CMPSCI 688 Lecture 7: Independence and Inference in Markov Networks

Benjamin M. Marlin

Department of Computer Science
University of Massachusetts Amherst

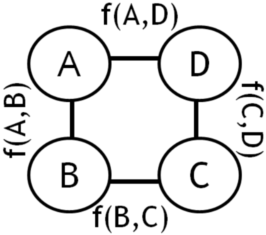
February 11, 2014

Markov Networks

A Markov network consists of an undirected graph $\mathcal{G}$ and a joint probability distribution $P(\mathbf{X})$, $\mathbf{X} = (X_1, ..., X_N)$.

- P is represented by a collection of local factors.
- The unnormalized joint distribution (Gibbs distribution) is obtained as a product of the local factors.
- Unlike Bayesian networks, the product of local factors needs to be explicitly normalized.

Example:



$$\frac{f(A,B)f(A,D)f(B,C)f(C,D)}{\sum_{A,B,C,D} f(A,B)f(A,D)f(B,C)f(C,D)}$$

Factorization Definition

The first definition of a Markov network is in terms of how the scope of the factors relate to the cliques in the graph.

Definition: Markov Network

Given an undirected graph $\mathcal{G}$ and a set of factors ϕ_k with scopes $\mathbf{D}_k$, the Gibbs distribution induced by the factors is a Markov network with respect to $\mathcal{G}$ if and only if each sub-graph of $\mathcal{G}$ induced by the scope of each factor $\mathbf{D}_k$ is a complete graph.

In short, if there is a potential covering a group of nodes, all of those nodes must be connected to each other in the graph.

Markov Properties

- Just like with Bayesian networks, Markov networks can be characterized in terms of the conditional independence relations implied by the network structure.
- Unlike Bayesian networks, which required a specialized separation criterion (D-separation), Markov networks use the standard graph-theoretic separation criterion.

Pairwise Markov Property

Pairwise Markov Property

The pairwise Markov property in a Markov network with underlying graph $\mathcal{G}$ states that two random variables X_i and X_j are conditionally independent given all of the remaining variables $\mathbf{X} - \{X_i, X_j\}$ if and only if there is no edge between X_i and X_j:

$$X_i \perp X_j | (\mathbf{X} - \{X_i, X_j\}) \Leftrightarrow (X_i, X_j) \notin \mathcal{G}$$

Local Markov Property

Local Markov Property

Define the set of variables $\mathbf{Z}$ to contain all the neighbors of X_i in $\mathcal{G}$ and $\mathbf{Y}$ to be all the remaining variables. In this case, $\mathbf{Z}$ is called the *Markov blanket* of X_i and it separates X_i from all the other nodes in the graph leading to:

$$X_i \perp \mathbf{Y} | \mathbf{Z}$$

Global Markov Property

Global Markov Property

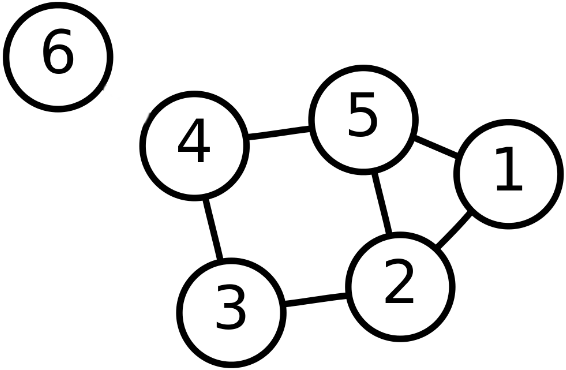
The global Markov property in a Markov network with underlying graph $\mathcal{G}$ states that two random variables X_i and X_j are conditionally independent given a third set of random variables $\mathbf{Z}$ if X_i and X_j are separated in $\mathcal{G}$ given $\mathbf{Z}$:

$$X_i \perp X_j | \mathbf{Z} \Leftrightarrow \text{sep}_{\mathcal{G}}(X_i, X_j | \mathbf{Z})$$

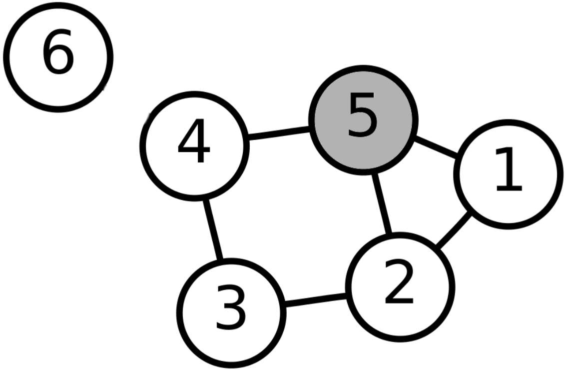
Paths and Active Paths

- **Separation:** We say that two variables X_i and X_j are separated in $\mathcal{G}$ given a third set $\mathbf{Z}$ if there are no active paths between them: $\text{sep}_{\mathcal{G}}(X_i, X_j|\mathbf{Z})$
- **Path:** We say that there is a path between X_i and X_j in $\mathcal{G}$ if we can move between X_i and X_j by following edges in $\mathcal{G}$.
- **Active Path:** We say that a path is active if it contains no nodes from from the set Z.

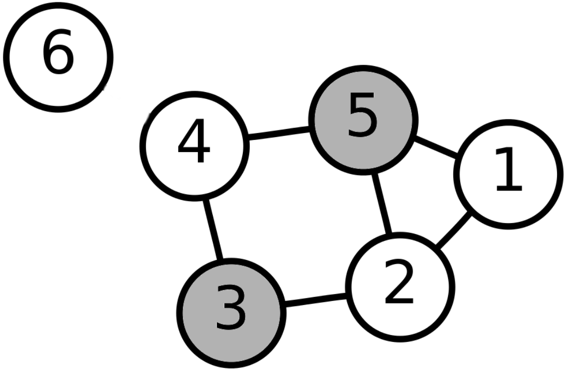
Example 1



Example 2



Example 3

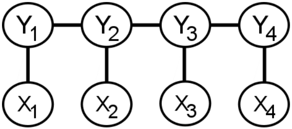


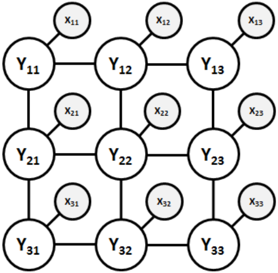
Inference in Markov Networks

- Given a Markov network, the main task we need to perform is answering probability queries.
- In a general probability query $P(\mathbf{Y}|\mathbf{X})$, we condition on a set of evidence variables $\mathbf{X}$, marginalize over a set of unobserved variables $\mathbf{Z}$ and compute the joint distribution over the remaining variables $\mathbf{Y}$. This process is called *probabilistic inference*.
- Conditioning is easy. The hard part of inference is marginalizing out all of variables that aren't involved in the query and computing the partition function.

Example: Conditional Random Fields

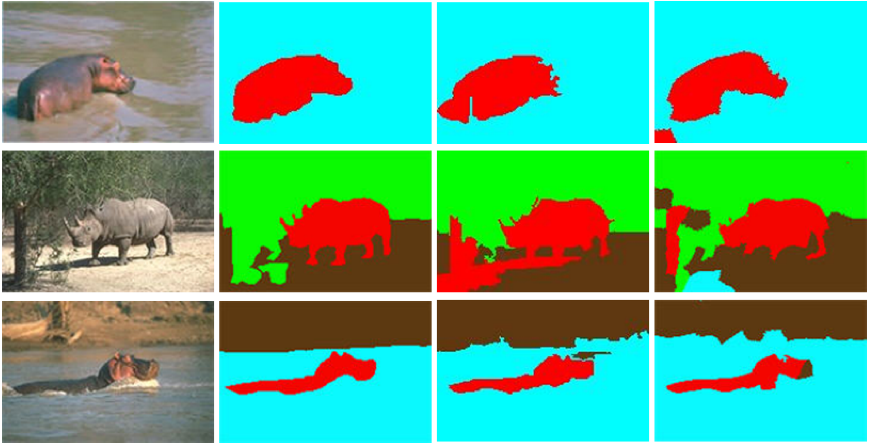
- Conditional random fields are one of the most important model classes in machine learning. They are essentially a special case of Markov networks where one group of nodes is always conditioned on.



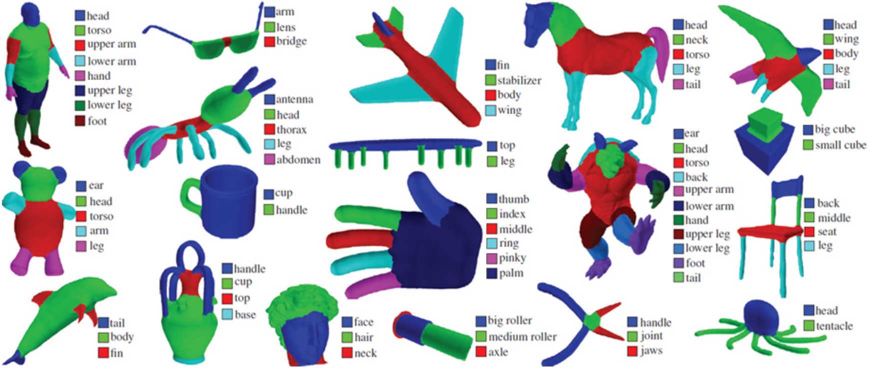


- The $\mathbf{Y}$ nodes are referred to as labels while the $\mathbf{X}$ nodes are referred to as feature or evidence nodes.

Example: Image Segmentation



Example: 3D Mesh Segmentation



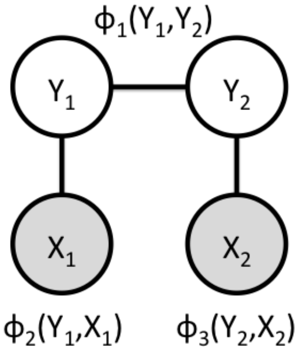
Conditioning

Conditioning on the evidence $\mathbf{X} = \mathbf{x}$ can be viewed as a specific transformation on the graph and the factors called *factor reduction*, which proceeds as follows:

1. $\mathcal{G}' \leftarrow \mathcal{G} - \mathbf{X}$
2. For all factors k
   - If $\mathbf{D}_k \cap \mathbf{X} = \emptyset$ then $\mathbf{D}'_k \leftarrow \mathbf{D}_k$ and $\phi'_k(\mathbf{D}'_k) \leftarrow \phi_k(\mathbf{D}_k)$
   - If $\mathbf{D}_k \cap \mathbf{X} = \mathbf{X}_k \neq \emptyset$, $\mathbf{D}'_k \leftarrow \mathbf{D}_k - \mathbf{X}_k$ and $\phi'_k(\mathbf{D}'_k) \leftarrow \phi_k(\mathbf{D}'_k, \mathbf{x}_k)$

If $\mathcal{G}$ and ϕ define a valid Markov network over $\mathbf{X}, \mathbf{Y}$, then after reduction, graph $\mathcal{G}'$ and factors ϕ' define a valid Markov network over $\mathbf{Y}$ with Gibbs distribution $P(\mathbf{Y}|\mathbf{X} = \mathbf{x})$.

Factor Reduction: Example



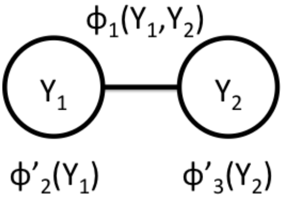
$\phi_1(Y_1,Y_2)$	Y_2=0	Y_2=1
Y_1=0	1	2
Y_1=1	7	2

$\phi_2(Y_1,X_1)$	X_1=0	X_1=1
Y_1=0	3	9
Y_1=1	4	1

$\phi_3(Y_2,X_2)$	X_2=0	X_2=1
Y_2=0	6	2
Y_2=1	2	7

Query: $P(Y_1,Y_2 \mid X_1=0, X_2=1)$

Factor Reduction: Step 1



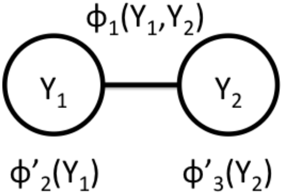
$\phi_1(Y_1,Y_2)$	Y_2=0	Y_2=1
Y_1=0	1	2
Y_1=1	7	2

$\phi_2(Y_1,X_1)$	X_1=0	X_1=1
Y_1=0	3	9
Y_1=1	4	1

$\phi_3(Y_2,X_2)$	X_2=0	X_2=1
Y_2=0	6	2
Y_2=1	2	7

Query: $P(Y_1,Y_2 \mid X_1=0, X_2=1)$

Factor Reduction: Step 2



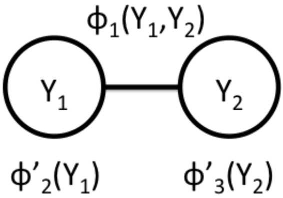
$\phi_1(Y_1,Y_2)$	$Y_2=0$	$Y_2=1$
$Y_1=0$	1	2
$Y_1=1$	7	2

$\phi_2(Y_1,X_1)$	$X_1=0$	$X_1=1$
$Y_1=0$	3	9
$Y_1=1$	4	1

$\phi_3(Y_2,X_2)$	$X_2=0$	$X_2=1$
$Y_2=0$	6	2
$Y_2=1$	2	7

Query: $P(Y_1,Y_2 \mid X_1=0, X_2=1)$

Factor Reduction: Step 2



$\phi_1(Y_1,Y_2)$	$Y_2=0$	$Y_2=1$
$Y_1=0$	1	2
$Y_1=1$	7	2

	$\phi'_2(Y_1)$
$Y_1=0$	3
$Y_1=1$	4

	$\phi'_3(Y_2)$
$Y_2=0$	2
$Y_2=1$	7

Query: $P(Y_1,Y_2 \mid X_1=0, X_2=1) \propto \phi_1(Y_1,Y_2)\, \phi'_2(Y_1)\, \phi'_3(Y_2)$

Marginalization

- The problem with marginalizing over many variables is that the computation involves sums with exponentially many terms.
- However, in the best case, these sums can be computed in linear time by distributing them into factor products. This is the basis for all computationally efficient exact inference algorithms.
- **Example:** If $Y_1, ..., Y_4$ are variables in a Markov network with factors $\phi_1(Y_1, Y_2)$, $\phi_2(Y_2, Y_3)$ and $\phi_3(Y_3, Y_4)$, then:

$$P(Y_1 = y_1) = \frac{\psi(1)}{\psi(1) + \psi(0)}$$

$$\psi(y_1) = \sum_{y_2} \sum_{y_3} \sum_{y_4} \phi_1(Y_1, Y_2)\phi_2(Y_2, y_3)\phi_3(y_3, y_4)$$

Exhaustive Factor Sum

- First, consider the exhaustive computation for $\psi(y_1)$ under the assumption that all variables are binary:

$$\psi(y_1) = \sum_{y_2} \sum_{y_3} \sum_{y_4} \phi_1(y_1, y_2)\phi_2(y_2, y_3)\phi_3(y_3, y_4)$$

- For each of the 2 values of Y_1, we compute a sum containing $2^3 = 8$ terms, each of which consists of a product of 3 terms.
- Multiplications: $2 \cdot 2^3 \cdot (3 - 1) = 32$.
- Additions: $2 \cdot (2^3 - 1) = 14$.

Efficient Factor Sum

Now, consider the efficient computation for $\psi(y_1)$ under the assumption that all variables are binary:

$$\psi(y_1) = \sum_{y_2} \phi_1(Y_1, y_2) \sum_{y_3} \phi_2(y_2, y_3) \sum_{y_4} \phi_3(y_3, y_4)$$

$$\tau_3(0) = \sum_{y_4} \phi_3(0, y_4) = \phi_3(0,0) + \phi_3(0,1)$$
$$\tau_3(1) = \sum_{y_4} \phi_3(1, y_4) = \phi_3(1,0) + \phi_3(1,1)$$

Multiplications: 0. Additions: 2.

Efficient Factor Sum

Now, consider the efficient computation for $\psi(y_1)$ under the assumption that all variables are binary:

$$\psi(y_1) = \sum_{y_2} \phi_1(y_1, y_2) \sum_{y_3} \phi_2(y_2, y_3)\tau_3(y_3)$$

$$\tau_2(0) = \sum_{y_3} \phi_2(0, y_3)\tau_3(y_3) = \phi_2(0,0)\tau_3(0) + \phi_2(0,1)\tau_3(1)$$
$$\tau_2(1) = \sum_{y_3} \phi_2(1, y_3)\tau_3(y_3) = \phi_2(1,0)\tau_3(0) + \phi_2(1,1)\tau_3(1)$$

Multiplications: 4. Additions: 2.

Efficient Factor Sum

Now, consider the efficient computation for $\psi(y_1)$ under the assumption that all variables are binary:

$$\psi(y_1) = \sum_{y_2} \phi_1(y_1, y_2)\tau_2(y_2)$$

$$\psi(1) = \sum_{y_2} \phi_1(1, y_2)\tau_2(y_2) = \phi_1(0,0)\tau_2(0) + \phi_1(0,1)\tau_2(1)$$
$$\psi(0) = \sum_{y_2} \phi_1(0, y_2)\tau_2(y_2) = \phi_1(1,0)\tau_2(0) + \phi_1(1,1)\tau_2(1)$$

Multiplications: 4. Additions: 2.

Efficient Factor Sum Complexity

- The exhaustive computation required 32 multiplications and 14 additions for a chain of length 4. For a chain of length N, this computation requires: $2^N \cdot (N-1)$ multiplications and $2 \cdot (2^{N-1} - 1)$ additions.
- The efficient computation required 8 multiplications and 6 additions! For a chain of length N, this computation requires: $4(N-2)$ multiplications and $2(N-1)$ additions.

Markov Networks in Practice

Deploying a Markov network in practice involves a number of different modeling choices subject to a variety of trade-offs, just as in Bayesian Networks:

- What variables to include?
- What graph structure to use?
- What parametric form to use for each factor?
- What parameters to use for each factor?

Markov Networks in Practice: Graph Structure

What graph structure to use?

- Structures like chains and trees are often selected for convenience (low storage complexity and fast inference).
- Unlike Bayesian networks, causality is not a useful guide for selecting graphs because Markov networks due to the lack of directionality.
- In the case of sequential and spatial processes, the variables are often associated with time stamps, indices in a sequence or locations in a metric space. In these cases the nearest neighbor graph is often used.
- We can also search for or learn structures (we'll see some of this later in the course).

Parameterizations

As with Bayesian networks, factors in discrete Markov networks can be parameterized in several different ways:

- **Standard:** The standard representation of a factor $\phi_k(\mathbf{D}_k)$ is simply a table of non-negative numbers with one element $\theta^k_{\mathbf{d}_k}$ for every $d_k \in Val(\mathbf{D}_k)$.

- **Log-Linear:** A log-linear representation of a factor $\phi_k(\mathbf{D}_k)$ associates a weight w_k and a fixed *feature function* $g_k(\mathbf{D}_k)$ with factor k. We define $\phi_k(\mathbf{D}_k) = \exp(w_k g_k(\mathbf{D}_k))$. The Ising model is thus a special case of a log-linear model where $g_k(X_i, X_j) = X_i X_j$.

Maximum Likelihood Learning in Markov Networks

Maximum likelihood estimation provides a default solution for parameter learning in Markov networks. The principles are exactly the same as for learning in Bayesian networks. In this case however, *the computational complexity of learning can be exponential in the number of variables.*

Review and Preview

- **Review:** We've discussed inference in Markov Networks, and described applications including CRFs.
- **To Do:** Assignment 2 due will be issued shortly.