Parameter Learning

Le Song

Machine Learning II: Advanced Topics
CSE 8803ML, Spring 2012
Sampling Methods

- **Direct Sampling**
  - Works only for easy distributions (multinomial, Gaussian etc.)

- **Rejection Sampling**
  - Create samples like direct sampling
  - Only count samples consistent with given evidence

- **Importance Sampling**
  - Create samples like direct sampling
  - Assign weights to samples

- **Gibbs Sampling**
  - Often used for high-dimensional problem
  - Use variables and its Markov blanket for sampling
Gibbs Sampling in formula

- **Gibbs sampling**
  - \( X = x^0 \)
  - For \( t = 1 \) to \( N \)
    - \( x_1^t \sim P(X_1 | x_2^{t-1}, ..., x_K^{t-1}) \)
    - \( x_2^t \sim P(X_2 | x_1^t, x_3^{t-1}, ..., x_K^{t-1}) \)
    - ...
    - \( x_K^t \sim P(X_K | x_1^t, ..., x_{K-1}^t) \)

- **Variants:**
  - Randomly pick variable to sample
  - sample block by block
    - \((x_1^t, x_2^t) \sim P(X_1, X_2 | x_3^{t-1}, ..., x_K^{t-1})\)

*For graphical models*
Only need to condition on the Variables in the Markov blanket
Gibbs Sampling: Image Segmentation

- Pairwise Markov random fields: \( P(X) = \frac{1}{Z} \prod_i \Psi(X_i) \prod_{ij} \Psi(X_i, X_j) \)

- Need conditional \( P(x_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_k) \)

\[
\frac{P(x_1, \ldots, x_k)}{P(x_1, \ldots, x_{i-1}, x_{i+1}, x_k)} = \frac{1}{Z} \prod_l \Psi(x_l) \prod_{lj} \Psi(x_l, x_j) \frac{1}{\sum_{x_i} \prod_l \Psi(x_l) \prod_{lj} \Psi(x_l, x_j)}
\]

- Terms without \( x_i \) will cancel out

- \( x_i \) is summed out in the denominator

\( \propto \Psi(x_i) \prod_{j \in N(i)} \Psi(x_i, x_j) \)
Convergence of Gibbs Sampling

- Not all samples $x^0, \ldots x^T$ are independent
- Consider a particular marginal $P(x_i|u_i)$

![Graph showing the convergence of Gibbs sampling with burn-in and empirical $P(x_i|u_i)$]

- True $P(x_i|u_i)$
- Empirical $\hat{P}(x_i|u_i)$
- Burn-in
- Take samples from here
Parameter Learning Example

- Estimate the probability $\theta$ of landing in heads using a biased coin

- Given a sequence of $N$ independently and identically distributed (iid) flips
  - Eg., $D = \{x_1, x_2, \ldots, x_N\} = \{1,0,1, \ldots, 0\}, x_i \in \{0,1\}$

- Model: $P(x|\theta) = \theta^x (1 - \theta)^{1-x}$
  - $P(x|\theta) = \begin{cases} 1 - \theta, & \text{for } x = 0 \\ \theta, & \text{for } x = 1 \end{cases}$

- Likelihood of a single observation $x_i$?
  - $P(x_i|\theta) = \theta^{x_i}(1 - \theta)^{1-x_i}$
Bayesian Parameter Estimation

- Bayesian treats the unknown parameters as a random variable, whose distribution can be inferred using Bayes rule:

  \[ P(\theta | D) = \frac{P(D|\theta)P(\theta)}{P(D)} = \frac{P(D|\theta)P(\theta)}{\int P(D|\theta)P(\theta)d\theta} \]

- Prior over \( \theta \), Beta distribution

  \[ P(\theta; \alpha, \beta) \propto \theta^{\alpha-1}(1 - \theta)^{\beta-1} \]

- Posterior distribution \( \theta \)

  \[ P(\theta | x_1, \ldots, x_N) = \frac{P(x_1,\ldots,x_N|\theta)P(\theta)}{P(x_1,\ldots,x_N)} \propto \theta^{nh}(1 - \theta)^{nt}\theta^{\alpha-1}(1 - \theta)^{\beta-1} = \theta^{nh+\alpha-1}(1 - \theta)^{nt+\beta-1} \]

- Posterior mean estimation:

  \[ \theta_{bayes} = \int \theta \ P(\theta|D)d\theta = C \int \theta \times \theta^{nh+\alpha-1}(1 - \theta)^{nt+\beta-1}d\theta = \frac{(nh+\alpha)}{N+\alpha+\beta} \]
MLE for Biased Coin

$$\theta_{ML} = \text{argmax}_\theta P(D|\theta)$$

Objective function, log likelihood

$$l(\theta; D) = \log P(D|\theta) = \log \theta^{n_h} (1 - \theta)^{n_t} = n_h \log \theta + (N - n_h) \log (1 - \theta)$$

We need to maximize this w.r.t. $\theta$

Take derivatives w.r.t. $\theta$

$$\frac{\partial l}{\partial \theta} = \frac{n_h}{\theta} - \frac{(N-n_h)}{1-\theta} = 0 \Rightarrow \hat{\theta}_{MLE} = \frac{n_h}{N} \text{ or } \hat{\theta}_{MLE} = \frac{1}{N} \sum_i x_i$$
How estimators should be used?

- $\theta_{MAP} = \text{argmax}_\theta P(\theta | D)$ is not Bayesian (even though it uses a prior) since it is a point estimate.

Consider predicting the future. A sensible way is to combine predictions based on all possible value of $\theta$, weighted by their posterior probability, this is called Bayesian prediction:

- $P(x_{new} | D) = \int P(x_{new}, \theta | D) d\theta$
- $= \int P(x_{new} | \theta, D)P(\theta | D) d\theta$
- $= \int P(x_{new} | \theta)P(\theta | D) d\theta$

A frequentist prediction will typically use a “plug-in” estimator such as ML/MAP

- $P(x_{new} | D) = P(x_{new} | \theta_{ML})$ or $P(x_{new} | D) = P(x_{new} | \theta_{MAP})$
Learning Graphical Models

The goal: given set of independent samples (assignments of random variables), find the best (the most likely) graphical model (both structure and the parameters)

\[(A,F,S,N,H) = (T,F,F,T,F)\]
\[(A,F,S,N,H) = (T,F,T,T,F)\]
\[...\]
\[(A,F,S,N,H) = (F,T,T,T)\]

<table>
<thead>
<tr>
<th>S</th>
<th>FA</th>
<th>TT</th>
<th>TF</th>
<th>FT</th>
<th>FF</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>0.9</td>
<td>0.7</td>
<td>0.8</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>0.1</td>
<td>0.3</td>
<td>0.2</td>
<td>0.8</td>
<td></td>
</tr>
</tbody>
</table>
MLE for General Bayesian Networks

- If we assume that the parameters for each CPT are globally independent, and all nodes are fully observed, then the log-likelihood function decomposes into a sum of local terms, one per node:
  \[ l(\theta; D) = \log P(D | \theta) \]
  \[ = \log \prod_i \prod_j P(x^i_j | pa^i_{X_j}, \theta_j) = \sum_i \sum_j \log P(x^i_j | pa^i_{X_j}, \theta_j) \]

- For each variable \( X_i \):
  \[ P_{MLE}(X_i = x_i | pa_{X_i} = u) = \frac{\#(X_i=x,Pa_{X_i}=u)}{\#(Pa_{X_i}=u)} \]

- Why?
Decomposable likelihood of directed model

\[ l(\theta; D) = \log P(D|\theta) = \]
\[ \sum_i \log P(a^i|\theta_a) + \sum_i \log P(f^i|\theta_f) + \sum_i \log P(s^i|a^i, f^i, \theta_s) + \sum_i \log P(h^i|s^i, \theta_h) \]

One term for each CPT; break up MLE problem into independent subproblems

Because the factorization of the distribution, we can estimate each CPT separately.
MLE for BNs with tabluar CPTs

- Assume each CPT is represented as a table (multinomial):
  \[ \theta_{ijk} = P(X_i = j | X_{\pi_i} = k) \]

  - Note that in case of multiple parents, \( X_{\pi_i} \) will have a composite state, and CPT will be a high dimensional table

  - The sufficient statistics are counts of family configurations
    \[ n_{ijk} = \#(X_i = j \& \& X_{\pi_i} = k) \]

- The log-likelihood is
  \[ L(\theta; D) = \log \prod_{ijk} \theta_{ijk}^{n_{ijk}} = \sum_{ijk} n_{ijk} \log \theta_{ijk} \]

- Using a Lagrange multiplier to enforce \( \sum_j \theta_{ijk} = 1 \), we get
  \[ \theta_{ijk}^{ML} = \frac{n_{ijk}}{\sum_{ij'k} n_{ij'k}} \]
MLE and Kullback-Leibler divergence

- KL divergence
  
  \[ D(Q(X) \| P(X)) = \sum_x Q(x) \log \frac{Q(x)}{P(x)} \]

- Empirical distribution
  
  \[ \hat{P}(X) = \frac{1}{N} \sum_{i=1}^{N} \delta(X, x_i) \]
  
  Where \( \delta(X, x_i) \) is a Kronecker delta function

\[ \text{Max}_\theta \text{MLE} = \text{Min}_\theta \text{KL} \]

\[ D(\hat{P}(X) \| P(X|\theta)) = \sum_x \hat{P}(x) \log \frac{\hat{P}(x)}{P(x|\theta)} \]

\[ = \sum_x \hat{P}(x) \log \hat{P}(x) - \sum_x \hat{P}(x) \log P(x|\theta) \]

\[ = \text{const.} - \frac{1}{N} \sum_{i=1}^{N} \log P(x_i|\theta) = \text{const.} - \frac{1}{N} l(\theta; D) \]
Bayesian estimator for tabular CPTs

- Factorization \( P(X = x) = \prod_i P(x_i \mid p_{aX_i}, \theta_i) \)

- Local CPT: multinomial distribution \( P(X_i = k \mid P_{aX_i} = j) = \theta_{kj} \)

- Put prior distribution over parameters \( P(\theta_a, \theta_b, \theta_s, \theta_h) \)
Global and Local Parameter Independence

- **Global** parameter independence
  - Parameters for all nodes in a GM
  - \( P(\theta_a, \theta_b, \theta_s, \theta_h) = P(\theta_a)P(\theta_b)P(\theta_s)P(\theta_h) \)

- **Local** Parameter Independence
  - Parameters in each node
  - \( P(X_i = k | P_a X_i = j) = \theta_{kj} \)
  - \( P(\theta_a) = \prod_{kj} P(\theta_{kj}) \)
Parameter independence Example

Provided all variables are observed, we can perform Bayesian update for each parameter independently.
Which distribution satisfy independence assumptions?

- Discrete DAG models:
  - \( X_i | Pa_{X_i} \sim \text{Multi}(\theta) \)
  - Dirichlet prior: \( P(\theta) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_k \theta_k^{\alpha_k} \)

- Gaussian DAG models:
  - \( X_i | Pa_{X_i} \sim \text{Normal}(\mu, \Sigma) \)
  - Normal prior: \( P(\mu | \nu, \Psi) \propto \exp \left( -\frac{1}{2} (\mu - \nu)' \Psi^{-1} (\mu - \nu) \right) \)
Parameter sharing

- Consider a time-invariant (stationary) first order Markov model
  - Initial state probability vector: \( \pi_j = P(X_1 = j) \)
  - State transition probability matrix: \( A_{ij} = P(X_t = j | X_{t-1} = i) \)

- The likelihood of one sequence
  \[ P(x_{1:T} | \theta) = P(x_1 | \pi) \prod_{t=2}^{T} P(x_t | x_{t-1}) \]

- The log-likelihood of \( N \) sequences
  \[ l(\theta; x_{1:T}) = \sum_{i=1}^{N} \log P(x_1^l | \pi) + \sum_{i=1}^{N} \sum_{t=2}^{T} \log P(x_t^l | x_{t-1}^l) \]

- Again, we can optimize each parameter separately
  - \( \pi \) can be simply estimated by the frequency
  - How about \( A \)?
Learning a Markov chain transition matrix

- $A$ is a stochastic matrix: $\sum_j A_{ij} = 1$
- Each row of $A$ is multinomial distribution
- MLE of $A_{ij}$ is the fraction of transitions from $i$ to $j$
  $$A_{ij}^{ML} = \frac{\#(i \to j)}{\#(i \to \cdot)}$$

Application:
- If the states of $X_t$ represent words, this is called a bigram language model

Small sample size problem:
- If $i \to j$ did not occur in data, we will have $A_{ij}^{ML} = 0$
- A standard hack: backoff smoothing $A_i^S = \lambda \eta + (1 - \lambda)A_i^{ML}$
Bayesian model for Markov chain

- Global and local parameter independence

- The posterior of $A_i\rightarrow.$ and $A_i'\rightarrow.$ is factorized despite the v-structure

- Assign a Dirichlet prior $\beta_i$ to each row of the transition matrix

\[
A_{ij}^{Bayes} = \frac{#(i\rightarrow j) + \beta_{ij}}{#(i\rightarrow \cdot) + \Sigma_j \beta_{ij}}
\]

\[
= \lambda \beta_{ij}' + (1 - \lambda) A_{ij}^{ML} \quad \text{where} \quad \lambda = \frac{\Sigma_j \beta_{ij}}{#(i\rightarrow \cdot) + \Sigma_j \beta_{ij}}
\]
MLE for Exponential Family

\[ P(X_1, ..., X_k | \theta) = \frac{1}{Z(\theta)} \exp(\sum_{ij} \theta_{ij} X_i X_j + \sum_i \theta_i X_i) \]
\[ = \frac{1}{Z(\theta)} \prod_{ij} \exp(\theta_{ij} X_i X_j) \prod_i \exp(\theta_i X_i) \]
\[ Z(\theta) = \sum_x \prod_{ij} \exp(\theta_{ij} X_i X_j) \prod_i \exp(\theta_i X_i) \]

\[ l(\theta, D) = \log(\prod_{l=1}^N \frac{1}{Z(\theta)} \prod_{ij} \exp(\theta_{ij} x_i^l x_j^l) \prod_i \exp(\theta_i x_i^l)) \]
\[ = \sum_i^N \left( \sum_{ij} \log(\exp(\theta_{ij} x_i^l x_j^l)) + \sum_i \log(\exp(\theta_i x_i^l)) - \log Z(\theta) \right) \]
\[ = \sum_i^N \left( \sum_{ij} \theta_{ij} x_i^l x_j^l + \sum_i \theta_i x_i^l - \log Z(\theta) \right) \]

*can be other feature function f(x_i)*
MLE for Exponential Family

\[ l(\theta, D) = \frac{1}{N} \sum_{l}^{N} \left( \sum_{ij} \theta_{ij} x_i^l x_j^l + \sum_{i} \theta_i x_i^l - \log Z(\theta) \right) \]

\[ \frac{\partial l(\theta,D)}{\partial \theta_{ij}} = \frac{1}{N} \sum_{l}^{N} \sum_{ij} x_i^l x_j^l - \frac{\partial \log Z(\theta)}{\partial \theta_{ij}} \]

\[ \frac{\partial l(\theta,D)}{\partial \theta_{ij}} = \frac{1}{N} \sum_{l}^{N} \sum_{ij} x_i^l x_j^l - \frac{1}{Z(\theta)} \frac{\partial Z(\theta)}{\partial \theta_{ij}} \]

\[ = \frac{1}{N} \sum_{l}^{N} x_i^l x_j^l - \frac{1}{Z(\theta)} \sum_{x} \prod_{ij} \exp(\theta_{ij}X_iX_j) \prod_i \exp(\theta_iX_i) \ X_iX_j \]

need to do inference
Moment Matching condition

\[ \frac{\partial l(\theta, D)}{\partial \theta_{ij}} = \frac{1}{N} \sum^N_l x_i^l x_j^l - \frac{1}{Z(\theta)} \sum x \prod_{ij} \exp(\theta_{ij} x_i x_j) \prod_i \exp(\theta_i x_i) \ x_i x_j \]

\text{empirical covariance matrix}

\text{covariance matrix from model}

\[ \tilde{P}(X_i, X_j) = \frac{1}{N} \sum_{l=1}^N \delta(X_i, x_i^l) \delta(X_j, x_j^l) \]

\[ \frac{\partial l(\theta, D)}{\partial \theta_{ij}} = E_{\tilde{P}(X_i, X_j)}[X_i X_j] - E_{P(X|\theta)}[X_i X_j] \]
MLE Learning Algorithm for Exponential models

- \( \max_\theta l(\theta, D) \) is a convex optimization problem.

- Can be solve by many methods, such as gradient descent, conjugate gradient.

- Initialize model parameters \( \theta \)

- Loop until convergence
  - Compute \( \frac{\partial l(\theta, D)}{\partial \theta_{ij}} = E_{\hat{P}(X_i X_j)}[X_i X_j] - E_{P(X|\theta)}[X_i X_j] \)
  - Update \( \theta_{ij} \leftarrow \theta_{ij} - \eta \frac{\partial l(\theta, D)}{\partial \theta_{ij}} \)