

Bayesian Networks and Markov Random Fields

1 Bayesian Networks

As before, we will use capital letters for random variables and lower case letters for values of those variables. A Bayesian network is a triple $\langle V, G, \mathcal{P} \rangle$ where V is a set of random variables X_1, \dots, X_n , G is a directed acyclic graph (DAG) whose nodes are the variables in V , and \mathcal{P} is a set of conditional probability tables as described below. The conditional probability tables determine a probability distribution over the values of the variables. If there is a directed edge in G from X_j to X_i then we will say that X_j is a parent of X_i and X_i is a child of X_j . To determine values for the variables one first selects values for variables that have no parents and then repeatedly picks a value for any node all of whose parents already have values. When we pick a value for a variable we look only at the values of the parents of that variable. We will write $P(x_i \mid \text{parents of } x_i)$ to abbreviate $P(x \mid x_{i_1}, \dots, x_{i_k})$ where x_{i_1}, \dots, x_{i_k} are the parents of x . For example, if x_7 has parents x_2 and x_4 then $P(x_7 \mid \text{parents of } x_7)$ abbreviates $P(x_7 \mid x_2, x_4)$. Formally, the probability distribution on the variables of a Bayesian network is determined by the following equation.

$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i \mid \text{parents of } x_i) \quad (1)$$

The conditional probabilities of the form $P(x_i \mid \text{parents of } x_i)$ are called conditional probability tables (CPTs). Suppose that each of the variables x_i has d possible values (this is not required in general). In this case if a variable x has k parents then $P(x \mid \text{parents of } x)$ has d^{k+1} values (with $(d-1)d^k$ degrees of freedom). These d^{k+1} values can be stored in a table with $k+1$ indices. Hence the term “table”. Note that the number of indices of the CPTs (conditional probability tables) is different for the different variables.

Note that an HMM is a Bayesian network with a variable for each hidden state and each observable token.

Bayesian networks are often used in medical diagnosis where variables represent the presence or absence of certain diseases or the presence or absence of certain measurable quantities such as blood sugar or presence of a certain protein in the blood.

In a Bayesian network the edges of the directed graph are often interpreted as “causation” with the parent causally influencing the child and parents getting assigned values temporally before children.

We are interested in “Bayesian inference” which means, intuitively, inferring

causes by observing their effects using Bayes' rule. In an HMM for example, we want to infer the hidden states from the observations that they cause.

In general we can formulate the inference problem as the problem of determining the probability of an unknown variable (a hidden cause) from observed values of other variables. In general we can consider the variables in any order.

$$P(x_5, x_7 \mid x_3, x_2) = \frac{P(x_5, x_7, x_2, x_3)}{P(x_2, x_3)}$$

So in general, for inference it suffices to be able to compute probabilities of the form $P(x_{i_1}, \dots, x_{i_k})$. We give an algorithm for doing this in section 3.

2 Inference in Bayesian Network is #P hard

A Boolean variable (also called Bernoulli variable) is a variable that has only the two possible values of 0 or 1. A disjunctive clause is a disjunction of literals where each literal is either a Boolean variable or the negation of a Boolean variable. For example we have that $(X_5 \vee \neg X_2 \vee X_3)$ is a clause with three literals. A 3SAT problem is a set of clauses with three literals in each clause. It is hard (in fact #P hard) to take a 3SAT problem and determine the number of complete assignments of values to variables that satisfy the clauses, i.e., that make every clause true.

Take X_1, \dots, X_n be independent Boolean (Bernoulli) variables with $P(X_i = 1) = 1/2$. Let Σ be a set of clauses over these variables where each clause has only three literals. Let C_j be a random variable which is 1 if the j th clause is satisfied. Since we can compute c_j from x_1, \dots, x_n using a (deterministic) conditional probability table having only three parents. Let A_j be a Boolean variable that is true if all of C_1, \dots, C_j are true. a_1 can be computed with a CPT from c_1 and for $j > 1$ we have that a_j can be computed using a CPT from a_{j-1} and c_j .

Now we have that $P(A_k = 1)$ is proportional to the number of truth assignments that satisfying all the clauses. This implies that computing the probability of a partial assignment in a Bayesian network is #P hard. It is widely believed that there are no polynomial time algorithms for #P hard problems.

3 Recursive Conditioning

Although inference in Bayesian networks is hard in general, there exist algorithms that work well when the Bayesian network has special structure.

- $\mathcal{D}(X)$ is the set of values that variable X can have.
- ρ ranges over partial assignments of values to variables — ρ assigns values to *some* of the variables in V .
- $\text{dom}(\rho)$ is the set of variables that are assigned values by ρ . For $X \in \text{dom}(\rho)$ we have that $\rho(X) \in \mathcal{D}(X)$.
- For $X \notin \text{dom}(\rho)$, and $x \in \mathcal{D}(X)$ we let $\rho[X := x]$ be the extension of ρ that assigns X the value x .
- σ ranges over total assignments of values to the variables in V . For $X \in V$ we have $\sigma(X) \in \mathcal{D}(X)$.
- $\sigma \sqsubseteq \rho$ means that the complete assignment σ is compatible with the partial assignment ρ . In other words, for $X \in \text{dom}(\rho)$ we have $\sigma(X) = \rho(X)$.

We can now write equation (1) as follows where T_i is the conditional probability table for the i th variable and $T_i(\sigma)$ is $P(x_i \mid \text{parents of } x_i)$ where the variable values are determined by σ .

$$P(\sigma) = \prod_i T_i(\sigma) \tag{2}$$

For a partial assignment ρ we have the following.

$$P(\rho) = \sum_{\sigma \sqsubseteq \rho} \prod_i T_i(\sigma) \tag{3}$$

Note that $T_i(\sigma)$ depends on only some of the variables assigned by σ . For example suppose we have seven variables X_1, \dots, X_7 arranged in an “inverted tree” where variable X_1 has parents X_2 and X_3 ; variable X_2 has parents X_4 and X_5 ; and variable X_3 has parents X_6 and X_7 . Now suppose that $\text{dom}(\rho) = \{X_1, X_2\}$, i.e., ρ assigns a value only to X_1 and X_2 . Then we can write equation (3) as follows.

$$\begin{aligned}
P(x_1, x_2) &= \sum_{x_3, x_4, x_5, x_6, x_7} \frac{[T_2(x_2, x_4, x_5)T_4(x_4)T_5(x_5)]}{[T_1(x_1, x_2, x_3), T_3(x_3, x_6, x_7)T_6(x_6)T_7(x_7)]} \\
&= \frac{\left[\sum_{x_4, x_5} T_2(x_2, x_4, x_5)T_4(x_4)T_5(x_5) \right]}{\left[\sum_{x_3, x_6, x_7} T_1(x_1, x_2, x_3)T_3(x_3, x_6, x_7)T_6(x_6)T_7(x_7) \right]} \\
&= \tilde{P}(\langle x_2 \rangle, \langle T_2, T_3, T_4 \rangle) \tilde{P}(\langle x_1, x_2 \rangle, \langle T_1, T_3, T_6, T_7 \rangle) \\
&\text{where} \\
\tilde{P}(\rho, \mathcal{T}) &= \sum_{\substack{\sigma \sqsubseteq \rho \\ \sigma \text{ only assigns to variables in } \mathcal{T}}} \Pi_{T \in \mathcal{T}} T(\sigma)
\end{aligned}$$

So we can summarize this as follows.

$$\begin{aligned}
P(x_1, x_2) &= \tilde{P}(\langle x_1, x_2 \rangle, \langle T_1, T_2, T_3, T_4, T_5, T_6, T_7 \rangle) \\
&= \tilde{P}(\langle x_2 \rangle, \langle T_2, T_3, T_4 \rangle) \tilde{P}(\langle x_1, x_2 \rangle, \langle T_1, T_3, T_6, T_7 \rangle)
\end{aligned}$$

To compute probabilities of the form $P(\rho)$ we can compute the quantities $\tilde{P}(\rho, \mathcal{T})$ where these quantities often factor. Recursive conditioning is defined by the following equation where $Y \notin \text{dom}(\rho)$.

$$\tilde{P}(\rho, \mathcal{T}) = \sum_{y \in \mathcal{D}(Y)} \tilde{P}(\rho_1[Y := y], \mathcal{T}_1) \cdots \tilde{P}(\rho_k[Y := y], \mathcal{T}_k)$$

For $i \neq j$ we must have that no variable $Z \notin \text{dom}(\rho) \cup \{Y\}$ appears in both \mathcal{T}_i and \mathcal{T}_j . Also, we require that $\rho_i[Y = y]$ is the restriction of $\rho[Y = y]$ to the variables occurring \mathcal{T}_i . See the above example. To make this algorithm efficient the computations of values for expressions of the form $\tilde{P}(\rho, \mathcal{T})$ must be memoized, i.e., stored in a table so that values can be reused if they are needed again. The choice of the variable $Y \notin \text{dom}(\rho)$ is important for the efficiency of the algorithm.

4 Markov Random Fields

A Bayesian network is a triple $\langle V, \mathcal{G} \rangle$ where V is a set of random variables X_1, \dots, X_n and \mathcal{G} is a set of functions such that for any total assignment σ to the variables in V , and for $\Gamma \in \mathcal{G}$, we have that $\Gamma(\sigma)$ is a non-negative real number. (Other authors require $\Gamma(\sigma) > 0$ but we will not require that here).

A Markov random field determines a probability distribution on assignments defined by the following equation.

$$P(\sigma) = \frac{1}{Z} \prod_{\Gamma \in \mathcal{G}} \Gamma(\sigma) \quad (4)$$

$$Z = \sum_{\sigma} \prod_{\Gamma \in \mathcal{G}} \Gamma(\sigma) \quad (5)$$

It is interesting to compare equation (4) to equation (2). One can see that a Bayesian network is a special case of a Markov random field in which \mathcal{G} is a set of conditional probability tables and $Z = 1$. In general for a Markov random field, and for $\Gamma \in \mathcal{G}$, one can identify a set of variables on which Γ depends. More formally, Γ depends on a variable X if there exists an assignment σ and a value $x \in \mathcal{D}(X)$ such that $\Gamma(\sigma) \neq \Gamma[\sigma[X = x]]$. We are typically interested in cases where for each $\Gamma \in \mathcal{G}$ we have that Γ depends on only a small number of variables, perhaps two or three.

In many applications, such as depth maps for vision, Markov random fields are more natural than Bayesian networks. We are now interested in inference for Markov random fields where we have the following.

$$\begin{aligned} P(x_2, x_5 \mid x_6, x_7) &= \frac{P(x_2, x_5, x_6, x_7)}{P(x_6, x_7)} \\ &= \frac{\sum_{\sigma \sqsubseteq \langle x_2, x_5, x_6, x_7 \rangle} \prod_{\Gamma \in \mathcal{G}} \Gamma(\sigma)}{\sum_{\sigma \sqsubseteq \langle x_6, x_7 \rangle} \prod_{\Gamma \in \mathcal{G}} \Gamma(\sigma)} \\ &= \frac{Z(\langle x_2, x_5, x_6, x_7 \rangle)}{Z(\langle x_6, x_7 \rangle)} \\ Z(\rho) &= \sum_{\sigma \sqsubseteq \rho} \prod_{\Gamma \in \mathcal{G}} \Gamma(\sigma) \end{aligned}$$

So for inference it now suffices to compute $Z(\rho)$ as defined above. Since

inference for Markov random fields generalizes inference for Bayesian networks, the inference problem for Markov random fields is also #P hard. But we can again use recursive conditioning which will work well when certain structure exists. We define $Z(\rho, \mathcal{G})$ as follows.

$$Z(\rho, \mathcal{G}) = \sum_{\sigma \sqsubseteq \rho} \prod_{\Gamma \in \mathcal{G}} \Gamma(\sigma)$$

where σ only assigns to variables in \mathcal{G}

Recursive conditioning for Markov random fields is defined by the following equation where $Y \notin \text{dom}(\rho)$.

$$Z(\rho, \mathcal{G}) = \sum_{y \in \mathcal{D}(Y)} Z(\rho_1[Y := y], \mathcal{G}_1) \cdots Z(\rho_k[Y := y], \mathcal{G}_k)$$

As with Bayesian networks, for $i \neq j$ we must have that no variable $Z \notin \text{dom}(\rho) \cup \{Y\}$ appears in both \mathcal{G}_i and \mathcal{G}_j . Also, we require that $\rho_i[Y = y]$ is the restriction of $\rho[Y = y]$ to the variables occurring \mathcal{G}_i . This algorithm is identical to that used for Bayesian networks and the same comments about memoization and the choice of $Y \in \text{dom}(\rho)$ apply.