Monte Carlo Methods

Lecture slides for Chapter 17 of *Deep Learning*

www.deeplearningbook.org

Ian Goodfellow

Last updated 2017-12-29

Roadmap

- Basics of Monte Carlo methods
- Importance Sampling
- Markov Chains

Randomized Algorithms

	Las Vegas	**Monte Carlo**
Type of Answer	Exact	Random amount of error
Runtime	Random (until answer found)	Chosen by user (longer runtime gives lesss error)

Estimating sums / integrals with samples

$$s = \sum_{\boldsymbol{x}} p(\boldsymbol{x}) f(\boldsymbol{x}) = E_p[f(\mathbf{x})] \tag{17.1}$$

$$s = \int p(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} = E_p[f(\mathbf{x})] \tag{17.2}$$

$$\hat{s}_n = \frac{1}{n} \sum_{i=1}^{n} f(\boldsymbol{x}^{(i)}). \tag{17.3}$$

Justification

- Unbiased:
  - The expected value for finite n is equal to the correct value
  - The value for any specific n samples will have random error, but the errors for different sample sets cancel out
- Low variance:
  - Variance is $O(1/n)$
  - For very large n, the error converges "almost surely" to 0

Roadmap

- Basics of Monte Carlo methods
- Importance Sampling
- Markov Chains

Non-unique decomposition

$$s = \int p(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} = E_p[f(\mathbf{x})] \tag{17.2}$$

Say we want to compute

$$\int a(\boldsymbol{x}) b(\boldsymbol{x}) c(\boldsymbol{x}) d\boldsymbol{x}.$$

Which part is p? Which part is f ?

p=a and f=bc? p=ab and f=c? etc.

No unique decomposition.

We can always pull part of any p into f.

Importance Sampling

$$p(\boldsymbol{x})f(\boldsymbol{x}) = q(\boldsymbol{x})\frac{p(\boldsymbol{x})f(\boldsymbol{x})}{q(\boldsymbol{x})} \tag{17.8}$$

This is our new p, meaning it is the distribution we will draw samples from

This ratio is our new f, meaning we will evaluate it at each sample

Why use importance sampling?

- Maybe it is feasible to sample from q but not from p
  - This is how GANs work
- A good q can reduce the variance of the estimate
- Importance sampling is still unbiased for every q

Optimal q

$$q^*(\boldsymbol{x}) = \frac{p(\boldsymbol{x})|f(\boldsymbol{x})|}{Z} \tag{17.13}$$

- Determining the optimal q requires solving the original integral, so not useful in practice
- Useful to understand intuition behind importance sampling
- This q minimizes the variance
- Places more mass on points where the weighted function is larger

Roadmap

- Basics of Monte Carlo methods
- Importance Sampling
- Markov Chains

Sampling from p or q

- So far we have assumed we can sample from p or q easily
- This is true when p or q has a *directed graphical model* representation
  - Use *ancestral sampling*
  - Sample each node given its parents, moving from roots to leaves

Sampling from undirected models

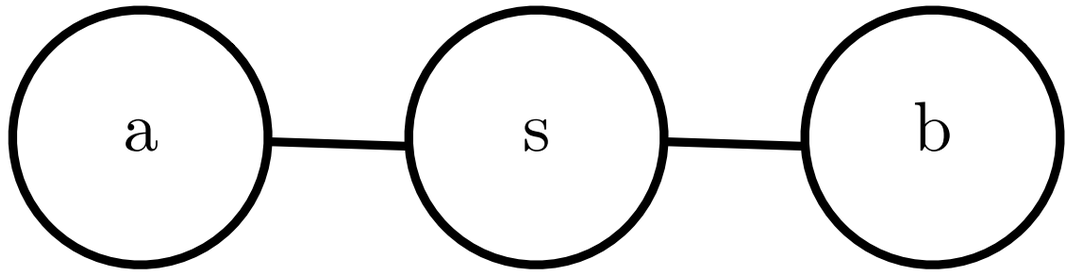
- Sampling from undirected models is more difficult
- Can't get a fair sample in one pass
- Use a Monte Carlo algorithm that incrementally updates samples, comes closer to sampling from the right distribution at each step
- This is called a *Markov Chain*

Simple Markov Chain: Gibbs sampling

- Repeatedly cycle through all variables
  - For each variable, randomly sample that variable given its *Markov blanket*
  - For an undirected model, the Markov blanket is just the neighbors in the graph
- *Block Gibbs* trick: conditionally independent variables may be sampled simultaneously

Gibbs sampling example

- Initialize a, s, and b
- For n repetitions
  - Sample a from P(a|s) and b from P(b|s)
  - Sample s from P(s|a,b)



Block Gibbs trick lets us sample a and b in parallel

Equilibrium

- Running a Markov Chain long enough causes it to *mix*
- After mixing, it samples from an *equilibrium distribution*
  - Sample before update comes from distribution $\pi(x)$
  - Sample after update is a different sample, but still from distribution $\pi(x)$

Downsides

- Generally infeasible to...
  - ...know ahead of time how long mixing will take
  - ...know how far a chain is from equilibrium
  - ...know whether a chain is at equilibrium
- Usually in deep learning we just run for n steps, for some n that we think will be big enough, and hope for the best

Trouble in Practice

- Mixing can take an infeasibly long time
- This is especially true for
  - High-dimensional distributions
  - Distributions with strong correlations between variables
  - Distributions with multiple highly separated modes

Difficult Mixing

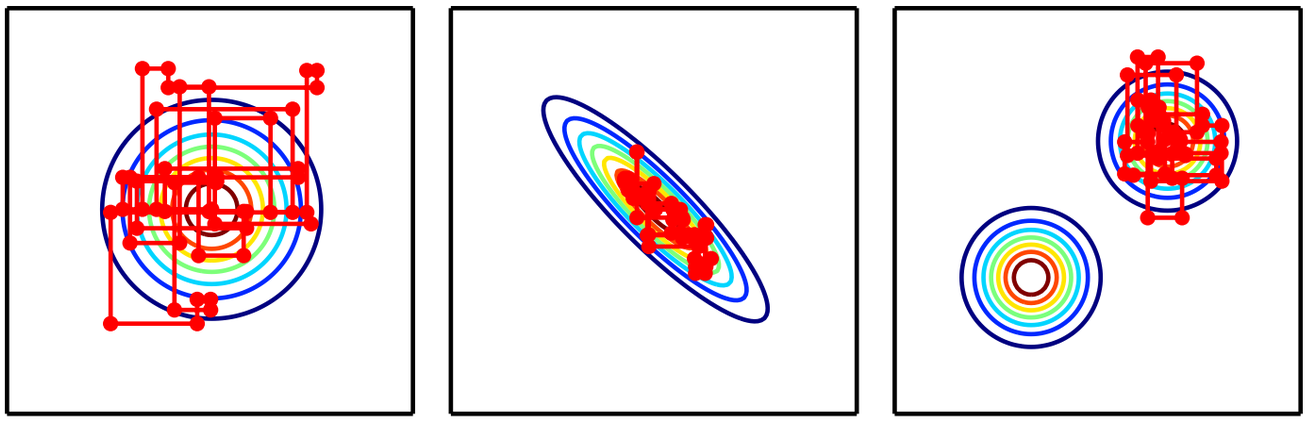


Figure 17.1: Paths followed by Gibbs sampling for three distributions, with the Markov chain initialized at the mode in both cases. *(Left)*A multivariate normal distribution with two independent variables. Gibbs sampling mixes well because the variables are independent. *(Center)*A multivariate normal distribution with highly correlated variables. The correlation between variables makes it difficult for the Markov chain to mix. Because the update for each variable must be conditioned on the other variable, the correlation reduces the rate at which the Markov chain can move away from the starting point. *(Right)*A mixture of Gaussians with widely separated modes that are not axis aligned. Gibbs sampling mixes very slowly because it is difficult to change modes while altering only one variable at a time.

Difficult Mixing in Deep Generative Models

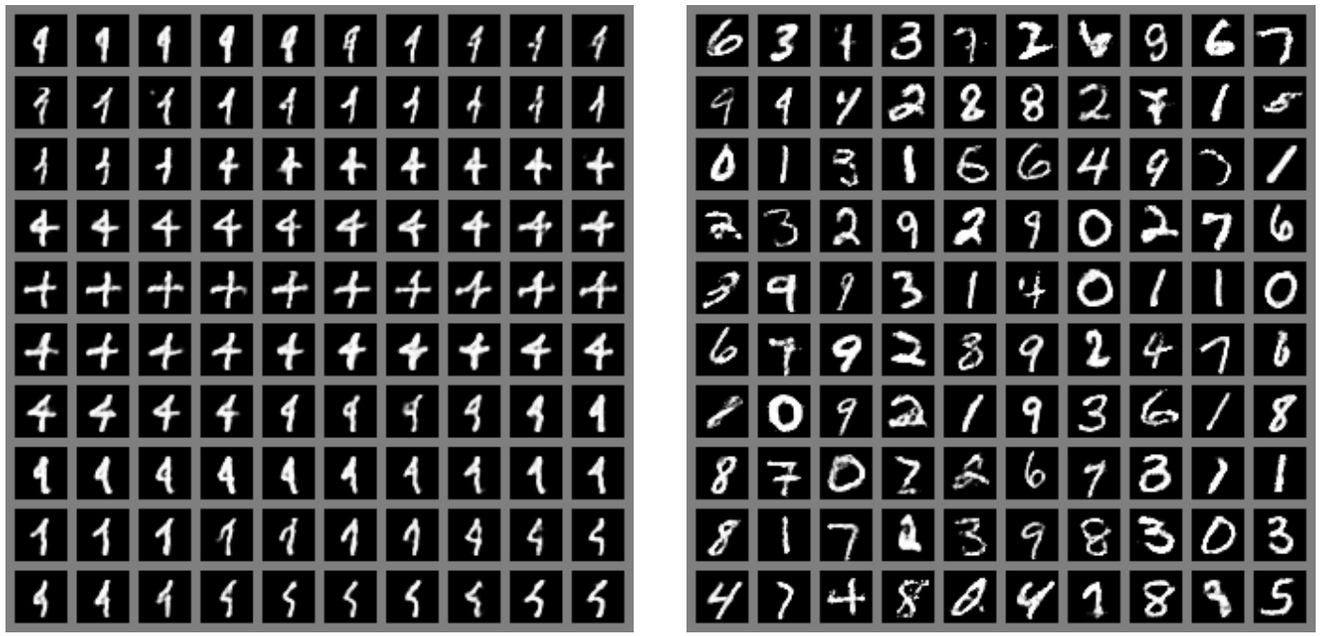


Figure 17.2: An illustration of the slow mixing problem in deep probabilistic models. Each panel should be read left to right, top to bottom. *(Left)*Consecutive samples from Gibbs sampling applied to a deep Boltzmann machine trained on the MNIST dataset. Consecutive samples are similar to each other. Because the Gibbs sampling is performed in a deep graphical model, this similarity is based more on semantic than raw visual features, but it is still difficult for the Gibbs chain to transition from one mode of the distribution to another, for example, by changing the digit identity. *(Right)*Consecutive ancestral samples from a generative adversarial network. Because ancestral sampling generates each sample independently from the others, there is no mixing problem.

For more information...