Bayesian Networks

Nicholas Ruozzi
University of Texas at Dallas

Structured Distributions

- We’ve seen two types of simple probability models that can be learned from data
  - Naive Bayes: assume attributes are independent given the label
  - Hidden Markov Models: assumes the hidden variables form a Markov chain and each observation is conditionally independent of the remaining variables given the corresponding latent variable
- Today: Bayesian networks
  - Generalizes both of these cases

Structured Distributions

- Consider a general joint distribution $p(X_1, \dots, X_n)$ over binary valued random variables

- If $X_1, \dots, X_n$ are all independent given a different random variable Y, then

$$p(x_1, \dots, x_n|y) = p(x_1|y) \dots p(x_n|y)$$

and

$$p(y, x_1, \dots, x_n) = p(y)p(x_1|y) \dots p(x_n|y)$$

- How much storage is needed to represent this model?

Structured Distributions

- Consider a different joint distribution $p(X_1, \dots, X_n)$ over binary valued random variables

- Suppose, for $i > 2, X_i$ is independent of $X_1, \dots, X_{i-2}$ given X_{i-1}

$$\begin{aligned} p(x_1, \dots, x_n) &= p(x_1)p(x_2|x_1) \dots p(x_n|x_1, \dots, x_{n-1}) \\ &= p(x_1)p(x_2|x_1)p(x_3|x_2) \dots p(x_n|x_{n-1}) \end{aligned}$$

- How much storage is needed to represent this model?

- This distribution corresponds to a Markov chain

Bayesian Network

- A **Bayesian network** is a directed graphical model that captures independence relationships of a given probability distribution
  - Directed acyclic graph (DAG), $G = (V, E)$
  - One node for each random variable
  - One conditional probability distribution per node
  - Directed edge represents a direct statistical dependence

Bayesian Network

- A **Bayesian network** is a directed graphical model that captures independence relationships of a given probability distribution
  - Encodes local Markov independence assumptions that each node is independent of its non-descendants given its parents
  - Corresponds to a **factorization** of the joint distribution

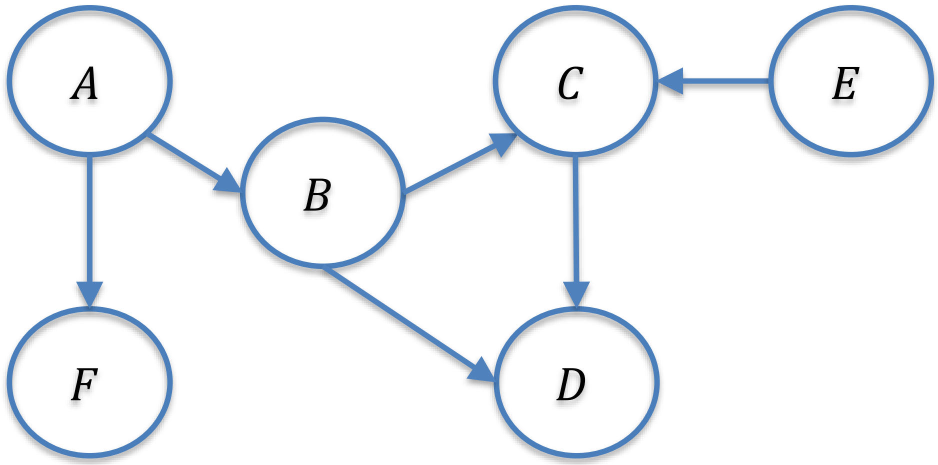
$$p(x_1, \ldots, x_n) = \prod_i p(x_i | x_{parents(i)})$$

Directed Chain

$$p(x_1, \dots, x_n) = p(x_1)p(x_2|x_1)p(x_3|x_2)\dots p(x_n|x_{n-1})$$

$$X_1 \rightarrow X_2 \rightarrow \cdots \rightarrow X_{n-1} \rightarrow X_n$$

Example:



- Local Markov independence relations?
- Joint distribution?

MLE for Bayesian Networks

- Given samples $x^{(1)}, \ldots, x^{(M)}$ from some unknown Bayesian network that factors over the directed acyclic graph G
  - The parameters of a Bayesian model are simply the conditional probabilities that define the factorization
  - For each $i \in G$ we need to learn $p(x_i | x_{parents(i)})$, create a variable $\theta_{x_i | x_{parents(i)}}$

$$\log l(\theta) = \sum_m \sum_{i \in V} \log \theta_{x_i^{(m)} | x_{parents(i)}^{(m)}}$$

MLE for Bayesian Networks

$$\log l(\theta) = \sum_{m} \sum_{i \in V} \log \theta_{x_i^{(m)} | x_{parents(i)}^{(m)}}$$

$$= \sum_{i \in V} \sum_{m} \log \theta_{x_i^{(m)} | x_{parents(i)}^{(m)}}$$

$$= \sum_{i \in V} \sum_{x_{parents(i)}} \sum_{x_i} \mathrm{N}_{x_\mathrm{i}, x_{\mathrm{parents(i)}}} \log \theta_{x_i | x_{parents(i)}}$$

MLE for Bayesian Networks

$$\log l(\theta) = \sum_{m} \sum_{i \in V} \log \theta_{x_i^{(m)} | x_{parents(i)}^{(m)}}$$

$$= \sum_{i \in V} \sum_{m} \log \theta_{x_i^{(m)} | x_{parents(i)}^{(m)}}$$

$$= \sum_{i \in V} \sum_{x_{parents(i)}} \sum_{x_i} \mathrm{N}_{x_{\mathrm{i}}, x_{\mathrm{parents(i)}}} \log \theta_{x_i | x_{parents(i)}}$$

$N_{x_i, x_{parents(i)}}$ **is the number of times** $(x_i, x_{parents(i)})$ **was observed in the training set**

MLE for Bayesian Networks

$$\log l(\theta) = \sum_{m} \sum_{i \in V} \log \theta_{x_i^{(m)} | x_{parents(i)}^{(m)}}$$

$$= \sum_{i \in V} \sum_{m} \log \theta_{x_i^{(m)} | x_{parents(i)}^{(m)}}$$

$$= \sum_{i \in V} \sum_{x_{parents(i)}} \sum_{x_i} \mathrm{N}_{x_{\mathrm{i}}, x_{\mathrm{parents(i)}}} \log \theta_{x_i | x_{parents(i)}}$$

Fix $x_{parents(i)}$ solve for $\theta_{x_i | x_{parents(i)}}$ for all x_i

(on the board)

MLE for Bayesian Networks

$$\theta_{x_i|x_{parents(i)}} = \frac{\mathrm{N}_{x_i, x_{\text{parents}(i)}}}{\sum_{x_i'} \mathrm{N}_{x_i', x_{\text{parents}(i)}}} = \frac{\mathrm{N}_{x_i, x_{\text{parents}(i)}}}{\mathrm{N}_{x_{\text{parents}(i)}}}$$

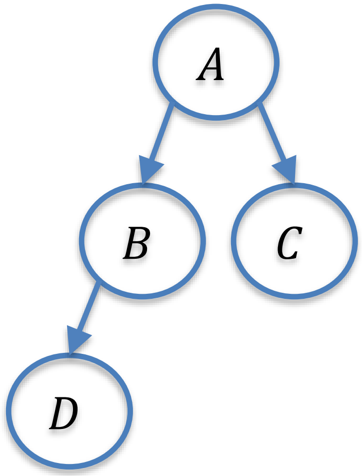
- This is just the empirical conditional probability distribution
  - Worked out nicely because of the factorization of the joint distribution
- Same as MLE for naive Bayes and HMMs (which are both BNs)

MLE for Bayesian Networks

- The previous slides have assumed that we are essentially given the structure (i.e., the DAG) of the network that we would like to learn
  - This may not be the case in practice: we may only be given samples and must learn both the parameters and the structure of the underlying network
  - But how do we decide which structures are better than others?

BN Structure Learning

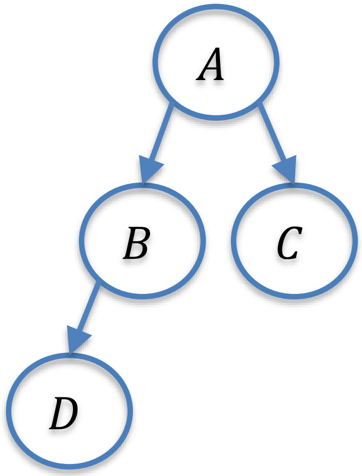
- The MLE of the conditional probability tables was given by the empirical probabilities



A	B	C	D
0	0	1	0
0	0	1	1
0	1	0	0
1	0	0	1
0	0	1	1

BN Structure Learning

- The MLE of the conditional probability tables was given by the empirical probabilities



A	B	C	D
0	0	1	0
0	0	1	1
0	1	0	0
1	0	0	1
0	0	1	1

A	P(A)
0	4/5
1	1/5

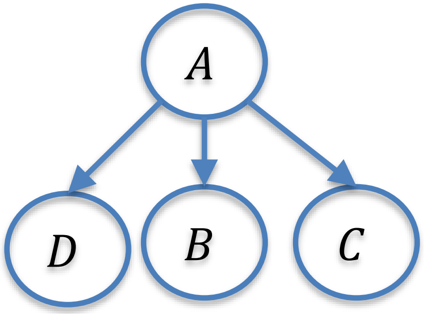
A	B	P(B|A)
0	0	3/4
0	1	1/4
1	0	1
1	1	0

B	D	P(D|B)
0	0	1/4
0	1	3/4
1	0	1
1	1	0

A	C	P(C|A)
0	0	1/4
0	1	3/4
1	0	1
1	1	0

BN Structure Learning

- The MLE of the conditional probability tables was given by the empirical probabilities



A	P(A)
0	4/5
1	1/5

A	B	P(B|A)
0	0	3/4
0	1	1/4
1	0	1
1	1	0

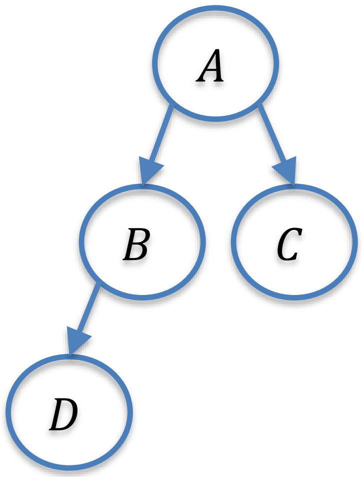
A	D	P(D|A)
0	0	1/2
0	1	1/2
1	0	0
1	1	1

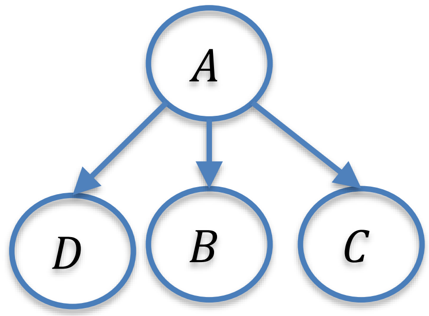
A	C	P(C|A)
0	0	1/4
0	1	3/4
1	0	1
1	1	0

A	B	C	D
0	0	1	0
0	0	1	1
0	1	0	0
1	0	0	1
0	0	1	1

BN Structure Learning

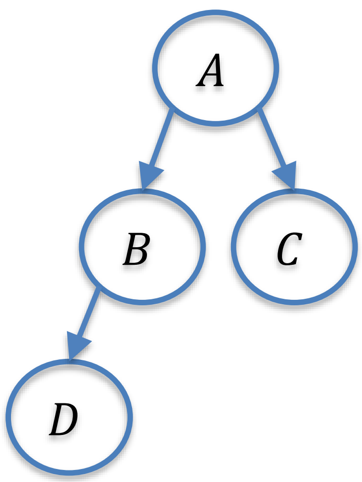
- Which model should be preferred?

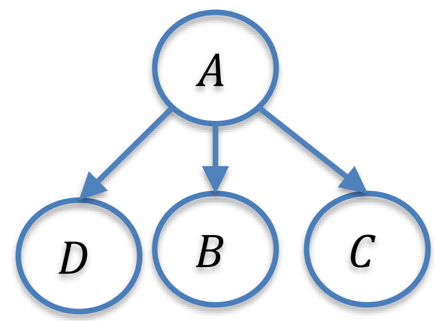




BN Structure Learning

- Which model should be preferred?

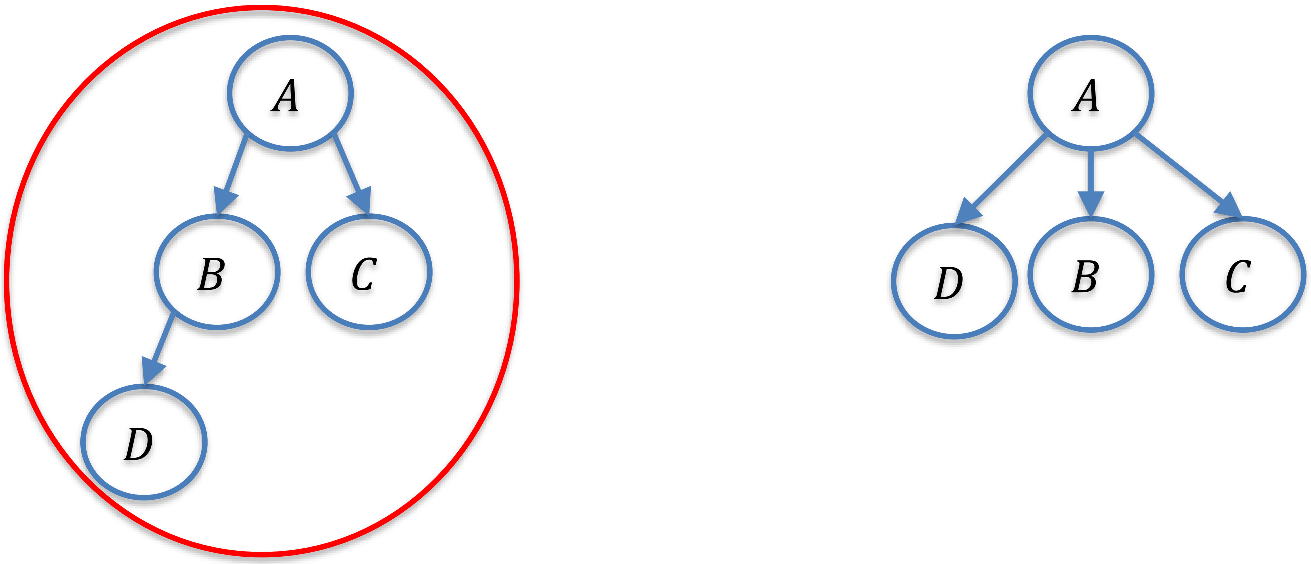




Which one has the highest log-likelihood given the data?

BN Structure Learning

- **Which model should be preferred?**



Which one has the highest log-likelihood given the data?

BN Structure Learning

- Determining the structure that maximizes the log-likelihood is not too difficult
  - A complete DAG always maximizes the log-likelihood!
  - This almost certainly results in overfitting
- Alternative is to attempt to learn simple structures
  - Optimize the log-likelihood over simple networks

Chow-Liu Trees

- Suppose that we want to find the best tree-structured BN that represents a given joint probability distribution
  - Find the tree-structured BN that maximizes the likelihood
- Let's consider the log-likelihood of a fixed tree T
  - Assume that the edges are directed so that each node has exactly one parent

Chow-Liu Trees

For a fixed tree:

$$\max_{\theta} \log l(\theta, T) = \sum_{i\in V(T)} \sum_{x_{\text{parent}(i)}} \sum_{x_i} \mathrm{N}_{x_{\mathrm{i}},x_{\text{parent}(i)}} \log \frac{\mathrm{N}_{x_i,x_{\text{parent}(i)}}}{N_{x_{\text{parent}(i)}}}$$

$$= \sum_{i\in V(T)} \left[\sum_{x_i} N_{x_i} \log N_{x_i} + \sum_{x_{\text{parent}(i)}} \sum_{x_i} \mathrm{N}_{x_{\mathrm{i}},x_{\text{parent(i)}}} \log \frac{\mathrm{N}_{x_i,x_{\text{parent}(i)}}}{N_{x_i} N_{x_{\text{parent}(i)}}} \right]$$

$$= \left[\sum_{i\in V} \sum_{x_i} N_{x_i} \log N_{x_i} \right] + \left[\sum_{(i,j)\in E(T)} \sum_{x_i,x_j} \mathrm{N}_{x_{\mathrm{i}},x_j} \log \frac{\mathrm{N}_{x_{\mathrm{i}},x_j}}{N_{x_i} N_{x_j}} \right]$$

Chow-Liu Trees

For a fixed tree:

$$\max_{\theta} \log l(\theta, T) = \sum_{i \in V(T)} \sum_{x_{\text{parent}(i)}} \sum_{x_i} \mathrm{N}_{x_i, x_{\text{parent}(i)}} \log \frac{\mathrm{N}_{x_i, x_{\text{parent}(i)}}}{N_{x_{\text{parent}(i)}}}$$

$$= \sum_{i \in V(T)} \left[\sum_{x_i} N_{x_i} \log N_{x_i} + \sum_{x_{\text{parent}(i)}} \sum_{x_i} \mathrm{N}_{x_\mathrm{i}, x_{\text{parent(i)}}} \log \frac{\mathrm{N}_{x_i, x_{\text{parent}(i)}}}{N_{x_i} N_{x_{\text{parent}(i)}}} \right]$$

$$= \left[\sum_{i \in V} \sum_{x_i} N_{x_i} \log N_{x_i} \right] + \left[\sum_{(i,j) \in E(T)} \sum_{x_i, x_j} \mathrm{N}_{x_\mathrm{i}, x_j} \log \frac{\mathrm{N}_{x_\mathrm{i}, x_j}}{N_{x_i} N_{x_j}} \right]$$

Doesn’t depend on the selected tree!

Chow-Liu Trees

For a fixed tree:

$$\max_{\theta} \log l(\theta, T) = \sum_{i \in V(T)} \sum_{x_{\text{parent}(i)}} \sum_{x_i} \mathrm{N}_{x_i, x_{\text{parent}(i)}} \log \frac{\mathrm{N}_{x_i, x_{\text{parent}(i)}}}{N_{x_{\text{parent}(i)}}}$$

$$= \sum_{i \in V(T)} \left[\sum_{x_i} N_{x_i} \log N_{x_i} + \sum_{x_{\text{parent}(i)}} \sum_{x_i} \mathrm{N}_{x_\mathrm{i}, x_{\text{parent(i)}}} \log \frac{\mathrm{N}_{x_i, x_{\text{parent}(i)}}}{N_{x_i} N_{x_{\text{parent}(i)}}} \right]$$

$$= \left[\sum_{i \in V} \sum_{x_i} N_{x_i} \log N_{x_i} \right] + \left[\sum_{(i,j) \in E(T)} \sum_{x_i, x_j} \mathrm{N}_{x_\mathrm{i}, x_j} \log \frac{\mathrm{N}_{x_\mathrm{i}, x_j}}{N_{x_i} N_{x_j}} \right]$$

This is the (empirical) **mutual information**, usually denoted $I(x_i; x_j)$

Chow-Liu Trees

- To maximize the log-likelihood, it then suffices to choose the tree T that maximizes

$$\max_T \sum_{i,j} I(x_i; x_j)$$

- This problem can be solved by finding the maximum weight spanning tree in the complete graph with edge weight w_{ij} given by the mutual information over the edge (i, j)
  - Greedy algorithm works: at each step, pick the largest remaining edge that does not form a cycle when added to the already selected edges

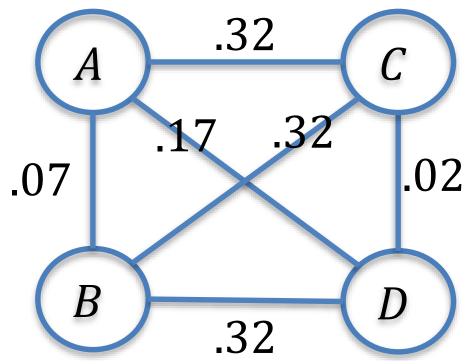
Chow-Liu Trees

- To use this technique for learning, we simply compute the mutual information for each edge using the empirical probability distributions and then find the max-weight spanning tree

- As a result, we can learn tree-structured BNs in polynomial time

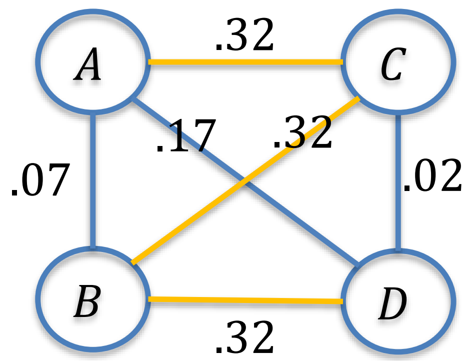
  – Can we generalize this to all DAGs?

Chow-Liu Trees: Example



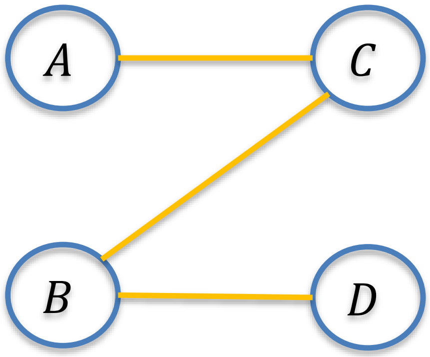
- Edge weights correspond to empirical mutual information for the earlier samples

Chow-Liu Trees: Example



- Edge weights correspond to empirical mutual information for the earlier samples

Chow-Liu Trees: Example



- Any directed tree (with one parent per node) over these edges maximizes the log-likelihood
  - Why doesn't the direction matter?