Lecture Notes for 6.824, Artificial Itelligence
 October, 1993

# Bayesian Networks

## 1 Introduction

Common sense reasoning is usually uncertain. If you see someone walking into a lecture with wet hair you might conclude that it is raining outside. This conclusion could, of course, be wrong. If you see a man and a woman at a playground with a small child you might conclude that these people are the child's parents and that they are married. You could, of course, be wrong. If you discover that there is no cereal in your cabinet, and you know you just put some there yesterday, you might conclude that your house mates ate it. You could be wrong. Although many AI researchers prefer other nonprobabilistic formalisms,[1] it is possible to construct and use probabilistic approaches to common sense reasoning. The advantages of the probabilistic approaches to common sense are described quite well in [Pearl, 1988].

As pointed out in the notes on common sense inference, to use probability theory as a foundation for common sense reasoning one must either reason from incomplete knowledge about probabilities (as in probabilistic logic) or find some way of representing a complete probability distribution over possible world histories. Here we introduce Bayesian networks as a instance of the second approach — Bayesian networks provide a complete probability distribution over world histories. Bayesian networks provide a formal model of some forms of common sense reasoning and, perhaps more importantly, also provide a formalism for reasoning about probabilities in a variety of applications such as speech recognition and medical diagnosis.

A Bayesian network models the causal structure of a nondeterministic

[1] For references to nonprobabilistic formalisms see the notes on general Bayesian inference.

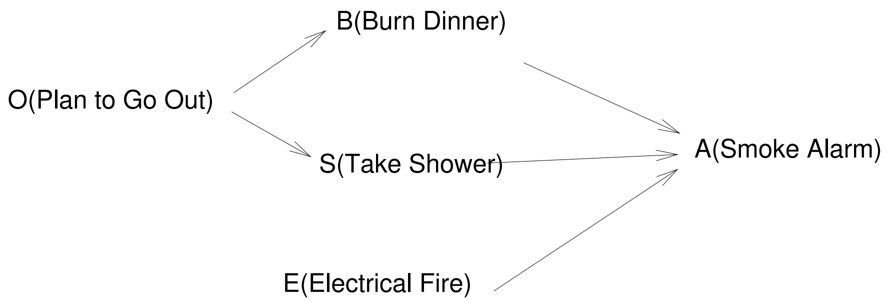


Figure 1: A Simple Bayesian Network

process. A Bayesian network is a directed graph — it consists of nodes and directed edges. The nodes can be thought of as representations of events and the edges can be viewed as describing causal relationships between events. For example, consider an apartment with a smoke detector. The smoke detector is near the bathroom and taking a shower often triggers the smoke detector (many smoke detectors detect steam). We can represent events and causal relationships between events in a direct graph as shown in figure 1.

The smoke alarm is likely to go off if someone burns dinner, takes a shower or there is an electrical fire. Each node in this network represents a Boolean variable which is true if a certain event takes place. The arrows are not intended to represent logical implication but rather to express influences — the smoke detector does not go off every time someone takes a shower. The above diagram expresses certain probabilistic relationships between the represented events. In particular it says that whether we have an electrical fire is independent of whether we take a shower. Bayesian networks of this form allow for some simple common sense reasoning. Let $O$ represent the statement that John plans to go out for the evening. Let $B$ represent the statement that John burns dinner. Let $S$ represent the statement that John takes a shower. Let $E$ represent the statement that there is an electrical fire, and let $A$ represent the statement that the smoke detector alarm goes off. Each of these is a random variable that can be either true or false. There is some probability distribution over the simultaneous assignments of values to these variables. The Bayesian network given above places constraints on this distribution. More precisely, the above network implies that truth values are assigned to these variables in a left to right order with the probability that a variable $Q$ is true being influenced only by the truth values of the variables at the origin of arcs leading into $Q$. For example, In the above diagram we have that $O$ is independent of $E$. This means that $P_r(E \wedge O) = P_r(E)P_r(O)$. However, the above diagram does not imply that $B$ is independent of $S$. If all we know is that $B$ is true then John must have decided to make dinner which means that it is likely that John did not plan to go out which means that it is unlikely that John decided to take a shower.

Many formalisms for uncertain reasoning have used statements about local conditional probabilities without being overly concerned with causality. Let $Q \leadsto R$ represent the statement that $P_r(R|Q) > P_r(R)$. Under natural probabilities for the Bayesian network described above we have $S \leadsto A$, i.e., taking a showing increases the likelihood that the alarm goes off. In this case the influence arc $S \leadsto A$ is in the same direction as the causal relationship — $S$ can cause $A$. But under natural probabilities we also have $A \leadsto E$. If all we now is that the alarm went off then the likelihood of an electrical fire is larger than if we did not know that the alarm went off. In this case the arc $A \leadsto E$ is in the opposite direction of the causal relationship — because $E$ causes $A$, $A$ provides evidence that $E$ occurred. Because influence arcs such as these do not incorporate causal information they can lead to incorrect inferences if used naively. For example, one might expect that from $S \leadsto A$ and $A \leadsto E$ we would have $S \leadsto E$, but this is clearly not the case. The Bayesian network formalism allows the correct inference. Given $A$ alone the probability of $E$ is increased. However, if we then learn $S$ we have an alternative explanation for $A$ so the the additional knowledge of $S$ reduces the likelihood of $E$. Bayesian networks provide more information than simple influence arcs.

## 2 The Bayesian Network Formalism

We are now ready to formally define the concept of a Bayesian network. A Bayesian network consists of a directed acyclic graph where each node in the graph is a random variable and where each node $x$ is associated with probabilistic information specifying how the value $x$ is to be nondeterministically derived from the values of the nodes on the arcs coming into $x$.

**Definition:** A *Bayesian network* is a directed acyclic graph (a DAG) where each node in the DAG is a random variable $X$ with a finite set of possible values $\{x_1, \ldots, x_n\}$ and each node $X$ is associated probability matrix $M^X$ which gives the probability that $X = x$ for each value possible value $X$ and each assignment of values to the variables at the origin of arcs leading to $X$. We use the notation $M^X_{x|y_1, \ldots, y_k}$ to represent the entry in the matrix $M^X$ giving the probability $P_r(X = x|Y_1 = y_1 \wedge \ldots \wedge Y_k = y_k)$ where $Y_1, \ldots, Y_k$ are all the nodes at the origin of arcs coming into the node $X$.

For example in the above Bayesian network the node $B$ is associated with a four element matrix $M^B$ where $M^B_{T|F}$ is $P_r(B = T|O = F)$ which we could also write as $P_r(B|\neg O)$. There is one entry in this matrix for each of the four possible assignments of true values to $O$ and $B$. The matrix $M^A$ has sixteen entries giving the probabilities for $A$ and $\neg A$ for each of the the possible truth assignments to the antecedent variables $B$, $S$, and $E$. The matrix $M^O$ contains two entries, one giving the prior probability of $O$ and one giving the prior probability of $\neg O$.

Note that $M^B_{T|T} + M^B_{F|T} = 1$, which could also be written as $P_r(B|O) + P_r(\neg B|O) = 1$. This means that we really need only store an entry for $P_r(B|O)$. For Boolean nodes this reduces the storage requirements by a factor of two. If there are $n$ possible values for each variable and the variable $X$ has $k$ antecedent variables then we need $n^k(n-1)$ entries to represent the matrix $M^X$.

# 3 Hidden Markov Processes as Bayesian Networks

A Bayesian network representing a run of $T$ steps of hidden Markov process is shown in figure 2. We let $q_1, \ldots, q_N$ be the internal states of the Markov model and $o_1, \ldots o_K$ be the alphabet of externally observable symbols. $S_t$ is the internal state at time $t$ and $O_t$ is the observed external symbol at time

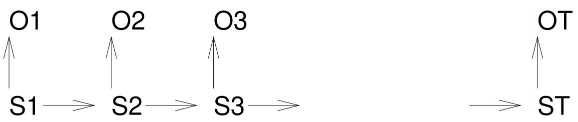


Figure 2: The Bayesian network for a hidden Markov process

t. A hidden Markov model is a triple $<\pi,\ a,\ b>$ where $\pi$ is a vector of $N$ numbers; $a$ is an $N \times N$ matrix; and $b$ is $N \times K$ matrix with the following interpretations.

$$\pi_i = P_r(S_1 = q_i)$$
$$a_{i,j} = P_r(S_{t+1} = q_j | S_t = q_i)$$
$$b_j(k) = P_r(O_t = o_k | S_t = q_j)$$

Each node in a Bayesian network represents a random variable with an associated matrix governing the generation of values for that node. In representing a hidden Markov process as a Bayesian network we must associate each node with the appropriate matrix. For example, the matrix $M^{S_t}$ should specify the probabilities of the values for $S_t$ given the values of the "predecessors" of $S_t$, i.e., those nodes at the origin of arcs leading to $S_t$. In the case of a hidden Markov process we have the following matrices.

$$M^{S_1}_{q_i} = \pi_i$$
$$M^{S_{t+1}}_{q_j|q_i} = a_{i,j}$$
$$M^{O_t}_{s_k|q_j} = b_j(k)$$

# 4 The Forward-Backward Procedure for Bayesian Trees

This section generalizes the forward-backward procedure used to compute conditional probabilities in hidden Markov processes to a procedure that can

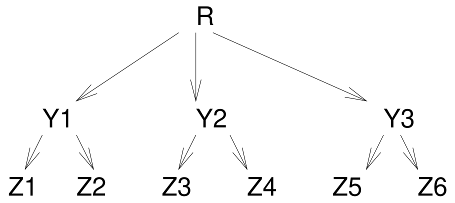


Figure 3: A tree structured Bayesian network

be applied to any tree structured Bayesian network — trees in which every arc is from a parent node to a child node and every node has at most one parent. A run of a hidden Markov process is a tree structured Bayesian network (see figure 2). A somewhat different tree structured Bayesian network is shown in figure 3. In the case of hidden Markov models our knowledge about the process consists of an assignment of values to the observation nodes $O_t$. The forward-backward procedure can be used to compute the probability of being in a particular internal state at a particular time given the complete sequence of observed external symbols. In the general case of a tree structured Bayesian network we are interested in computing the probabilities of values for nodes given values for certain other nodes in the network. For example, for the Bayesian tree shown in figure 3 we might have observations for the variables $Z_1$, $Y_2$ and $Z_6$. Figure 4 highlights the observed nodes by placing circles around them. We are interested in constructing an algorithm for computing the probabilities for values of unobserved nodes given values for other observed nodes. It turns out that we can assume without loss of generality that all observations occur at leaf nodes. For example, to model the observed value of $Y_2$ we add a leaf node $Y_2'$ such that the probability matrix $M^{Y_2'}$ is such that $Y_2'$ is deterministically assigned the same value as $Y_2$, i.e., $M^{Y_2'}_{x|y}$ is 1 if $x = y$ and 0 otherwise. Figure 5 shows the result of introducing a new leaf node to model the observed value of the nonleaf node $Y_2$.

Consider a Bayesian tree and let $O$ be an assignment of observed values to some (but not all) of the leaf nodes in that tree. We can think of $O$ as a body of "evidence" about a particular run of the Bayesian process. Formally, $O$ is a partial mapping from nodes to values. Let $X$ be a node of the tree. Each node $X$ divides the observations $O$ into to disjoint sets, denoted $O_{\geq X}$ and

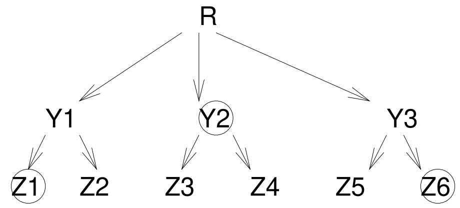


Figure 4: Observations in a Bayesian network

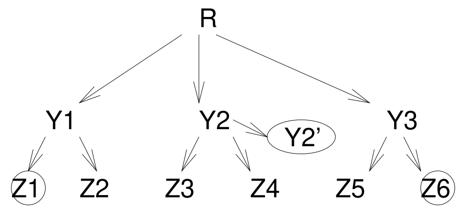


Figure 5: Moving observations to leaf nodes

$O_{<X}$. The observation set $O_{\geq X}$ is the restriction of the partial assignment $O$ to those nodes that are equal to $X$ or connected to $X$ via children of $X$. In the example shown in figure 5 we have that $O_{\geq Y_1}$ is the restriction of $O$ to $Z_1$; $O_{\geq Z_6}$ is the restriction of $O$ to $Z_6$; and $O_{\geq Z_2}$ is the empty assignment. In general $O_{\geq X}$ is a subset of the evidence $O$. The empty assignment (the lack of any evidence) is treated as the universally true proposition. Hence, for the example shown in figure 5, we have $P_r(O_{\geq Z_3}|\Phi) = 1$ and $P_r(\Phi|O_{\geq Z_3}) = P_r(\Phi)$. The observation set $O_{<X}$ is the restriction of the partial assignment $O$ to those nodes that are connected to $X$ via the parent of $X$. In the example shown in figure 5 we have that $O_{<Y_1}$ is the assignment of the observed values to $Y_2'$ and $Z_6$. We also have that $O_{<Y_2}$ is the restriction of $O$ to $Z_1$ and $Z_6$.

We now define the forward and backward numbers of the froward-backward procedure for Bayesian trees. For any node $X$ and possible value $x$ for $X$ we define the forward number $\alpha_X(x)$ and the backward number $\beta_X(x)$ as follows.

$$\alpha_X(x) \equiv P_r(O_{<X} \wedge X = x)$$

$$\beta_X(x) \equiv P_r(O_{\geq X}|X = x)$$

Note that the total observed data $O$ is the union of $O_{\geq X}$ and $O_{<X}$. Also note that once the value of $X$ is known, the observations $O_{<X}$ and $O_{\geq X}$ are independent. So we have the following formula for $P_r(X = x|O)$.

$$P_r(X = x|O) = \frac{P_r(O_{<X} \wedge X = x)P_r(O_{\geq X}|X = x)}{P_r(O)} = \frac{\alpha_X(x)\beta_X(x)}{P_r(O)}$$

The term $P_r(O)$ can be viewed as a normalizing constant in the above equation — it is simply the sum over all possible values of $x$ of $\alpha_X(x)\beta_X(x)$. Given the goal of computing probabilities of the form $P_r(X = x|O)$ we now see that it is sufficient to compute the $\alpha$ and $\beta$ numbers for all the nodes in the network. In a tree structured network the $\beta$ numbers can be computed from the leaves of the tree up. Given the $\beta$ numbers, the $\alpha$ numbers can be computed from the root of the tree down. We use the notation $O(X) = \perp$ to indicate that $X$ is not one of the nodes whose value has been observed. We use the notation $O(X) = x$ to indicate that the node $X$ is observed to have the value $x$. The definition of $\beta_X(x)$ implies the following equalities in the case where $X$ is a leaf node.

$$\beta_X(x) = \begin{cases} 1 & \text{if } X \text{ is a leaf node and } O(X) = \perp \\ 1 & \text{if } X \text{ is a leaf node and } O(X) = x \\ 0 & \text{if } X \text{ is a leaf node, } O(X) \neq \perp \text{ and } O(X) \neq x \end{cases}$$

Figure 6 shows a nonleaf node with its children and the observations below those children. To compute $\beta_X(x)$ for internal (nonleaf) nodes we treat the children of $X$ independently. Consider the arc $X \rightarrow Y$. For each such arc in the network we define $\beta_{X \rightarrow Y}(x)$ as follows.

$$\beta_{X \rightarrow Y}(x) \equiv P_r(O_{\geq Y}|X = x)$$

We can compute $\beta_{X \rightarrow Y}(x)$ using the following identity.

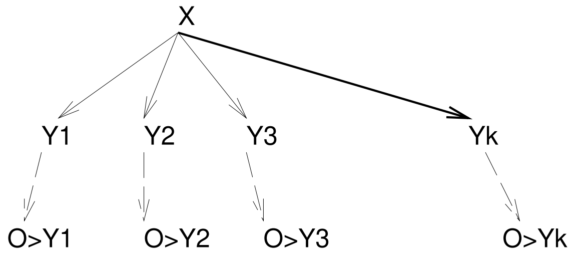


Figure 6: Computing $\beta_X(x)$

$$\beta_{X \to Y}(x) = \sum_y P_r(Y = y | X = x) P_r(O_{\geq Y} | Y = y) = \sum_y M^Y_{y|x} \beta_Y(y)$$

The number $\beta_{X \to Y}(x)$ expresses a part of $\beta_X(x)$ — that part due to the child $Y$. Since the children behave independently we can now write an expression for $\beta_X(x)$ as the product over all children $Y$ of the probability that the data below $Y$ is generated given that $X = x$. We let $C(X)$ be all the children nodes of $X$.

$$\beta_X(x) = \prod_{Y \in C(x)} \beta_{X \to Y}(x)$$

The above equations allow the $\beta$ numbers to be computed starting at the leaves of the tree and moving up toward the root. The reader might want to compare this "backward" computation with the backward phase of the forward-backward procedure used for hidden Markov models.

We now give a way of computing $\alpha_X(x)$ from the root of the tree down using the $\beta$ numbers that are computed from the bottom of the tree up. We are again assuming that all observed nodes are leaf nodes. First, if $X$ is the root node then we have the following.

$$\text{If X is the root node} \quad \alpha_X(x) = M^X_x$$

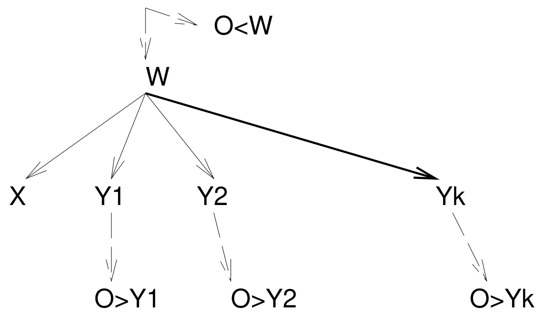


Figure 7: Computing $\alpha_X(x)$

Now suppose that $X$ is a node other than the root node, let $W$ be the parent of $X$, and let $S(X)$ be the set of sibling nodes of $X$, i.e., those children of $W$ other than $X$. This situation is shown in figure 7. The observations $O_{<X}$ can be expressed as $O_{<W} \cup O_{\geq Y_1} \cup \ldots \cup O_{\geq Y_k}$ where $Y_1, \ldots, Y_k$ are the members of $S(X)$. Figure 7 shows the various collections of observations that make up $O_{<X}$. We now have the following.

$$\alpha_X(x) = P_r(O_{<X} \wedge X = x) = \sum_w \alpha_W(w) M^X_{x|w} \prod_{Y \in S(X)} \beta_{W \rightarrow Y}(w)$$

The product over an empty set is by convention 1, so if $X$ has no siblings then $\alpha_X(x) = \sum_w \alpha_W(w) M^X_{x|w}$ where $W$ is the parent of $X$.

# References

[Pearl, 1988] Judea Pearl. *Probilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.