COMS 4995 Lecture 12: Reversible Models

Richard Zemel

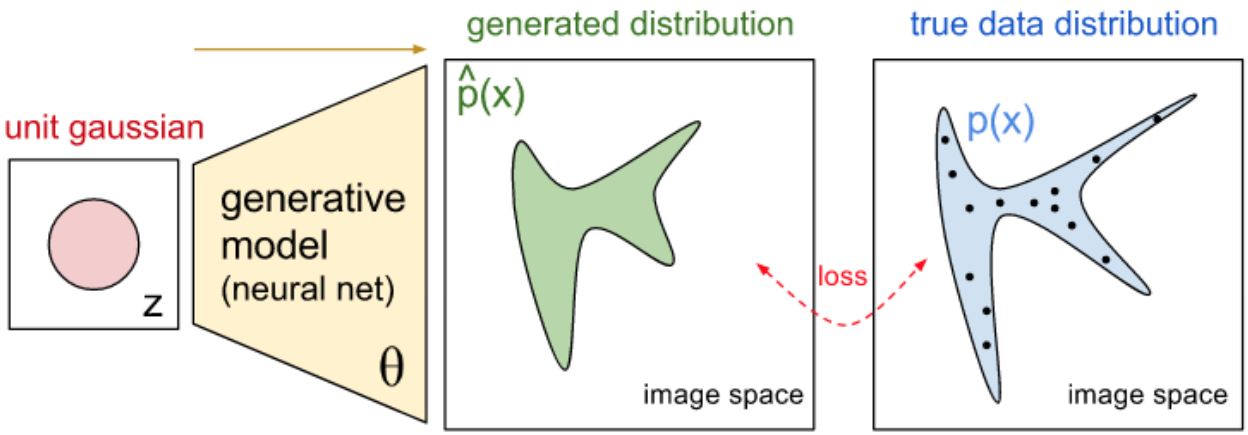
Overview

- In generative modeling, we'd like to train a network that models a distribution, such as a distribution over images.
- We have seen a few approaches to generative modeling:
  - Autoregressive models
  - Generative adversarial networks (last lecture)
  - Reversible architectures (this lecture)
  - Variational autoencoders (next lecture)

  All four approaches have different pros and cons.

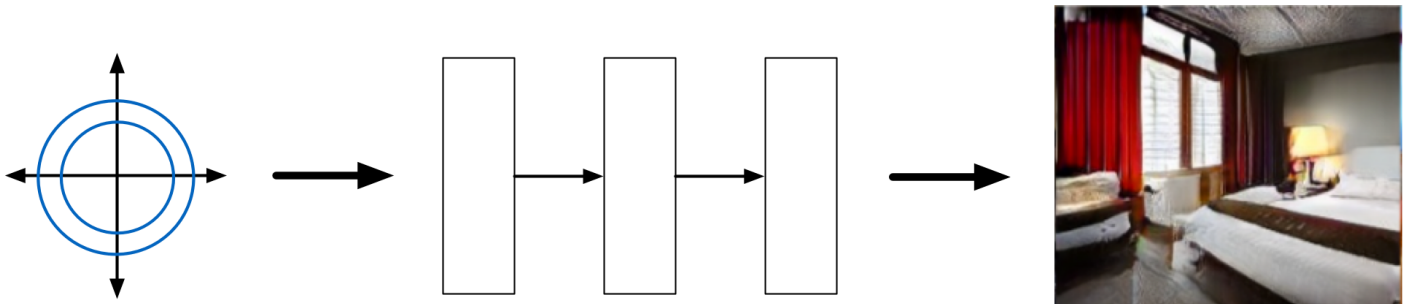
Generator Networks

- Start by sampling the code vector **z** from a fixed, simple distribution (e.g. spherical Gaussian)
- The generator network computes a differentiable function G mapping **z** to an **x** in data space



https://blog.openai.com/generative-models/

Generator Networks



Each dimension of the code vector is sampled independently from a simple distribution, e.g. Gaussian or uniform.

This is fed to a (deterministic) generator network.

The network outputs an image.

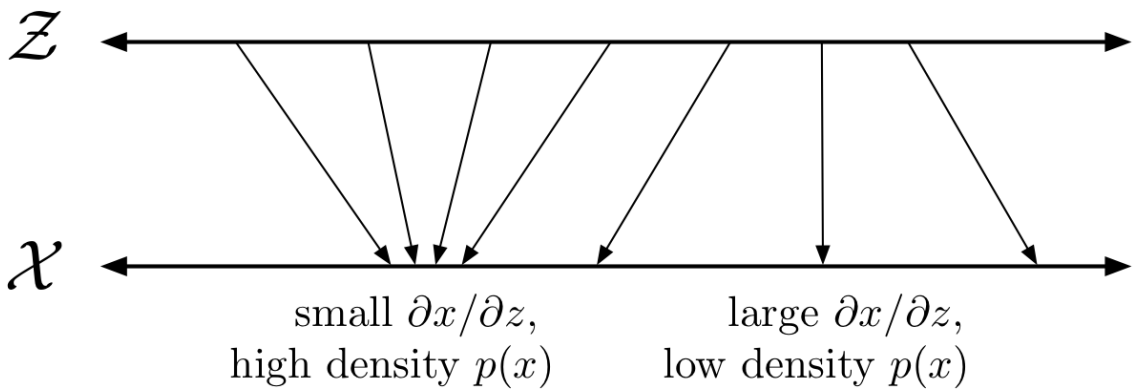
- We have seen how to learn generator networks by training a discriminator in GANs.
- Problem:
  - Learning can be very unstable. Need to tune many hyperparameters.
  - No direct evaluation metric to assess the trained generator networks.
- Idea: learn the generator directly via change of variables. (Calculus!)

Change of Variables Formula

- Let f denote a differentiable, bijective mapping from space $\mathcal{Z}$ to space $\mathcal{X}$. (I.e., it must be 1-to-1 and cover all of $\mathcal{X}$.)
- Since f defines a one-to-one correspondence between values $\mathbf{z} \in \mathcal{Z}$ and $\mathbf{x} \in \mathcal{X}$, we can think of it as a change-of-variables transformation.
- Change-of-Variables Formula from probability theory: if $\mathbf{x} = f(\mathbf{z})$, then

$$p_X(\mathbf{x}) = p_Z(z) \left| \det \left(\frac{\partial \mathbf{x}}{\partial \mathbf{z}} \right) \right|^{-1}$$

- Intuition for the Jacobian term:



Change of Variables Formula

- Suppose we have a generator network which computes the function f. It's tempting to apply the change-of-variables formula in order to compute the density $p_X(\mathbf{x})$.
- I.e., compute $\mathbf{z} = f^{-1}(\mathbf{x})$

$$p_X(\mathbf{x}) = p_Z(z) \left| \det \left(\frac{\partial \mathbf{x}}{\partial z} \right) \right|^{-1}$$

- Problems?

Change of Variables Formula

- Suppose we have a generator network which computes the function f. It's tempting to apply the change-of-variables formula in order to compute the density $p_X(\mathbf{x})$.
- I.e., compute $\mathbf{z} = f^{-1}(\mathbf{x})$

$$p_X(\mathbf{x}) = p_Z(z) \left| \det \left(\frac{\partial \mathbf{x}}{\partial z} \right) \right|^{-1}$$

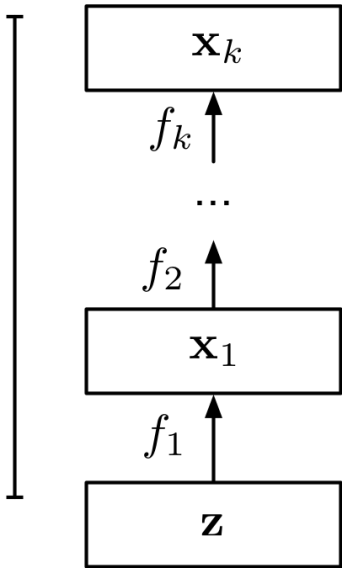
- Problems?
  - It needs to be differentiable, so that the Jacobian $\partial \mathbf{x} / \partial z$ is defined.
  - The mapping f needs to be invertible, with an easy-to-compute inverse.
  - We need to be able to compute the (log) determinant.
- Differentiability is easy (just use a differentiable activation function), but the other requirements are trickier.

Reversible Blocks

- Now let's define a reversible block which is invertible and has a tractable determinant.

- Such blocks can be composed.
  - Inversion: $f^{-1} = f_1^{-1} \circ \cdots \circ f_k^{-1}$
  - Determinants: $\left|\frac{\partial \mathbf{x}_k}{\partial \mathbf{z}}\right| = \left|\frac{\partial \mathbf{x}_k}{\partial \mathbf{x}_{k-1}}\right| \cdots \left|\frac{\partial \mathbf{x}_2}{\partial \mathbf{x}_1}\right| \left|\frac{\partial \mathbf{x}_1}{\partial \mathbf{z}}\right|$

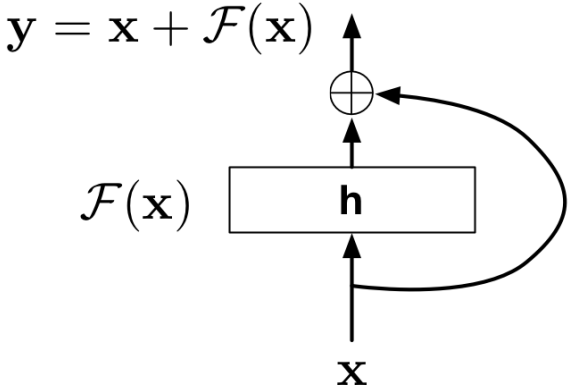
$$f = f_k \circ \cdots \circ f_1$$



Reversible Blocks

- Recall the residual blocks:

$$\mathbf{y} = \mathbf{x} + \mathcal{F}(\mathbf{x})$$

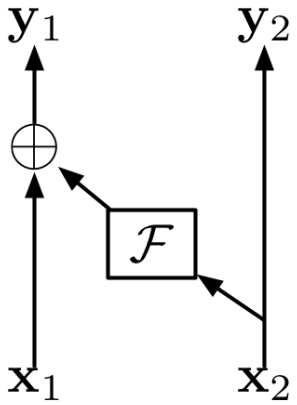


- Reversible blocks are a variant of residual blocks. Divide the units into two groups, $\mathbf{x}_1$ and $\mathbf{x}_2$.

$$\mathbf{y}_1 = \mathbf{x}_1 + \mathcal{F}(\mathbf{x}_2)$$
$$\mathbf{y}_2 = \mathbf{x}_2$$

- Inverting a reversible block:

$$\mathbf{x}_2 = \mathbf{y}_2$$
$$\mathbf{x}_1 = \mathbf{y}_1 - \mathcal{F}(\mathbf{x}_2)$$



Reversible Blocks

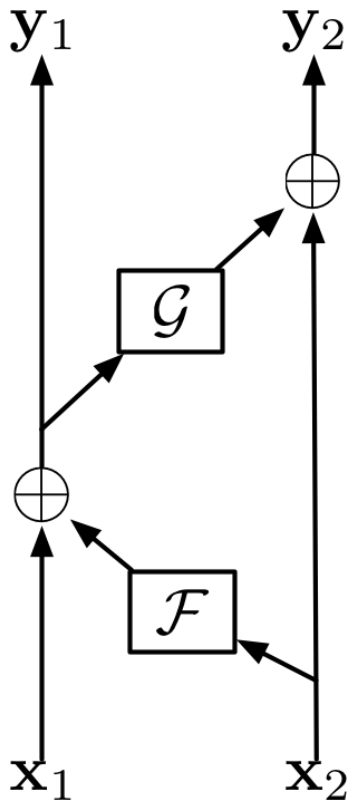
Composition of two reversible blocks, but with $\mathbf{x}_1$ and $\mathbf{x}_2$ swapped:

- Forward:

$$\mathbf{y}_1 = \mathbf{x}_1 + \mathcal{F}(\mathbf{x}_2)$$
$$\mathbf{y}_2 = \mathbf{x}_2 + \mathcal{G}(\mathbf{y}_1)$$

- Backward:

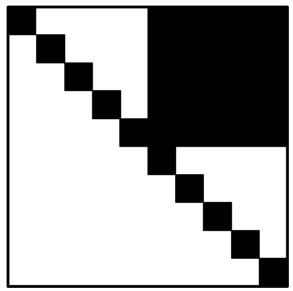
$$\mathbf{x}_2 = \mathbf{y}_2 - \mathcal{G}(\mathbf{y}_1)$$
$$\mathbf{x}_1 = \mathbf{y}_1 - \mathcal{F}(\mathbf{x}_2)$$



Volume Preservation

- It remains to compute the log determinant of the Jacobian.
- The Jacobian of the reversible block:

$$\mathbf{y}_1 = \mathbf{x}_1 + \mathcal{F}(\mathbf{x}_2) \qquad \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{pmatrix} \mathbf{I} & \frac{\partial \mathcal{F}}{\partial \mathbf{x}_2} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$$
$$\mathbf{y}_2 = \mathbf{x}_2$$



- This is an upper triangular matrix. The determinant of an upper triangular matrix is the product of the diagonal entries, or in this case, 1.
- Since the determinant is 1, the mapping is said to be volume preserving.

Nonlinear Independent Components Estimation

- We've just defined the reversible block.
  - Easy to invert by subtracting rather than adding the residual function.
  - The determinant of the Jacobian is 1.
- Nonlinear Independent Components Estimation (NICE) trains a generator network which is a composition of lots of reversible blocks.
- We can compute the likelihood function using the change-of-variables formula:

$$p_X(\mathbf{x}) = p_Z(\mathbf{z}) \left| \det \left(\frac{\partial \mathbf{x}}{\partial \mathbf{z}} \right) \right|^{-1} = p_Z(\mathbf{z})$$

- We can train this model using maximum likelihood. I.e., given a dataset $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$, we maximize the likelihood

$$\prod_{i=1}^{N} p_X(\mathbf{x}^{(i)}) = \prod_{i=1}^{N} p_Z(f^{-1}(\mathbf{x}^{(i)}))$$

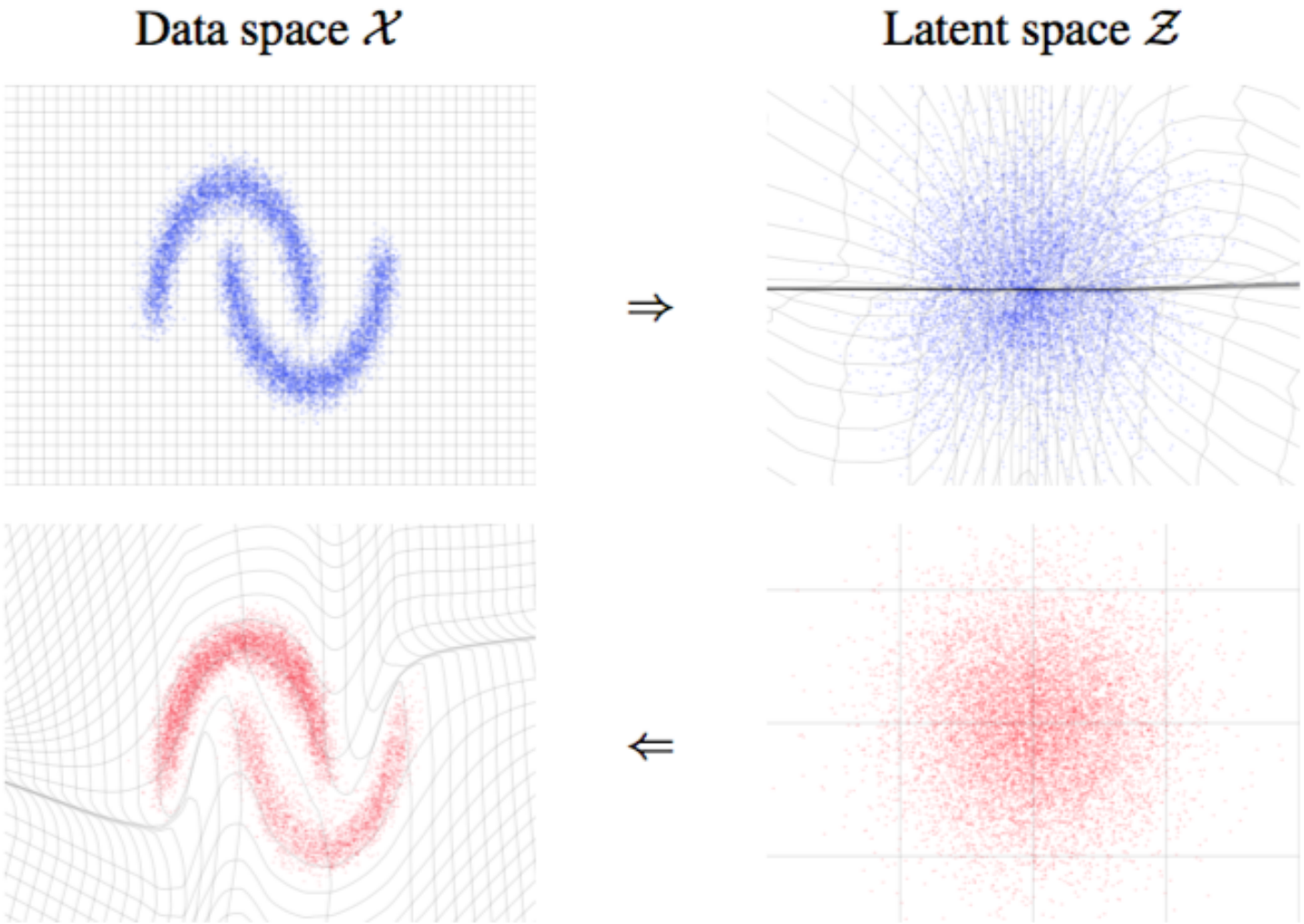
Nonlinear Independent Components Estimation

- Likelihood:

$$p_X(\mathbf{x}) = p_Z(\mathbf{z}) = p_Z(f^{-1}(\mathbf{x}))$$

- Remember, p_Z is a simple, fixed distribution (e.g. independent Gaussians)
- Intuition: train the network such that f^{-1} maps each data point to a high-density region of the code vector space $\mathcal{Z}$.
  - Without constraints on f, it could map everything to $\mathbf{0}$, and this likelihood objective would make no sense.
  - But it can't do this because it's volume preserving.

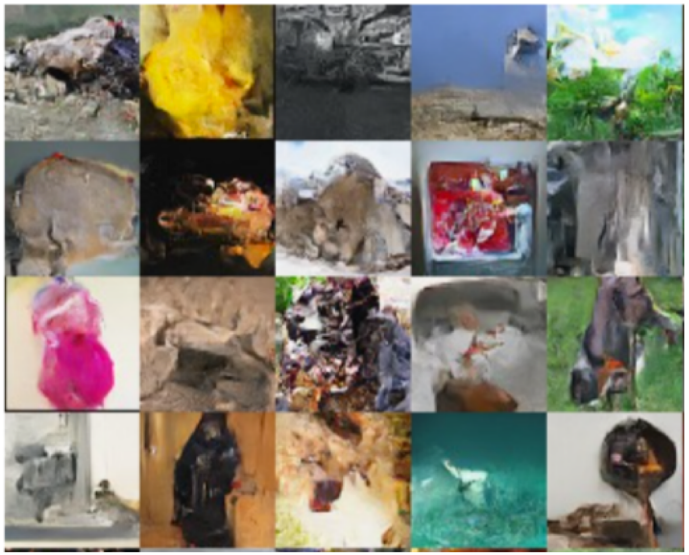
Nonlinear Independent Components Estimation



Dinh et al., 2016. Density estimation using RealNVP.

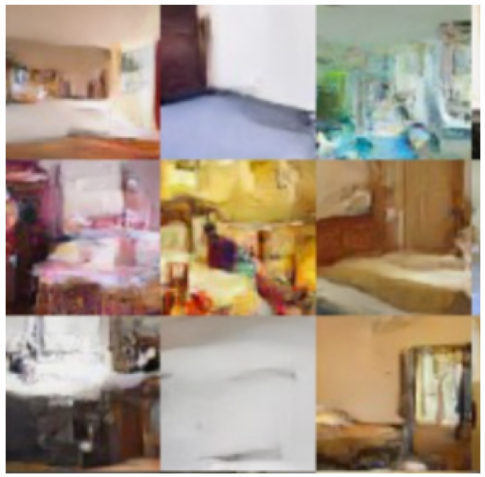
Nonlinear Independent Components Estimation

Samples produced by RealNVP, a model based on NICE.



ImageNet

celebrities



bedrooms

Dinh et al., 2016. Density estimation using RealNVP.

RevNets (optional)

- A side benefit of reversible blocks: you don't need to store the activations in memory to do backprop, since you can reverse the computation.
  - I.e., compute the activations as you need them, moving backwards through the computation graph.
- Notice that reversible blocks look a lot like residual blocks.
- Can use this to design a reversible residual network (RevNet) architecture which is like a ResNet, but with reversible blocks instead of residual blocks.
  - Matches state-of-the-art performance on ImageNet, but without the memory cost of activations!
  - Gomez et al., NIPS 2017. "The reversible residual network: backprop without storing activations".