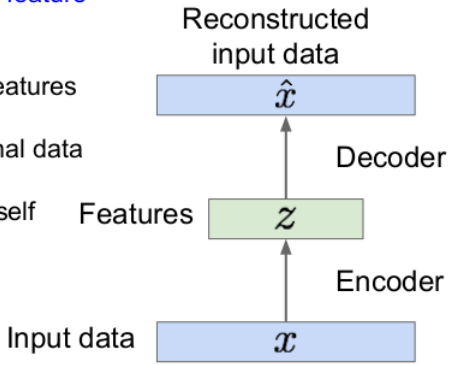


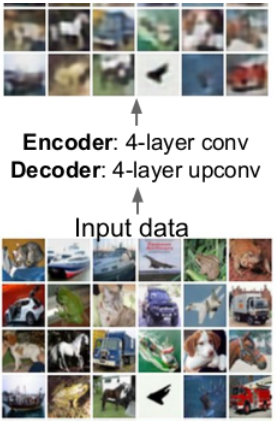


Autoencoders – Recap

How to learn this feature representation?

Train such that features can be used to reconstruct original data
"Autoencoding" - encoding input itself

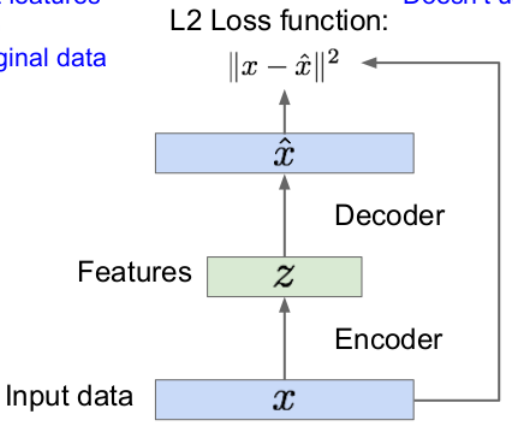


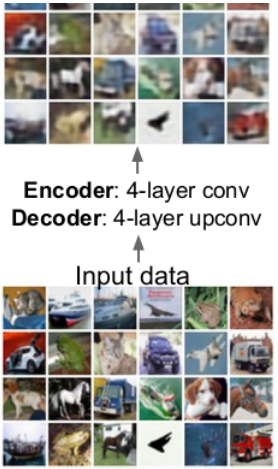


Autoencoders – Recap

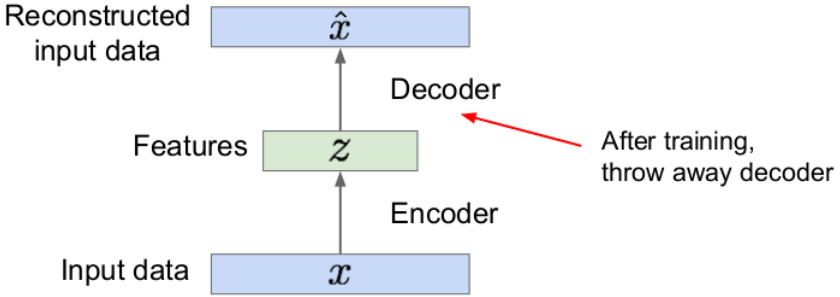
Train such that features can be used to reconstruct original data

Doesn't use labels!



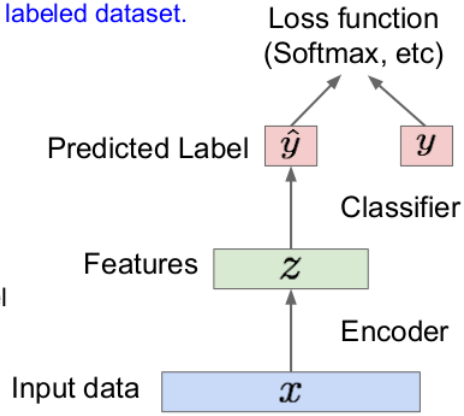


Autoencoders – Recap



Autoencoders – Recap

Transfer from large, unlabeled dataset to small, labeled dataset.



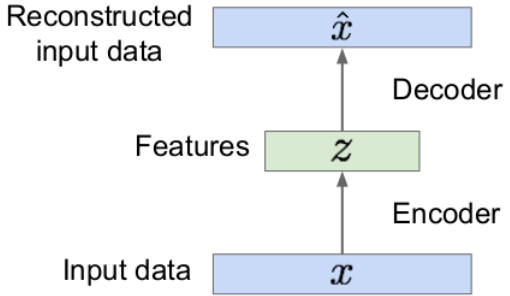
Encoder can be used to initialize a **supervised** model

Fine-tune encoder jointly with classifier

bird plane
dog deer truck

Train for final task (sometimes with small data)

Autoencoders – Recap



Autoencoders can reconstruct data, and can learn features to initialize a supervised model

Features capture factors of variation in training data.

But we can't generate new images from an autoencoder because we don't know the space of z.

How do we make autoencoder a **generative model**?

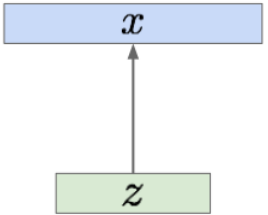
Variational Autoencoders

Probabilistic spin on autoencoders - will let us sample from the model to generate data!

Assume training data $\{x^{(i)}\}_{i=1}^{N}$ is generated from the distribution of unobserved (latent) representation **z**

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

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



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

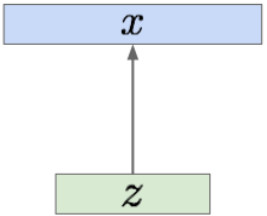
Variational Autoencoders

Probabilistic spin on autoencoders - will let us sample from the model to generate data!

Assume training data $\{x^{(i)}\}_{i=1}^{N}$ is generated from the distribution of unobserved (latent) representation **z**

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

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



Intuition (remember from autoencoders!): **x** is an image, **z** is latent factors used to generate **x:** attributes, orientation, etc.

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

Variational Autoencoders

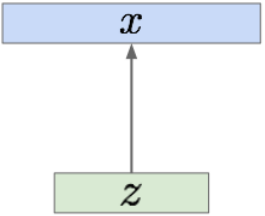
We want to estimate the true parameters θ^* of this generative model given training data x.

Sample from true conditional

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

Sample from true prior

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



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

Variational Autoencoders

We want to estimate the true parameters θ^* of this generative model given training data x.

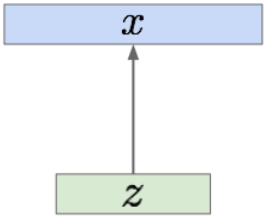
How should we represent this model?

Sample from true conditional

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

Sample from true prior

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

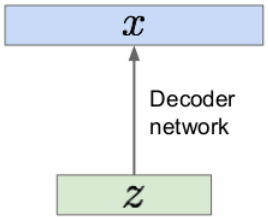


Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

Variational Autoencoders

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

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



We want to estimate the true parameters θ^* of this generative model given training data x.

How to train the model?

Learn model parameters to maximize likelihood of training data

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

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

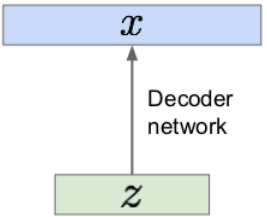
Variational Autoencoders

Sample from true conditional

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

Sample from true prior

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



We want to estimate the true parameters θ^* of this generative model given training data x.

How to train the model?

Learn model parameters to maximize likelihood of training data

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

Q: What is the problem with this?

Intractable!

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

Variational Autoencoders: Intractability

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

✔

Simple Gaussian prior

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

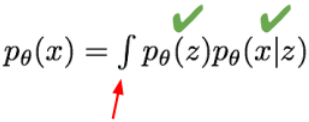
Variational Autoencoders: Intractability

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

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

Variational Autoencoders: Intractability

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



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

Variational Autoencoders: Intractability

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

Posterior density: $p_\theta(z|x) = p_\theta(x|z) p_\theta(z) / p_\theta(x)$

Intractable data likelihood

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

Variational Autoencoders

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

Posterior density also intractable: $p_\theta(z|x) = p_\theta(x|z) p_\theta(z) / p_\theta(x)$

Solution: In addition to modeling $p_\theta(x|z)$, learn $q_\phi(z|x)$ that approximates the true posterior $p_\theta(z|x)$.

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

Variational inference is to approximate the unknown posterior distribution from only the observed data x

Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

Variational Autoencoders

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && \text{(Bayes' Rule)}\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && \text{(Multiply by constant)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] && \text{(Logarithms)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}
$$

Decoder network gives $p_\theta(x|z)$, can compute estimate of this term through sampling (need some trick to differentiate through sampling).

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.

Variational Autoencoders

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

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

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

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

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

We want to maximize the data likelihood

Decoder network gives $p_\theta(x|z)$, can compute estimate of this term through sampling.

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.

Variational Autoencoders

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

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

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

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

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}$$

We want to maximize the data likelihood

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

Variational Autoencoders

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

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

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

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

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}$$

Decoder: reconstruct the input data

Encoder: make approximate posterior distribution close to prior

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

Variational Autoencoders

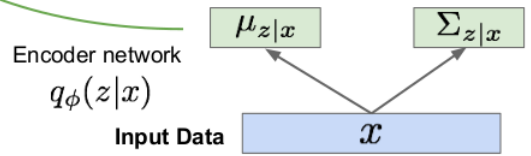
Putting it all together: maximizing the likelihood lower bound

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

$$D_{KL}(\mathcal{N}(\mu_{z|x}, \Sigma_{z|x})||\mathcal{N}(0, I))$$

Have analytical solution

Make approximate posterior distribution close to prior

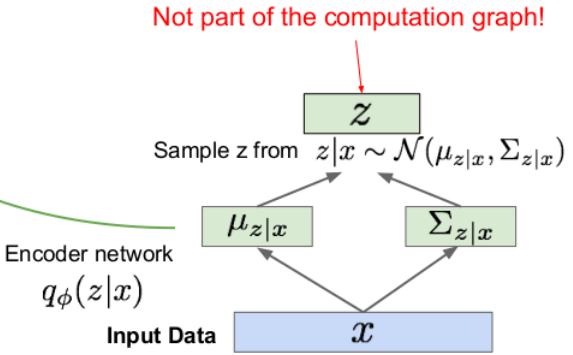


Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

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

Make approximate posterior distribution close to prior



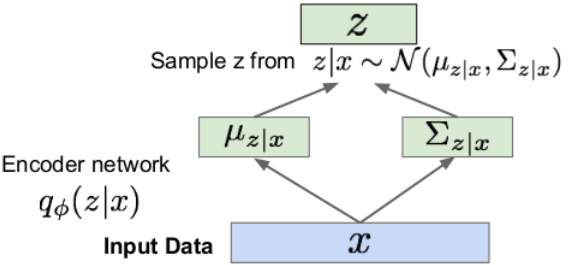
Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

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

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

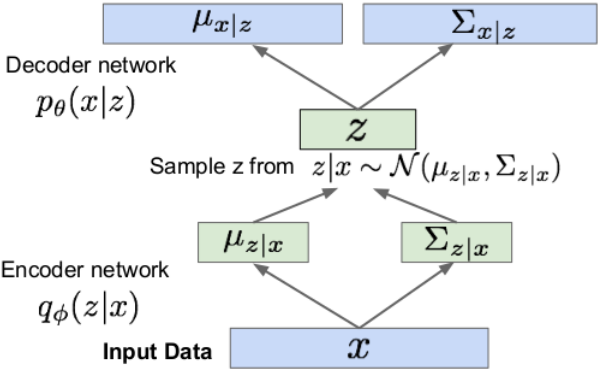
$$z = \mu_{z|x} + \epsilon\sigma_{z|x}$$



Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

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

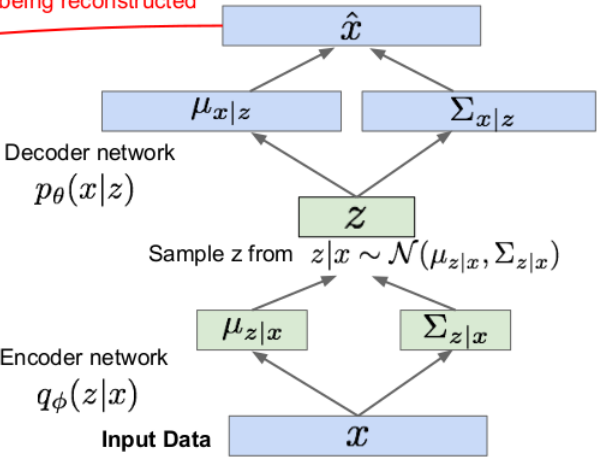


Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

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

Maximize likelihood of original input being reconstructed

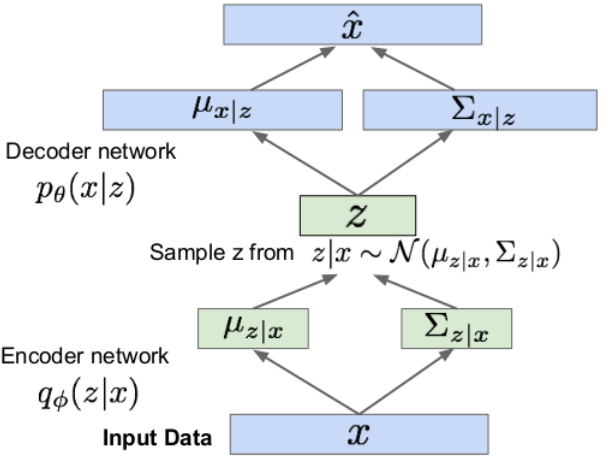


Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

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

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



Variational Autoencoders: Generating Dat

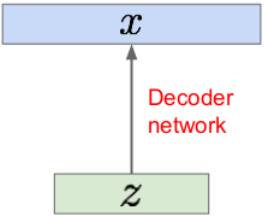
Our assumption about data generation process

Sample from true conditional

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

Sample from true prior

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



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

Variational Autoencoders: Generating Dat

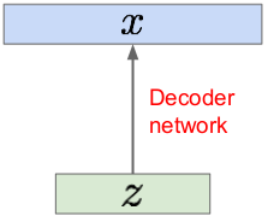
Our assumption about data generation process

Sample from true conditional

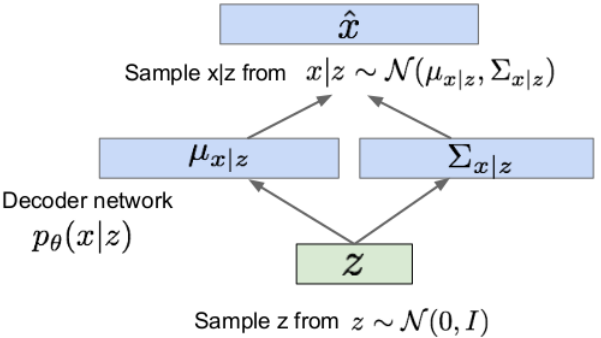
$p_{\theta^*}(x \mid z^{(i)})$

Sample from true prior

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



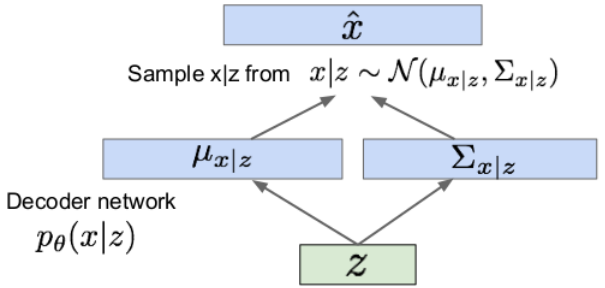
Now given a trained VAE: use decoder network & sample z from prior!

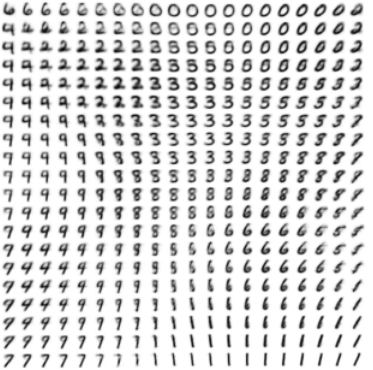


Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

Variational Autoencoders: Generating Dat

Use decoder network. Now sample z from prior!

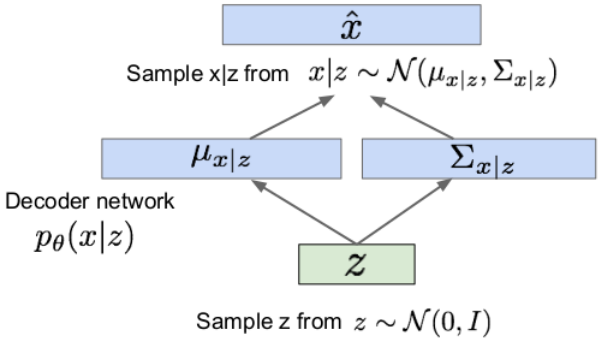




Kingma and Welling, “Auto-Encoding Variational Bayes”, ICLR 2014

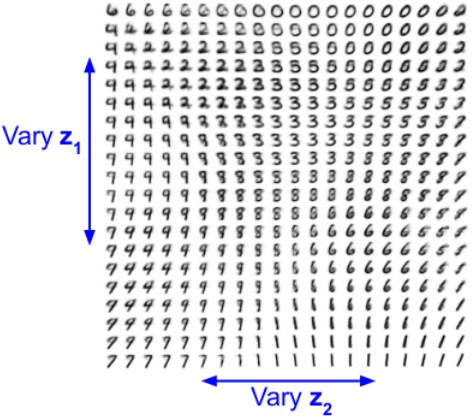
Variational Autoencoders: Generating Dat

Use decoder network. Now sample z from prior!



Kingma and Welling, "Auto-Encoding Variational Bayes", ICLR 2014

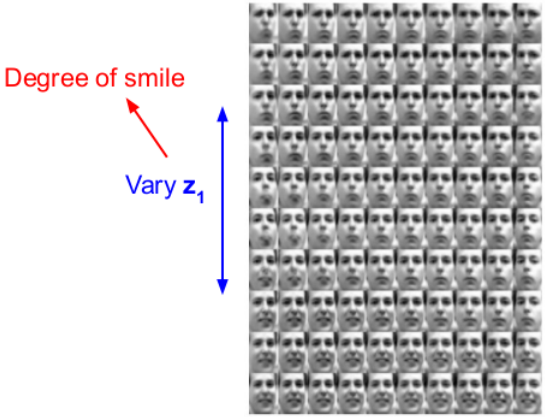
Data manifold for 2-d **z**



Variational Autoencoders: Generating Dat

Diagonal prior on **z**
=> independent
latent variables

Different
dimensions of **z**
encode
interpretable factors
of variation



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

Variational Autoencoders

Probabilistic spin to traditional autoencoders => allows generating data
Defines an intractable density => derive and optimize a (variational) lower bound

Pros:

- Principled approach to generative models
- Interpretable latent space.
- Allows inference of q(z|x), can be useful feature representation for other tasks

Cons:

- Maximizes lower bound of likelihood: okay, but not as good evaluation as PixelRNN/PixelCNN
- Samples blurrier and lower quality compared to state-of-the-art (GANs)

Active areas of research:

- More flexible approximations, e.g. richer approximate posterior instead of diagonal Gaussian, e.g., Gaussian Mixture Models (GMMs), Categorical Distributions.
- Learning disentangled representations.

References

1. Deep Learning by Ian Goodfellow, Yoshua Bengio, Aaron Courville
   https://www.deeplearningbook.org/
2. Deep Learning with Python by Francois Chollet.
   https://livebook.manning.com/book/deep-learning-with-python/

Thank You!