Monte Carlo Methods

Sargur N. Srihari
srihari@cedar.buffalo.edu

Topics

1. Sampling and Monte Carlo Methods
2. Importance Sampling
3. Markov Chain Monte Carlo Methods
4. Gibbs Sampling
5. Mixing between separated modes

What are Monte Carlo Algorithms?

- We cannot obtain precise answers to many problems in machine learning
  - We can use either deterministic approximate algorithms or Monte Carlo approximations
    - Both approaches are common in machine learning
- Monte Carlo algorithms return answers with random amount of error
  - For a fixed computational budget, a Monte Carlo algorithm can provide an approximate answer
    - Las Vegas algorithms return exact answers but use random amount of resources

1. Why Sampling?

- Many ML algorithms are based on drawing samples from some probability distribution
  - and using these samples to form a Monte Carlo estimate of some desired quantity
- Sampling provides a flexible way to approximate many sums and integrals at reduced cost
  - Sometimes for speedup of a costly but tractable sum, e.g., subsample training cost with minibatches
  - In other cases, learning algorithms require us to approximate an intractable sum or integral
    - E.g., gradient of the log partition function of an undirected model

Basics of MC sampling

- When a sum or integral is intractable
  - E.g., has exponential no of terms and no exact simplification is known
  - It can often be approximated using MC sampling
- The idea is to view the sum or integral as an expectation under some distribution and to approximate the expectation by a corresponding average

Summation→Expectation→Average

- Sum or integral to approximate is

$$s = \sum_{\boldsymbol{x}} p(\boldsymbol{x}) f(\boldsymbol{x}) \quad \text{or} \quad s = \int p(\boldsymbol{x}) f(\boldsymbol{x}) \, d\boldsymbol{x}$$

- Rewriting expression as an expectation

$$s = E_p[f(\boldsymbol{x})] \quad \text{or} \quad s = E_p[f(\boldsymbol{x})]$$

  - with the constraint that p is a probability distribution (for the sum) or a pdf (for the integral)

- Approximate s by drawing n samples $\boldsymbol{x}^{(1)},..\boldsymbol{x}^{(n)}$ from p and then forming the empirical average

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

Justification of Approximation

- The sample average approximation is justified by a few different properties
  1. The estimator $\hat{s}$ is unbiased, since

$$E[\hat{s}_n] = \frac{1}{n}\sum_{i=1}^{n} E\left[f\left(\boldsymbol{x}^{(i)}\right)\right] \quad \text{since } \hat{s}_n \text{ is the sample average } \hat{s}_n = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}^{(i)})$$

$$= \frac{1}{n}\sum_{i=1}^{n} s \quad \text{since } s = E_p[f(\boldsymbol{x})]$$

$$= s$$

  2. The law of large numbers states that if the samples $\boldsymbol{x}^{(i)}$ are i.i.d. then the average converges almost surely to the expected value

$$\lim_{n\to\infty} \hat{s}_n = s$$

Provided the variance of the individual terms $\mathrm{Var}[f(\boldsymbol{x}^{(i)})]$ is bounded

Variance of estimate

- Consider variance of $\hat{s}_n$ as n increases
  - $\mathrm{Var}[\hat{s}_n]$ converges & decreases to 0 if $\mathrm{Var}[f(\boldsymbol{x}^{(i)})] < \infty$:

$$\mathrm{Var}[\hat{s}_n] = \frac{1}{n^2}\sum_{i=1}^{n}\mathrm{Var}[f(\boldsymbol{x})] = \frac{Var[f(\boldsymbol{x})]}{n}$$

- Result tells how to estimate error in MC average
  - Equivalently expected error of the approximation
  - Compute empirical average of the $f(\boldsymbol{x}^{(i)})$ and their empirical variance to determine estimator of $\mathrm{Var}[\hat{s}_n]$
    - Central Limit Theorem tells us that distribution of the average $\hat{s}_n$ has a normal distribution with mean s and variance $\mathrm{Var}[\hat{s}_n]$. This allows us to estimate confidence intervals around the estimate $\hat{s}_n$ using the Normal cdf

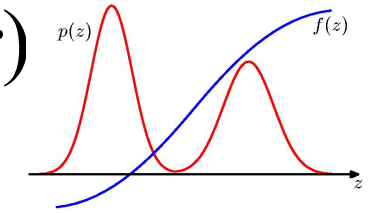
Sampling from base distribution $p(\boldsymbol{x})$

- Sampling from $p(\boldsymbol{x})$ is not always possible
- In such a case use importance sampling
- A more general approach is Monte Carlo Markov chains
  - To form a sequence of estimators that converge towards the distribution of interest

2. Importance Sampling

- Principal reason for sampling $p(\boldsymbol{x})$ is evaluating expectation of some $f(\boldsymbol{x})$

$$E[f] = \int f(\boldsymbol{x}) p(\boldsymbol{x}) \, d\boldsymbol{x}$$



- Given samples $\boldsymbol{x}^{(i)}$, $i=1,..,n$, from $p(\boldsymbol{x})$, the finite sum approximation is

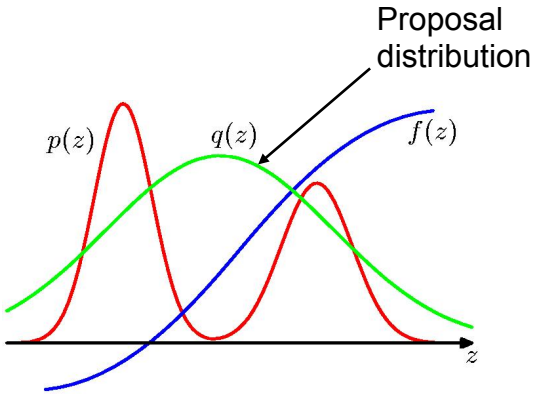
$$\hat{f} = \frac{1}{n} \sum_{i=1}^{n} f(\boldsymbol{x}^{(i)})$$

- But drawing samples $p(\boldsymbol{x})$ may be impractical
- Importance sampling uses:
  - a proposal distribution– like rejection sampling
    - But all samples are retained
  - Assumes that for any $\boldsymbol{x}$, $p(\boldsymbol{x})$ can be evaluated

Determining Importance weights

- Samples $\{\boldsymbol{x}^{(i)}\}$ are drawn from simpler dist. $p(\boldsymbol{x})$

$$E[f] = \int f(\boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}$$
$$= \int f(\boldsymbol{x}) \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})} q(\boldsymbol{x}) d\boldsymbol{x}$$
$$= \frac{1}{n} \sum_{l=1}^{n} \frac{p(\boldsymbol{x}^{(i)})}{q(\boldsymbol{x}^{(i)})} f(\boldsymbol{x}^{(i)})$$



- Samples are weighted by ratios

$r_l = p(\boldsymbol{x}^{(i)}) \ / \ q(\boldsymbol{x}^{(i)})$

  - Known as importance weights
    - Which corrects the bias introduced by wrong distribution

Importance Sampling: Choice of $p(\boldsymbol{x})$

- More generally, wish to evaluate the expression using sampling:

$$s = \sum_{\boldsymbol{x}} p(\boldsymbol{x}) f(\boldsymbol{x}) \qquad s = \int p(\boldsymbol{x}) f(\boldsymbol{x}) \, d\boldsymbol{x}$$

- Which part of the integrand has the role of the probability $p(\boldsymbol{x})$
  - [from which we sample $\boldsymbol{x}^{(1)}, .. \boldsymbol{x}^{(n)}$]
- And which part has role of $f(\boldsymbol{x})$ whose expected value (under the probability distribution) is estimated as

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

Decomposition of Integrand

- In the equation $s = \sum_{\boldsymbol{x}} p(\boldsymbol{x}) f(\boldsymbol{x})$ $s = \int p(\boldsymbol{x}) f(\boldsymbol{x}) \, d\boldsymbol{x}$
- There is no unique decomposition because $p(\boldsymbol{x}) f(\boldsymbol{x})$ can be rewritten as

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

  - where we now sample from q and average

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

  - In many cases the problem $s = \int p(\boldsymbol{x}) f(\boldsymbol{x}) \, d\boldsymbol{x}$ is specified naturally as expectation of $f(\boldsymbol{x})$ given distribution $p(\boldsymbol{x})$
    - But it may not be optimal in no of samples required

Derivation of Optimal $q^*(\boldsymbol{x})$

- Any Monte Carlo estimator $\hat{s}_p = \frac{1}{n} \sum_{i=1, \boldsymbol{x}^{(i)} \sim p}^{n} f(\boldsymbol{x}^{(i)})$
  - can be transformed into an importance sampling estimator $\hat{s}_q = \frac{1}{n} \sum_{i=1, \boldsymbol{x}^{(i)} \sim q}^{n} \frac{p(\boldsymbol{x}^{(i)}) f(\boldsymbol{x}^{(i)})}{q(\boldsymbol{x}^{(i)})}$ using $p(\boldsymbol{x}) f(\boldsymbol{x}) = q(\boldsymbol{x}) \frac{p(\boldsymbol{x}) f(\boldsymbol{x})}{q(\boldsymbol{x})}$
    - It can be readily seen that the expected value of the estimator does not depend on q: $E_q[\hat{s}_q] = E_q[\hat{s}_p] = s$
  - The variance of an importance sampling estimator is sensitive to the choice of q. The variance is $Var[\hat{s}_p] = Var\left[\frac{1}{n} \frac{p(\boldsymbol{x}) f(\boldsymbol{x})}{q(\boldsymbol{x})}\right]$
    - The minimum variance occurs when q is $q^*(\boldsymbol{x}) = \frac{p(\boldsymbol{x}) |f(\boldsymbol{x})|}{Z}$
    - where Z is the normalization constant chosen so that $q^*(\boldsymbol{x})$ sums or integrates to one

Choice of suboptimal $q(\boldsymbol{x})$

- Any choice of q is valid and q^* is the optimal one (yields minimum variance)
- Sampling from q^* is usually infeasible
- Other choices of q can be feasible, reducing variance somewhat

Biased Importance Sampling (BIS)

- Another approach is BIS
  - Has advantage of not requiring normalized p or q.
- For discrete variables, BIS estimator is

$$\hat{s}_{BIS} = \frac{\sum_{i=1}^{n} \frac{p(\boldsymbol{x}^{(i)})}{q(\boldsymbol{x}^{(i)})} f(\boldsymbol{x}^{(i)})}{\sum_{i=1}^{n} \frac{p(\boldsymbol{x}^{(i)})}{q(\boldsymbol{x}^{(i)})}} = \frac{\sum_{i=1}^{n} \frac{p(\boldsymbol{x}^{(i)})}{\tilde{q}(\boldsymbol{x}^{(i)})} f(\boldsymbol{x}^{(i)})}{\sum_{i=1}^{n} \frac{p(\boldsymbol{x}^{(i)})}{\tilde{q}(\boldsymbol{x}^{(i)})}} = \frac{\sum_{i=1}^{n} \frac{\tilde{p}(\boldsymbol{x}^{(i)})}{\tilde{q}(\boldsymbol{x}^{(i)})} f(\boldsymbol{x}^{(i)})}{\sum_{i=1}^{n} \frac{\tilde{p}(\boldsymbol{x}^{(i)})}{\tilde{q}(\boldsymbol{x}^{(i)})}},$$

where $p\hat{}$ and $q\hat{}$ are unnormalized forms of p and q
and $\boldsymbol{x}^{(i)}$ are samples from q

Effect of choice of q

- Good q → efficient Monte Carlo estimation
- Poor choice of q → efficiency much worse
  - Looking at $Var[\hat{s}_p] = Var\left[\frac{1}{n}\frac{p(\boldsymbol{x})f(\boldsymbol{x})}{q(\boldsymbol{x})}\right]$ if there are samples for which $\frac{p(\boldsymbol{x})f(\boldsymbol{x})}{q(\boldsymbol{x})}$ is large then the variance of the estimator can get large
    - Happens when $q(\boldsymbol{x})$ is tiny while neither $p(\boldsymbol{x})$ or $q(\boldsymbol{x})$ is small enough to cancel it
  - The q distribution is usually chosen to be a very simple distribution so that it is easy to sample from
    - When $\boldsymbol{x}$ is high dimensional this simplicity causes it to match p of $p|f|$ poorly
    - Very small or very large ratios are possible when $\boldsymbol{x}$ is high dimensional

Importance sampling in Deep learning

- Used in many ML algorithms incl. deep learning
- Examples
  - To accelerate training in neural language models with large vocabulary
    - Other neural nets with large no of outputs
  - Estimate partition function (normalize prob. distribution)
  - Estimate log-likelihood in deep directed models such as variational autoencoder
  - Estimate gradient in SGD where most of the cost comes from a small no of misclassified samples
    - Sampling more difficult examples can reduce variance of gradient

3. Markov Chain Monte Carlo Methods

- In many cases we wish to use a Monte Carlo technique but there is no tractable method for drawing exact samples from $p_{\text{model}}(\boldsymbol{x})$ or from a good (low variance) importance sampling distribution $q(\boldsymbol{x})$
- In deep learning this happens most often when $p_{\text{model}}(\boldsymbol{x})$ is represented by an undirected model
- In this case we use a mathematical tool called a Markov chain to sample from $p_{\text{model}}(\boldsymbol{x})$
- Algorithms that use Markov chains to perform Monte Carlo estimates are called MCMC

MCMC and Energy-Based Models

- Guarantees for MCMC are when model does not assign zero probability to any state
- Thus convenient to present these techniques as sampling from an energy-based model (EBM)

$$p(\boldsymbol{x}) \propto \exp(-E(\boldsymbol{x}))$$

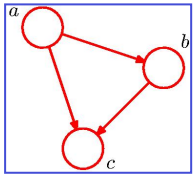
$$p(\mathbf{x}) = \frac{1}{Z}\tilde{p}(\mathbf{x})$$

$$\tilde{p}(\mathbf{x}) = \prod_{C \in G} \phi(C)$$

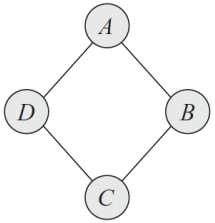
$$Z = \int \tilde{p}(\mathbf{x})\, d\mathbf{x}$$

$$\tilde{p}(\mathbf{x}) = \exp(-E(\mathbf{x}))$$

Need more than ancestral sampling



- Ancestral Sampling for directed acyclic graphs:
  - Start with lowest numbered node
  - Draw a sample from the distribution $p(x_1)$ which we call $\hat{x}_1$
  - Work through each of the nodes in order
    - For node n we draw a sample from conditional distribution $p(x_n|pa_n)$
  - Defines an efficient single pass algorithm
- Not so simple in EBMs: chicken-egg problem
  - In order to sample from A we need to draw from $p(A|B,D)$
  - In order to sample from B we need to sample from $p(B|A,C)$



- Core idea of Markov chain
  - Have a state $\boldsymbol{x}$ that begins with an arbitrary value
  - Over time we repeatedly update $\boldsymbol{x}$
  - Eventually $\boldsymbol{x}$ becomes a fair sample from $p(\boldsymbol{x})$
  - Markov chain is defined by random state $\boldsymbol{x}$ and transition distribution $T(\boldsymbol{x}'|\ \boldsymbol{x})$

Theoretical understanding of MCMC

- Reparameterize the problem
- Restrict attention to the case where r.v. $\boldsymbol{x}$ has countably many states
  - Represent the state as an integer x
- The different states are drawn from some distribution $q^{(t)}(x)$ where t is no of time steps
- Our goal is for $q^{(t)}(x)$ to converge to $p(x)$
  - If we have chosen T correctly then the stationary distribution q will be equal to the distribution p we wish to sample from
  - We describe how to choose T next

4. Gibbs Sampling

- The conceptually simplest approach for drawing samples from an undirected graph
- Suppose we have a graphical model over an n-dimensional vector of random variables $\mathbf{x}$
- We iteratively visit each variable x_i and draw a sample conditioned on all the other variables, i.e., from $p(\mathrm{x}_i|\mathrm{x}_{-i})$
- Due to the separation properties of the graphical model, we can equivalently condition on only the neighbors of x_i

Gibbs Sampling with M variables

- Initialize first sample: $\{z_i,\ i=1,..,M\}$
- For $t=1,..,T,\ T =$ no of samples
  - Sample $z_1^{(t+1)} \sim p(z_1|z_2^{(t)},z_3^{(t)},\ldots,\ z_M^{(t)})$
  - Sample $z_2^{(t+1)} \sim p(z_2|z_1^{(t+1)},z_3^{(t)},\ldots,\ z_M^{(t)})$
  - .....
  - Sample $z_j^{(t+1)} \sim p(z_j|z_1^{(t+1)},..z_{j-1}^{(t+1)},\ z_{j+1}^{(t)}\ \ldots,\ z_M^{(t)})$
  - .....
  - Sample $z_M^{(t+1)} \sim p(z_M|z_1^{(t+1)},z_2^{(t+1)},\ldots,\ z_{M-1}^{(t+1)})$
- $p(z_j|z_{-j})$ is called a *full conditional* for variable j