Graphical Models (III)

Learning

Eric Xing
Carnegie Mellon University
June 4, 2007

Learning Graphical Models

The goal:

Given set of independent samples \textit{(assignments} of random variables), find the \textbf{best} (the most likely?) graphical model (both the graph and the CPDs)

\begin{align*}
(B,E,A,C,R) &= (T,F,T,T,F) \\
(B,E,A,C,R) &= (T,F,F,T,F) \\
\ldots &
\end{align*}

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>$P(A \mid E,B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>b</td>
<td>0.9 0.1</td>
</tr>
<tr>
<td>e</td>
<td>b</td>
<td>0.2 0.8</td>
</tr>
<tr>
<td>e</td>
<td>b</td>
<td>0.9 0.1</td>
</tr>
<tr>
<td>e</td>
<td>b</td>
<td>0.01 0.99</td>
</tr>
</tbody>
</table>
Learning Graphical Models

- Scenarios:
  - completely observed GMs
    - directed √
    - undirected √
  - partially observed GMs
    - directed √
    - undirected (an open research topic)

- Estimation principles:
  - Maximal likelihood estimation (MLE) √
  - Bayesian estimation
  - Maximal conditional likelihood
  - Maximal "Margin"

- We use learning as a name for the process of estimating the parameters, and in some cases, the topology of the network, from data.

Score-based approach

Data

\( (x_1^{(1)}, \ldots, x_n^{(1)}) \)
\( (x_1^{(2)}, \ldots, x_n^{(2)}) \)
\[ \vdots \]
\( (x_1^{(M)}, \ldots, x_n^{(M)}) \)

Possible structures

Learn parameters

Score struc/param

Maximum likelihood 10^{-5}
Bayesian 10^{-3}
Conditional likelihood 10^{-15}
Margin ...

Eric Xing
ML Parameter Est. for completely observed GMs of given structure

- The data:
  \[\{(z^{(1)}, x^{(1)}), (z^{(2)}, x^{(2)}), (z^{(3)}, x^{(3)}), \ldots (z^{(N)}, x^{(N)})\}\]

The basic idea underlying MLE

- Likelihood (for now let’s assume that the structure is given):
  \[L(\theta \mid X) = p(X \mid \theta) = p(X_1 \mid \theta_1)p(X_2 \mid \theta_2)p(X_3 \mid X_1, X_2, \theta_3)\]

- Log-Likelihood:
  \[l(\theta \mid X) = \log p(X \mid \theta) = \log p(X_1 \mid \theta_1) + \log p(X_2 \mid \theta_2) + \log p(X_3 \mid X_1, X_2, \theta_3)\]

- Data log-likelihood
  \[l(\theta \mid DATA) = \log \prod_{x} p(X_x \mid \theta) = \sum_{x} \log p(X_{x,1} \mid \theta_1) + \sum_{x} \log p(X_{x,2} \mid \theta_2) + \sum_{x} \log p(X_{x,3} \mid X_{x,1}, X_{x,2}, \theta_3)\]

- MLE
  \[\{\theta_1, \theta_2, \theta_3\}_{MLE} = \arg \max l(\theta \mid DATA)\]

  \[\theta'_1 = \arg \max \sum_{x} \log p(X_{x,1} \mid \theta_1), \quad \theta'_2 = \arg \max \sum_{x} \log p(X_{x,2} \mid \theta_2), \quad \theta'_3 = \arg \max \sum_{x} \log p(X_{x,3} \mid X_{x,1}, X_{x,2}, \theta_3)\]
Example 1: conditional Gaussian

- The completely observed model:
  - \( Z \) is a class indicator vector
    \[
    Z = \begin{bmatrix}
    Z_1^z \\
    Z_2^z \\
    \vdots \\
    Z_M^z
    \end{bmatrix}, \quad \text{where } Z^z = \{0,1\}, \text{ and } \sum Z^z = 1
    \]
    and a datum is in class \( i \) w.p. \( \pi_i 
    \]
    \[
    p(z^i = 1 | \pi) = \pi_i = \pi_1^z \times \pi_2^z \times \ldots \times \pi_M^z
    \]
    \[
    p(z) = \prod \pi_m^z
    \]
  - \( X \) is a conditional Gaussian variable with a class-specific mean
    \[
    p(x | z^m = 1, \mu, \sigma) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{ \frac{1}{2\sigma^2} (x - \mu_m)^2 \right\}
    \]
    \[
    p(x | z, \mu, \sigma) = \prod_{m} N(x | \mu_m, \sigma)^{z_m}
    \]

- Data log-likelihood
  \[
  l(\theta | D) = \log \prod_{z,x} p(z, x) = \log \prod_{z} p(z | \pi) p(x | z, \mu, \sigma)
  \]
  \[
  = \sum_{z} \log p(z | \pi) + \sum_{z} \log p(x | z, \mu, \sigma)
  \]
  \[
  = \sum_{z} \log \prod_{m} \pi_m^{z_m} + \sum_{z} \log \prod_{m} N(x | \mu_m, \sigma)^{z_m}
  \]
  \[
  = \sum_{z} \sum_{m} z_m \log \pi_m - \sum_{z} \sum_{m} \frac{1}{2\sigma^2} (x - \mu_m)^2 + C
  \]
  - MLE
    \[
    \pi_m^* = \arg \max_{\pi} l(\theta | D), \quad \Rightarrow \frac{\partial}{\partial \pi_m} l(\theta | D) = 0, \forall m, \quad \text{s.t. } \sum \pi_m = 1
    \]
    \[
    \Rightarrow \pi_m^* = \frac{\sum z_m x}{N} = \frac{n_m}{N}
    \]
    \( \pi_m^* \) the fraction of samples of class \( m \)
    \[
    \mu_m^* = \arg \max_{\mu} l(\theta | D), \quad \Rightarrow \frac{\partial}{\partial \mu_m} l(\theta | D) = 0, \forall m
    \]
    \[
    \mu_m^* = \frac{\sum z_m x}{\sum z_m} = \frac{\sum z_m x}{n_m}
    \]
    \( \mu_m^* \) the average of samples of class \( m \)
Example 2: HMM: two scenarios

- **Supervised learning**: estimation when the “right answer” is known
  - **Examples**:
    - GIVEN: a genomic region $x = x_1,...,x_{1,000,000}$ where we have good (experimental) annotations of the CpG islands
    - GIVEN: the casino player allows us to observe him one evening, as he changes dice and produces 10,000 rolls

- **Unsupervised learning**: estimation when the “right answer” is unknown
  - **Examples**:
    - GIVEN: the porcupine genome; we don’t know how frequent are the CpG islands there, neither do we know their composition
    - GIVEN: 10,000 rolls of the casino player, but we don’t see when he changes dice

- **QUESTION**: Update the parameters $\theta$ of the model to maximize $P(x | \theta)$ - Maximal likelihood (ML) estimation

Recall definition of HMM

- Transition probabilities between any two states
  - $p(y_i^t = 1 | y_{i-1}^t = 1) = a_{ij}$,
  - or $p(y_i | y_{i-1} = 1) \sim \text{Multinomial}(a_1, a_2, ..., a_M)$ $\forall i \in I$.

- Start probabilities
  - $p(y_1) \sim \text{Multinomial}(\pi_1, \pi_2, ..., \pi_M)$

- Emission probabilities associated with each state
  - $p(x_i | y_i^t = 1) \sim \text{Multinomial}(b_1, b_2, ..., b_N)$ $\forall i \in I$.
  - or in general: $p(x_i | y_i^t = 1) \sim f(x_i | \theta)$ $\forall i \in I$. 
Supervised ML estimation

- Given $x = x_1 \ldots x_N$ for which the true state path $y = y_1 \ldots y_N$ is known,
- Define:
  - $A_{ij} = \# \text{ times state transition } i \rightarrow j \text{ occurs in } y$
  - $B_{ik} = \# \text{ times state } i \text{ in } y \text{ emits } k \text{ in } x$
- We can show that the maximum likelihood parameters $\theta$ are:
  - $A_{ij}^{ML} = \frac{\#(i \rightarrow j)}{\#(i \rightarrow \bullet)} = \frac{\sum_{t=1}^{T} y_{t+1}^i y_{t+2}^j}{\sum_{t=1}^{T} y_{t+1}^i}$
  - $B_{ik}^{ML} = \frac{\#(i \rightarrow k)}{\#(i \rightarrow \bullet)} = \frac{\sum_{t=1}^{T} y_{t+1}^i x_{t+1}^k}{\sum_{t=1}^{T} y_{t+1}^i}$

  - What if $x$ is continuous? We can treat $\{x_{t+1}, y_{t+1}; t = 1: T, n = 1: N\}$ as $\mathbb{R}^N \times T$ observations of, e.g., a Gaussian, and apply learning rules for Gaussian ...

Supervised ML estimation, ctd.

- **Intuition:**
  - When we know the underlying states, the best estimate of $\theta$ is the average frequency of transitions & emissions that occur in the training data.

- **Drawback:**
  - Given little data, there may be overfitting:
    - $P(x|\theta)$ is maximized, but $\theta$ is unreasonable
    - 0 probabilities – VERY BAD

- **Example:**
  - Given 10 casino rolls, we observe
    - $x = 2, 1, 5, 6, 1, 2, 3, 6, 2, 3$
  - Then:
    - $a_{FF} = 1; \quad a_{FL} = 0$
    - $b_{F1} = b_{F2} = 0; b_{F3} = 0.2; b_{F4} = 0; b_{F5} = b_{F6} = 0.1$
Pseudocounts

- Solution for small training sets:
  - Add pseudocounts
    \[ A_{ij} = \# \text{ times state transition } i \rightarrow j \text{ occurs in } y + R_{ij} \]
    \[ B_{ik} = \# \text{ times state } i \text{ in } y \text{ emits } k \text{ in } x + S_{ik} \]
  - \( R_{ij}, S_{ij} \) are pseudocounts representing our prior belief
  - Total pseudocounts: \( R_i = \sum_j R_{ij}, S_i = \sum_k S_{ik} \),
    - "strength" of prior belief,
    - total number of imaginary instances in the prior

- Larger total pseudocounts \( \Rightarrow \) strong prior belief
- Small total pseudocounts: just to avoid 0 probabilities --- smoothing
- This is equivalent to Bayesian est. under a uniform prior with "parameter strength" equals to the pseudocounts

MLE for general BNs

- If we assume the parameters for each CPD are globally independent, and all nodes are fully observed, then the log-likelihood function decomposes into a sum of local terms, one per node:

\[
L(\theta, D) = \log p(D | \theta) = \log \prod \left( \prod p(x_n | x_{-n}, \theta) \right) = \sum \left( \sum p(x_n | x_{-n}, \theta) \right)
\]
Example: A directed model

- Consider the distribution defined by the directed acyclic GM:

\[ p(x \mid \theta) = p(x_1 \mid \theta_1) p(x_2 \mid x_1, \theta_1) p(x_3 \mid x_2, \theta_2) p(x_4 \mid x_3, x_4, \theta_4) \]

- This is exactly like learning four separate small BNs, each of which consists of a node and its parents.

\[ \begin{align*}
X_1 & \quad \rightarrow \quad X_2 \\
X_2 & \quad \rightarrow \quad X_3 \\
X_3 & \quad \rightarrow \quad X_4 \\
X_4 & \quad \rightarrow \quad X_1
\end{align*} \]

MLE for BNs with tabular CPDs

- Assume each CPD is represented as a table (multinomial) where

\[ \theta_{ijk} \overset{\text{def}}{=} p(X_i = j \mid X_{-i} = k) \]

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

- The sufficient statistics are counts of family configurations

\[ n_{ijk} \overset{\text{def}}{=} \sum x_i^j x_{-i}^k \]

- The log-likelihood is

\[ L(\theta; D) = \log \prod_{i,j,k} \theta_{ijk}^{n_{ijk}} = \sum_{i,j,k} n_{ijk} \log \theta_{ijk} \]

- Using a Lagrange multiplier to enforce \( \sum_j \theta_{ijk} = 1 \), we get:

\[ \theta_{ijk}^{ML} = \frac{n_{ijk}}{\sum_{i,j,k} n_{ijk}} \]
ML Structural Learning for completely observed GMs

Data

\[(s_0^m, ..., s_n^m)\]
\[(q_0^m, ..., q_n^m)\]

Information Theoretic Interpretation of ML

\[\ell(\theta_G, D; G) = \log p(D | \theta_G, G)\]
\[= \log \prod_s \left( \prod_x p(x, \theta_{x(G)}, \theta_{e(x(G))}) \right)\]
\[= \sum_s \left( \sum_x \log p(x, \theta_{x(G)}, \theta_{e(x(G))}) \right)\]
\[= M \sum_i \left( \sum_{x, x_{x(G)}} \frac{\text{count}(x_i, x_{x(G)})}{M} \log p(x_i | x_{x(G)}, \theta_{e(x(G))}) \right)\]
\[= M \sum_i \left( \sum_{x, x_{x(G)}} p(x_i | x_{x(G)}, \theta_{e(x(G))}) \log p(x_i | x_{x(G)}, \theta_{e(x(G))}) \right)\]

From sum over data points to sum over count of variable states
Information Theoretic Interpretation of ML (con'd)

\[ \ell(\theta, G; D) = \log \hat{p}(D | \theta, G) \]
\[ = M \sum_i \left( \sum_{x_i, x_{\neg i}} \hat{p}(x_i, x_{\neg i} | G) \log \frac{\hat{p}(x_i, x_{\neg i} | G, \theta_G)}{\hat{p}(x_i, x_{\neg i})} \right) \]
\[ = M \sum_i \left( \sum_{x_i, x_{\neg i}} \hat{p}(x_i, x_{\neg i} | G) \log \frac{\hat{p}(x_i, x_{\neg i} | G, \theta_G) \hat{p}(x_i)}{\hat{p}(x_i, x_{\neg i}) \hat{p}(x_i)} \right) \]
\[ = M \sum_i \left( \sum_{x_i, x_{\neg i}} \hat{p}(x_i, x_{\neg i} | G) \log \frac{\hat{p}(x_i, x_{\neg i} | G, \theta_G)}{\hat{p}(x_i, x_{\neg i})} \right) - M \sum_i \hat{H}(x_i) \]

Decomposable score and a function of the graph structure

Structural Search

- How many graphs over \( n \) nodes? \( O(2^n) \)
- How many trees over \( n \) nodes? \( O(n!) \)
- But it turns out that we can find exact solution of an optimal tree (under MLE)!
  - Trick: in a tree each node has only one parent!
  - Chow-liu algorithm
Chow-Liu tree learning algorithm

- Objective function:
  \[ \ell(\theta, G; D) = \log \hat{p}(D | \theta, G) = M \sum_i \hat{I}(x_i, x_{x_i(G_i)}) - M \sum_i \hat{H}(x_i) \]
  \[ C(G) = M \sum_i \hat{I}(x_i, x_{x_i(G_i)}) \]

- Chow-Liu:
  - For each pair of variable \( x_i \) and \( x_j \)
    - Compute empirical distribution:
      \[ \hat{p}(x_i, x_j) = \frac{\text{count}(x_i, x_j)}{M} \]
    - Compute mutual information:
      \[ \hat{I}(x_i, x_j) = \sum_{x_i, x_j} \hat{p}(x_i, x_j) \log \frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i) \hat{p}(x_j)} \]
  - Define a graph with node \( x_1, \ldots, x_n \)
    - Edge \((i,j)\) gets weight \( \hat{I}(x_i, x_j) \)

Chow-Liu algorithm (con’d)

- Objective function:
  \[ \ell(\theta, G; D) = \log \hat{p}(D | \theta, G) = M \sum_i \hat{I}(x_i, x_{x_i(G_i)}) - M \sum_i \hat{H}(x_i) \]
  \[ C(G) = M \sum_i \hat{I}(x_i, x_{x_i(G_i)}) \]

- Chow-Liu:
  - Optimal tree BN
    - Compute maximum weight spanning tree
    - Direction in BN: pick any node as root, do breadth-first-search to define directions
    - I-equivalence:
      \[ C(G) = I(A, B) + I(A, C) + I(C, D) + I(C, E) \]
Structure Learning for general graphs

- **Theorem:** The problem of learning a BN structure with at most $d$ parents is NP-hard for any (fixed) $d \geq 2$

- **Most structure learning approaches use heuristics**
  - Exploit score decomposition
  - Two heuristics that exploit decomposition in different ways
    - Greedy search through space of node-orders
    - Local search of graph structures

Order search versus graph search

- **Order search advantages**
  - For fixed order, optimal BN –more "global"optimization
  - Space of orders much smaller than space of graphs

- **Graph search advantages**
  - Not restricted to $k$ parents
  - Especially if exploiting CPD structure, such as CSI
  - Cheaper per iteration
  - Finer moves within a graph
Bayesian model averaging

- Probabilistic statements of $\Theta$ are conditioned on the values of the observed variables $A_{\text{obs}}$ and prior $p(\chi)$

$$p(\Theta; \chi)$$

$$\Theta_{\text{Bayes}} = \int_{\Theta} p(\Theta | A, \chi) d\Theta$$

$p(\Theta | A; \chi) \propto p(A | \Theta) p(\Theta; \chi)$

posterior likelihood prior

Learning partially observed GMs

- The data:

$$\{(x^{(1)}), (x^{(2)}), (x^{(2)}) \ldots (x^{(N)})\}$$
Gaussian Mixture Models (GMMs)

- Consider a mixture of $K$ Gaussian components:

$$p(x_n | \mu, \Sigma) = \sum_k \pi_k N(x_n | \mu_k, \Sigma_k)$$

- This model can be used for unsupervised clustering.
  - This model (fit by AutoClass) has been used to discover new kinds of stars in astronomical data, etc.

---

Gaussian Mixture Models (GMMs)

- Consider a mixture of $K$ Gaussian components:
  - $Z$ is a latent class indicator vector:
    $$p(z_n) = \text{multi}(z_n : \pi) = \prod_k (\pi_k)^{z_n}$$
  - $X$ is a conditional Gaussian variable with a class-specific mean/covariance
    $$p(x_n | z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma_k|^{1/2}} \exp\left\{ \frac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right\}$$

- The likelihood of a sample:
  $$p(x_n | \mu, \Sigma) = \sum_k p(z_n^k = 1 | \pi) p(x_n | z_n^k = 1, \mu, \Sigma) = \sum_k \pi_k \prod_k (\pi_k)^{z_n^k} N(x_n | \mu_k, \Sigma_k)^{z_n^k}$$
Why is Learning Harder?

- In fully observed iid settings, the log likelihood decomposes into a sum of local terms (at least for directed models).
  \[ \ell(\theta; D) = \log p(x, z | \theta) = \log p(z | \theta_z) + \log p(x | z, \theta_x) \]

- With latent variables, all the parameters become coupled together via marginalization.
  \[ \ell_c(\theta; D) = \log \sum_z p(x, z | \theta) = \log \sum_z p(z | \theta_z) p(x | z, \theta_x) \]

Toward the EM algorithm

- Recall MLE for completely observed data.

- Data log-likelihood
  \[ \ell(\theta; D) = \log \prod_n p(z_n, x_n) = \log \prod_n p(z_n | \pi) p(x_n | z_n, \mu, \sigma) \]
  \[ = \sum_n \log \prod_k \pi_k^{z_n^k} + \sum_n \log \prod_k N(x_n; \mu_k, \sigma_k) \]
  \[ = \sum_n \sum_k z_n^k \log \pi_k \cdot \sum_n \sum_k \frac{1}{2\sigma_k} (x_n - \mu_k)^2 + C \]

- MLE
  \[ \hat{\theta}_{k,MLE} = \arg \max_{\theta_k} \ell(\theta; D) \]
  \[ \hat{\mu}_{k,MLE} = \arg \max_{\mu_k} \ell(\theta; D) \]
  \[ \Rightarrow \hat{\mu}_{k,MLE} = \sum_n z_n^k x_n / \sum_n z_n^k \]

- What if we do not know \( z_n \)?
Expectation-Maximization

- **Start:**
  - "Guess" the centroid $\mu_k$ and covariance $\Sigma_k$ of each of the K clusters
- **Loop**

Example: Gaussian mixture model

- A mixture of K Gaussians:
  - $Z$ is a latent class indicator vector
    $$ p(z_n) = \text{multi}(z_n; \pi) = \prod_k (\pi_k)^{z_{nk}} $$
  - $X$ is a conditional Gaussian variable with class-specific mean/covariance
    $$ p(x_n | z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^d \Sigma_k} \exp \left\{ -\frac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right\} $$
  - The likelihood of a sample:
    $$ p(x_n, z_n; \mu, \Sigma) = \sum_{z_n} p(x_n | z_n) p(z_n; \pi) $$
    $$ = \sum_k \prod_{n} \left( (\pi_k)^{z_{nk}} \frac{1}{(2\pi)^d \Sigma_k} \exp \left\{ -\frac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right\} \right) $$
  - The expected complete log likelihood
    $$ \langle \xi(\theta; x, z) \rangle = \sum_n \langle \log p(z_n; \pi) \rangle_{p(z_n | x)} + \sum_n \langle \log p(x_n | z_n, \mu, \Sigma) \rangle_{p(x_n | z_n)} $$
    $$ = \sum_n \sum_k (z_{nk}^k \log \pi_k) - \frac{1}{2} \sum_n \sum_k (z_{nk}^k) (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) + \log |\Sigma_k| + C $$
E-step

- We maximize \( \mathcal{L}(\theta) \) iteratively using the following iterative procedure:
  - **Expectation step**: computing the expected value of the sufficient statistics of the hidden variables (i.e., \( z \)) given current est. of the parameters (i.e., \( \pi \) and \( \mu \)).

\[
\tau_{nk}^{(t)} = \{ z_n^k \} = p(z_n^k = 1 | x, \mu^{(t)}, \Sigma^{(t)}) = \frac{\pi_k^{(t)} N(x_n, \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_{l} \pi_l^{(t)} N(x_n, \mu_l^{(t)}, \Sigma_l^{(t)})}
\]

- Here we are essentially doing inference

M-step

- We maximize \( \mathcal{L}(\theta) \) iteratively using the following iterative procedure:
  - **Maximization step**: compute the parameters under current results of the expected value of the hidden variables

\[
\pi'_k = \arg \max \mathcal{L}(\theta), \quad \Rightarrow \frac{\partial}{\partial \pi_k} \mathcal{L}(\theta) = 0, \forall k, \text{ s.t. } \sum_k \pi'_k = 1
\]

\[
\pi'_k = \frac{\sum \{ z_n^k \} / N} {\sum \{ z_n^k \} / N} = \frac{\{ n_k \} / N} {\{ n_k \} / N}
\]

\[
\mu'_k = \arg \max \mathcal{L}(\theta), \quad \Rightarrow \frac{\partial}{\partial \mu_k} \mathcal{L}(\theta) = \frac{\sum z_{nk}^{(t)} x_n}{\sum z_{nk}^{(t)}}
\]

\[
\Sigma'_k = \arg \max \mathcal{L}(\theta), \quad \Rightarrow \frac{\partial}{\partial \Sigma_k} \mathcal{L}(\theta) = \frac{\sum z_{nk}^{(t)} (x_n - \mu_k^{(t)}) (x_n - \mu_k^{(t)})^T}{\sum z_{nk}^{(t)}}
\]

- This is isomorphic to MLE except that the variables that are hidden are replaced by their expectations (in general they will be replaced by their corresponding "sufficient statistics")
Theory underlying EM

- What are we doing?

- Recall that according to MLE, we intend to learn the model parameter that would have maximize the likelihood of the data.

- But we do not observe $z$, so computing
  \[ \ell_c(\theta; D) = \log \sum_z p(x, z \mid \theta) = \log \sum_z p(z \mid \theta_z) p(x \mid z, \theta_z) \]
  is difficult!

- What shall we do?

Complete & Incomplete Log Likelihoods

- Complete log likelihood
  Let $X$ denote the observable variable(s), and $Z$ denote the latent variable(s).
  If $Z$ could be observed, then
  \[ \ell_c(\theta; x, z) = \log p(x, z \mid \theta) \]
  - Usually, optimizing $\ell_c()$ given both $z$ and $x$ is straightforward (c.f. MLE for fully observed models).
  - Recalled that in this case the objective for, e.g., MLE, decomposes into a sum of factors, the parameter for each factor can be estimated separately.
  - But given that $Z$ is not observed, $\ell_c()$ is a random quantity, cannot be maximized directly.

- Incomplete log likelihood
  With $z$ unobserved, our objective becomes the log of a marginal probability:
  \[ \ell_c(\theta; x) = \log p(x \mid \theta) = \log \sum_z p(x, z \mid \theta) \]
  - This objective won’t decouple
Expected Complete Log Likelihood

- For any distribution \( q(z) \), define expected complete log likelihood:
  \[
  \langle \ell(\theta; x, z) \rangle_q = \sum_z q(z | x, \theta) \log p(x, z | \theta)
  \]
  - A deterministic function of \( \theta \)
  - Linear in \( \ell() \) --- inherit its factorizability
  - Does maximizing this surrogate yield a maximizer of the likelihood?

- Jensen’s inequality
  \[
  \ell(\theta; x) = \log p(x | \theta) \\
  = \log \sum_z p(x, z | \theta) \\
  = \log \sum_z q(z | x) \frac{p(x, z | \theta)}{q(z | x)} \\
  = \sum_z q(z | x) \log \frac{p(x, z | \theta)}{q(z | x)} \\
  \Rightarrow \ell(\theta; x) \geq \langle \ell(\theta; x, z) \rangle_q + H_q
  \]

Lower Bounds and Free Energy

- For fixed data \( x \), define a functional called the free energy:
  \[
  F(q, \theta) = \sum_z q(z | x) \log \frac{p(x, z | \theta)}{q(z | x)} \leq \ell(\theta; x)
  \]
  - The EM algorithm is coordinate-ascent on \( F \):
    - E-step: \( q^{t+1} = \arg \max_q F(q, \theta^t) \)
    - M-step: \( \theta^{t+1} = \arg \max_\theta F(q^{t+1}, \theta) \)
E-step: maximization of expected \( \ell_c \) w.r.t. \( q \)

- **Claim:**
  
  \[ q^{t+1} = \arg \max_q F(q, \theta^t) = p(z | x, \theta^t) \]
  
  - This is the posterior distribution over the latent variables given the data and the parameters. Often we need this at test time anyway (e.g. to perform classification).

- **Proof (easy):** this setting attains the bound \( \mathcal{L}(\theta, x) \geq R(q, \theta) \)
  
  \[
  F(p(z | x, \theta^t), \theta^t) = \sum_z p(z | x, \theta^t) \log \frac{p(x, z | \theta^t)}{p(z | x, \theta^t)} \\
  = \sum_z q(z | x) \log p(x | \theta^t) \\
  = \log p(x | \theta^t) = \mathcal{L}(\theta^t; x)
  \]

- Can also show this result using variational calculus or the fact that \( \mathcal{L}(\theta, x) - F(q, \theta) = \text{KL}(q || p(z | x, \theta)) \)

**E-step \equiv plug in posterior expectation of latent variables**

- Without loss of generality: assume that \( p(x, z | \theta) \) is a generalized exponential family distribution:
  
  \[
  p(x, z | \theta) = \frac{1}{Z(\theta)} h(x, z) \exp \left\{ \sum_i \theta_i f_i(x, z) \right\}
  \]

- Special cases: if \( p(x | z) \) are GLIMs, then \( f_i(x, z) = \eta_i^T(z) \xi_i(x) \)

- The expected complete log likelihood under \( q^{t+1} = p(z | x, \theta^t) \) is
  
  \[
  \left\langle \ell_c(\theta^t; x, z) \right\rangle_{q^{t+1}} = \sum_z q(z | x, \theta^t) \log p(x, z | \theta^t) - A(\theta) \\
  = \sum_i \theta^t_i \left\langle f_i(x, z) \right\rangle_{q(z|x, \theta^t)} - A(\theta) \\
  = \sum_i \theta^t_i \left\langle \eta_i(z) \right\rangle_{q(z|x, \theta^t)} \xi_i(x) - A(\theta)
  \]
M-step: maximization of expected $\ell_c$ w.r.t. $\theta$

- Note that the free energy breaks into two terms:

$$F(q, \theta) = \sum_z q(z | x) \log \frac{p(x, z | \theta)}{q(z | x)}$$

$$= \sum_z q(z | x) \log p(x, z | \theta) - \sum_z q(z | x) \log q(z | x)$$

$$= \langle \ell_c(\theta; x, z) \rangle_q + H_q$$

- The first term is the expected complete log likelihood (energy) and the second term, which does not depend on $\theta$, is the entropy.

- Thus, in the M-step, maximizing with respect to $\theta$ for fixed $q$ we only need to consider the first term:

$$\theta^{t+1} = \arg \max_\theta \langle \ell_c(\theta; x, z) \rangle_{q^{t+1}} = \arg \max_\theta \sum_z q(z | x) \log p(x, z | \theta)$$

- Under optimal $q^{t+1}$, this is equivalent to solving a standard MLE of fully observed model $p(x, z | \theta)$, with the sufficient statistics involving $z$ replaced by their expectations w.r.t. $p(z | x, \theta)$.

Example: HMM

- **Supervised learning**: estimation when the “right answer” is known
  - **Examples**:
    - GIVEN: a genomic region $x = x_1 \ldots x_{1,000,000}$ where we have good (experimental) annotations of the CpG islands
    - GIVEN: the casino player allows us to observe him one evening, as he changes dice and produces 10,000 rolls

- **Unsupervised learning**: estimation when the “right answer” is unknown
  - **Examples**:
    - GIVEN: the porcupine genome; we don’t know how frequent are the CpG islands there, neither do we know their composition
    - GIVEN: 10,000 rolls of the casino player, but we don’t see when he changes dice

- **QUESTION**: Update the parameters $\theta$ of the model to maximize $p(x | \theta)$ - -- Maximal likelihood (ML) estimation
The Baum Welch algorithm

- The complete log likelihood
  \[ \zeta(\theta; x, y) = \log p(x, y) = \log \prod_{t} p(y_{nt} | y_{nt-1}) \prod_{t} p(x_{nt} | x_{nt-1}) \]

- The expected complete log likelihood
  \[ \langle \zeta(\theta; x, y) \rangle = \sum_{\pi} \left( \sum_{t} \log \pi_{nt} \right) + \sum_{t} \sum_{t'=t+1} \left( \log a_{nt'} \left(y_{nt'}, y_{nt} \right) + \sum_{t} \sum_{t'=t+1} \left( \log b_{nt'} \left(y_{nt'} \right) \right) \right) \]

- EM
  - The E step
    \[ \gamma_{nt} = \left( y_{nt} \right) = p(y_{nt} = 1 | x_n) \]
    \[ \xi_{nt} = \left( y_{nt-1} y_{nt} \right) = p(y_{nt-1} = 1, y_{nt} = 1 | x_n) \]
  - The M step ("symbolically" identical to MLE)
    \[ \pi_{nt}^{ML} = \frac{\gamma_{nt}^{ML}}{\sum_{n=1}^{N} \gamma_{nt}^{ML}} \]
    \[ a_{nt'}^{ML} = \frac{\sum_{n=1}^{N} \gamma_{nt'}^{ML} y_{nt}^{ML}}{\sum_{n=1}^{N} \sum_{n'=1}^{N} \gamma_{nt'}^{ML} y_{nt'}^{ML}} \]
    \[ b_{nt'}^{ML} = \frac{\sum_{n=1}^{N} \sum_{n'=1}^{N} \gamma_{nt'}^{ML} x_{nt'}^{ML}}{\sum_{n=1}^{N} \sum_{n'=1}^{N} \gamma_{nt'}^{ML} x_{nt'}^{ML}} \]

Unsupervised ML estimation

- Given \( x = x_1...x_N \) for which the true state path \( y = y_1...y_N \) is unknown,

  **EXPECTATION MAXIMIZATION**

  0. Starting with our best guess of a model \( M \), parameters \( \theta \)
  1. Estimate \( A_{ij}, B_{ih} \) in the training data
     - How? \( A_{ij} = \sum_{n=1}^{N} \gamma_{nt} \gamma_{nt+1} \gamma_{nt+1} \gamma_{nt} \gamma_{nt} \)
     - \( B_{ih} = \sum_{n=1}^{N} \gamma_{nt} x_{nt} \)
  2. Update \( \theta \) according to \( A_{ij}, B_{ih} \)
     - Now a "supervised learning" problem
  3. Repeat 1 & 2, until convergence

  **This is called the Baum-Welch Algorithm**

  We can get to a provably more (or equally) likely parameter set \( \theta \) each iteration
EM for general BNs

while not converged
   % E-step
   for each node $i$
      $ESS_i = 0$ % reset expected sufficient statistics
   for each data sample $n$
      do inference with $X_{n,H}$
      for each node $i$
         $ESS_i = S_i(x_{n,i}, x_{n,:i}) p(x_{n,i} | x_{n,:-H})$
   % M-step
   for each node $i$
      $\theta_i := \text{MLE}(ESS_i)$

Summary: EM Algorithm

- A way of maximizing likelihood function for latent variable models. Finds MLE of parameters when the original (hard) problem can be broken up into two (easy) pieces:
  1. Estimate some “missing” or “unobserved” data from observed data and current parameters.
  2. Using this “complete” data, find the maximum likelihood parameter estimates.

- Alternate between filling in the latent variables using the best guess (posterior) and updating the parameters based on this guess:
  - E-step: $q^{t+1} = \arg \max_q F(q, \theta^t)$
  - M-step: $\theta^{t+1} = \arg \max_\theta F(q^{t+1}, \theta)$

- In the M-step we optimize a lower bound on the likelihood. In the E-step we close the gap, making bound=likelihood.
Learning completely observed undirected GMs

- The data:

\[ \{(z^{(1)}, x^{(1)}), (z^{(2)}, x^{(2)}), (z^{(3)}, x^{(3)}), \ldots (z^{(N)}, x^{(N)})\} \]

MLE for undirected graphical models

- For **directed** graphical models, the log-likelihood decomposes into a sum of terms, one per family (node plus parents).
- For **undirected** graphical models, the log-likelihood **does not** decompose, because the normalization constant \( Z \) is a function of all the parameters

\[
P(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{c \in C} \psi_c(x_c) \quad Z = \sum_{x_1, \ldots, x_n} \prod_{c \in C} \psi_c(x_c)
\]

- In general, we will need to do inference (i.e., marginalization) to learn parameters for undirected models, even in the fully observed case.
Feature-based Clique Potentials

- So far we have discussed the most general form of an undirected graphical model in which cliques are parameterized by general potential functions $\psi_c(x_c)$.

$$P(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{c \in C} \psi_c(x_c)$$

- But for large cliques these general potentials are exponentially costly for inference and have exponential numbers of parameters that we must learn from limited data.

- One solution: change the graphical model to make cliques smaller. But this changes the dependencies, and may force us to make more independence assumptions than we would like.

- Another solution: keep the same graphical model, but use a less general parameterization of the clique potentials.

- This is the idea behind feature-based models.

Features

- Consider a clique $x_c$ of random variables in a UGM, e.g. three consecutive characters $c_1c_2c_3$ in a string of English text.

- How would we build a model of $p(c_1c_2c_3)$?
  - If we use a single clique function over $c_1c_2c_3$, the full joint clique potential would be huge: $26^3 - 1$ parameters.
  - However, we often know that some particular joint settings of the variables in a clique are quite likely or quite unlikely, e.g. ing, ate, ion, ?ed, qu?, jkx, zzz,...

- A “feature” is a function which is vacuous over all joint settings except a few particular ones on which it is high or low.
  - For example, we might have $f_{\text{ing}}(c_1c_2c_3)$ which is 1 if the string is ‘ing’ and 0 otherwise, and similar features for ‘?ed’, etc.

- We can also define features when the inputs are continuous. Then the idea of a cell on which it is active disappears, but we might still have a compact parameterization of the feature.
Features as Micropotentials

- By exponentiating them, each feature function can be made into a "micropotential". We can multiply these micropotentials together to get a clique potential.

- Example: a clique potential \( \psi(c_1, c_2, c_3) \) could be expressed as:

\[
\psi_c(c_1, c_2, c_3) = e^{\theta_{c_1} f_{c_1}(c_1)} \times e^{\theta_{c_2} f_{c_2}(c_2)} \times \ldots \\
= \exp \left\{ \sum_{k=1}^{K} \theta_k f_k(c_1, c_2, c_3) \right\}
\]

- This is still a potential over 26^3 possible settings, but only uses \( K \) parameters if there are \( K \) features.

- By having one indicator function per combination of \( x_c \), we recover the standard tabular potential.

Combining Features

- Each feature has a weight \( \theta_k \) which represents the numerical strength of the feature and whether it increases or decreases the probability of the clique.

- The marginal over the clique is a generalized exponential family distribution, actually, a GLIM:

\[
p(c_1, c_2, c_3) \propto \exp \left\{ \theta_{c_1} f_{c_1}(c_1, c_2, c_3) + \theta_{c_2} f_{c_2}(c_1, c_2, c_3) + \ldots \right\}
\]

- In general, the features may be overlapping, unconstrained indicators or any function of any subset of the clique variables:

\[
\psi_c(x_c) \overset{\text{def}}{=} \exp \left\{ \sum_{k \in I_c} \theta_k f_k(x_c) \right\}
\]

- How can we combine feature into a probability model?
Feature Based Model

- We can multiply these clique potentials as usual:
  \[
p(x) = \frac{1}{Z(\theta)} \prod_{\mathcal{Z}} \psi_{\mathcal{Z}}(x_{\mathcal{Z}}) = \frac{1}{Z(\theta)} \exp \left\{ \sum_{i \in \mathcal{I}_c} \theta_i f_i(x_{\mathcal{Z}}) \right\}
  \]

- However, in general we can forget about associating features with cliques and just use a simplified form:
  \[
p(x) = \frac{1}{Z(\theta)} \exp \left\{ \sum_{i} \theta_i f_i(x_{\mathcal{Z}}) \right\}
  \]

- This is just our friend the exponential family model, with the features as sufficient statistics.

MLE of Feature Based UGMs

- Scaled likelihood function
  \[
  \tilde{\ell}(\theta; D) = \ell(\theta; D) / N = \frac{1}{N} \sum_{n} \log p(x_n | \theta)
  \]
  \[= \sum_{x} \tilde{p}(x) \log p(x | \theta)
  \]
  \[= \sum_{x} \tilde{p}(x) \sum_{i} \theta_i f_i(x) - \log Z(\theta)
  \]

- Instead of optimizing this objective directly, we attack its lower bound
  
  - The logarithm has a linear upper bound ...
    \[\log Z(\theta) \leq \mu Z(\theta) - \log \mu - 1\]
  - This bound holds for all \(\mu\); in particular, for \(\mu = Z^{-1}(\theta^{*})\)
  - Thus we have
    \[\tilde{\ell}(\theta; D) \geq \sum_{x} \tilde{p}(x) \sum_{i} \theta_i f_i(x) - \frac{Z(\theta)}{Z(\theta^{*})} - \log Z(\theta^{*}) + 1\]
Generalized Iterative Scaling (GIS)

- Lower bound of scaled loglikelihood
  \[ \tilde{\ell}(\theta; D) \geq \sum_x \tilde{p}(x) \sum_i \theta_i f_i(x) - \frac{Z(\theta)}{Z(\theta^*)} \log Z(\theta^*) + 1 \]

- Define \( \Delta \theta_i^{(r)} \) \( \overset{\text{def}}{=} \theta_i^{(r)} - \theta_i^{(r-1)} \)
  \[ \tilde{\ell}(\theta; D) \geq \sum_x \tilde{p}(x) \sum_i \theta_i f_i(x) - \frac{1}{Z(\theta^*)} \sum_i \exp \left[ \sum_l \theta_l f_l(x) \right] \log Z(\theta^*) + 1 \]

- Relax again
  - Assume \( f_i(x) \geq 0 \), \( \sum_i f_i(x) = 1 \)
  - Convexity of exponential:
    \[ \exp \left( \sum_i \pi_i x_i \right) \leq \sum_i \pi_i \exp(x_i) \]

- We have:
  \[ \tilde{\ell}(\theta; D) \geq \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \sum_x p(x | \theta^{(r)}) \sum_i f_i(x) \exp(\Delta \theta_i^{(r)}) - \log Z(\theta^*) + 1 \overset{\text{def}}{=} \Lambda(\theta) \]

GIS

- Lower bound of scaled loglikelihood
  \[ \tilde{\ell}(\theta; D) \geq \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \sum_x p(x | \theta^{(r)}) \sum_i f_i(x) \exp(\Delta \theta_i^{(r)}) - \log Z(\theta^*) + 1 \overset{\text{def}}{=} \Lambda(\theta) \]

- Take derivative:
  \[ \frac{\partial \Lambda}{\partial \theta} = \sum_i \sum_x \tilde{p}(x) f_i(x) - \exp(\Delta \theta_i^{(r)}) \sum_x p(x | \theta^{(r)}) f_i(x) \]

- Set to zero
  \[ \exp(\Delta \theta_i^{(r)}) = \frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p(x | \theta^{(r)}) f_i(x)} Z(\theta^*) \]
  - where \( p^{(r)}(x) \) is the unnormalized version of \( p(x | \theta^{(r)}) \)

- Update
  \[ \theta^{(r+1)} = \theta^{(r)} + \Delta \theta^{(r)} \Rightarrow p^{(r+1)}(x) = p^{(r)}(x) e^{\Delta \theta_i^{(r)} f_i(x)} \]
  \[ p^{(r+1)}(x) = \frac{p^{(r)}(x)}{Z(\theta)} \prod_i \left( \frac{\tilde{p}(x) f_i(x)}{\sum_j \tilde{p}(x) f_j(x)} Z(\theta) \right) \]
  \[ = \frac{p^{(r)}(x)}{Z(\theta^*)} \prod_i \left( \frac{\tilde{p}(x) f_i(x)}{\sum_j \tilde{p}(x) f_j(x)} \right) \]
  \[ = p^{(r)}(x) \prod_i \left( \frac{\tilde{p}(x) f_i(x)}{\sum_j \tilde{p}(x) f_j(x)} \right) \]
Example: Conditional Random Fields

\[ p_s(y|x) = \frac{1}{Z(\theta, x)} \exp\left\{ \sum_c \theta_c f_c(x, y) \right\} \]

- Allow arbitrary dependencies on input
- Clique dependencies on labels
- Use approximate inference for general graphs

Alternative Learning Strategy

- Recall that in CRF
  - We predict based on:
    \[ y^* | x = \arg \max_y p_s(y|x) = \frac{1}{Z(\theta, x)} \exp\left\{ \sum_c \theta_c f_c(x, y) \right\} \]
  - And we learn based on:
    \[ \theta^*_{\{y_a, x_a\}} = \arg \max_{\theta} \prod_a p_s(y_a | x_a) = \prod_a \frac{1}{Z(\theta, x_a)} \exp\left\{ \sum_c \theta_c f_c(x, y_a) \right\} \]
- MaxMargin:
  - We predict based on:
    \[ y^* | x = \arg \max_y \sum_c \theta_c f_c(x, y) = \arg \max_y w^T F(x, y) \]
  - And we learn based on:
    \[ w^*_{\{y_a, x_a\}} = \arg \max_w \left( \max_{y' \neq y_a} w^T (F(y_a, x_a) - F(y', x_a)) \right) \]
Max-Margin Learning

\[
\text{max} \quad \frac{1}{2} \|w\| - \sum_n \xi_n \\
\text{s.t.} \quad w^T (F(y_n, x_n) - F(y'_n, x_n)) \geq \xi_n + \Delta(y'_n, y_n) \quad \forall n, y'_n \in \mathcal{Y} \setminus y_n \\
\xi_n \geq 0
\]

- **Solutions:**
  - Convex optimization (akin to SVM) with exponentially many constrains
  - Many algorithms and heuristics exist
    - Interior-point methods
    - Iterative active-support elimination
    - Inference based on GM
    - …

Open Problems

- **Unsupervised CRF learning and MaxMargin Learning**
  - We want to recognize a pattern that is maximally different from the rest!

  - What does margin or conditional likelihood mean in these cases? Given only \(\{X_n\}\), how can we define the cost function?
    \[
p_y(y|x) = \frac{1}{Z(\theta, x)} \exp\left\{\sum_i \theta_i f_i(x, y_i)\right\}
    \]
    
    \[
    \text{margin} = w^T (F(y_n, x_n) - F(y'_n, x_n))
    \]
  - Algorithmic challenge