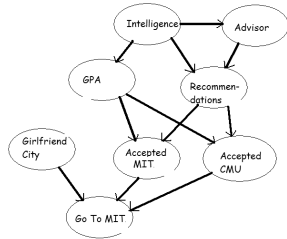


Extensions of Bayesian Networks

Ethan Howe,
James Lenfestey,
Tom Temple



Outline

- Intro to Dynamic Bayesian Nets (Tom)
- Exact inference in DBNs with demo (Ethan)
- Approximate inference and learning (Tom)
- Probabilistic Relational Models (James)

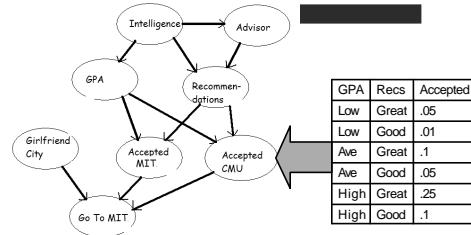
Reasoning under Uncertainty

- How do we use prior knowledge and new observations to judge what is likely and what is not?

$$P(s \mid \text{observations, prior knowledge})$$

- But that is a very large joint distribution

Bayesian Network



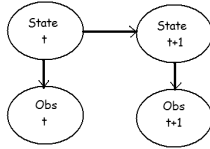
Bayesian Network

- A Bayesian Network is a Directed Acyclic Graph (DAG) with variables as nodes.
- Edges go from parent to child such that
 - Each child, x , has a conditional probability table $P(x \mid \text{parents}(x))$ that defines the affects that x feels from its parents.
- Intuitively, these edges codify *direct relationships* between nodes in a cause-effect manner.
- The lack of an edge between nodes implies that they are conditionally independent.

Features of Bayesian Networks

- Arbitrarily descriptive; allows encapsulation of all the available prior knowledge
- The model makes no distinction between observed variables and inferred variables
- The DAG restriction is somewhat limiting

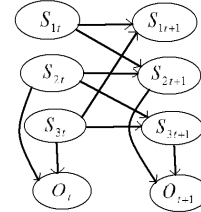
HMM as a Bayes Net



	S1	S2	S3	S4	...
S1	.8	.2	.2	.01	...
S2	.01	.5	.1	.02	...
S3	.05	.1	.6	.01	...
S4	.01	.01	.01	.9	...
...

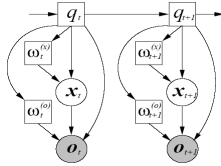
Dynamic Bayesian Networks

- DBNs are BNs with a temporal regularity analogous to the HMM
- The full representation of a DBN grows with time.



Hybrid State in DBNs

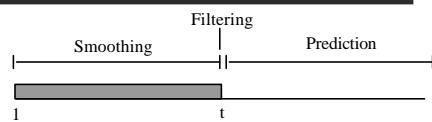
- For basis functions, parent variables are used as function parameters
- Usually the function is a mixture of Gaussians with known means and variances and the parent variables decides the mixture weights.



Exact Inference in Bayesian Networks

- Queries of the network are fielded by “summing out” all of the non-query variables
- A variable is summed out by collecting all of the factors that contain it into a new factor. Then sum over all of the possible states of the variable to be eliminated.

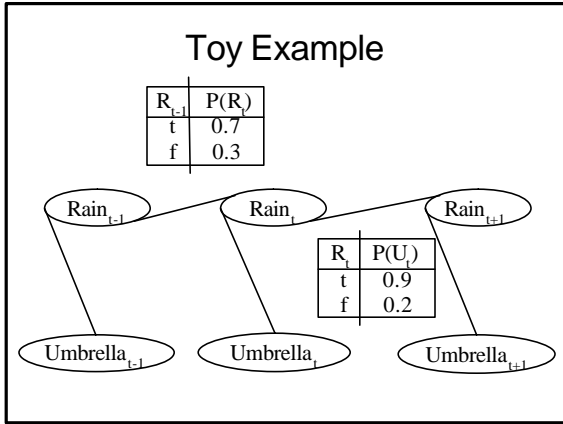
Temporal Inference



- Filtering
 - Probabilities of current states
- Prediction
 - Probabilities of future states
- Smoothing
 - Probabilities of past states

Notation

- X_{t+1} : set of hidden variables in the t+1 time slice
- x_{t+1} : set of values for those hidden variables at t+1
- e_{t+1} : set of evidence at time t+1
- $e_{1:t}$: set of evidence from all times from 1 to t
- a : normalization constant



Markov Assumptions

- First-order Markov process
 - $P(X_t | X_{0:t-1}) = P(X_t | X_{t-1})$
- Markov assumption of evidence
 - $P(E_t | X_{0:t}, E_{0:t-1}) = P(E_t | X_t)$

Prediction: Separating Time Slices

$$P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1}) \text{ (divide evidence)}$$

$$= a P(e_{t+1} | X_{t+1}, e_{1:t}) P(X_{t+1} | e_{1:t}) \text{ (Bayes' rule)}$$

$$= a P(e_{t+1} | X_{t+1}) P(X_{t+1} | e_{1:t}) \text{ (Markov property of evidence)}$$

Predict Next Time Slice from Current

$$P(X_{t+1} | e_{1:t+1})$$

$$= a P(e_{t+1} | X_{t+1}) \sum_x P(X_{t+1} | x_t, e_{1:t}) P(x_t | e_{1:t})$$

$$= a P(e_{t+1} | X_{t+1}) \sum_x P(X_{t+1} | x_t) P(x_t | e_{1:t})$$

Example: Umbrella Seen on Day 1

$$P(R_0) = \{0.5, 0.5\}$$

$$P(R_1 | u_1) = a P(u_1 | R_1) \sum P(R_1 | r_0) P(r_0)$$

$$= a \{0.9, 0.2\} (0.5 \{0.7, 0.3\} + 0.5 \{0.3, 0.7\})$$

$$= a \{0.45, 0.1\}$$

$$= \{0.818, 0.182\}$$

Example: Umbrella Seen Again on Day 2

$$P(R_2 | u_1) = \sum P(R_2 | r_1) P(r_1 | u_1)$$

$$P(R_2 | u_1, u_2) \stackrel{a}{=} a P(u_2 | R_2) P(R_2 | u_1)$$

$$= a \{0.9, 0.2\} (0.818 \{0.7, 0.3\} + 0.182 \{0.3, 0.7\})$$

$$= a \{0.565, 0.075\} = \{0.883, 0.117\}$$

Encapsulate Prediction

- Repeating actions at each step of time
- Formalize as procedure
 - $P(\mathbf{X}_{t+1} | \mathbf{e}_{1:t+1}) = \text{FORWARD}(\text{previous state set, new evidence set})$
- **Only need to pass message from previous time step**
 - $f_{1:t+1} = a \text{ FORWARD}(f_{1:t}, e_{t+1})$
- Can forget about previous times

Smoothing: Break into data before and after

$$\begin{aligned}
 P(\mathbf{X}_k | \mathbf{e}_{1:t}) &= P(\mathbf{X}_k | \mathbf{e}_{1:k}, \mathbf{e}_{k+1:t}) \quad (\text{divide evidence}) \\
 &= a P(\mathbf{X}_k | \mathbf{e}_{1:k}) P(\mathbf{e}_{k+1:t} | \mathbf{X}_k, \mathbf{e}_{1:k}) \quad (\text{Bayes' rule}) \\
 &= a P(\mathbf{X}_k | \mathbf{e}_{1:k}) P(\mathbf{e}_{k+1:t} | \mathbf{X}_k) \quad (\text{Markov Indep.}) \\
 &= a \mathbf{f}_{1:k} \mathbf{b}_{k+1:t}
 \end{aligned}$$

Backwards message

$$\begin{aligned}
 P(\mathbf{X}_{k+1:t} | \mathbf{e}_k) \\
 &= \sum_{\mathbf{X}_{k+1}} P(\mathbf{e}_{k+1:t} | \mathbf{X}_k, \mathbf{x}_{k+1}) P(\mathbf{x}_{k+1} | \mathbf{X}_k) \quad (\text{cond. on } \mathbf{X}_{k+1}) \\
 &= \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1:t} | \mathbf{x}_{k+1}) P(\mathbf{x}_{k+1} | \mathbf{X}_k) \quad (\text{Markov indep.}) \\
 &= \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1}, \mathbf{e}_{k+2:t} | \mathbf{x}_{k+1}) P(\mathbf{x}_{k+1} | \mathbf{X}_k) \\
 &= \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1} | \mathbf{x}_{k+1}) P(\mathbf{e}_{k+2:t} | \mathbf{x}_{k+1}) P(\mathbf{x}_{k+1} | \mathbf{X}_k)
 \end{aligned}$$

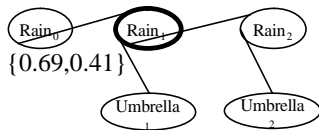
Backwards procedure

- Repeating actions at each step of time
- Formalize as procedure
 - $P(\mathbf{e}_{k+1:t} | \mathbf{X}_k) = \text{BACKWARD}(\text{next state set, future evidence set})$
- **Message passing goes backwards in time**
 - $\mathbf{b}_{k+1:t} = \text{BACKWARD}(\mathbf{b}_{k+2:t}, \mathbf{e}_{k+1:t})$

Example: Was it really raining day 1?

$$\begin{aligned}
 P(u_2 | R_1) &= \sum_{r_2} P(u_2 | r_2) P(u_{3,2} | r_2) P(r_2 | R_1) \\
 &= (0.9 * 1 * \{0.7, 0.3\}) + (0.2 * 1 * \{0.3, 0.7\}) \\
 &= \{0.69, 0.41\}
 \end{aligned}$$

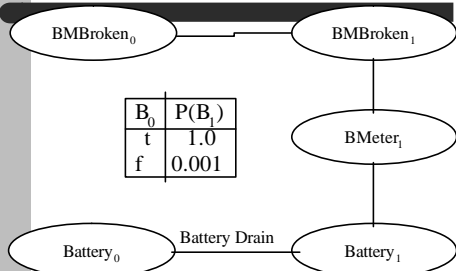
$$\begin{aligned}
 P(R_1 | u_{1:2}) &= a \mathbf{f}_{1:1} \mathbf{b}_{2:2} \\
 &= a \{0.818, 0.182\} \{0.69, 0.41\} \\
 &= \{0.883, 0.117\}
 \end{aligned}$$



Forward-Backwards Algorithm

- Save FORWARD values at each time step
- Each time new data arrives, recompute BACKWARD values.
- Time Complexity $O(t)$
- Space Complexity $O(t)$
- Both can be bounded if only care about last k slices.

Demo: Battery Meter Model



Exact Inference in DBNs

- There are some additional difficulties with exact inference in DBNs
 - The factors grow to include all the states which sharply decreases the advantage over HMMs
 - Constant memory requirement

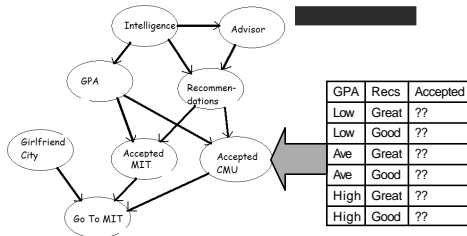
Approximate Inference

- While a number of inference algorithms exist for the standard Bayes Net, few of them adapt well to the DBN.
- One that does adapt is Particle filtering, due to its ingenious use of resampling
 - Particle filtering deals well with hybrid state
- Sampling has trouble with unlikely events

Learning

- The three components of a BN
 - the probabilities
 - the structure
 - the variables
 can all be learn automatically, though with varying degrees of complexity

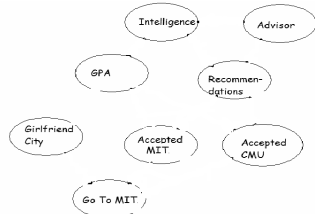
Probability learning



How do we generate the Network?

- The network contains variables, edges and probabilities all of which need to be known ahead of time.
- Given all the variables and their relationships, it is relatively easy to estimate the probabilities from sample data, even if that sample data does not include every variable.

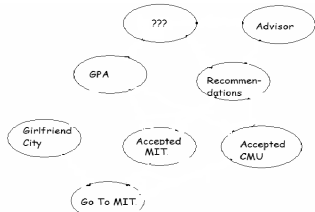
Structure Learning



Structure Learning

- Given the variables only, we can guess a structure, learn parameters for that structure and then rate it based on how well it “predicts” the data.
- Care must be taken not to overfit.
- Search the structure space with greedy-ascent on this rating
- Randomization helps avoid local maxima

Hidden Variable Learning

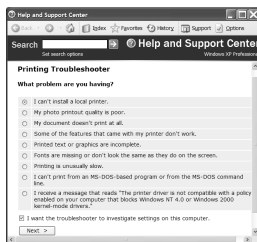


Hidden Variable Learning

- Given a subset of the variables, generate new variables, structure and parameters.
- Greedy ascent on the output of Structure learning.
- Some risk of overfit

Applications

- Windows printer troubleshooter and goddamn paper-clip



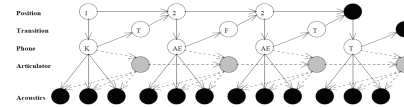
More exciting applications

- Speech and Vision recognition
 - Fewer parameters than traditional HMMs mean better learning
 - Allows for easier experimentation with intuitive (i.e. hand coded) hierarchical models

In Depth Training

- Since the DBN model doesn't make a distinction between observed variables and hidden variables, it can learn a model with access to data that we don't have during prediction.
- For example, in Speech Recognition,
 - We know that sounds are made by the pose of the lips mouth and tongue
 - While training we can measure the pose

Isolated Word Recognition



	WER	# Param.
Acoustics Only (baseline)	9.8%	31488
2 Discrete Articulatory Values	8.5%	62976
4 Discrete Articulatory Values	7.7%	126690
8 Discrete Articulatory Values	8.4%	257070

Recall a few important points

- Bayesian networks are an arbitrarily expressive model for joint probability distributions
- DBNs can be more compact than HMMs and therefore easier to learn and faster to use
- Dynamic Bayesian networks allow for reasoning in a temporal model
 - Tolerant of missing data, likewise able to exploit bonus data
 - Computationally, the compactness advantage is often lost in exact reasoning
- Particle Filters are an alternative for exploiting this compactness as long as the important probabilities are large enough to be sampled from.