L11 – Undirected Graphical Models

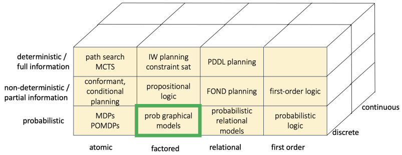
Barber 4.1, 4.2, 4.4, 5.1 (Notice that we are changing texts.)

What you should know after this lecture

- How a factor graph represents a distribution
- Relationship between factor graphs and Bayes nets
- How to use the sum-product algorithm to compute marginals in a factor graph

Probabilistic reasoning about partially-specified world states

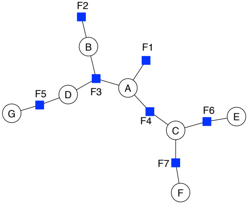
Factored states
Discrete-valued factors
Probability over possible worlds!

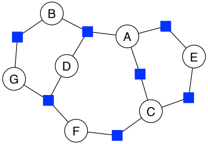


Undirected models

- Directed models (Bayes nets) are good for many problems, particularly when there is a causal interpretation of the arrows. (Though causality is not necessary)
- Relationship between pixels in an image or adjacent plots of property is not independent but there’s no sensible way to assign a direction.
- Can make graphical models with nodes and undirected arcs: Markov random fields
- We will skip that step and go straight to a formalism called factor graphs that can represent both directed and undirected models.

Factor graphs





Undirected bipartite graph: factors only connect to variables

- Round nodes are random variables V
- Square nodes are <u>factors</u> ϕ: tables specifying, for each tuple of value of the connected variables, a non-negative number
- Represent a probability distribution (e.g. left graph above)

$$P((a, b, c, d, e, f)) = \frac{1}{Z}\phi_1(a)\phi_2(b)\phi_3(a, b, d)\phi_4(a, c)\phi_5(d, g)\phi_6(c, e)\phi_7(c, f)$$

where Z is a normalizer

$$Z = \sum_{a,b,c,d,e,f} \phi_1(a)\phi_2(b)\phi_3(a, b, d)\phi_4(a, c)\phi_5(d, g)\phi_6(c, e)\phi_7(c, f)$$

Bayes nets to factor graphs

- Variable nodes are the same
- Add one factor for each CPT
- Connect it to the "output" node and all parents
- Note that, for this construction $Z = 1$ (no need to normalize!)
  Prove this to yourself by recalling the probability distribution represented by a Bayes net.

Independence relations in factor graphs

- The Markov blanket of a node V consists of all nodes that are connected to any factor connected to V.
- The Markov blanket of A in our example is $\{\mathrm{B}, \mathrm{D}, \mathrm{C}\}$
- A node V is not independent of any node in its MB
- A node V is conditionally independent of the rest of the graph, conditioned on $\mathrm{mb}(V)$
- There are some sets of independence relations that are describable by a Bayes net but not describable by a factor graph (and vice versa)

Inference in factor graphs

Some inference problems:

- Joint distribution: In a factor graph, use table multiplication to compute a big table

$$\frac{1}{Z}\prod_k \phi_k$$

  where Z is the sum of all table entries
- Marginal distribution: $P(Y)$ where $Y \subset \mathcal{V}$
- Conditional probability: $P(Y \mid E = e)$, where $Y \subset \mathcal{V}$, $E \subset \mathcal{V}$, and $Y \cap E = \emptyset$; and e is the observed values of the variables in E. Note that it is not necessary that $Y \cup E = \mathcal{V}$.
- Most probable assignment (MAP):

$$\text{argmax}\, y P(Y = y \mid E = e) \enspace .$$

  Note that the MAP of a set of variables is not necessarily the set of MAPs of the individual variables.

Computing all the individual marginals

- This method only applies if your factor graph does not have any cycles!
- Awesome algorithm with many names: belief propagation, sum-product, message passing
- Runs in time $O(N \cdot |T^*|)$ where N is the number of nodes and $|T^*|$ is number of entries in the largest table (exponential in the number of variables it is connected to).
- Can parallelize the computation.

Belief propagation idea

- Pick an arbitrary variable $V_i \in \mathcal{V}$ to be the root node
- Let $N(V)$ be the factors connected to V, $N(\phi)$ vars connected to ϕ

$$
\begin{aligned}
P(V_i) &= \sum_{\mathcal{V} \setminus V_i} P(\bar{v}) \\
&= \sum_{\mathcal{V} \setminus V_i} \prod_j \phi_j(\bar{v}) \\
&= \sum_{\mathcal{V} \setminus V_i} \prod_{\phi \in N(V_i)} F_\phi(\bar{v}) \\
&= \prod_{\phi \in N(V_i)} \sum_{V \in N(\phi) \setminus V_i} F_\phi(\bar{v}) \\
&= \prod_{\phi \in N(V_i)} \mu_{\phi \to V}(v)
\end{aligned}
$$

where F_ϕ is the product of all the factors in the subtree attached to factor ϕ

- Recursive algorithm passes messages from leaves up to root, and then back down again

Factor-to-variable messages

$\mu_{\phi \to V}(v)$ expresses the ϕ subtree's preference over the vector of possible values v for variable V

Let $N(\phi)$ be the set of variables connected to factor ϕ

$$\mu_{\phi \to V}(v) = \sum_{W \in N(\phi) \setminus V} \phi(v, \bar{w}) \prod_{W \in N(\phi) \setminus V} \mu_{W \to \phi}(w)$$

Base case if ϕ is a leaf:

$$\mu_{\phi \to V}(v) = \phi(v)$$

Think of $\mu_{\phi \to V}$ as representing $P(V)$ if all subtrees except ϕ were cut off.

Slight abuse of notation: $\prod$ is multiplying tables, $\sum$ is marginalizing out variables.

Variable-to-factor messages

$\mu_{V \to \phi}(v)$ expresses the V subtree's preference over the vector of possible values v for variable V

$$\mu_{V \to \phi}(v) = \prod_{\psi \in N(V) \setminus \phi} \mu_{\psi \to V}(v)$$

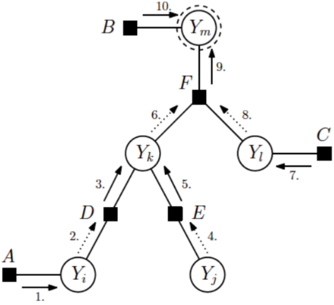
Base case if X_i is a leaf:

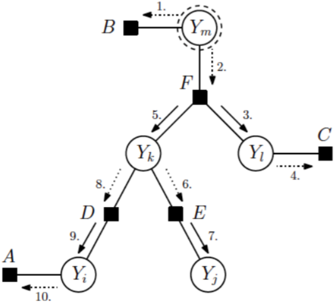
$$\mu_{V \to \phi}(v) = 1$$

Think of $\mu_{V \to \phi}$ as representing $P(V)$ if factor ϕ were cut off.

Sum-Product

1. Select V_i as root
2. Recursively compute $P(V_i) \propto \prod_{\phi \in N(V_i)} \mu_{\phi \to V_i}$
3. Pass messages back down the tree, at each node computing marginal $P(V_j) \propto \prod_{\phi \in N(V_j)} \mu_{\phi \to V_j}$





Recall that $\propto$ means "proportional to," and we generally need to normalize to get a distribution.

Handling evidence

To compute $P(V \mid E = e)$, add a new potential for every variable $V_i \in E$ that assigns 1 to $V_i = e_i$ and 0 to all other values for V_i. Then run sum-product.

More than marginal!

Easy to compute $P(V_i, V_j)$ if they are connected in the graph via one factor ϕ:

$$P(V_i, V_j) \propto \phi \prod_{\phi_i \in N(V_i) \setminus \phi} \mu_{\phi_i \to V_i} \prod_{\phi_j \in N(V_j) \setminus \phi} \mu_{\phi_j \to V_j} \prod_{V_k \in N(\phi) \setminus \{V_i, V_j\}} \mu_{V_k \to \phi}$$

Multiply everything coming into V_i, V_j, and ϕ from elsewhere, and normalize

If they aren't neighbors, then for each value $V_i = v_i$, compute

$$P(V_i = v_i, V_j = v_j) = P(V_i = v_i \mid V_j = v_j) P(V_j = v_j)$$

using tools we have already established.

Handling loopy factor graphs

Exact inference is exponential in the number of variables in the "tree width" (largest group of variables that has to be considered jointly)

1. Cutset conditioning: pick a subset of nodes C such that, if they were removed, the remaining graph would be a tree. Iterate over assignments to C, do inference, and then reassemble the answers.
2. Variable elimination: iteratively,
    - Pick a variable V (efficiency depends on how you do this)
    - Define new $\phi' = \sum_{v} \prod_{\phi \in N(V)} \phi$
    - Remove V and all $\phi \in N(V)$ from graph
    - Add ϕ' (defined on all neighboring variables)
    - Until you have a tree (or one big table!)
3. Junction tree alg : complicated!

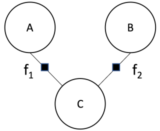
Approximation methods

1. Keep iterating belief propagation. It might converge...
2. Sampling : later!

Next time

- Approximate inference via sampling
- Finding the most likely assignment
- Temporal models

Sum-Product Practice



$$
\begin{aligned}
P(C) &= \sum_A \sum_B P(A, B, C) \\
&= 1/Z \prod_i \mu_{f_i \to q} \\
\mu_{f_1 \to C}(C) &= \sum_A \phi_{f_1}(C, A) \\
\mu_{f_2 \to C}(C) &= \sum_A \phi_{f_2}(B, A)
\end{aligned}
$$

- Suppose

$$
\phi_{f_1}(C, A) = \begin{bmatrix} A & C & \phi_{f_1}(A, C) \\ T & T & 0.05 \\ T & F & 0.45 \\ F & T & 0.45 \\ F & F & 0.05 \end{bmatrix}
$$

Then

$$
\mu_{f_1 \to C} = \begin{bmatrix} C & \mu \\ T & .5 \\ F & .5 \end{bmatrix}
$$

- Suppose

$$
\phi_{f_2}(B, A) = \begin{bmatrix} B & C & \phi_{f_1}(A, C) \\ T & T & 0.0 \\ T & F & 0.5 \\ F & T & 0.0 \\ F & F & 0.5 \end{bmatrix}
$$

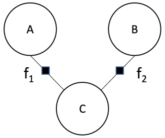
Then

$$
\mu_{f_2 \to C} = \begin{bmatrix} C & \mu \\ T & 0 \\ F & 1 \end{bmatrix}
$$

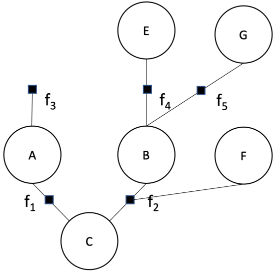
- Which means

$$
\begin{aligned}
P(C) &= (1/Z)(\mu_{f_1 \to C} \times \mu_{f_2 \to C}) \\
&= 1/Z \begin{bmatrix} C & \mu \\ T & .5 \\ F & .5 \end{bmatrix} \times \begin{bmatrix} C & \mu \\ T & 0 \\ F & 1 \end{bmatrix} \qquad = \begin{bmatrix} C & P(C) \\ T & 0 \\ F & 1 \end{bmatrix} \text{ where } Z = 0.5
\end{aligned}
$$

Sum-Product Practice



$$P(C) = 1/Z \prod_{i=1}^{2} \mu_{f_i \to C}$$
$$\mu_{f_1 \to C} = \sum_{A} \phi_{f_1} \mu_{A \to f_1}$$
$$\mu_{A \to f_1} = \mathbf{1}$$
$$\mu_{f_2 \to C} = \sum_{A} \phi_{f_2} \mu_{B \to f_2}$$
$$\mu_{B \to f_2} = \mathbf{1}$$



$$\mu_{f_3 \to A} = f_3$$
$$\mu_{A \to f_1} = \mu_{f_3 \to A}$$
$$\mu_{f_1 \to C} = \sum_{A} f_1 \cdot \mu_{A \to f_1}$$
$$\mu_{E \to f_4} = \mathbf{1}$$
$$\mu_{f_4 \to B} = \sum_{E} f_4 \cdot \mu_{E \to f_4}$$
$$\mu_{G \to f_5} = \mathbf{1}$$
$$\mu_{f_5 \to B} = \sum_{G} f_5 \cdot \mu_{G \to f_5}$$
$$\mu_{B \to f_2} = \mu_{f_4 \to B} \cdot \mu_{f_5 \to B}$$
$$\mu_{F \to f_2} = \mathbf{1}$$
$$\mu_{f_2 \to C} = \sum_{B,F} f_2 \cdot \mu_{B \to f_2} \cdot \mu_{F \to f_2}$$
$$P(C) \propto \mu_{f_1 \to C} \cdot \mu_{f_2 \to C}$$

Note that, to do the backward pass, you **do not** pass $P(C)$ back out. So $\mu_{C\to f_1} = \mu_{f_2\to C}$. Similarly $\mu_{C\to f_2} = \mu_{f_1\to C}$. And then $\mu_{f_2\to F} = \sum_{C,B} f_2 \cdot \mu_{C\to f_2} \cdot \mu_{B\to f_2}$.