Generative and Discriminative Approaches to Graphical Models
CMSC 35900 Topics in AI
Lecture 2

Yasemin Altun

January 26, 2007
Elimination algorithm finds single marginal probability $p(f)$ by
- Choosing an elimination ordering $I$ such that $f$ is the last node
- Keep track of active potentials
- Eliminate a node $i$ by removing all active potentials referencing $i$, take product and sum over $x_i$, put this message on the active list.

Consider trees with multiple queries. This is a special case of Junction Tree Algorithm, Sum-Product algorithm.
- Choose a root node arbitrarily
- A natural elimination order is depth-first traversal of the tree.
Elimination of $j$ node.

If $j$ is an evidence node $\phi^E(x_j) = 1[x_j = \bar{x}_j]$ else $\phi^E(x_j) = 1$

$$m_{ji}(x_i) = \sum_{x_j} \left( \phi^E(x_j) \phi(x_i, x_j) \prod_{k \in c(j)} m_{kj}(x_j) \right)$$

The marginal of root $x_r$ is proportional to its messages

$$p(x_r|\bar{x}_e) = \phi^E(x_r) \prod_{k \in c(r)} m_{kr}(x_r)$$
Fig a) Choosing 1 as root, 2 has all the messages from nodes below it

Fig b) To compute $p(x_2)$, $m_{12}(x_2)$ is needed

Fig c) One pass from leaves to root and one pass backward gives the messages required to compute all node marginals
Hidden Markov Models are directed sequence models.

- Let $\mathbf{x} = (x_1, \ldots, x_l)$ and $\mathbf{y} = (y_1, \ldots, y_l)$ be sequences.
- $x_i \in \mathcal{O}$, where $\mathcal{O}$ is a set of possible observations.
- $y_i \in \mathcal{Y}$, where $\mathcal{Y}$ is a set of possible labels.
- $\pi_\sigma = p(y_1 = \sigma; \theta)$ initial transition.
- $T_{\sigma, \sigma'} = p(y_{i+1} = \sigma | y_i = \sigma'; \theta)$ transition probabilities for $\sigma, \sigma' \in \mathcal{Y}$.
- $O_{\sigma, u} = p(x_i = u | y_i = \sigma; \theta)$ observation probabilities for $\sigma \in \mathcal{Y}, u \in \mathcal{O}$.

$$P(\mathbf{x}, \mathbf{y}; \theta) = \pi_{y_1} \prod_{i=1}^{N} O_{y_i, x_i} \prod_{i=1}^{N-1} T_{y_{i+1}, y_i}$$
Set $y_i$ as the root, $x$ are evidence.

Forward pass

$$\alpha_i(\sigma) = m_{(i-1)i}(x_i = \sigma)$$

$$p(\bar{x}_{1:i}, y_i = \sigma) = \alpha_i(\sigma) = \left(\sum_{\sigma' \in Y} \alpha_{i-1}(\sigma') T_{\sigma,\sigma'}\right) O_{\sigma,\bar{x}_i}$$

Backward pass

$$p(\bar{x}_{i+1:l}|y_i = \sigma) = \beta_i(\sigma) = \sum_{\sigma' \in Y} T_{\sigma',\sigma} O_{\sigma',\bar{x}_{i+1}} \beta_{i+1}(\sigma')$$

Combining messages

$$P(y_i = \sigma|x_{1:l}) = \frac{p(\bar{x}_{1:l}|y_i)p(y_i)}{p(\bar{x}_{1:l})} \propto p(\bar{x}_{1:i}|y_i)p(\bar{x}_{i+1:l}|y_i)p(y_i) = \alpha_i(\sigma)\beta_i(\sigma)$$
Data structure:
- Generate a tree whose nodes are the maximal cliques. Then, we get \( p(x_C | x_{V \setminus C}) \) using the sum-product. We can marginalize over this to get individual marginal probabilities.
- Marginalization over different cliques of node \( i \) should yield to same \( p(x_i | x_{V \setminus i}) \).
- There are multiple clique tree. We need a data structure that preserves global consistency via local consistency.
- A junction tree is a clique tree for each clique pair \( C_i, C_j \), sharing some variables \( C_s \), \( C_s \) is included in all the nodes of the clique tree on the path from \( C_i \) and \( C_j \).
- There exists a junction tree of a graph iff the graph is triangulated.

Elimination ordering
- Message-Passing Protocol: A clique \( C_i \) can send a message to neighbor \( C_j \) if it has received messages from all its other neighbors.
Moralize if the graph is directed

Triangulate the graph $G$ (using variable elimination algorithm)

Construct a junction tree $J$ from the triangulated graph via maximal weight spanning tree.

Initialize the potential of cliques $C$ of $J$ as the product of potentials from $G$ all assigned factors from the model.

Propagate local messages on the junction tree
Same formulation can be used exactly for finding MAP

Multiplication distributes over max as well as sum

\[ \max(ab, ac) = a \max(b, c) \]

We can do MAP computation using exactly the same algorithm replacing sums with max

\[
\begin{align*}
\max p(x) &= \max_{x_1} \max_{x_2} \max_{x_3} \max_{x_4} \max_{x_5} p(x_1) p(x_2|x_1) p(x_3|x_1) p(x_4|x_2) p(x_5|x_3) p(x_6|x_2, x_5) \\
&= \max_{x_1} p(x_1) \max_{x_2} p(x_2|x_1) \max_{x_3} p(x_3|x_1) \max_{x_4} p(x_4|x_2) \max_{x_5} p(x_5|x_3) p(x_6|x_2, x_5)
\end{align*}
\]
Parameter Learning: Given the graph structure, how to get the conditional probability distributions $P_\theta(X_i|X_{\Pi_i})$?

- Parameters $\theta$ are unknown constants
- Given fully observed training data $D$
- Find their estimates maximizing the (penalized) log-likelihood

$$\hat{\theta} = \arg\max_{\theta} \log P(D|\theta)(-\lambda R(\theta))$$

Parameter estimation with fully observed data
Parameter estimation with latent variables, Expectation Maximization (EM)

Structure Learning: Given fully observed training data $D$, how to get the graph $G$ and its parameters $\theta$?
- After studying the discriminative framework
Random variable $X = (X_1, \ldots, X_n)$. The data $D = (x^1, \ldots, x^m)$ is a set of $m$ IID observations.

Maximum likelihood estimate (MLE) can be found by maximizing the log probability of the data. In Bayes nets,

$$
\ell(\theta; D) = \log p(D|\theta) = \log \prod_{j=1}^{m} p(x^j|\theta) = \sum_{j=1}^{m} \sum_{i=1}^{n} \log p(x^j_i|x^j_{\pi_j}, \theta_i)
$$

Consider $X_i$ be discrete rv with $K$ possible values. Then, $p(x_i|x_{\pi_i}, \theta_i) = \theta_i(x_i, x_{\pi_i})$ is a multinomial distribution st

$$
\sum_{x_i} \theta_i(x_i, x_{\pi_i}) = 1.
$$

$$
\ell(\theta; D) = \sum_{i} \sum_{x_{i,\pi_i}} N(x_{i,\pi_i}) \log \theta_i(x_i, x_{\pi_i})
$$

$$
N(x_{i,\pi_i}) = \sum_{j=1}^{m} 1[x^j_{i,\pi_i} = x_{i,\pi_i}] \text{ observed count of } x_{i,\pi_i} \text{ assignment in data}
$$
Parameter Estimation with Complete Observations

\[ \ell(\theta; D) = \sum_i \sum_{x_i, \pi_i} N(x_i, \pi_i) \log \theta_i(x_i, x_{\pi_i}) \]

- Estimation of \( \theta_i \) is independent of \( \theta_k \) for \( k \neq i \).
- Estimating of \( \theta_i \), ignore data associated with nodes other than \( i \) and \( \pi_i \) and maximize \( N(x_i, \pi_i) \log \theta_i(x_i, x_{\pi_i}) \) wrt \( \theta_i \).
- Add a Lagrangian term for normalization and solve for \( \theta_i \).
- ML estimate is the relative frequency.

\[ \hat{\theta}_i(x_i, x_{\pi_i}) = \frac{N(x_i, x_{\pi_i})}{N(\pi_i)} \]
Hidden Markov Models are directed sequence models.

Let \( x = (x_1, \ldots, x_l) \) and \( y = (y_1, \ldots, y_l) \) be sequences.

- \( x_i \in \mathcal{O} \), where \( \mathcal{O} \) is a set of possible observations.
- \( y_i \in \mathcal{Y} \), where \( \mathcal{Y} \) is a set of possible labels.
- \( \pi_\sigma = p(y_1 = \sigma; \theta) \) initial transition.
- \( T_{\sigma, \sigma'} = p(y_{i+1} = \sigma | y_i = \sigma'; \theta) \) transition probabilities for \( \sigma, \sigma' \in \mathcal{Y} \)
- \( O_{\sigma, u} = p(x_i = u | y_i = \sigma; \theta) \) observation probabilities for \( \sigma \in \mathcal{Y}, u \in \mathcal{O} \)

\[
P(x, y; \theta) = \pi y_1 \prod_{i=1}^{N} O_{y_i, x_i} \prod_{i=1}^{N-1} T_{y_{i+1}, y_i}
\]
Parameter Estimation in HMMs

\[ P(x, y; \theta) = \pi_{y_1} \prod_{i=1}^{N} O_{y_i, x_i} \prod_{i=1}^{N-1} T_{y_{i+1}, y_i} \]

- MLE estimate
  - \( \pi^*_\sigma = \frac{\sum_{j=1}^{m} 1[y^j_1=\sigma]}{m} \)
  - \( T^*_{\sigma, \sigma'} = \frac{\sum_{j=1}^{m} \sum_{i=1}^{l_j-1} 1[y^j_{i+1}=\sigma \land y^j_i=\sigma']}{\sum_{j=1}^{m} \sum_{i=1}^{l_j-1} y^j_i=\sigma'} \)
  - \( O^*_{\sigma, u} = \frac{\sum_{j=1}^{m} \sum_{i=1}^{l_j} 1[y^j_i=\sigma \land x^j_i=u]}{\sum_{j=1}^{m} \sum_{i=1}^{l_j} y^j_i=\sigma} \)
In MRFs, $Z$ couples the parameters. There is no closed formed solution of $\theta$.

For potential functions $\phi_C(x_C) = \exp(\langle f(x_C), \theta_C \rangle)$

$$\ell(\theta; D) = \sum_{j=1}^{m} \log p(x^j|\theta)$$

$$= \sum_{j} \left( \sum_{C} \langle f(x_C), \theta_C \rangle - \log Z_\theta \right)$$

$$= \sum_{C} N(x_C) \langle f(x_C), \theta_C \rangle - m \log Z_\theta$$

Take the derivative of $\ell$ wrt $\theta$, equate to 0 and solve for $\theta$.

The derivative of $\log Z$ wrt $\theta_C$ is the expectation $E_{x \sim p(\theta)}[f(x_C)]$. 

$$p(x|\theta) = \frac{\prod_{C} \phi(x_C)}{\sum_{x' \in Y^n} \prod_{C} \phi(x'_C)}$$
When there are latent variables, we need to marginalize over the latent variables, which leads to coupling between parameters.

Let $X$ be observed variables, $Z$ latent variables.

$$\ell(\theta; D) = \log \sum_z p(x, z|\theta)$$

Expectation Maximization (EM) Algorithm is a general approach to maximum likelihood parameter estimation (MLE) with latent(hidden) variables.

EM is a coordinate-descent algorithm to minimize Kullback-Leibler (KL) divergence
Since $Z$ is not observed, the log-likelihood of data is a marginal over $Z$ which does not decompose.

\[
\ell(\theta; x) := \log p(x|\theta) = \log \sum_z p(x, z|\theta)
\]

Replace this with expected log-likelihood wrt the averaging distribution $q(z|x)$. This is linear in

\[
l_c(\theta; x, z) := \log p(x, z|\theta).
\]

This optimization is a lower bound on $l(\theta; x)$. Using Jensen’s inequality

\[
\ell(\theta; x) = \log \sum_z p(x, z|\theta) = \log \sum_z q(z|x) \frac{p(x, z|\theta)}{q(z|x)} \\
\leq \sum_z q(z|x) \log \frac{p(x, z|\theta)}{q(z|x)} = \mathcal{L}(q, \theta)
\]
Until convergence

- Maximize wrt $q$ (E step):
  \[ q^{t+1} = \arg\max_q \mathcal{L}(q, \theta^t) \]

- Maximize wrt $\theta$ (M step):
  \[ \theta^{t+1} = \arg\max_\theta \mathcal{L}(q^{t+1}, \theta) \]
Until convergence

Maximize wrt $q$ (E step): $q^{t+1} = \arg\max_q \mathcal{L}(q, \theta^t)$
Maximize wrt $\theta$ (M step): $\theta^{t+1} = \arg\max_\theta \mathcal{L}(q^{t+1}, \theta)$

$$\arg\max_\theta \mathcal{L}(q, \theta^t) = \arg\max_\theta \sum_z q(z|x) \log p(x, z|\theta)$$
$$- \sum_z q(z|x) \log q(z|x)$$
$$= \arg\max_\theta \langle \ell_c(\theta; x, z) \rangle_q$$
Until convergence

Maximize wrt $q$ (E step): $q^{t+1} = \text{argmax}_q \mathcal{L}(q, \theta^t)$

$$
\ell(\theta; x) - \mathcal{L}(q, \theta) = \sum_z q(z|x) \log p(x|\theta) - \sum_z q(z|x) \log \frac{p(x, z|\theta)}{q(z|x)}
= D(q(z|x) || p(z|x, \theta))
$$

Minimizing $\ell(\theta; x) - \mathcal{L}(q, \theta)$ is equivalent to maximizing $\mathcal{L}(q, \theta)$.

KL divergence is minimized when $q(z|x) = p(z|x, \theta^t)$

Intuitively, given $p(x, z|\theta)$, $p(z|x, \theta^t)$ is the best guess for latent variables conditioned on $x$.

Maximize wrt $\theta$ (M step): $\theta^{t+1} = \text{argmax}_\theta \mathcal{L}(q^{t+1}, \theta)$

M step increases a lower bound on likelihood

E step closes the gap to yield $\ell(\theta^t; x) = \mathcal{L}(q^{t+1}, \theta^t)$
EM for HMMs

- **E step**: Get expected counts using Forward-Backward Algorithm.
  \[ q^t = p(z|x, \theta^{t-1}) \]

- **M step**: Find MLE estimate wrt the expected counts (relative frequency).
  \[ \theta^t = \arg\max_{\theta} \langle \ell_c(\theta; x, z) \rangle_{q^t} \]
Maximum Entropy Markov Models (MEMM) [McCFrePer00] (Generalization to Bayes Nets is trivial.)

a) HMM: \( P(\mathbf{x}, \mathbf{y}; \theta) = \pi_{y_1} \prod_{i=1}^{N} O_{y_i, x_i} \prod_{i=1}^{N-1} T_{y_{i+1}, y_i} \)

b) MEMM: \( P(\mathbf{y}|\mathbf{x}; \theta) = \pi_{y_1, x_1} \prod_{i=1}^{N-1} T_{y_{i+1}, y_i, x_{i+1}} \)
Forward-Backward algorithm

\[ \alpha_i(\sigma) = \sum_{\sigma' \in \mathcal{Y}} \alpha_{i-1}(\sigma') T_{\sigma, \sigma', \bar{x}_i} \]

\[ \beta_i(\sigma) = \sum_{\sigma' \in \mathcal{Y}} T_{\sigma', \sigma, \bar{x}_{i+1}} \beta_{i+1}(\sigma') \]

Representation for \( \sigma, \bar{\sigma} \in \mathcal{Y}, u \in \mathcal{O} \)

\[ T_{\sigma, \sigma', \bar{u}} = \frac{1}{Z(\bar{u}, \sigma')} \exp \left( \langle \theta, f(\bar{u}, \sigma, \sigma') \rangle \right) \]

\[ Z(\bar{u}, \sigma') = \sum_{\sigma} \exp \left( \langle \theta, f(\bar{u}, \sigma, \sigma') \rangle \right) \]

Coupling between \( \theta \) due to \( Z \) results in no-closed-form solution. Use some convex optimization technique.

Note the difference between undirected models where \( Z \) is a sum over all possible values of all nodes (\( \mathcal{Y}^n \)). Here, it is a sum over \( \mathcal{Y} \).
Generative vs Discriminative Approaches

Generative framework: HMMs model $p(x, y)$

- **Advantages:**
  - Efficient learning algorithm. Relative frequency.
  - Can handle missing observation (simply latent variable)
  - Naturally incorporates prior knowledge

- **Disadvantages:**
  - Harder problem $p(y|x)p(y)$ vs $p(y|x)$
  - Questionable independence assumption $x_i \perp x_j|y_i, \forall j \neq i$
  - Limited representation. Overlapping features are problematic.
    - Assuming independence violates the model.
      \[ p(x_i|y_i) = \prod_k p(f_k(x_i)|y_i) \]
      \[ f_1(u) = \text{Is } u \text{ the word "Brown"?}, f_2(u) = \text{Is } u \text{ capitalized?} \]
    - Conditioning increase the number of parameters.
      \[ p(x_i|y_i) = \prod_k p(f_k(x_i)|y_i, f_1(x_i), \ldots, f_{k-1}(x_i)) \]
    - Increased complexity to capture dependency between $y_i$ and $x_{i-1}$, $p(x_i|y_i, y_{i-1})$. 
Generative vs Discriminative Approaches

Discriminative learning: MEMM model $p(y|x)$

- Advantages:
  - Solves more direct problem.
  - Richer representation via kernels or feature selection.
    Conditioning on $x$, overlapping features are trivial.
  - No independence assumption on observations.
  - Lower asymptotic error.

- Disadvantages:
  - Expensive learning. There is no closed form solution as in HMMs. Therefore, we need Forward-Backward algorithm.
  - Missing values in observation is problematic due to conditioning.
  - Requires more data ($O(d)$) to reach its asymptotic error, whereas generative models require $O(\log d)$, for $d$ number of parameters [NgJor01].