Learning in Graphical Models

Le Song
Lecture 20, Nov 6, 2012

Reading: Chap 8, C. Bishop Book
Message passing algorithm

\[ m_{ji}(X_i) \propto \sum_{X_j} \Psi(X_i, X_j) \Psi(X_j) \prod_{s \in N(j) \setminus i} m_{sj}(X_j) \]

- **Product of incoming messages**
- **Multiply by local potentials**
- **Sum out** \( X_j \)

\( X_j \) can send message when incoming messages from \( N(j) \setminus i \) arrive
Recall Variable Elimination Algorithm

- Choose an ordering in which the query node $f$ is the final node
- Eliminate node $i$ by removing all potentials containing $i$, take sum/product over $x_i$
- Place the resultant factor back

For a **Tree** graphical model:

- Choose query node $f$ as the root of the tree
- View tree as a directed tree with edges pointing towards $f$
- Elimination of each node can be considered as **message-passing** directly along tree branches, rather than on some transformed graphs
- Thus, we can use the tree itself as a data-structure to inference
How about general graph?

- Trees are nice
  - Can just compute two messages for each edge
  - Order computation along the graph
  - Associate intermediate results with edges

- General graph is not so clear
  - Different elimination generate different cliques and factor size
  - Computation and immediate results not associated with edges
  - Local computation view is not so clear

Can we make them tree like or treat them as trees?
Message passing for loopy graph

- Local message passing for trees guarantees the consistency of local marginals
  - $P(X_i)$ computed is the correct one
  - $P(X_i, X_j)$ computed is the correct one
  - ...

- For loopy graphs, no consistency guarantees for local message passing
Inference for loopy graph models is NP-hard in general

Treat loopy graphs locally as if they were trees

Iteratively estimate the marginal
- Read in messages
- Process messages
- Send updated out messages

Repeat for all variables until convergence
Message update schedule

- **Synchronous update:**
  - $X_j$ can send message when incoming messages from $N(j) \setminus i$ arrive
  - Slow
  - Provably correct for tree, may converge for loopy graphs

- **Asynchronous update:**
  - $X_j$ can send message when there is a change in any incoming messages from $N(j) \setminus i$
  - Fast
  - Not easy to prove convergence, but empirically it often works
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** \( \theta \)

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

- **Posterior mean estimation**:

  \[ \theta_{bayes} = \int \theta \ P(\theta | D) d\theta = C \int \theta \times \theta^{n_h+\alpha-1}(1 - \theta)^{n_t+\beta-1} d\theta = \]

  \[ \frac{(n_h+\alpha)}{N+\alpha+\beta} \]
MLE for Biased Coin

\[ \theta_{ML} = \arg \max_\theta P(D|\theta) \]

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. \( \theta \)

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

\[ \theta_{MAP} = \arg\max_{\theta} P(\theta | D) \] 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 \textbf{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

\[
P(x_{new} | D) = P(x_{new} | \theta_{ML}) \text{ or } P(x_{new} | D) = P(x_{new} | \theta_{MAP})
\]
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)

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

<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>
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 \( X_i \):

\[ P_{MLE}(X_i = x | pa_{X_i} = u) = \frac{\#(X_i = x, Pa_{X_i} = u)}{\#(Pa_{X_i} = u)} \]

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
  \[ \theta_{ijk}^{ML} = \frac{n_{ijk}}{\sum_{j'} n_{ij'k}} \]
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
  $$P(X_i = k | Pa_{X_i} = j) = \theta_{kj}$$
  $$P(\theta_a) = \prod_{kj} P(\theta_{kj})$$
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

\[
P(x_{1:T} | \theta) = P(x_1 | \pi) \prod_{t=2}^{T} P(x_t | x_{t-1}, A)
\]

The log-likelihood of \( N \) sequences

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

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 a multinomial distribution
- MLE of $A_{ij}$ is the fraction of transitions from $i$ to $j$
  
  $$A_{ij}^{ML} = \frac{\#(i \rightarrow j)}{\#(i \rightarrow \cdot)}$$

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

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

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

\[ = \frac{1}{Z(\theta)} \prod_{ij} \exp(\theta_{ij} X_i X_j) \prod_i \exp(\theta_i X_i) \]

\[ Z(\theta) = \sum_x \prod_{ij} \exp(\theta_{ij} X_i X_j) \prod_i \exp(\theta_i X_i) \]

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

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

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

*can be other feature function* \( f(x_i) \)

*Term \( \log Z(\theta) \) does not decompose!*
Derivatives of log likelihood

\[ l(\theta, D) = \frac{1}{N} \sum_1^N \left( \sum_{ij} \theta_{ij} x_i^l x_j^l + \sum_i \theta_i x_i^l - \log Z(\theta) \right) \]

\[ \frac{\partial l(\theta, D)}{\partial \theta_{ij}} = \frac{1}{N} \sum_1^N \sum_{ij} x_i^l x_j^l - \frac{\partial \log Z(\theta)}{\partial \theta_{ij}} \]

\[ = \frac{1}{N} \sum_1^N x_i^l x_j^l - \frac{1}{Z(\theta)} \sum_{i} \Pi_{ij} \exp(\theta_{ij} X_i X_j) \Pi_i \exp(\theta_i X_i) \quad X_i X_j \]

A convex problem
Can find global optimum

need to do inference
Moment matching condition

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

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

Moment matching: \[ \frac{\partial l(\theta, D)}{\partial \theta_{ij}} = E_{\hat{P}(X_i, X_j)}[X_i X_j] - E_{P(X|\theta)}[X_i X_j] \]
Optimize MLE for undirected models

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

- Can be solved 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 \( \theta_{ij} \leftarrow \theta_{ij} - \eta \frac{\partial l(\theta, D)}{\partial \theta_{ij}} \)

- Or use the gradient equation for fixed point iteration: iterative proportional fitting
Exponential Family

- Random variable $X$, $P(X|\theta) = \frac{1}{Z(\theta)} h(X) \exp(\theta^\top T(X))$
  - $Z(\theta) = \int h(X) \exp (\theta^\top T(X)) dX$

- $P(X|\theta) = h(X) \exp(\theta^\top T(X) - A(\theta))$

- **Base measure**
- **Canonical parameter** $\theta$
- **Sufficient statistics**
- **Log-partition function** $A(\theta) = \log Z(\theta)$

- Example: Bernoulli, multinomial, Gaussian, Poisson, Gamma, ...
Multivariate Gaussian

\[ P(X|\theta) = h(X)\exp(\theta^\top T(X) - A(\theta)) \]

Random variable \( X \in R^k \)

\[ P(X|\mu, \Sigma) = \frac{1}{k} \exp \left( -\frac{1}{2} (X - \mu)^\top \Sigma^{-1} (X - \mu) \right) \]
\[ = \frac{1}{k} \exp \left( -\frac{1}{2} \text{tr}(\Sigma^{-1}XX^\top) + \mu^\top \Sigma^{-1}X - \frac{1}{2} \mu^\top \Sigma^{-1} \mu - \frac{1}{2} \log |\Sigma| \right) \]

Exponential family representation

\[ \theta = (\Sigma^{-1}\mu; -\frac{1}{2} \text{vec} (\Sigma^{-1})) \]
\[ T(X) = (X; \text{vec}(XX^\top)) \]
\[ A(\theta) = \frac{1}{2} \mu^\top \Sigma^{-1} \mu + \frac{1}{2} \log |\Sigma| \]
\[ h(X) = (2\pi)^{\frac{k}{2}} \]
Multinomial distribution

\[ P(X|\theta) = h(X) \exp(\theta^\top T(X) - A(\theta)) \]

- Multinomial distribution with \( k \) values
  - Binary vector \( X \in \{0,1\}^K \), \( X \sim \text{multi}(X|\theta) \)
  - \( \sum_k X_k = 1, \sum_k \theta_k = 1 \)

\[ P(X|\theta) = \theta_1^{X_1} \theta_2^{X_2} \ldots \theta_K^{X_K} = \exp\left(\sum_{k=1}^K X_k \ln \theta_k\right) \]
  - \( = \exp\left(\sum_{k=1}^{K-1} X_k \ln \theta_k + X_K \ln \theta_K\right) \)
  - \( = \exp\left(\sum_{k=1}^{K-1} X_k \ln \theta_k + (1 - \sum_{k=1}^{K-1} X_k) \log \theta_K\right) \)
  - \( = \exp\left(\sum_{k=1}^{K-1} X_k \log \left(\frac{\theta_k}{\theta_K}\right) + \log \theta_K\right) \)

\[ \theta = \left(\log \frac{\theta_k}{\theta_K}; 0\right), T(X) = X, A(\theta) = -\log(\theta_K), h(X) = 1 \]
Why exponential family?

- Moment generating property: we can easily compute moments of any exponential family distribution by taking the derivatives of the log normalizer

- Mean:
  \[
  \frac{dA}{d\theta} = \frac{d}{d\theta} \log Z(\theta) = \frac{1}{Z(\theta)} \frac{d}{d\theta} Z(\theta)
  \]
  \[
  = \frac{1}{Z(\theta)} \frac{d}{d\theta} \int h(X) \exp(\theta^T T(X)) \, dX
  \]
  \[
  = \int T(X) \frac{h(X) \exp(\theta^T T(X))}{Z(\theta)} \, dX
  \]
  \[
  = E_{P(X|\theta)}[T(X)]
  \]

- Variance:
  \[
  \frac{d^2 A}{d\theta^2} = E_{P(X|\theta)}[T^2(X)] - E_{P(X|\theta)}^2[T(X)] = Var[T(X)]
  \]
MLE for exponential family

- For iid data, the log-likelihood is
  \[
  l(\theta, D) = \log(\prod_{i=1}^{N} h(x_i)\exp(\theta^T T(x_i) - A(\theta))) \\
  = \sum_l \log h(x_l) + \theta^T \sum_l T(x_l) - NA(\theta)
  \]

- Take derivatives and set of zero:
  \[
  \frac{\partial l(\theta,D)}{\partial \theta} = \sum_l T(x_l) - N \frac{\partial A(\theta)}{\partial \theta} = 0 \\
  \frac{1}{N} \sum_l T(x_l) = \frac{\partial A(\theta)}{\partial \theta} = E_{P(X|\theta)}[T(X)]
  \]

- This is moment matching condition for exponential family