

Readings:

K&F: 11.3, 11.5
Yedidia et al. paper from the class website
Chapter 9, 20 – Jordan
K&F: 16.2

Unifying Variational and GBP Learning Parameters of MNs EM for BNs

Graphical Models – 10708

Carlos Guestrin

Carnegie Mellon University

November 15th, 2006

1

Revisiting Mean-Fields

$\ln Z = \underbrace{F[P_{\mathcal{F}}, Q]}_{\text{constant}} + D(Q||P_{\mathcal{F}})$ $F[P_{\mathcal{F}}, Q] = \sum_{\phi \in \mathcal{F}} E_Q[\ln \phi] + H_Q(\mathcal{X})$

■ Choice of Q: $Q(x) = \prod_i Q_i(x_i)$

■ Optimization problem:

$\max_{Q_i} \sum_{\phi \in \mathcal{F}} E_Q[\ln \phi] + \sum_i H_{Q_i}(X_i)$ ← decomposes

s.t. $\sum_{x_i} Q_i(x_i) = 1$

$Q_i(x_i) \geq 0$

$\max_Q F[P_{\mathcal{F}}, Q] = \sum_{\phi \in \mathcal{F}} E_Q[\ln \phi] + \sum_j H_{Q_j}(X_j), \quad \forall i, \sum_{x_i} Q_i(x_i) = 1$

Interpretation of energy functional

- Energy functional: $F[P_{\mathcal{F}}, Q] = \sum_{\phi \in \mathcal{F}} E_Q[\ln \phi] + H_Q(\mathcal{X})$

- Exact if $P=Q$: $\ln Z = F[P_{\mathcal{F}}, Q] + D(Q||P_{\mathcal{F}})$
Q ∈ P *maximized* *→ 0*

- View problem as an approximation of entropy term:

estimate expectation w.r.t. Q

approximating $H_Q(\mathcal{X}) \approx H_P(\mathcal{X})$

interpretation: $H_P(\mathcal{X}) \approx \sum_i H_{Q_i}(x_i)$

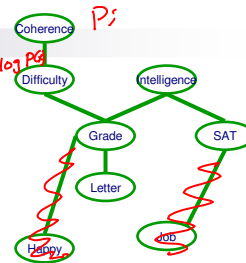
Entropy of a tree distribution

- Entropy term: $H_P(\mathcal{X}) = -E_P[\log P(\mathcal{X})] = -\sum_{\mathcal{X}} P(\mathcal{X}) \log P(\mathcal{X})$

- Joint distribution: $P(\mathcal{X}) = \prod_{i,j} \psi_{ij}(x_i, x_j)$

- Decomposing entropy term:

$$P(\mathcal{X}) = \frac{P(CD) \cdot P(DG) \cdot P(GI) \cdot P(JS) \cdot P(GL)}{P(CD) \cdot P(G) \cdot P(G) \cdot P(LJ)}$$



- More generally: $H_P(\mathcal{X}) = \sum_{(i,j) \in E} H(X_i, X_j) - \sum_i (d_i - 1) H(X_i)$
 □ d_i number neighbors of X_i

Loopy BP & Bethe approximation

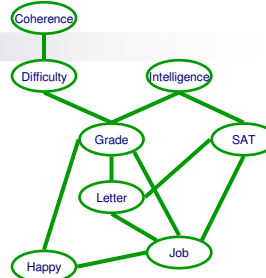
■ Energy functional: $F[P_{\mathcal{F}}, Q] = \sum_{\phi \in \mathcal{F}} E_Q[\ln \phi] + H_Q(\mathcal{X})$

■ **Bethe approximation of Free Energy:**

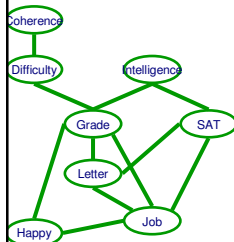
□ use entropy for trees, but loopy graphs:

$$\tilde{F}[P_{\mathcal{F}}, Q] = \sum_{(i,j) \in E} E_{\pi_{ij}}[\ln \phi_{ij}] + \sum_{(i,j) \in E} H_{\pi_{ij}}(X_i, X_j) - \sum_i (d_i - 1) H_{\pi_i}(X_i)$$

■ **Theorem:** If Loopy BP converges, resulting π_{ij} & π_i are stationary point (usually local maxima) of Bethe Free energy!



GBP & Kikuchi approximation



■ Exact Free energy: Junction Tree

$$F[P_{\mathcal{F}}, Q] = \sum_{(i,j) \in E} E_{\pi_{ij}}[\ln \phi_{ij}] + \sum_i H_{\pi_{C_i}}(C_i) - \sum_{(i,j) \in \mathcal{T}} H_{\pi_{S_{ij}}}(S_{ij})$$

■ Bethe Free energy:

$$\tilde{F}[P_{\mathcal{F}}, Q] = \sum_{(i,j) \in E} E_{\pi_{ij}}[\ln \phi_{ij}] + \sum_{(i,j) \in E} H_{\pi_{ij}}(X_i, X_j) - \sum_i (d_i - 1) H_{\pi_i}(X_i)$$

■ Kikuchi approximation: Generalized cluster graph

- spectrum from Bethe to exact
- entropy terms weighted by counting numbers
- see Yedidia et al.

■ **Theorem:** If GBP converges, resulting π_{C_i} are stationary point (usually local maxima) of Kikuchi Free energy!



What you need to know about GBP

- Spectrum between Loopy BP & Junction Trees:
 - More computation, but typically better answers
- If satisfies RIP, equations are very simple
- General setting, slightly trickier equations, but not hard
- Relates to variational methods: Corresponds to local optima of approximate version of energy functional

10.708 – ©Carlos Guestrin 2006

7

Announcements

- Tomorrow's recitation
 - Khalid on learning Markov Networks

10.708 – ©Carlos Guestrin 2006

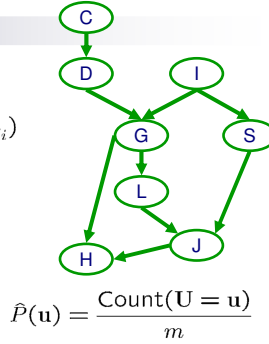
8

Learning Parameters of a BN

- Log likelihood decomposes:

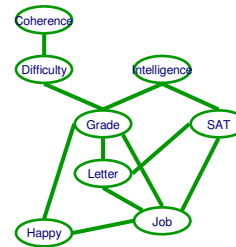
$$\ell(\mathcal{D} : \theta) = \log P(\mathcal{D} | \theta) = m \sum_i \sum_{x_i, \text{Pa}_{x_i}} \hat{P}(x_i, \text{Pa}_{x_i}) \log P(x_i | \text{Pa}_{x_i})$$

- Learn each CPT independently
- Use counts



Log Likelihood for MN

- Log likelihood of the data:

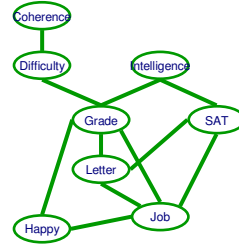


Log Likelihood doesn't decompose for MNs

$$\hat{P}(\mathbf{u}) = \frac{\text{Count}(\mathbf{U} = \mathbf{u})}{m}$$

- Log likelihood:

$$\ell(\mathcal{D} : \theta) = \log P(\mathcal{D} | \theta, \mathcal{G}) = m \sum_i \sum_{\mathbf{c}_i} \hat{P}(\mathbf{c}_i) \log \psi_i(\mathbf{c}_i) - m \log Z$$



- A concave problem

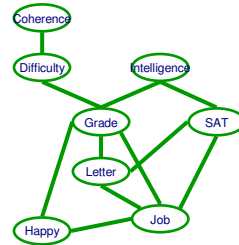
- Can find global optimum!!

- Term $\log Z$ doesn't decompose!!

Derivative of Log Likelihood for MNs

$$\hat{P}(\mathbf{u}) = \frac{\text{Count}(\mathbf{U} = \mathbf{u})}{m}$$

$$\ell(\mathcal{D} : \theta) = \log P(\mathcal{D} | \theta, \mathcal{G}) = m \sum_i \sum_{\mathbf{c}_i} \hat{P}(\mathbf{c}_i) \log \psi_i(\mathbf{c}_i) - m \log Z$$



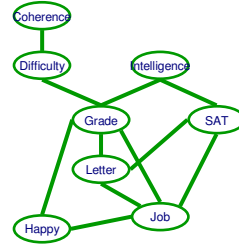
Derivative of Log Likelihood for MNs

$$\hat{P}(\mathbf{u}) = \frac{\text{Count}(\mathbf{U} = \mathbf{u})}{m}$$

$$\ell(\mathcal{D} : \theta) = \log P(\mathcal{D} | \theta, \mathcal{G}) = m \sum_i \sum_{\mathbf{c}_i} \hat{P}(\mathbf{c}_i) \log \psi_i(\mathbf{c}_i) - m \log Z$$

- Derivative:

$$\frac{\partial \ell}{\partial \psi_i(\mathbf{c}_i)} = \frac{m \hat{P}(\mathbf{c}_i)}{\psi_i(\mathbf{c}_i)} - \frac{m P_{\mathcal{F}}(\mathbf{c}_i)}{\psi_i(\mathbf{c}_i)}$$



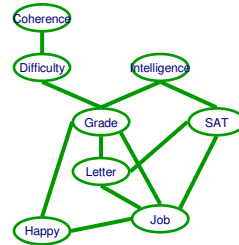
- Setting derivative to zero
- Can optimize using gradient ascent
 - Let's look at a simpler solution

Iterative Proportional Fitting (IPF)

$$\hat{P}(\mathbf{u}) = \frac{\text{Count}(\mathbf{U} = \mathbf{u})}{m}$$

$$\frac{\partial \ell}{\partial \psi_i(\mathbf{c}_i)} = \frac{m \hat{P}(\mathbf{c}_i)}{\psi_i(\mathbf{c}_i)} - \frac{m P_{\mathcal{F}}(\mathbf{c}_i)}{\psi_i(\mathbf{c}_i)}$$

- Setting derivative to zero:
- Fixed point equation:
- Iterate and converge to optimal parameters
 - Each iteration, must compute:



Log-linear Markov network (most common representation)

- **Feature** is some function $\phi[\mathbf{D}]$ for some subset of variables \mathbf{D}
 - e.g., indicator function
- **Log-linear model** over a Markov network H :
 - a set of features $\phi_1[\mathbf{D}_1], \dots, \phi_k[\mathbf{D}_k]$
 - each \mathbf{D}_i is a subset of a clique in H
 - two ϕ 's can be over the same variables
 - a set of weights w_1, \dots, w_k
 - usually learned from data
 - $$P(X_1, \dots, X_n) = \frac{1}{Z} \exp \left[\sum_{i=1}^k w_i \phi_i(\mathbf{D}_i) \right]$$

10.708 – ©Carlos Guestrin 2006

15

Learning params for log linear models 1 – Generalized Iterative Scaling

$$P(X_1, \dots, X_n) = \frac{1}{Z} \exp \left[\sum_{i=1}^k w_i \phi_i(\mathbf{D}_i) \right]$$

- IPF generalizes easily if:
 - $\phi_i(\mathbf{x}) \geq 0$
 - $\sum_i \phi_i(\mathbf{x}) = 1$
- Update rule: $w_i^{(t+1)} = w_i^{(t)} + \log \left(\frac{\sum_{\mathbf{x}} \hat{P}(\mathbf{x}) \phi_i(\mathbf{x})}{\sum_{\mathbf{x}} P^{(t)}(\mathbf{x}) \phi_i(\mathbf{x})} \right)$
- Must compute:
- If conditions violated, equations are not so simple...
 - c.f., Improved Iterative Scaling [Berger '97]

10.708 – ©Carlos Guestrin 2006

16

Learning params for log linear models 2 – Gradient Ascent

$$P(X_1, \dots, X_n) = \frac{1}{Z} \exp \left[\sum_{i=1}^k w_i \phi_i(\mathbf{D}_i) \right]$$

- Log-likelihood of data:

- Compute derivative & optimize
 - usually with conjugate gradient ascent
 - You will do an example in your homework! ☺

10.708 – ©Carlos Guestrin 2006

17

What you need to know about learning MN parameters?

- BN parameter learning easy
- MN parameter learning doesn't decompose!

- Learning requires inference!

- Apply gradient ascent or IPF iterations to obtain optimal parameters
 - applies to both tabular representations and log-linear models

10.708 – ©Carlos Guestrin 2006

18

Thus far, fully supervised learning

- We have assumed fully supervised learning:

- Many real problems have missing data:

10.708 – ©Carlos Guestrin 2006

19

The general learning problem with missing data

- Marginal likelihood – \mathbf{x} is observed, \mathbf{z} is missing:

$$\begin{aligned}\ell(\theta : \mathcal{D}) &= \log \prod_{j=1}^m P(\mathbf{x}_j | \theta) \\ &= \sum_{j=1}^m \log P(\mathbf{x}_j | \theta) \\ &= \sum_{j=1}^m \log \sum_{\mathbf{z}} P(\mathbf{x}_j, \mathbf{z} | \theta)\end{aligned}$$

10.708 – ©Carlos Guestrin 2006

20

E-step

- \mathbf{x} is observed, \mathbf{z} is missing
- Compute probability of missing data given current choice of θ
 - $Q(\mathbf{z}|\mathbf{x}_j)$ for each \mathbf{x}_j
 - e.g., probability computed during classification step
 - corresponds to “classification step” in K-means

$$Q^{(t+1)}(\mathbf{z} | \mathbf{x}_j) = P(\mathbf{z} | \mathbf{x}_j, \theta^{(t)})$$

Jensen's inequality

$$\ell(\theta : \mathcal{D}) = \sum_{j=1}^m \log \sum_{\mathbf{z}} P(\mathbf{z} | \mathbf{x}_j) P(\mathbf{x}_j | \theta)$$

- **Theorem:** $\log \sum_{\mathbf{z}} P(\mathbf{z}) f(\mathbf{z}) \geq \sum_{\mathbf{z}} P(\mathbf{z}) \log f(\mathbf{z})$

Applying Jensen's inequality

- Use: $\log \sum_{\mathbf{z}} P(\mathbf{z}) f(\mathbf{z}) \geq \sum_{\mathbf{z}} P(\mathbf{z}) \log f(\mathbf{z})$

$$\ell(\theta^{(t)} : \mathcal{D}) = \sum_{j=1}^m \log \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} | \mathbf{x}_j) \frac{P(\mathbf{z}, \mathbf{x}_j | \theta^{(t)})}{Q^{(t+1)}(\mathbf{z} | \mathbf{x}_j)}$$

The M-step maximizes lower bound on weighted data

- Lower bound from Jensen's:

$$\ell(\theta^{(t)} : \mathcal{D}) \geq \sum_{j=1}^m \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} | \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j | \theta^{(t)}) + H(Q^{(t+1)})$$

- Corresponds to weighted dataset:

- $\langle \mathbf{x}_1, \mathbf{z}=1 \rangle$ with weight $Q^{(t+1)}(\mathbf{z}=1 | \mathbf{x}_1)$
- $\langle \mathbf{x}_1, \mathbf{z}=2 \rangle$ with weight $Q^{(t+1)}(\mathbf{z}=2 | \mathbf{x}_1)$
- $\langle \mathbf{x}_1, \mathbf{z}=3 \rangle$ with weight $Q^{(t+1)}(\mathbf{z}=3 | \mathbf{x}_1)$
- $\langle \mathbf{x}_2, \mathbf{z}=1 \rangle$ with weight $Q^{(t+1)}(\mathbf{z}=1 | \mathbf{x}_2)$
- $\langle \mathbf{x}_2, \mathbf{z}=2 \rangle$ with weight $Q^{(t+1)}(\mathbf{z}=2 | \mathbf{x}_2)$
- $\langle \mathbf{x}_2, \mathbf{z}=3 \rangle$ with weight $Q^{(t+1)}(\mathbf{z}=3 | \mathbf{x}_2)$
- ...

The M-step

$$\ell(\theta^{(t)} : \mathcal{D}) \geq \sum_{j=1}^m \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} | \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j | \theta^{(t)}) + H(Q^{(t+1)})$$

- Maximization step:

$$\theta^{(t+1)} \leftarrow \arg \max_{\theta} \sum_{j=1}^m \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} | \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j | \theta)$$

- Use expected counts instead of counts:

- If learning requires $\text{Count}(\mathbf{x}, \mathbf{z})$
- Use $E_{Q^{(t+1)}}[\text{Count}(\mathbf{x}, \mathbf{z})]$

10.708 - ©Carlos Guestrin 2006

25

Convergence of EM

- Define potential function $F(\theta, Q)$:

$$\ell(\theta : \mathcal{D}) \geq F(\theta, Q) = \sum_{j=1}^m \sum_{\mathbf{z}} Q(\mathbf{z} | \mathbf{x}_j) \log \frac{P(\mathbf{z}, \mathbf{x}_j | \theta)}{Q(\mathbf{z} | \mathbf{x}_j)}$$

- EM corresponds to coordinate ascent on F

- Thus, maximizes lower bound on marginal log likelihood
- As seen in Machine Learning class last semester

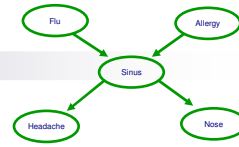
10.708 - ©Carlos Guestrin 2006

26

Data likelihood for BNs

- Given structure, log likelihood of fully observed data:

$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G})$$



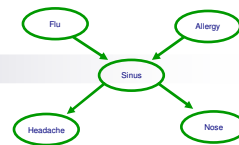
10.708 – ©Carlos Guestrin 2006

27

Marginal likelihood

- What if S is hidden?

$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G})$$



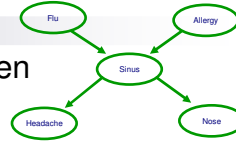
10.708 – ©Carlos Guestrin 2006

28

Log likelihood for BNs with hidden data

- Marginal likelihood – \mathbf{O} is observed, \mathbf{H} is hidden

$$\begin{aligned}\ell(\theta : \mathcal{D}) &= \sum_{j=1}^m \log P(\mathbf{o}^{(j)} | \theta) \\ &= \sum_{j=1}^m \log \sum_{\mathbf{h}} P(\mathbf{h}, \mathbf{o}^{(j)} | \theta)\end{aligned}$$



10.708 – ©Carlos Guestrin 2006

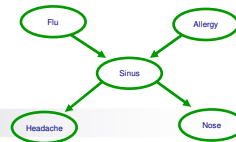
29

E-step for BNs

- E-step computes probability of hidden vars \mathbf{h} given \mathbf{o}

$$Q^{(t+1)}(\mathbf{x} | \mathbf{o}) = P(\mathbf{x} | \mathbf{o}, \theta^{(t)})$$

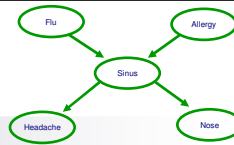
- Corresponds to inference in BN



10.708 – ©Carlos Guestrin 2006

30

The M-step for BNs

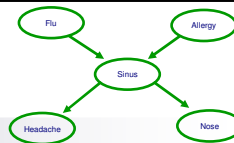


- Maximization step:

$$\theta^{(t+1)} \leftarrow \arg \max_{\theta} \sum_{\mathbf{x}} Q^{(t+1)}(\mathbf{h} | \mathbf{o}) \log P(\mathbf{h}, \mathbf{o} | \theta)$$

- Use expected counts instead of counts:
 - If learning requires $\text{Count}(\mathbf{h}, \mathbf{o})$
 - Use $E_{Q^{(t+1)}}[\text{Count}(\mathbf{h}, \mathbf{o})]$

M-step for each CPT



- M-step decomposes per CPT

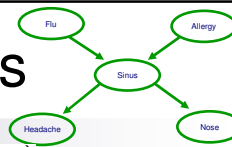
- Standard MLE:

$$P(X_i = x_i | \text{Pa}_{X_i} = \mathbf{z}) = \frac{\text{Count}(X_i = x_i, \text{Pa}_{X_i} = \mathbf{z})}{\text{Count}(\text{Pa}_{X_i} = \mathbf{z})}$$

- M-step uses expected counts:

$$P(X_i = x_i | \text{Pa}_{X_i} = \mathbf{z}) = \frac{\text{ExCount}(X_i = x_i, \text{Pa}_{X_i} = \mathbf{z})}{\text{ExCount}(\text{Pa}_{X_i} = \mathbf{z})}$$

Computing expected counts



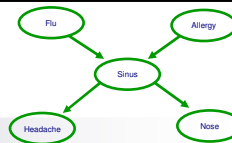
$$P(X_i = x_i | \text{Pa}_{X_i} = \mathbf{z}) = \frac{\text{ExCount}(X_i = x_i, \text{Pa}_{X_i} = \mathbf{z})}{\text{ExCount}(\text{Pa}_{X_i} = \mathbf{z})}$$

- M-step requires expected counts:
 - For a set of vars \mathbf{A} , must compute $\text{ExCount}(\mathbf{A}=\mathbf{a})$
 - Some of \mathbf{A} in example j will be observed
 - denote by $\mathbf{A}_O = \mathbf{a}_O^{(j)}$
 - Some of \mathbf{A} will be hidden
 - denote by \mathbf{A}_H
- Use inference (E-step computes expected counts):
 - $\text{ExCount}^{(t+1)}(\mathbf{A}_O = \mathbf{a}_O^{(j)}, \mathbf{A}_H = \mathbf{a}_H) \leftarrow P(\mathbf{A}_H = \mathbf{a}_H | \mathbf{A}_O = \mathbf{a}_O^{(j)}, \boldsymbol{\theta}^{(t)})$

10.708 - ©Carlos Guestrin 2006

33

Data need not be hidden in the same way



- When data is fully observed
 - A data point is
- When data is partially observed
 - A data point is
- But unobserved variables can be different for different data points
 - e.g.,
- Same framework, just change definition of expected counts
 - $\text{ExCount}^{(t+1)}(\mathbf{A}_O = \mathbf{a}_O^{(j)}, \mathbf{A}_H = \mathbf{a}_H) \leftarrow P(\mathbf{A}_H = \mathbf{a}_H | \mathbf{A}_O = \mathbf{a}_O^{(j)}, \boldsymbol{\theta}^{(t)})$

10.708 - ©Carlos Guestrin 2006

34

What you need to know

- EM for Bayes Nets
- E-step: inference computes expected counts
 - Only need expected counts over X_i and \mathbf{Pa}_{X_i}
- M-step: expected counts used to estimate parameters
- Hidden variables can change per datapoint

- Use labeled and unlabeled data → some data points are complete, some include hidden variables