Complexity of Inference

+ 

Monte Carlo Methods
COMPUTATIONAL COMPLEXITY
OF INFERENCE
Question: In order to prove that a decision problem is NP-Hard, we must...

A. ... reduce our decision problem to a known NP-Hard problem.
B. ... reduce a known NP-Hard problem to our decision problem.

Answer:
Complexity Classes

- An algorithm runs in **polynomial time** if its runtime is a polynomial function of the input size (e.g. $O(n^k)$ for some fixed constant $k$)
- The class **P** consists of all problems that can be solved in polynomial time

- A problem for which the answer is binary (e.g. yes/no) is called a **decision problem**
- The class **NP** contains all decision problems where ‘yes’ answers can be verified (proved) in polynomial time
- A problem is **NP-Hard** if given an $O(1)$ oracle to solve it, every problem in NP can be solved in polynomial time (e.g. by reduction)
- A problem is **NP-Complete** if it belongs to both the classes NP and NP-Hard

Figure from https://en.wikipedia.org/wiki/NP-completeness
Complexity Classes

- A problem for which the answer is a nonnegative integer is called a **counting problem**.
- The class $\#P$ contains the counting problems that align to decision problems in NP.
  - Really this is the class of problems that count the number of accepting paths in a Turing machine that is nondeterministic and runs in polynomial time.
- A problem is **$\#P$-Hard** if given an $O(1)$ oracle to solve it, every problem in $\#P$ can be solved in polynomial time (e.g. by reduction).
- A problem is **$\#P$-Complete** if it belongs to both the classes $\#P$ and $\#P$-Hard.
- There are no known polytime algorithms for solving $\#P$-Complete problems. If we found one it would imply that $P = NP$.

**Examples of $\#P$-Hard problems**
- $\#\text{SAT}$, i.e. how many satisfying solutions for a given SAT problem?
- How many solutions for a given DNF formula?
- How many solutions for a 2-SAT problem?
- How many perfect matchings for a bipartite graph?
- How many graph colorings (with $k$ colors) for a given graph $G$?

Examples from https://en.wikipedia.org/wiki/%E2%99%A5P-complete
5. Inference

Three Tasks:

1. **Marginal Inference (#P-Hard)**
   Compute marginals of variables and cliques
   \[
   p(x_i) = \sum_{x': x'_i = x_i} p(x' | \theta) \quad \bigg| \quad p(x_C) = \sum_{x': x'_C = x_C} p(x' | \theta)
   \]

2. **Partition Function (#P-Hard)**
   Compute the normalization constant
   \[
   Z(\theta) = \sum_x \prod_{C \in \mathcal{C}} \psi_C(x_C)
   \]

3. **MAP Inference (NP-Hard)**
   Compute variable assignment with highest probability
   \[
   \hat{x} = \arg\max_x p(x | \theta)
   \]
3-SAT

Background:

• Formulas
  – *Def*: a **literal** is a binary variable or its negation, e.g. $x_1$ is a positive literal and $\neg x_1$ is a negative literal, where $x_1 \in \{0, 1\}$
  – *Def*: a **clause** is a disjunction of literals, e.g. $(\neg x_1 \lor x_2 \lor \neg x_3)$
  – *Def*: a formula is in **conjunctive normal form (CNF)** if it is a conjunction of clauses, e.g.
    $$(\neg x_1 \lor x_2 \lor \neg x_3) \land (x_2 \lor x_4 \lor \neg x_6) \land (x_1 \lor \neg x_3 \lor \neg x_5)$$

• The 3-SAT Problem
  – *Given*: a CNF formula where each clause has at most 3 literals
  – *Goal*: report the satisfiability of the formula, i.e. whether there is a satisfying assignment to the variables that makes the entire formula true
Computational Complexity of MAP Inference

- **Claim:** MAP inference is NP-Hard
- **Proof Sketch:**
  
  **Overview:** we reduce 3-SAT (known to be NP-Hard) to the MAP Inference problem

1. Construct a factor graph as follows:
   a. add a variable $x_i$ to the factor graph for each variable in 3-SAT
   b. add a variable $c_l$ to the factor graph for each clause in 3-SAT
   c. add a factor $\Psi(c_l, x_i, x_j, x_k)$ for each clause $c_l(x_i, x_j, x_k)$
   d. let the factor $\Psi(c_l, x_i, x_j, x_k) = 1$ if $c_l(x_i, x_j, x_k) = \text{true}$ and $\Psi(x_i, x_j, x_k) = 0$ otherwise

2. Run MAP inference to obtain the most probable assignment

3. Return true if all the clause variables are true; and false otherwise
#-SAT

**Background:**

- **The 3-SAT Problem**
  - **Given:** a CNF formula where each clause has at most 3 literals
  - **Goal:** report the satisfiability of the formula, i.e. whether there is a **satisfying assignment** to the variables that makes the entire formula true

- **The #-SAT Problem**
  - **Given:** a CNF formula where each clause has at most 3 literals
  - **Goal:** report the **number of satisfying assignments** of the formula
Computational Complexity of Marginal Inference

• **Claim:** Marginal inference is #P-Hard

• **Proof Sketch:**
  
  **Overview:** we reduce #\text{-}\text{SAT} (known to be #P-Hard) to the marginal inference problem

  1. Construct a factor graph as follows:
     a. ...left as an exercise...
  2. Run marginal inference
  3. Return the number of satisfying assignments by...
     a. ...left as an exercise...
APPROXIMATE MARGINAL INFERENCE
1. **Data**

\[ \mathcal{D} = \{ x^{(n)} \}_{n=1}^{N} \]

2. **Model**

\[ p(x \mid \theta) = \frac{1}{Z(\theta)} \prod_{C \in \mathcal{C}} \psi_C(x_C) \]

3. **Objective**

\[ \ell(\theta ; \mathcal{D}) = \sum_{n=1}^{N} \log p(x^{(n)} \mid \theta) \]

5. **Inference**

1. **Marginal Inference**

\[ p(x_C) = \sum_{x' : x'_C = x_C} p(x' \mid \theta) \]

2. **Partition Function**

\[ Z(\theta) = \sum_{x} \prod_{C \in \mathcal{C}} \psi_C(x_C) \]

3. **MAP Inference**

\[ \hat{x} = \arg\max_{x} p(x \mid \theta) \]

4. **Learning**

\[ \theta^* = \arg\max_{\theta} \ell(\theta ; \mathcal{D}) \]
A Few Problems for a Factor Graph

Suppose we already have the parameters of a Factor Graph...

1. How do we compute the probability of a specific assignment to the variables?
   \[ P(T=t, H=h, A=a, C=c) \]

2. How do we draw a sample from the joint distribution?
   \[ t, h, a, c \sim P(T, H, A, C) \]

3. How do we compute marginal probabilities?
   \[ P(A) = \ldots \]

4. How do we draw samples from a conditional distribution?
   \[ t, h, a \sim P(T, H, A \mid C = c) \]

5. How do we compute conditional marginal probabilities?
   \[ P(H \mid C = c) = \ldots \]
Marginals by Sampling on Factor Graph

Suppose we took many samples from the distribution over taggings:

\[ p(x) = \frac{1}{Z} \prod_{\alpha} \psi_{\alpha}(x_{\alpha}) \]

Sample 1:  
\[ n \rightarrow v \rightarrow p \rightarrow d \rightarrow n \]

Sample 2:  
\[ n \rightarrow n \rightarrow v \rightarrow d \rightarrow n \]

Sample 3:  
\[ n \rightarrow v \rightarrow p \rightarrow d \rightarrow n \]

Sample 4:  
\[ v \rightarrow n \rightarrow p \rightarrow d \rightarrow n \]

Sample 5:  
\[ v \rightarrow n \rightarrow v \rightarrow d \rightarrow n \]

Sample 6:  
\[ n \rightarrow v \rightarrow p \rightarrow d \rightarrow n \]

Suppose we took many samples from the distribution over taggings:

\[ p(x) = \frac{1}{Z} \prod_{\alpha} \psi_{\alpha}(x_{\alpha}) \]

Sample 1:  
\[ n \rightarrow v \rightarrow p \rightarrow d \rightarrow n \]

Sample 2:  
\[ n \rightarrow n \rightarrow v \rightarrow d \rightarrow n \]

Sample 3:  
\[ n \rightarrow v \rightarrow p \rightarrow d \rightarrow n \]

Sample 4:  
\[ v \rightarrow n \rightarrow p \rightarrow d \rightarrow n \]

Sample 5:  
\[ v \rightarrow n \rightarrow v \rightarrow d \rightarrow n \]

Sample 6:  
\[ n \rightarrow v \rightarrow p \rightarrow d \rightarrow n \]

\[ \sqrt[n]{\frac{1}{Z}} \prod_{\alpha} \psi_{\alpha}(x_{\alpha}) \]

\[ \sqrt[n]{\frac{1}{Z}} \prod_{\alpha} \psi_{\alpha}(x_{\alpha}) \]

\[ \sqrt[n]{\frac{1}{Z}} \prod_{\alpha} \psi_{\alpha}(x_{\alpha}) \]

\[ \sqrt[n]{\frac{1}{Z}} \prod_{\alpha} \psi_{\alpha}(x_{\alpha}) \]

\[ \sqrt[n]{\frac{1}{Z}} \prod_{\alpha} \psi_{\alpha}(x_{\alpha}) \]

\[ \sqrt[n]{\frac{1}{Z}} \prod_{\alpha} \psi_{\alpha}(x_{\alpha}) \]

\[ \sqrt[n]{\frac{1}{Z}} \prod_{\alpha} \psi_{\alpha}(x_{\alpha}) \]
Marginals by Sampling on Factor Graph

The marginal $p(X_i = x_i)$ gives the probability that variable $X_i$ takes value $x_i$ in a random sample.
Marginals by Sampling on Factor Graph

Estimate the marginals as:

Sample 1:

Sample 2:

Sample 3:

Sample 4:

Sample 5:

Sample 6:

<START>
MONTE CARLO METHODS
Monte Carlo Methods

Whiteboard

– Problem 1: Generating samples from a distribution
– Problem 2: Estimating expectations
– Why is sampling from p(x) hard?
– Example: estimating plankton concentration in a lake
– Algorithm: Uniform Sampling
– Example: estimating partition function of high dimensional function
Properties of Monte Carlo

Estimator: \[
\int f(x)P(x) \, dx \approx \hat{f} \equiv \frac{1}{S} \sum_{s=1}^{S} f(x^{(s)}), \quad x^{(s)} \sim P(x)
\]

Estimator is unbiased:
\[
\mathbb{E}_{P(\{x^{(s)}\})}[\hat{f}] = \frac{1}{S} \sum_{s=1}^{S} \mathbb{E}_{P(x)}[f(x)] = \mathbb{E}_{P(x)}[f(x)]
\]

Variance shrinks \( \propto 1/S \):
\[
\text{var}_{P(\{x^{(s)}\})}[\hat{f}] = \frac{1}{S^2} \sum_{s=1}^{S} \text{var}_{P(x)}[f(x)] = \text{var}_{P(x)}[f(x)] / S
\]

“Error bars” shrink like \( \sqrt{S} \)
A dumb approximation of $\pi$

$$P(x, y) = \begin{cases} 
1 & 0 < x < 1 \text{ and } 0 < y < 1 \\
0 & \text{otherwise}
\end{cases}$$

$$\pi = 4 \int \int \mathbb{1} ((x^2 + y^2) < 1) P(x, y) \, dx \, dy$$

octave:1> S=12; a=rand(S,2); 4*mean(sum(a.*a,2)<1)
ans = 3.3333
octave:2> S=1e7; a=rand(S,2); 4*mean(sum(a.*a,2)<1)
ans = 3.1418
Aside: don’t always sample!

“Monte Carlo is an extremely bad method; it should be used only when all alternative methods are worse.”

— Alan Sokal, 1996

Example: numerical solutions to (nice) 1D integrals are fast

octave:1> 4 * quadl(@(x) sqrt(1-x.^2), 0, 1, tolerance)

Gives $\pi$ to 6 dp’s in 108 evaluations, machine precision in 2598.

(NB Matlab’s quadl fails at zero tolerance)
Sampling from distributions

Draw points uniformly under the curve:

Probability mass to left of point $\sim$ Uniform[0,1]
Sampling from distributions

How to convert samples from a Uniform[0,1] generator:

\[ h(y) = \int_{-\infty}^{y} p(y') \, dy' \]

Draw mass to left of point:
\[ u \sim \text{Uniform}[0,1] \]

Sample:
\[ y(u) = h^{-1}(u) \]

Although we can’t always compute and invert \( h(y) \)
Rejection Sampling

Whiteboard:

– Example: Rejection Sampling with a rectangular proposal
Rejection sampling

Sampling underneath a \( \tilde{P}(x) \propto P(x) \) curve is also valid

Draw underneath a simple curve \( k\tilde{Q}(x) \geq \tilde{P}(x) \):
- Draw \( x \sim Q(x) \)
- height \( u \sim \text{Uniform}[0, k\tilde{Q}(x)] \)

Discard the point if above \( \tilde{P} \), i.e. if \( u > \tilde{P}(x) \)
Importance sampling

Computing \( \tilde{P}(x) \) and \( \tilde{Q}(x) \), then *throwing* \( x \) *away* seems wasteful. Instead rewrite the integral as an expectation under \( Q \):

\[
\int f(x) P(x) \, dx = \int f(x) \frac{P(x)}{Q(x)} Q(x) \, dx, \quad (Q(x) > 0 \text{ if } P(x) > 0)
\]

\[
\approx \frac{1}{S} \sum_{s=1}^{S} f(x^{(s)}) \frac{P(x^{(s)})}{Q(x^{(s)})}, \quad x^{(s)} \sim Q(x)
\]

This is just simple Monte Carlo again, so it is unbiased.

Importance sampling applies when the integral is not an expectation. Divide and multiply any integrand by a convenient distribution.
Importance sampling (2)

Previous slide assumed we could evaluate $P(x) = \tilde{P}(x)/\mathcal{Z}_P$

$$\int f(x) P(x) \, dx \approx \frac{\mathcal{Z}_Q}{\mathcal{Z}_P} \frac{1}{S} \sum_{s=1}^{S} f(x^{(s)}) \frac{\tilde{P}(x^{(s)})}{\tilde{Q}(x^{(s)})} \tilde{r}(s), \quad x^{(s)} \sim Q(x)$$

$$\approx \frac{1}{S} \sum_{s=1}^{S} f(x^{(s)}) \frac{\tilde{r}(s)}{\frac{1}{S} \sum_{s'} \tilde{r}(s')} \equiv \sum_{s=1}^{S} f(x^{(s)}) w^{(s)}$$

This estimator is **consistent** but **biased**

**Exercise:** Prove that $\mathcal{Z}_P/\mathcal{Z}_Q \approx \frac{1}{S} \sum_{s} \tilde{r}(s)$
• Sums and integrals, often expectations, occur frequently in statistics
• **Monte Carlo** approximates expectations with a sample average
• **Rejection sampling** draws samples from complex distributions
• **Importance sampling** applies Monte Carlo to ‘any’ sum/integral
Pitfalls of Monte Carlo

Rejection & importance sampling scale badly with dimensionality

Example:

\[ P(x) = \mathcal{N}(0, \mathbb{I}), \quad Q(x) = \mathcal{N}(0, \sigma^2\mathbb{I}) \]

Rejection sampling:
Requires \( \sigma \geq 1 \). Fraction of proposals accepted = \( \sigma^{-D} \)

Importance sampling:

Variance of importance weights = \( \left( \frac{\sigma^2}{2^{1/\sigma^2}} \right)^{D/2} - 1 \)

Infinite / undefined variance if \( \sigma \leq 1/\sqrt{2} \)