Learning Latent Variable Models

Volodymyr Kuleshov

Cornell Tech

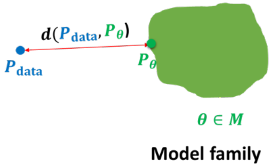
Lecture 7

Announcements

- Assignment 1 is due on Thursday.
- The Gradescope system is now up. The code is M45WYY.
- Presentation topics are up.
- I will send confirmations to the students who emailed me about presentation topics over the past week.

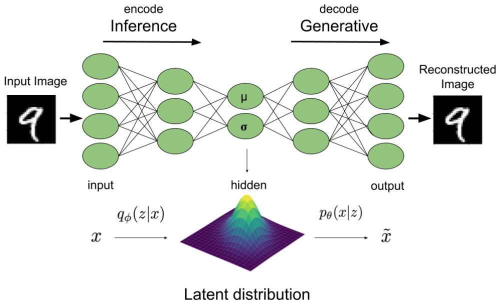
Lecture Outline

Recap: Autoregressive Models



1. Autoregressive models: $p_\theta(\mathbf{x}) = \prod_{i=1}^{n} p_\theta(x_i | \mathbf{x}_{<i})$
   - Chain rule based factorization is fully general
   - Compact representation via *conditional independence* and/or *neural parameterizations*
2. Autoregressive models Pros:
   - Easy to evaluate likelihoods
   - Easy to train
3. Autoregressive models Cons:
   - Requires an ordering
   - Generation is sequential
   - Cannot learn features in an unsupervised way

Recap: Latent Variable Models Models

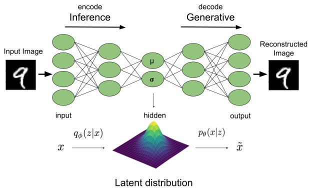


Variational Autoencoders: $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z}$

$$\mathcal{L}(\mathbf{x}; \theta, \phi) = E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z}; \theta)] - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z}))$$

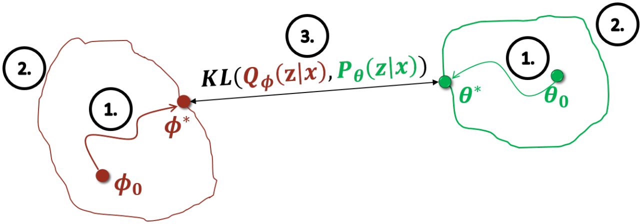
- Infinite mixture of Gaussians. means are parametrized by deep net.
- Objective has a natural auto-encoder interpretation.

Recap: Latent Variable Models Models



1. Latent Variable Models Pros:
   - Easy to build flexible models
   - Suitable for unsupervised learning
2. Latent Variable Models Cons:
   - Hard to evaluate likelihoods
   - Hard to train via maximum-likelihood
   - Fundamentally, the challenge is that posterior inference $p(\mathbf{z} \mid \mathbf{x})$ is hard. Typically requires variational approximations

Research Directions



Improving variational learning via:

1. Better optimization techniques
2. More expressive approximating families
3. Alternate loss functions

Model families - Encoder

Augmenting variational posteriors

- Monte Carlo methods: Importance Sampling (Burda et al., 2015), MCMC (Salimans et al., 2015, Hoffman, 2017, Levy et al., 2018), Sequential Monte Carlo (Maddison et al., 2017, Le et al., 2018, Naesseth et al., 2018), Rejection Sampling (Grover et al., 2018)
- Normalizing flows (Rezende & Mohammed, 2015, Kingma et al., 2016)

Model families - Decoder

- Powerful decoders $p(\mathbf{x}|\mathbf{z};\theta)$ such as DRAW (Gregor et al., 2015), PixelCNN (Gulrajani et al., 2016)
- Parameterized, learned priors $p(\mathbf{z};\theta)$ (Nalusnick et al., 2016, Tomczak & Welling, 2018, Graves et al., 2018)

Variational objectives

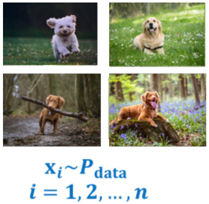
Tighter ELBO does not imply:

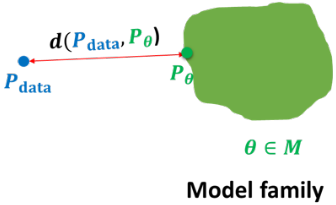
- Better samples: Sample quality and likelihoods are uncorrelated (Theis et al., 2016)
- Informative latent codes: Powerful decoders can ignore latent codes due to tradeoff in minimizing reconstruction error vs. KL prior penalty (Bowman et al., 2015, Chen et al., 2016, Zhao et al., 2017, Alemi et al., 2018)

Alternatives to the reverse-KL divergence:

- Renyi's alpha-divergences (Li & Turner, 2016)
- Integral probability metrics such as maximum mean discrepancy, Wasserstein distance (Dziugaite et al., 2015; Zhao et. al, 2017; Tolstikhin et al., 2018)

Can We Get Best of Both Worlds?





- Model families:
  - Autoregressive Models: $p_\theta(\mathbf{x}) = \prod_{i=1}^{n} p_\theta(x_i|\mathbf{x}_{<i})$
  - Variational Autoencoders: $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z})d\mathbf{z}$
- Autoregressive models provide tractable likelihoods but no direct mechanism for learning features
- Variational autoencoders can learn feature representations (via latent variables $\mathbf{z}$) but have intractable marginal likelihoods
- **Key question**: Can we design a latent variable model with tractable likelihoods? Yes! Use normalizing flows.

Simple Prior to Complex Data Distributions

- Desirable properties of any model distribution:
  - Analytic density
  - Easy-to-sample
- Many simple distributions satisfy the above properties e.g., Gaussian, uniform distributions
- Unfortunately, data distributions could be much more complex (multi-modal)
- **Key idea**: Map simple distributions (easy to sample and evaluate densities) to complex distributions (learned via data) using **invertible** change of variables transformations.

Lecture Outline

Example: Change of Variables

- Let Z be a uniform random variable $\mathcal{U}[0, 2]$ with density p_Z. What is $p_Z(1)$? $\frac{1}{2}$
- Let $X = 4Z$, and let p_X be its density. What is $p_X(4)$?
- $p_X(4) = p(X = 4) = p(4Z = 4) = p(Z = 1) = p_Z(1) = 1/2$
- **This is incorrect.** Clearly, X is uniform in $[0, 8]$, so $p_X(4) = 1/8$
- Probability mass functions are not probability distributions (measures).
- Transformations need to preserve total *volume* of probability mass.

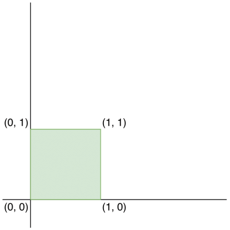
Example: Change of Variables

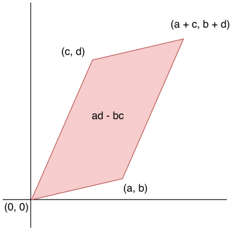
- **Change of variables (1D case)**: If $X = f(Z)$ and $f(\cdot)$ is monotone with inverse $Z = f^{-1}(X) = h(X)$, then:

$$p_X(x) = p_Z(h(x))|h'(x)|$$

- Previous example: If $X = 4Z$ and $Z \sim \mathcal{U}[0, 2]$, what is $p_X(4)$?
- Note that $h(X) = X/4$
- $p_X(4) = p_Z(1)h'(4) = 1/2 \times 1/4 = 1/8$
- We have expanded the support of the distribution by 4. Hence, we need to decrease the mass at each point by 4 to preserve the volume.
- Generalizes to higher dimensions via determinants of transformations

Review: Determinants and Volumes (in 2D)





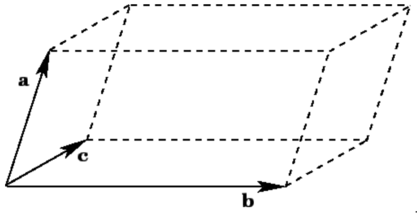
- Matrix $A = \begin{pmatrix} a & c \\ b & d \end{pmatrix}$ maps a unit square to a parallelogram, e.g.:

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a & c \\ b & d \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

- The volume of the parallelotope is equal to the determinant of the transformation A

$$\det(A) = \det \begin{pmatrix} a & c \\ b & d \end{pmatrix} = ad - bc$$

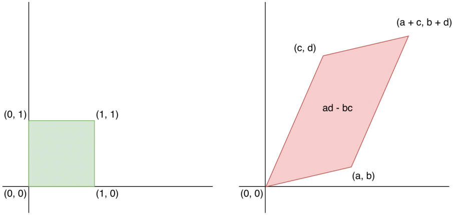
Review: Determinants and Volumes (in 3D)



- The volume formula still holds in 3D.
- Note that if two vectors are colinear, we get a plane, which has volume zero in 3D. The determinant is zero and the matrix is singular.

Review: Determinants and Volumes (in n-D)

- In general, the matrix A maps the unit hypercube $[0,1]^n$ to a parallelotope
- Hypercube and parallelotope are generalizations of square/cube and parallelogram/parallelopiped to higher dimensions



- Determinant $\det(A)$ still gives volume of the n-D shape.

Determinants and Volumes for Changing Variables

- Let Z be a uniform random vector in $[0,1]^n$
- Let $X = AZ$ for a square invertible matrix A, with inverse $W = A^{-1}$. How is X distributed?
- The volume of the parallelotope is equal to the determinant of the transformation A

$$\det(A) = \det \begin{pmatrix} a & c \\ b & d \end{pmatrix} = ad - bc$$

- X is uniformly distributed over the parallelotope. Hence, we have

$$\begin{aligned} p_X(\mathbf{x}) &= p_Z(W\mathbf{x}) \left|\det(W)\right| \\ &= p_Z(W\mathbf{x}) / \left|\det(A)\right| \end{aligned}$$

Change of Variables Formula (General Case)

- For linear transformations specified via A, change in volume is given by the determinant of A
- For non-linear transformations $\mathbf{f}(\cdot)$, the *linearized* change in volume is given by the determinant of the Jacobian of $\mathbf{f}(\cdot)$.

The Jacobian

Consider a vector valued function $f : \mathbb{R}^n \to \mathbb{R}^m$, with:

- $\mathbf{x} = (x_1, \cdots, x_n)$
- $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \cdots, f_m(\mathbf{x}))$

The Jacobian is defined as:

$$J = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \cdots & \cdots & \cdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

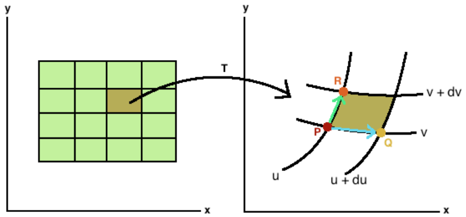
This generalizes the gradient to multi-variate functions.

Change of Variables Formula (General Case)

- For linear transformations specified via A, change in volume is given by the determinant of A
- For non-linear transformations $\mathbf{f}(\cdot)$, the *linearized* change in volume is given by the determinant of the Jacobian of $\mathbf{f}(\cdot)$.
- **Change of Variables Formula** (General case): The mapping between Z and X, given by $\mathbf{f} : \mathbb{R}^n \mapsto \mathbb{R}^n$, is invertible such that $X = \mathbf{f}(Z)$ and $Z = \mathbf{f}^{-1}(X)$.

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

Change of Variables Formula (General Case): Intuition



- We are interested in mapping a small volume between (v, u) and $(v + dv, u + du)$.
- For sufficiently small du, dv, the function can be linearized, and becomes the linear mapping specified by the Jacobian.

Change of Variables Formula (General Case): Observations

- **Change of variables (General case)**: The mapping between Z and X, given by $\mathbf{f} : \mathbb{R}^n \mapsto \mathbb{R}^n$, is invertible such that $X = \mathbf{f}(Z)$ and $Z = \mathbf{f}^{-1}(X)$.

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

- Note 1: $\mathbf{x}, \mathbf{z}$ need to be continuous and have the same dimension. For example, if $\mathbf{x} \in \mathbb{R}^n$ then $\mathbf{z} \in \mathbb{R}^n$
- Note 2: For any invertible matrix A, $det(A^{-1}) = det(A)^{-1}$

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

Change of Variables Formula (General Case): 2D Example

- Let Z_1 and Z_2 be continuous random variables with joint density p_{Z_1,Z_2}.
- Let $u : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ be a transformation with inverse $v : \mathbb{R}^2 \rightarrow \mathbb{R}^2$.
- Let $X_1 = u_1(Z_1, Z_2)$ and $X_2 = u_2(Z_1, Z_2)$ Then, $Z_1 = v_1(X_1, X_2)$ and $Z_2 = v_2(X_1, X_2)$

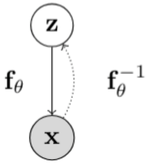
$$
\begin{aligned}
& p_{X_1,X_2}(x_1, x_2) \\
&= p_{Z_1,Z_2}(v_1(x_1, x_2), v_2(x_1, x_2)) \left| \det \begin{pmatrix} \frac{\partial v_1(x_1,x_2)}{\partial x_1} & \frac{\partial v_1(x_1,x_2)}{\partial x_2} \\ \frac{\partial v_2(x_1,x_2)}{\partial x_1} & \frac{\partial v_2(x_1,x_2)}{\partial x_2} \end{pmatrix} \right| \text{(inverse)} \\
&= p_{Z_1,Z_2}(z_1, z_2) \left| \det \begin{pmatrix} \frac{\partial u_1(z_1,z_2)}{\partial z_1} & \frac{\partial u_1(z_1,z_2)}{\partial z_2} \\ \frac{\partial u_2(z_1,z_2)}{\partial z_1} & \frac{\partial u_2(z_1,z_2)}{\partial z_2} \end{pmatrix} \right|^{-1} \text{(forward)}
\end{aligned}
$$

Lecture Outline

1. Recap and Motivation for Normalizing Flows
    - Autoregressive Models
    - Latent Variable Models
    - Research Directions in LVMs
2. Volume-Preserving Transformations
    - The Determinant
    - Change of Variables Formula
3. Normalizing Flows
    - Representation and Learning
    - Composing Simple Transformations
    - Triangular Jacobians

Normalizing Flow Models: Representation

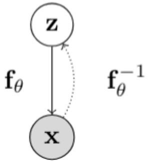
- Consider a directed, latent-variable model over observed variables X and latent variables Z



- In a **normalizing flow model**, the mapping between Z and X, given by $\mathbf{f}_\theta : \mathbb{R}^n \mapsto \mathbb{R}^n$, is deterministic and invertible such that $X = \mathbf{f}_\theta(Z)$ and $Z = \mathbf{f}_\theta^{-1}(X)$

Normalizing Flow Models: Learning

- In a **normalizing flow model**, the mapping between Z and X, given by $\mathbf{f}_\theta : \mathbb{R}^n \mapsto \mathbb{R}^n$, is deterministic and invertible such that $X = \mathbf{f}_\theta(Z)$ and $Z = \mathbf{f}_\theta^{-1}(X)$



- We want to learn $p_X(\mathbf{x};\theta)$ using the principle of maximum likelihood.
- Using change of variables, the marginal likelihood $p(\mathbf{x})$ is given by

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

- Note 1: Unlike in a VAE, we are computing the marginal likelihood exactly!
- Note 2: $\mathbf{x}, \mathbf{z}$ need to be continuous and have the same dimension.

Normalizing Flow Models: Constructing f.

We need to construct a density transformation that is:

- Invertible, so that we can apply the change of variables formula.
- Expressive, so that we can learn complex distributions.
- Computationally tractable, so that we can optimize and evaluate it.

Strategy:

- Start with a simple distribution for $\mathbf{z}_0$ (e.g., Gaussian)
- Apply sequence of M **simple** invertible transformations with $\mathbf{x} \triangleq \mathbf{z}_M$

$$\mathbf{z}_m := \mathbf{f}_\theta^m \circ \cdots \circ \mathbf{f}_\theta^1(\mathbf{z}_0) = \mathbf{f}_\theta^m(\mathbf{f}_\theta^{m-1}(\cdots(\mathbf{f}_\theta^1(\mathbf{z}_0)))) \triangleq \mathbf{f}_\theta(\mathbf{z}_0)$$

- By change of variables

$$p_X(\mathbf{x};\theta) = p_Z\left(\mathbf{f}_\theta^{-1}(\mathbf{x})\right) \prod_{m=1}^{M} \left|\det\left(\frac{\partial(\mathbf{f}_\theta^m)^{-1}(\mathbf{z}_m)}{\partial \mathbf{z}_m}\right)\right|$$

(Note: determininant of composition equals product of determinants)

Example: Planar Flows

- Planar flow (Rezende and Mohamed, 2016). Invertible transformation

$$\mathbf{x} = \mathbf{f}_\theta(\mathbf{z}) = \mathbf{z} + \mathbf{u}h(\mathbf{w}^T\mathbf{z} + b)$$

  parameterized by $\theta = (\mathbf{w}, \mathbf{u}, b)$ where $h(\cdot)$ is a non-linearity

- Absolute value of the determinant of the Jacobian is given by

$$\left|\det\frac{\partial \mathbf{f}_\theta(\mathbf{z})}{\partial \mathbf{z}}\right| = \left|\det(I + h'(\mathbf{w}^T\mathbf{z} + b)\mathbf{u}\mathbf{w}^T)\right|$$

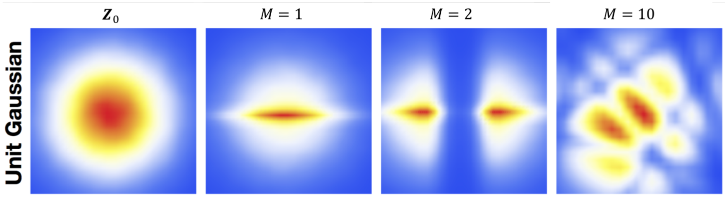
$$= \left|1 + h'(\mathbf{w}^T\mathbf{z} + b)\mathbf{u}^T\mathbf{w}\right|$$

(matrix determinant lemma)

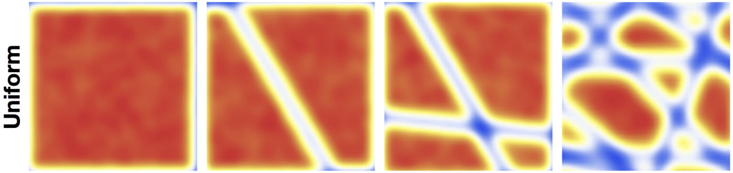
- Need to restrict parameters and non-linearity for the mapping to be invertible. For example, $h = tanh()$ and $h'(\mathbf{w}^T\mathbf{z} + b)\mathbf{u}^T\mathbf{w} \geq -1$

Example: Planar Flows

- Base distribution: Gaussian



- Base distribution: Uniform



- 10 planar transformations can transform simple distributions into a more complex one

Normalizing Flows: Recap

Normalizing: Change of variables gives a normalized density after applying an invertible transformation.
Flow: The function f makes the probability mass smoothly flow from a simple distribution over the space to one that is complex.

- Transformations need to be invertible, hence $\dim(X) = \dim(Z)$.
- Complex transformations can be composed from simple ones:

$$\mathbf{z}_m := \mathbf{f}_\theta^m \circ \cdots \circ \mathbf{f}_\theta^1(\mathbf{z}_0) = \mathbf{f}_\theta^m(\mathbf{f}_\theta^{m-1}(\cdots(\mathbf{f}_\theta^1(\mathbf{z}_0)))) \triangleq \mathbf{f}_\theta(\mathbf{z}_0)$$

- Learning via **maximum likelihood** over the dataset $\mathcal{D}$

$$\max_\theta \log p_X(\mathcal{D};\theta) = \sum_{\mathbf{x}\in\mathcal{D}} \log p_Z\left(\mathbf{f}_\theta^{-1}(\mathbf{x})\right) + \log\left|\det\left(\frac{\partial \mathbf{f}_\theta^{-1}(\mathbf{x})}{\partial \mathbf{x}}\right)\right|$$

Normalizing Flows: Learning and Inference Recap

- **Exact likelihood evaluation** via inverse tranformation $\mathbf{x} \mapsto \mathbf{z}$ and change of variables formula
- **Sampling** via forward transformation $\mathbf{z} \mapsto \mathbf{x}$

$$\mathbf{z} \sim p_Z(\mathbf{z}) \quad \mathbf{x} = \mathbf{f}_\theta(\mathbf{z})$$

- **Latent representations** inferred via inverse transformation (no inference network required!)

$$\mathbf{z} = \mathbf{f}_\theta^{-1}(\mathbf{x})$$

Challenges in Building Flow Models

- Complex, invertible transformations with tractable evaluation:
  - Likelihood evaluation requires efficient evaluation of $\mathbf{x} \mapsto \mathbf{z}$ mapping
  - Sampling requires efficient evaluation of $\mathbf{z} \mapsto \mathbf{x}$ mapping
- Computing likelihoods also requires the evaluation of determinants of $n \times n$ Jacobian matrices, where n is the data dimensionality
  - Computing the determinant for an $n \times n$ matrix is $O(n^3)$: prohibitively expensive within a learning loop!

Key idea: Choose tranformations so that the resulting Jacobian matrix has special structure. For example, the determinant of a triangular matrix is the product of the diagonal entries, i.e., an $O(n)$ operation

Triangular Jacobian

$$\mathbf{x} = (x_1, \cdots, x_n) = \mathbf{f}(\mathbf{z}) = (f_1(\mathbf{z}), \cdots, f_n(\mathbf{z}))$$

$$J = \frac{\partial \mathbf{f}}{\partial \mathbf{z}} = \begin{pmatrix} \frac{\partial f_1}{\partial z_1} & \cdots & \frac{\partial f_1}{\partial z_n} \\ \cdots & \cdots & \cdots \\ \frac{\partial f_n}{\partial z_1} & \cdots & \frac{\partial f_n}{\partial z_n} \end{pmatrix}$$

Suppose $x_i = f_i(\mathbf{z})$ only depends on $\mathbf{z}_{\leq i}$. Then

$$J = \frac{\partial \mathbf{f}}{\partial \mathbf{z}} = \begin{pmatrix} \frac{\partial f_1}{\partial z_1} & \cdots & 0 \\ \cdots & \cdots & \cdots \\ \frac{\partial f_n}{\partial z_1} & \cdots & \frac{\partial f_n}{\partial z_n} \end{pmatrix}$$

has lower triangular structure. Determinant can be computed in **linear time**. Similarly, the Jacobian is upper triangular if x_i only depends on $\mathbf{z}_{\geq i}$
Next lecture: Designing invertible transformations!