Lecture 20:

Variational Methods

Kevin Murphy
29 November 2004

Inference

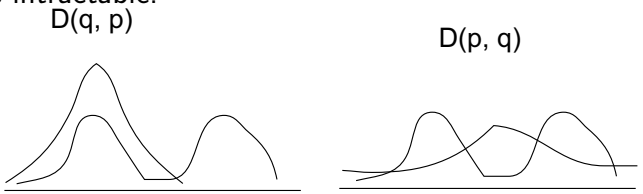
- Inference means computing $P(h_i|v)$, where h are the hidden variables v are the visible variables.
- For discrete (eg binary) hidden nodes, exact inference takes $O(2^w)$ time, where w is the induced width of the graph.
- For continuous hidden nodes, exact (closed-form) inference is only possible in rare circumstances eg. jointly Gaussian models (Kalman filters, etc.).
- We will first consider various approximations for approximate inference for discrete variables.
- For continuous or mixed discrete/cts variables
  - Extend ADF/ PF from online inference in chains to offline inference in general graphs: (ADF $\rightarrow$ EP, PF $\rightarrow$ NBP)
  - Or use MCMC (eg Gibbs)

Variational inference

- Let us try to find an approximation $Q(h)$ which is as close as possible to $P(h|v)$.
- We usually measure closeness using Kullback Leibler divergence:

$$D(Q,P) \stackrel{\text{def}}{=} \int_h Q(h) \log \frac{Q(h)}{P(h|v)} = E_{H\sim Q} \log \frac{Q(H)}{P(H|v)}$$

- This is different than minimizing $D(P,Q) = E_{H\sim P} \log \frac{Q(H)}{P(H|v)}$ which is intractable.



- We will endow Q with free (variational) parameters, and minimize $\min_\xi D(Q(h,\xi), P(h|v))$.

Variational Free Energy

- Minimizing $D(Q,P) \stackrel{\text{def}}{=} \int_h Q(h) \log \frac{Q(h)}{P(h|v)}$ is hard, since $P(h|v)$ is intractable. But for a Bayes net, $P(h,v)$ is easy (product of CPDs).
- So we minimize the free energy:

$$\begin{aligned} F(Q,P) &\stackrel{\text{def}}{=} D(Q,P) - \log P(v) \\ &= \int_h Q(h) \log \frac{Q(h)}{P(h|v)} - \int_h Q(h) \log P(v) = \int_h Q(h) \log \frac{Q(h)}{P(h,v)} \end{aligned}$$

- Since $D(Q,P) \geq 0$, we have $F(Q,P) \geq -\log P(v)$, so minimizing F is maximizing an upper bound on the log likelihood.
- Alternative derivation: use Jensen's inequality:

$$\begin{aligned} \log P(v) &= \log \int_h P(h,v) = \log \int_h Q(h) \frac{P(h,v)}{Q(h)} \\ &\geq \int_h Q(h) \log \frac{P(h,v)}{Q(h)} = -F(Q,P) \end{aligned}$$

Exact inference

- Let us minimize $F(Q, P)$ subject only to the constraint that $\sum_h Q(H) = 1$:

$$J = \sum_h Q(h) \log Q(h) - \sum_h Q(h) \log P(h|v) + \lambda(\sum_h Q(h) - 1)$$

- Derivative:

$$\frac{\partial J}{\partial Q(h')} = \log Q(h') + \frac{Q(h')}{Q(h')} - \log P(h'|v) + \lambda$$

- Solving $\frac{\partial J}{\partial Q(h')} = 0$ yields $Q(h) = P(h|v)$.

Pairwise MRFs

- For ease of explanation, I will often assume the model can be written as an MRF with pairwise potentials (one per edge):

$$P(x|y) = \frac{1}{Z} \prod_{<ij>} \psi_{ij}(x_i, x_j) \prod_i \psi_{ii}(x_i)$$

- Any Bayes net/ Markov net/ factor graph can be converted into this form, by creating extra "meganodes".

Viterbi approximation

- The Viterbi approximation is to assume that all the posterior probability mass is assigned to a single (MAP) assignment $\hat{h}$: $Q(h) = \delta(h, \hat{h})$.
- i.e., we associate every hidden variable with a single value.
- For GMs with low treewidth, we can find $\hat{h}$ efficiently.
- In general, we can use iterative techniques.

Iterative Conditional Modes (ICM)

- ICM assigns each variable to its MAP estimate, holding all the others constant:

$$h_i := \arg\max_{h_i} P(h_i|h \setminus h_i, v) = \arg\max_{h_i} \psi_{ii}(h_i) \prod_{j \in N_i} \psi_{ij}(h_i, h_j)$$

  where the h_j's are in i's Markov blanket.
- K-means clustering is an example of ICM, where h are the assignment variables for each data point to a cluster, and the value of the cluster centers (means).
- ICM is very greedy and often gets stuck in local optima.

Gibbs sampling

- Gibbs sampling is a stochastic version of ICM, where instead of picking the best state, we sample a state:

$$h_i \sim P(h_i|h \setminus h_i, v) = \frac{F(h_i)}{\sum_{h_i} F(h_i)}$$

where

$$F(h_i) = \psi_{ii}(h_i) \prod_{j \in N_i} \psi_{ij}(h_i, h_j)$$

- This is less greedy than ICM, but can be much slower.

Mean field method

- The mean field method is like a deterministic version of Gibbs sampling, where we replace samples with expected values.
- We make a fully factorized approximation: $Q(x) = \prod_i b_i(x_i)$. So the mean field free energy is

$$F_{MF}(\{b_i\}) = -\sum_{<ij>} \sum_{x_i, x_j} b_i(x_i) b_j(x_j) \log \psi_{ij}(x_i, x_j) + \sum_i \sum_{x_i} b_i(x_i)[\log b_i(x_i) - \log \psi_{ii}(x_i)]$$

- We want to minimize $F_{MF}(b_i)$ subject to $\sum_{x_i} b_i(x_i) = 1$.
- Hence we iteratively update

$$b_i(x_i) \propto \psi_{ii}(x_i) \exp\left(\sum_{j \in N_i} \sum_{x_j} b_j(x_j) \log \psi_{ij}(x_i, x_j)\right)$$

Mean field Boltzmann machines

- The Boltzmann machine (stochastic Hopfield network) is a pairwise MRF where nodes are binary (either $S_i \in \{0, 1\}$ or $h_i \in \{-1, +1\}$), and potentials have the restricted form $\psi_{ij}(S_i, S_j) = \exp \theta_{ij} S_i S_j$ and $\phi_{ii}(S_i) = \exp \theta_{i0} S_i$:

$$P(s) = \frac{1}{Z} \exp\left(\sum_{i<j} \theta_{ij} S_i S_j + \sum_i \theta_{i0} S_i\right)$$

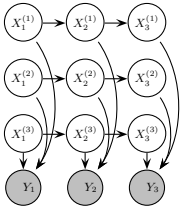
- The mean field approximation is $Q(h|v) = \prod_i \mu_i^{S_i} (1 - \mu_i)^{1 - S_i}$, where $\mu_i = E(S_i) = P(S_i = 1|v)$.
- Minimizing $D(P, Q)$ yields the mean field update equations:

$$\mu_i = \sigma(\sum_j \theta_{ij} \mu_j + \theta_{i0})$$

Structured variational approximations

- Meanfield assumes Q is fully factorized.
- We can model correlations by exploiting *tractable substructure*.
- e.g., decompose factorial HMM into product of chains

$$Q(X_{1:T}^{1:N}) = \prod_{i=1}^{N} Q(X_{1:N}^{i})$$



Loopy belief propagation

- Structured variational approximations remove some edges from the graph and replace their effect with variational parameters.
- An alternative is to leave all the original edges intact, but only capture their effect locally.
- Recall that the free energy is

$$F = \sum_h Q(h) \log Q(h) - \sum_k \sum_{h_{C_k}} Q(h_{C_k}) \log \psi_k(h_{C_k}, v_{C_k})$$

- The second term is the expected value of a local factor, and is easy to compute for any $Q(h)$.
- But the entropy term is intractable for general $Q(h)$.
- We will show how loopy belief propagation can minimize an approximation to this.

Bethe free energy

- The Bethe approximation is

$$Q(h) \approx \frac{\prod_{<ij>} b_{ij}(h_i, h_j)}{\prod_i b_i(h_i)^{d_i - 1}}$$

  where d_i is the degree of h_i (ie., number of factors it appears in).
- So the Bethe free energy is

$$F_{MF}(\{b_i, b_{ij}\}) = -\sum_{<ij>} \sum_{x_i, x_j} b_{ij}(x_i, x_j)[\log b_{ij}(x_i, x_j) - \log \phi_{ij}$$
$$-\sum_i (d_i - 1) \sum_{x_i} b_i(x_i)[\log b_i(x_i) - \log \psi_i(x_i)]$$

  where $\phi_{ij}(x_i, x_j) = \psi_{ij}(x_i, x_j)\psi_{ii}(x_i)\psi_{jj}(x_j)$.
- Loopy belief propagation is a way to minimize this subject to the constraints $\sum_{x_i} b_{ij}(x_i, x_j) = b_j(x_j)$ and $\sum_{x_i} b_i(x_i) = 1$.

Loopy belief propagation minimizes Bethe free energy

- In LBP, we iteratively update our beliefs by message passing

$$m_{ij}(x_j) \propto \sum_{x_i} \psi_{ij}(x_i, x_j)\psi_{ii}(x_i) \prod_{k \in N_i \setminus j} m_{ki}(x_i)$$
$$b_i(x_i) \propto \psi_{ii}(x_i) \prod_{k \in N_i} m_{ki}(x_i)$$

- The messages m_{ij} are $exp(\lambda_{ij})$, where λ_{ij} is the Lagrange multiplier enforcing the marginalization constraint while minimizing F_{bethe}.
- LBP sometimes called "sum-product algorithm".

Discrete Message passing/ belief propagation

- Consider an MRF with one potential per edge

$$P(X) = \frac{1}{Z} \prod_{<ij>} \psi_{ij}(X_i, X_j) \prod_i \phi_i(X_i)$$

- We can generalize the forwards-backwards algorithm as follows:

$$m_{ij}(x_j) = \sum_{x_i} \phi_i(x_i)\psi_{ij}(x_i, x_j) \prod_{k \in N_i \setminus \{j\}} m_{ji}(x_i)$$
$$b_i(x_i) \propto \phi_i(x_i) \prod_{j \in N_i} m_{ji}(x_i)$$

- If all potentials, messages and beliefs are discrete:

$$m_{ij} = \psi_{ij}^T \phi_i . * \prod_k m_{ki}, \quad b_i \propto \phi_i . * \prod_{j \in N_i} m_{ji}$$

Complexity of discrete BP

- If all potentials, messages and beliefs are discrete:

$$m_{ij} = \psi_{ij}^T \phi_i . * \prod_k m_{ki}, \quad b_i \propto \phi_i . * \prod_{j \in N_i} m_{ji}$$

- If there are K states, each message takes $O(K^2)$ time to compute.
- For certain kinds of potentials (e.g., $\psi_{ij}(i,j) = exp(||u_i - u_j||^2)$ where $u_i, u_j \in \mathbb{R}$), the messages can be computed in $O(K \log K)$ or even $O(K)$ time.
- For general potentials, once can use multipole methods and fancy data structures (like kd-trees) to do this in $O(K)$ or $O(K \log K)$ time. See Nando's NIPS workshop on "fast methods".

Loopy belief propagation

- The BP equations are exact if the graph is a chain or a tree (assuming we can implement sum and product operators analytically).
- What happens if BP is applied to graphs with loops?
- If may not convergence, and even if it does, it may be wrong.
- However, in practice, it often works well (e.g., error correcting codes).

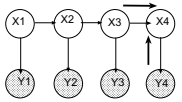
Msg Type	Algo	Correct if conv?	Suff cond for conv?
Discrete	$\sum \prod$	No	No
Discrete	$\max \prod$	Strong local opt.	No
Gaussian	$\sum \prod$	Means - yes, covs - no	Yes
General	$\sum \prod$	?	?

Summary so far

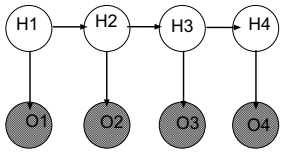
- For discrete state spaces, we have the following ranking of algorithms from best to worst (in terms of accuracy/ speed):
  - Loopy belief propagation
  - Mean field
  - Iterative conditional modes
  - Gibbs sampling
- What about continuous state spaces?

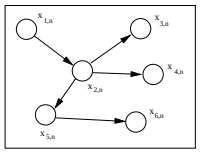
Message passing for general state spaces

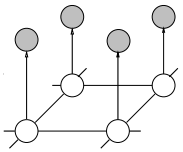
- Filtering on chains is equivalent to message passing in a left-to-right fashion.



- Smoothing on chains involves a forward and a backwards pass.
- Inference on trees involves an upwards and a downwards pass.
- Inference on loopy graphs involves parallel message passing.







Gaussian Message passing/ belief propagation

- Consider an MRF with one potential per edge

$$P(X) = \frac{1}{Z} \prod_{<ij>} \psi_{ij}(X_i, X_j) \prod_i \phi_i(X_i)$$

where $\psi_{ij} = \exp(X_i^T V_{ij} X_j)$ and $\phi_i = \exp(\frac{1}{\sigma_i}(X_i - \mu_i)^2)$.

- The BP equations are as before:

$$m_{ij}(x_j) \propto \sum_{x_i} \phi_i(x_i)\psi_{ij}(x_i, x_j) \prod_{k \in N_i \setminus \{j\}} m_{ji}(x_i)$$

$$b_i(x_i) \propto \phi_i(x_i) \prod_{j \in N_i} m_{ji}(x_i)$$

- Since a Gaussian times a Gaussian is a Gaussian, and the marginal of a Gaussian is another Gaussian, we can implement these equations in closed form (generalization of the Kalman filter).

General Message passing/ belief propagation

- In general, how can we implement these equations?

$$m_{ij}(x_j) \propto \sum_{x_i} \phi_i(x_i)\psi_{ij}(x_i, x_j) \prod_{k \in N_i \setminus \{j\}} m_{ji}(x_i)$$

$$b_i(x_i) \propto \phi_i(x_i) \prod_{j \in N_i} m_{ji}(x_i)$$

- This depends on the form of the potentials ψ and ϕ, and the form of the beliefs b (from which the form of the messages m can be inferred), just as in filtering for state-space models.

Expectation Propagation (EP)

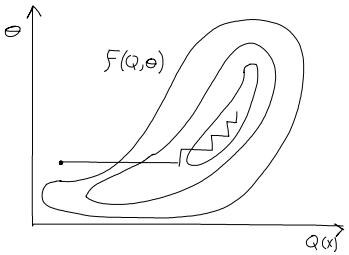
- Cross between ADF (assumed density filtering) and BP.
- Suppose potentials/ beliefs are mixtures of K Gaussians. Number of mixture components of posterior belief is K^d for a node with d neighbors; need to project back to K Gaussians (moment matching).
- The tractable messages are inferred by dividing the new belief by the old belief.
- EP is iterated ADF.
- Advantages of iterating:
  - Errors made earlier in the sequence can be recovered from.
  - Less dependence on the order in which data arrives.
- Multiple forward-backwards passes are necessary, even for chains/ trees, because the message computations are not exact.

Non-parametric belief propagation (NPBP)

- Cross between particle filtering and BP.

EM

- If all the hidden variables are discrete, and all the parameters are continuous, we can use the approximation $Q(h,\theta) = Q(h)\delta(\theta,\hat{\theta})$, where $Q(h)$ is a general posterior on h and a delta function on the parameters.
- The EM algorithm consists of minimizing $F(Q, P_\theta)$ using coordinate ascent.
- E-step: minimize wrt $Q(h)$ = computing $P(h|v,\hat{\theta})$.
- M-step: minimize wrt $\delta(\theta,\hat{\theta})$.



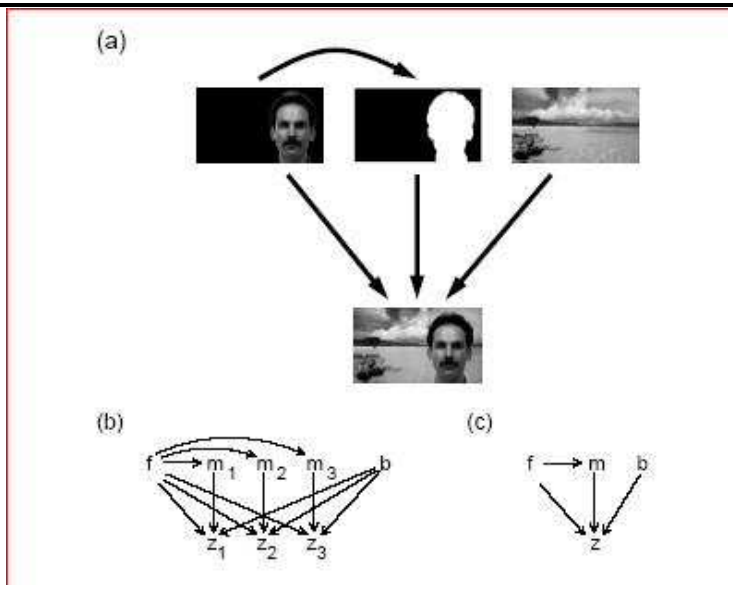
EM variants

- Standard EM: $Q(h,\theta) = P(h|v,\hat{\theta})\delta(\theta,\hat{\theta})$.
- Variational EM: use a variational approximation (eg mean field) in the E-step.
- Stochastic EM: use Monte Carlo in the E-step.
- Incremental EM: update parameters after each training case (online) instead of after all data (batch).
- "Generalized EM": do a partial M-step (eg. gradient step).
- Variational Bayes EM (ensemble learning): replace point estimates of parameters with distributions in the M-step.
- CG-EM: alternate between conjugate gradient and EM.

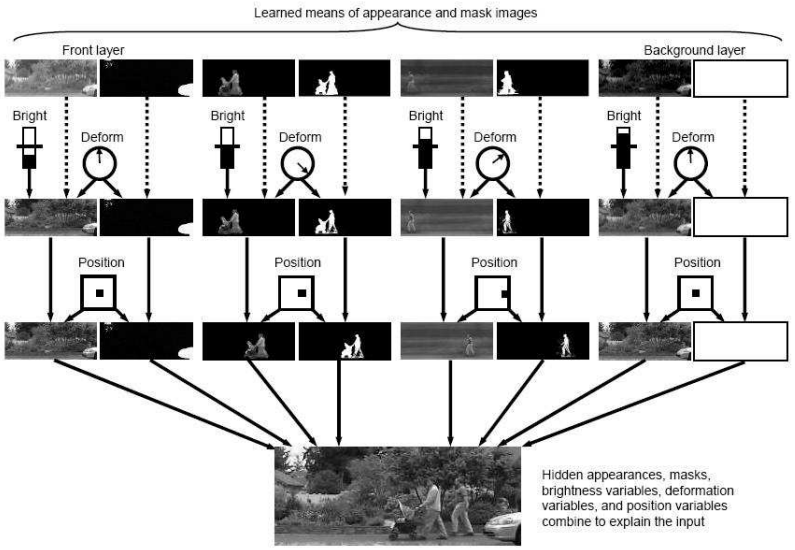
Comparison of methods

- "A comparison of algorithms for inference and learning in PGMs", Frey and Jojic, PAMI 2004 to appear

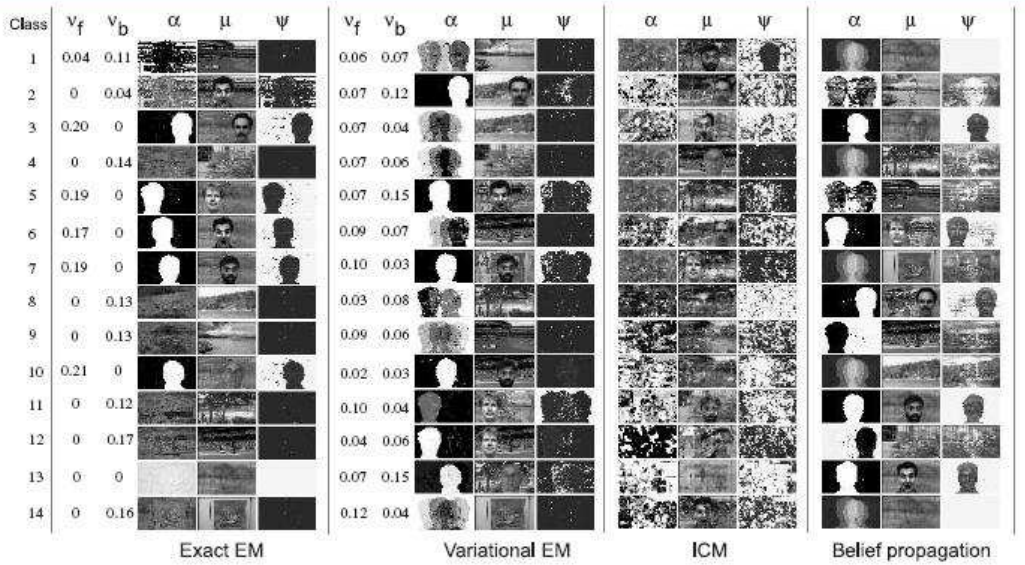
Layered model of foreground + background



Multiple layers plus continuous deformations



Comparison of EM: exact, mean field, ICM, BP



Comparison of EM: exact, mean field, ICM, BP