Introduction into Bayesian networks

Mgr. Libor Vaněk

Table of contents

1. Introduction into Bayes' theorem

Classical statistical models do not permit introduction of prior knowledge into the model. For most of the purposes this is desired behavior as it prevents introduction of extraneous data that might skew the experimental results. However there are times when it's useful to leverage prior knowledge as input into further evaluation process.

Bayes' Theorem was developed by the Rev. Thomas Bayes, an 18th century mathematician and theologian, and was first published in 1763. It can be expressed as:

$$P(H|E,c) = \frac{P(H|c) * P(E|H,c)}{P(E|c)}$$

We update our belief in hypothesis *H* given on additional evidence *E* with background context *c*. Left-hand term – *P(H|E,c)* – is known as "posterior probability" or the probability of H after considering the effect of *E* on *c*. The term *P(H|c)* is called the "prior probability of *H* given *c* alone". The term *P(E|H,c)* is called the "likelihood" and gives the probability of the evidence assuming the hypothesis *H* and the background information *c* is true. Finally, the last term *P(E|c)* is independent of *H* and can be regarded as a normalizing or scaling factor.

2. Bayesian networks

2.1. Motivation

The idea of conditional probability has proved to be very useful in real world. There are countless examples where probability of one event is conditional on the probability of a previous one. Although it is possible to use the sum and product rules of probability theory to anticipate this factor of conditionality, this in many cases leads in to NP-hard calculations.

The prospect of managing a scenario with 5 discrete random variables (2^5-1=31 discrete parameters) might be manageable. An expert system for monitoring patients with 37 variables that results in a joint distribution of over 2^{37} parameters would not be manageable at all.

Bayesian network can be, for example, used to represent the probabilistic relationships between diseases and symptoms. Given symptoms, the network can compute the probabilities of the presence of various diseases.

Using a Bayesian network can also save considerable amounts of space, if the dependencies in the joint distribution are sparse. For example storing the conditional probabilities of 10 two-valued variables using a table requires storage space for $2^{10} = 1024$ values. If the local distribution of no variable depends on more than 3 parent variables, the Bayesian network representation only needs to store at most $10 * 2^3 = 80$ values.

Another advantage of Bayesian networks is that it is intuitively easier for a human to understand (a sparse set of) direct dependencies and local distributions than complete joint distribution.

2.2. Introduction

Bayesian networks are directed acyclic graphs whose nodes represent variables, and whose missing edges encode conditional independencies between the variables. Nodes represent random variables – they may be observable quantities, latent variables, unknown parameters or hypotheses. Each node is associated with a probability function that takes as input a particular set of values for the node's parent variables and gives the probability of the variable represented by the node. If the parents are *m* Boolean variables then the probability function could be represented by a table of 2^m entries, one entry for each possible combination of its parents being true or false.

Efficient algorithms exist that perform inference and learning in Bayesian networks. Bayesian networks that model sequences of variables (e.g. speech signals or protein sequences) are called dynamic Bayesian networks. Generalizations of Bayesian networks that can represent and solve decision problems under uncertainty are called influence diagrams.

2.3. Definition

There are several equivalent definitions of a Bayesian network. For all the following, let *G = (V, E)* be a directed acyclic graph (or DAG), and let $X = (X_v)_{v \in V}$ be a set of random variables indexed by V.

2.3.1. Factorization definition

X is a Bayesian network with respect to *G* if its joint probability density function (with respect to a product measure) can be written as a product of the individual density functions, conditional on their parent variables

$$p(x) = \prod_{v \in V} p\big(x_v \mid x_{\mathrm{pa}(v)}\big)$$

where *pa(v)* is the set of parents of *v* – or in other words hose vertices pointing directly to *v* via a single edge).

For any set of random variables, the probability of any member of a joint distribution can be calculated from conditional probabilities using the chain rule as

$$\mathrm{P}(X_1 = x_1, \ldots, X_n = x_n) = \prod_{v=1}^{n} \mathrm{P}(X_v = x_v \mid X_{v+1} = x_{v+1}, \ldots, X_n = x_n)$$

Compare this with the definition above, which can be written as:

$$\mathrm{P}(X_1 = x_1, \ldots, X_n = x_n) = \prod_{v=1}^{n} \mathrm{P}(X_v = x_v \mid X_j = x_j$$ for each X_j which is a parent of X_v*)*

The difference between the two expressions is the conditional independence of the variables from any of their non-descendents, given the values of their parent variables.

2.3.2. Markov blanket

The Markov blanket for node *A* is a set of nodes composed of *A*'s parents, *A*'s children and children's other parents. Therefore Markov blanket contains all the variables that shield the node *A* from the rest of the network. This means that the Markov blanket of a node is the only knowledge needed to predict the behavior of that node.

The values of the parents and children of a node evidently give information about that node. However, its children's parents also have to be included, because they can be used to explain away the node in question.

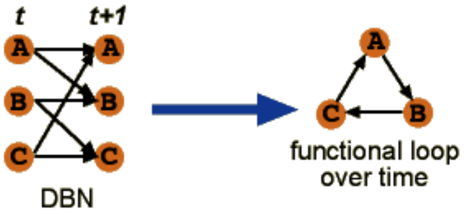
X is a Bayesian network with respect to G if every node is conditionally independent of all other nodes in the network, given its Markov blanket.

2.4. Dynamical Bayesian Networks

Bayesian networks do have some limitations for functional network inference. First, due to mathematical properties of the joint probability distribution, it is possible to have a group of BNs which represent exactly the same joint probability distribution, having the same conditional dependence and independence relationships, but which differ in the direction of some of their edges. Such a group is called an equivalence class of Bayesian networks (the BNs below represent an equivalence class). This creates problems in assigning direction of causation to an interaction from an edge in a Bayesian network.

Second, the restriction of the BN to be acyclic (also due to mathematical properties of the joint probability distribution) is a problem for biology-specific models, because feedback loops are a common biological feature. A BN could not model a feedback loop because it cannot have loops, or cycles.

Fortunately, both of these limitations can be overcome by using dynamic Bayesian networks (DBNs). A DBN consists of representing all variables at two (or more) points in time. Edges are drawn from the variables at the earlier time to those at the later time.



In this way, cycles over time can be represented using an underlying acyclic DBN. For example, in the DBN on the left above, we see that A at time t influences B at time $t+1$, B influences C, and C influences A. This represents a loop over time (on right), but the DBN has no loops. Additionally, there is no ambiguity over direction of edges—even if an equivalence class exists for this BN, we know the correct biological interpretation is that influence travels forward in time, not into the past.

2.4.1. Dynamic Bayesian Network limitations

Granularity

The modeling technique is unable to describe a problem in which the resolution of events over time varies.

The simple solution of setting the time interval between *t* and *t+1* to the shortest overall period between events leads to increased computational cost during all other parts of the evaluation.

Abstract Temporal Relationships

Abstract concepts such as precedence ("A comes before B.") cannot be expressed in this kind of model.

3. Bayesian Networks examples

3.1. Rainy day tomorrow?

Given a situation where it might rain today, and might rain tomorrow, what is the probability that it will rain on both days? Rains on two consecutive days are not independent events with isolated probabilities. If it rains on one day, it is more likely to rain the next. Solving such a problem involves determining the chances that it will rain today, and then determining the chance that it will rain tomorrow conditional on the probability that it will rain today. These are known as "joint probabilities." Suppose that P(rain today) = 0.20 and P(rain tomorrow given that it rains today) = 0.70. The probability of such joint events is determined by:

$$P(E_1, E_2) = P(E_1)P(E_2|E_1)$$

which can also be expressed as:

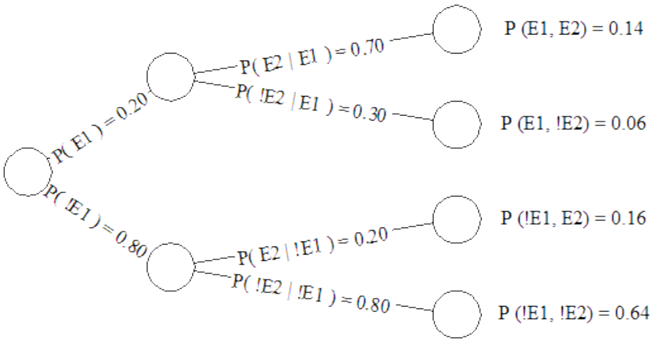
$$P(E_2|E_1) = \frac{P(E_1, E_2)}{P(E_1)}$$

Working out the joint probabilities for all eventualities, the results can be expressed in a table format:

	Rain tomorrow	**No rain tomorrow**	**Marginal probability of rain tomorrow**
Raid today	0.14	0.06	0.20
No raid today	0.16	0.64	0.80
Marginal probability of raid tomorrow	0.30	0.70	

From the table, it is evident that the joint probability of rain over both days is 0.14. There was a great deal of other information that had to be brought into the calculations before such a determination was possible. With only two discrete binary variables four calculations were required.

This same scenario can be expressed using a Bayesian Network Diagram as shown ("!" is used to denote logical "not").



One of the attractive properties of Bayesian networks is that once they form directed acyclic graph they can be browsed using depth-first algorithm. Then we have to calculate only the branches we are interested in - in our case P(E1), P(E2|E1) and P(E2,E1).

We can also utilize the graph both visually and algorithmically to determine which parameters are independent of each other. Instead of calculating four joint probabilities, we can use the independence of the parameters to limit our calculations to two. It is self-evident that the probabilities of rain on the second day having rained on the first are completely autonomous from the probabilities of rain on the second day having not rained on the first.

At the same time as emphasizing parametric indifference, Bayesian Networks also provide a parsimonious representation of conditionality among parametric relationships. While the probability of rain today and the probability of rain tomorrow are two discrete events (it cannot rain both today and tomorrow at the same time), there is a conditional relationship between them (if it rains today, the lingering weather systems and residual moisture are more likely to result in rain tomorrow). For this reason, the directed edges of the graph are connected to show this dependency.

3.2. Burglary alarm

Friedman and Goldszmidt suggest looking at Bayesian Networks as a "story". They offer the example of a story containing five random variables: "Burglary", "Earthquake", "Alarm", "Neighbour Call", and "Radio Announcement". In such a story, "Burglary" and "Earthquake" are independent, and "Burglary" and "Radio Announcement" are independent given "Earthquake." This is to say that there is no event that affects both burglaries and earthquakes. As well, "Burglary" and "Radio Announcements" are independent given "Earthquake" - meaning that while a radio announcement might result from an earthquake, it will not result as a repercussion from a burglary.

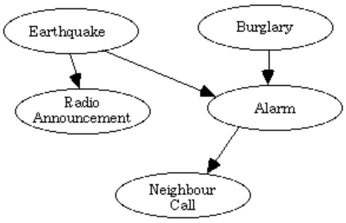
Because of the independence among these variables, the probability of P(A,R,E,B) (The joint probability of an alarm, radio announcement, earthquake and burglary) can be reduced from:

$$P(A,R,E,B)=P(A|R,E,B)*P(R|E,B)*P(E|B)*P(B)$$

involving 15 parameters to 8:

$$P(A,R,E,B) = P(A|E,B)*P(R|E)*P(E)*P(B)$$

This significantly reduced the number of joint probabilities involved. This can be represented as a Bayesian Network:



Using a Bayesian Network offers many advantages over traditional methods of determining causal relationships. Independence among variables is easy to recognize and isolate while conditional relationships are clearly delimited by a directed graph edge: two variables are independent if all the paths between them are blocked (given the edges are directional). Not all the joint probabilities need to be calculated to make a decision; extraneous branches and relationships can be ignored (One can make a prediction of a radio announcement regardless of whether an alarm sounds). By optimizing the graph, every node can be shown to have at most *k* parents. The algorithmic routines required can then be run in $O(2^k n)$ instead of $O(2^n)$ time. In essence, the algorithm can run in linear time (based on the number of edges) instead of exponential time (based on the number of parameters).

Associated with each node is a set of conditional probability distributions. For example, the "Alarm" node might have the following probability distribution:

Probability Distribution for the Alarm Node given the events of "Earthquakes" and "Burglaries" ("!" denotes "not")			
Earthquake	**Burglary**	**P(A|E,B)**	**P(!A|E,B)**
E	B	0.90	0.10
E	!B	0.20	0.80
!E	B	0.90	0.10
!E	!B	0.01	0.99

For example, should there be both an earthquake and a burglary, the alarm has a 90% chance of sounding. With only an earthquake and no burglary, it would only sound in 20% of the cases. A burglary unaccompanied by an earthquake would set off the alarm 90% of the time, and the chance of a false alarm given no antecedent event should only have a probability of 0.1% of the time. Obviously, these values would have to be determined a posteriori.

4. Inference and learning

One of the main usages of Bayesian networks is, based on a newly introduced evidence, to update the probability that a hypothesis may be true.

There are three main inference tasks for Bayesian networks:

4.1. Inferring unobserved variables

Because a Bayesian network is a complete model for the variables and their relationships, it can be used to answer probabilistic queries about them. For example, the network can be used to find out updated knowledge of the state of a subset of variables when other variables (the evidence variables) are observed. This process of computing the posterior distribution of variables given evidence is called probabilistic inference.

The posterior gives a universal sufficient statistic for detection applications, when one wants to choose values for the variable subset that minimize some expected loss function, for instance the probability of decision error. A Bayesian network can thus be considered a mechanism for automatically applying Bayes' theorem to complex problems.

The most common exact inference methods are: variable elimination, which eliminates (by integration or summation) the non-observed non-query variables one by one by distributing the sum over the product; clique tree propagation, which caches the computation so that many variables can be queried at one time and new evidence can be propagated quickly; and recursive conditioning, which allows for a space-time tradeoff and matches the efficiency of variable elimination when enough space is used. All of these methods have complexity that is exponential in the network's tree width. The most common approximate inference algorithms are stochastic MCMC (Markov Chain Monte Carlo) simulation, mini-bucket elimination that generalizes loopy belief propagation, and variation methods.

4.2. Parameter learning

In order to fully specify the Bayesian network and thus fully represent the joint probability distribution, it is necessary to specify for each node X the probability distribution for X conditional upon X 's parents. The distribution of X conditional upon its parents may have any form. It is common to work with discrete or Gaussian distributions since that simplifies calculations. Sometimes only constraints on a distribution are known; one can then use the principle of maximum entropy to determine a single distribution, the one with the greatest entropy given the constraints.

Often these conditional distributions include parameters that are unknown and must be estimated from data, sometimes using the maximum likelihood approach. Direct maximization of the likelihood (or of the posterior probability) is often complex when there are unobserved variables. A classical approach to this problem is the expectation-maximization algorithm which alternates computing expected values of the unobserved variables conditional on observed data, with maximizing the complete likelihood (or posterior) assuming that previously computed expected values are correct. Under mild regularity conditions this process converges on maximum likelihood (or maximum posterior) values for parameters.

A more fully Bayesian approach to parameters is to treat parameters as additional unobserved variables and to compute a full posterior distribution over all nodes conditional upon observed data, then to integrate out the parameters. This approach can be expensive and lead to large dimension models, so in practice classical parameter-setting approaches are more common.

4.3. Structure learning

In the simplest case, a Bayesian network is specified by an expert and is then used to perform inference. In other applications the task of defining the network is too complex for humans. In this case the network structure and the parameters of the local distributions must be learned from data.

Automatically learning the graph structure of a Bayesian network is a challenge pursued within machine learning. The basic idea goes back to a recovery algorithm developed by Rebane and Pearl and rests on the distinction between the three possible types of adjacent triplets allowed in a directed acyclic graph (DAG):

1. $X \rightarrow Y \rightarrow Z$
2. $X \leftarrow Y \rightarrow Z$
3. $X \rightarrow Y \leftarrow Z$

Type 1 and type 2 represent the same dependencies (X and Z are independent given Y) and are, therefore, indistinguishable. Type 3, however, can be uniquely identified, since X and Z are marginally independent and all other pairs are dependent. Thus, while the skeletons (the graphs stripped of arrows) of these three triplets are identical, the directionality of the arrows is partially identifiable.

The same distinction applies when X and Z have common parents, except that one must first condition on those parents. Algorithms have been developed to systematically determine the skeleton of the underlying graph and, then, orient all arrows whose directionality is dictated by the conditional independencies observed.

An alternative method of structural learning uses optimization-based search. It requires a scoring function and a search strategy. A common scoring function is posterior probability of the structure given the training data. The time requirement of an exhaustive search returning back a structure that maximizes the score is super-exponential in the number of variables.

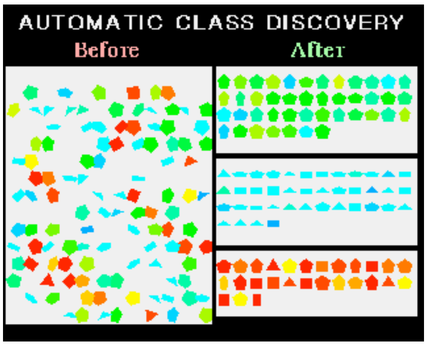
A local search strategy makes incremental changes aimed at improving the score of the structure. A global search algorithm like MCMC can avoid getting trapped in local minima.

5. Practical Uses for Bayesian Networks

5.1. AutoClass

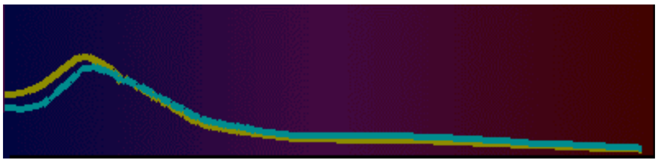
The National Aeronautic and Space Administration have a large investment in Bayesian research. NASA's Ames Research Center is interested in deep-space exploration and knowledge acquisition. In gathering data from deep-space observatories and planetary probes, an apriori imposition of structure or pattern expectations is inappropriate. Researchers do not always know what to expect or even have hypotheses for which to test when gathering such data. Bayesian inference is useful because it allows the inference system to construct its own potential systems of meaning upon the data. Once any implicit network is discovered within the data, the juxtaposition of this network against other data sets allows for quick and efficient testing of new theories and hypotheses.

The AutoClass project is an attempt to create Bayesian applications that can automatically interpolate raw data from interplanetary probes, and deep space explorations. A graphical example of AutoClass's capabilities is displayed bellow - it's an AutoClass interpolation of raw data with no predefined categories.

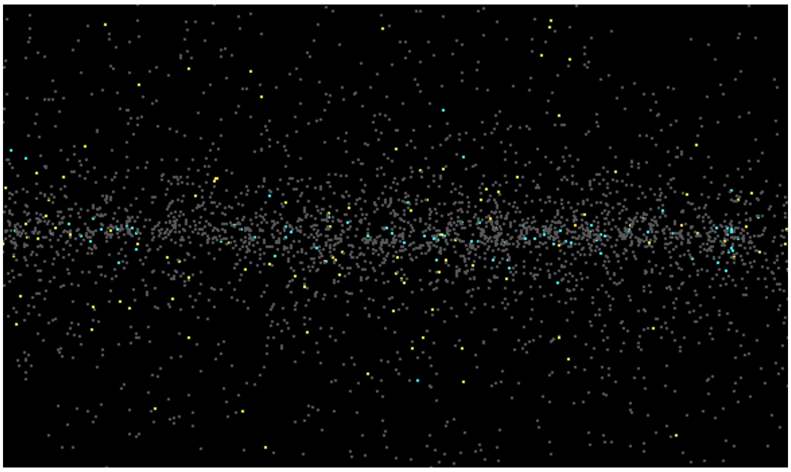


An AutoClass interpolation of raw data with no predefined categories. Sorted data is grouped by colour and shape. The top area is sorted into green-blue shapes, the middle into blues, and the bottom into red-orange-yellow shapes.

An applied example of AutoClass's capabilities was the input of infrared spectra. Although no differences among this spectra were initially suspected, AutoClass successfully distinguished two subgroups of stars.



The difference is confirmed by looking at their positions on this map of the galaxy (one subgroup is clearly located near galactic plane while the other seems to be distributed more uniformly)



5.2. Introduction of Search Heuristics

Searching for a solution to a problem is usually an NP-hard problem resulting in a combinatorial explosion of possible solutions to investigate. This problem is often ameliorated through the use of heuristics, or sub-routines to make "intelligent" choices along the decision tree. An appropriately defined heuristic can quicken the search by eliminating obviously unsuccessful paths from the search tree. An inappropriately defined heuristic might eliminate the successful solutions and result in no evident solution.

Bayesian networks can replace heuristic methods by introducing a method where the probabilities are updated continually during search.

One class of search algorithms called stochastic searching utilizes what are known as "Monte-Carlo" procedures. These procedures are non-deterministic and do not guarantee a solution to a problem. As such they are very fast, and repeated use of these algorithms will add evidence that a solution does not exist even though they never prove that such a solution is non-existent.

5.3. Lumiere

The Lumiere project at Microsoft Research was initiated in 1993 with the goal of developing methods and an architecture for reasoning about the goals and needs of software users as they work with software. At the heart of Lumiere are Bayesian models that capture the uncertain relationships between the goals and needs of a user and observations about program state, sequences of actions over time, and words in a user's query (when such a query has been made).

Ancestors of Lumiere included earlier research on probabilistic models of user goals to support the task of custom-tailoring information displayed to pilots of commercial aircraft, and related work on user modeling for the decision-theoretic control of displays that led to systems that modulate data displayed to flight engineers at the NASA Mission Control Center.

Early on in the Lumiere project, studies were performed in the Microsoft usability labs to investigate key issues in determining how best to assist a user as they worked. The studies were aimed at exploring how experts in specific software applications worked to understand problems that users might be having with software from the user's behaviors.

The Office Assistant in the Office '97 and Office 2003 product suites was based in spirit on the Lumiere and on prior research efforts that had led to the Answer Wizard help retrieval system in Office '95. Office committed to a character-based assistant. Users were able to choose one of several assistants each of whom had a variety of behavioral patterns - all of whom draw their ability to interpret context and natural language queries from Bayesian user models.

6. Limitations of Bayesian Networks

In spite of their remarkable power and potential to address inferential processes, there are some inherent limitations and liabilities to Bayesian networks.

In reviewing the Lumiere project, one potential problem that is seldom recognized is the remote possibility that a system's user might wish to violate the distribution of probabilities upon which the system is built. While an automated help desk system that is unable to embrace unusual or unanticipated requests is merely frustrating, an automated navigation system that is unable to respond to some previously unforeseen event might put an aircraft and its occupants in mortal peril. While these systems can update their goals and objectives based on prior distributions of goals and objectives among sample groups, the possibility that a user will make a novel request for information in a previously unanticipated way must also be accommodated.

Two other problems are more serious. The first is the computational difficulty of exploring a previously unknown network. To calculate the probability of any branch of the network, all branches must be calculated. While the resulting ability to describe the network can be performed in linear time, this process of network discovery is an NP-hard task which might either be too costly to perform, or impossible given the number and combination of variables.

The second problem centers on the quality and extent of the prior beliefs used in Bayesian inference processing. A Bayesian network is only as useful as this prior knowledge is reliable. Either an excessively optimistic or pessimistic expectation of the quality of these prior beliefs will distort the entire network and invalidate the results. Related to this concern is the selection of the statistical distribution induced in modeling the data. Selecting the proper distribution model to describe the data has a notable effect on the quality of the resulting network.

7. Conclusion

These concerns aside, Bayesian networks have incredible power to offer assistance in a wide range of endeavors. They support the use of probabilistic inference to update and revise belief values. Bayesian networks readily permit qualitative inferences without the computational inefficiencies of traditional joint probability determinations. In doing so, they support complex inference modeling including rational decision making systems, value of information and sensitivity analysis. As such, they are useful for causality analysis and through statistical induction they support a form of automated learning. This learning can involve parametric discovery, network discovery, and causal relationship discovery.

7.1. Usage of Bayesian Networks in my thesis

My thesis will focus on effective generation of test cases, running them and evaluating results - all in context of metamodel repository. I plan to use Bayesian algorithms and networks for several important parts of overall work.

7.1.1. Model verification

Testing results (test success/failure) will be evaluated by software "probes" injected into standard environment - e.g. monitoring transferred messages or RPC calls.

These probes are to be used to monitor also real-world traffic. Gathered data can be then used in combination of Bayesian structure learning to identify potential discrepancies from manually created model.

7.1.2. Test case generation and evaluation

Typical model consists of several more or less interconnected entities with attributes. Based only on the model itself there is (except some very rare special cases) virtually infinite number of potential test cases.

Bayesian network will be trained from real-world traffic and/or manually to generate the most real-world-like test cases. Also it can be used to evaluate tests failures inter-dependencies to help to identify potential problem (e.g. which entity in model is most probably to fail).