Bayesian Network Learning System based on Neural Networks

Yoichi Motomura
Electrotechnical Laboratory
motomura@etl.go.jp

Isao Hara
Electrotechnical Laboratory
hara@etl.go.jp

Abstract

In real world applications like autonomous robots moving in real environments , it is necessary to cope with uncertain events and observations. Therefore, modeling uncertain relationships among many kinds of variables and predicting such variables are important issues. Bayesian network is a probabilistic model that represents conditional dependency among random variables.

In this paper, Bayesian network that learns conditional probabilities by neural networks is introduced. Real world problems include non-linear, multi-dimensional and continuous random variables. However, most of conventional Bayesian networks can handle discrete variables. Well known Bayesian network for continuous variables exists[6] but it can handle only linear dependency between child and parent nodes. Therefore, we apply neural networks as flexible representations of the conditional probabilities of Bayesian network. The model is constructed and trained with samples collected in the real environment. Learning conditional probability is realized by the back propagation algorithm. Constructing appropriate graph structure of Bayesian network is still a hard problem[4].

Hence we have to find good graph structures by fitting with statistical data. For this purpose, we develop Bayesian network software, "BAYONET". This system can connect to major database software, operate the databases, learn from samples in such large databases, and find better graph structure by interacting with users or automatic graph switching algorithms. After constructing Bayesian network in the system, it is applied to estimation of uncertain events which can not be observed directly. As an example, a robot control architecture using Bayesian network is shown.

1 Introduction

In real world environments, the robot's view of the world is filled with uncertainty which is caused by sensor noises, incomplete observations, and dynamic changes in the environment. Thus, an autonomous intelligent system requires the ability to predict and estimate such uncertain factors. Probabilistic reasoning by Bayesian network [2, 3] is a promising approach.

Bayesian networks represent conditional dependency among random variables by conditional probabilities. We have to give conditional probabilities to estimate uncertain variables. One possible approach is that we get probabilities from statistical data sampled in the real world environment. Moreover, it is necessary to handle many kinds of variables in the real world domain. We propose to use neural networks to represent conditional probabilities in Bayesian networks. Using neural networks can provide a general framework for representing and learning non-linear conditional probabilities mixed with continuous, discrete, multi-valued, and multi-dimensional variables.

In section 2, we propose a Bayesian network that learns conditional probabilities by feed forward neural networks for real world applications. In section 3, implementation of this model as Bayesian network learning system written in Java is explained. In order to realize learning from large database, the system uses JDBC(Java DataBase Connectivity) and it can connect to major SQL databases. We introduce structure finding properties for learning Bayesian networks as a data analyzing tool. In section 4, an example for robotics application is described briefly.

2 Bayesian network based on neural networks

2.1 Bayesian networks

Bayesian network (also called belief network) is a probabilistic reasoning model that represents the dependency among random variables and gives a concise specification of joint probability distributions. This model is defined as a directed acyclic graph (DAG), which has conditional probabilities in each node. Nodes represent random variables in a domain. Directed links in the graph represent dependency between nodes.

To simplify, let us consider the simplest network $X \rightarrow Y$. X is called *parent node* and Y is called *child node*. The child node has a conditional probability $P(Y|X)$. When information about X is given, *Belief* that $Y = y$ (or posterior probability distribution of Y given $P(X)$ is computed by a marginalizing operation over the sample space of X, Ω_X,

$$BEL(Y = y) = \alpha \int_{x \in \Omega_X} P(y|x)\, dP(x). \quad (1)$$

On the other hand, when we have $P(Y)$ at first, *Belief* that $X = x$ is updated by,

$$\begin{aligned} BEL(X = x) &= \int_{y \in \Omega_Y} P(x|y)\, dP(y) \\ &= \beta \int_{y \in \Omega_Y} P(y|x)\, dP(y) \cdot P(x), \quad (2) \end{aligned}$$

where α, β are normalizing constants. These calculations can be regarded as belief propagation through the links X and Y. For more complex Bayesian networks, probabilistic inference is executed by belief propagation through all links in the network[2, 3].

In most conventional Bayesian networks, random variables take discrete values and conditional probabilities are represented by a table of possible combinations of parent values and child values. However, for many kinds of variables like continuous or multi-dimensional variables, we need more flexible representation for conditional probabilities in Bayesian networks.

2.2 Bayesian network based on neural networks

One solution for the problem of representing conditional probabilities with many kinds of variables is using neural networks. We call this model *Bayesian network based on neural networks*. A neural network can handle both discrete variables and continuous variables in the same manner. In the case that Y is a discrete random variable, $Y = (y_1, y_2, \cdots, y_K)$, k neurons can represent the probability vector of Y, $P(y_1), P(y_2), \cdots, P(y_k)$ with normalization to make the sum equals to 1. Let's introduce a feed forward neural network that has input neurons to represent X, output neurons to represent Y, and hidden neurons. Then, we can represent a conditional probability $P(Y|X = x)$ as the following,

$$f_k(x) = g\left(\sum_j v_{jk} g\left(\sum_i w_{ij} x_i + b_j\right) + b_k\right)$$

$$g(x) = \frac{1}{1 + \exp(-x)}$$

$$P(y_k|x) = f_k(x) / \sum_k f_k(x).$$

Here, v, w, b are connection weights of the neural network.

When Y is a continuous random variable, we approximate the probability distribution of Y as a Gaussian distribution parametrized by μ, σ, and represent $P(y|x)$ as,

$$f_\mu(x) = \sum_j v_j^\mu g\left(\sum_i w_{ij}^\mu x_i + b_j^\mu\right) + b^\mu$$

$$f_\sigma(x) = \sum_j v_j^\sigma g\left(\sum_i w_{ij}^\sigma x_i + b_j^\sigma\right) + b^\sigma$$

$$P(y|x) = \frac{1}{\sqrt{2\pi} f_\sigma(x)} \exp\left(-\frac{(y - f_\mu(x))^2}{2 f_\sigma(x)^2}\right)$$

Using such neural networks to represent each conditional probability of each child node, we can construct Bayesian network with neural networks. For example in Figure 1, the child node C of the Bayesian network can be represented by the output neurons, and the parent nodes A and B are represented by the input neurons of the neural network(left side). Conditional

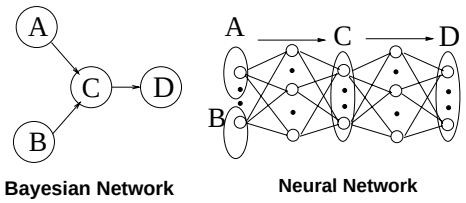

Figure 1: Structure of BNNN

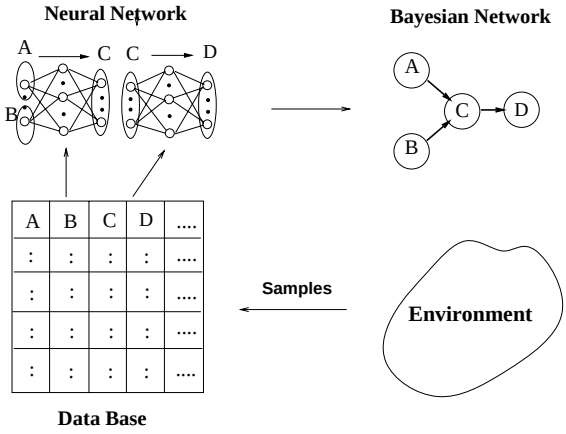

Figure 2: Learning of BNNN from database

dependency between C and D is also modeled by input and output neurons of another neural network(right side). Thus, each child node of the Bayesian network has its own neural network. The number of input neurons of the neural network is determined by the the dimension of the parent nodes and the number of output neurons is determined by the dimension of the child node. For discrete variables, integral operations, equation (1) and (2) can be computed in the straight forward way. In the case of continuous variables, we can use Monte Carlo sampling and numerical integral operations to approximate.

2.3 Learning conditional probabilities

If we have enough data set of parent nodes and child nodes, the neural network can be trained to approximate conditional probability $P(Y|X)$ for the child node Y given X by the back propagation method. Each neural network is trained with corresponding data set of parent and child nodes. Learning continuous conditional probability $P(Y|X)$ is regarded as an approximation of nonlinear functions, $\mu_Y(X)$ and $\sigma_Y(X)$. $\mu_Y(X)$ is given by mean of Y given X. The training set of $\sigma_Y(X)$ can be given by computing standard deviation of Y given X.

After training the neural network successfully, it can represent the conditional dependency between corresponding parent nodes and child nodes. If there is no conditional dependency between the nodes, learning will fail. This means that this training scheme can also

be applied to detect conditional dependency between the variables.

2.4 Finding Network Structure

Finding structure of Bayesian network is an important but hard problem [4]. The difficulties of this problem are caused by the following reasons.

- Computational cost of searching graph structures is NP hard.
- In order to evaluate the goodness of each structure, learning conditional probabilities has to be completed.
- Currently, suitable evaluation criteria for this problem is discussed, but still there are many controversial issues.

As a resort at this moment, we propose the method that keeps possible tree structures for each child node (local trees) and selects better local tree structure with many kinds of algorithms using an information criterion like MDL(Minimal Description Length). Tree selecting algorithms decide appropriate local graph structure independently, then it can be assumed that this result makes whole structure optimal.

3 Implementation

We have developed a system to implement the model described in the previous section. The name of our system is *BAYONET*[1].

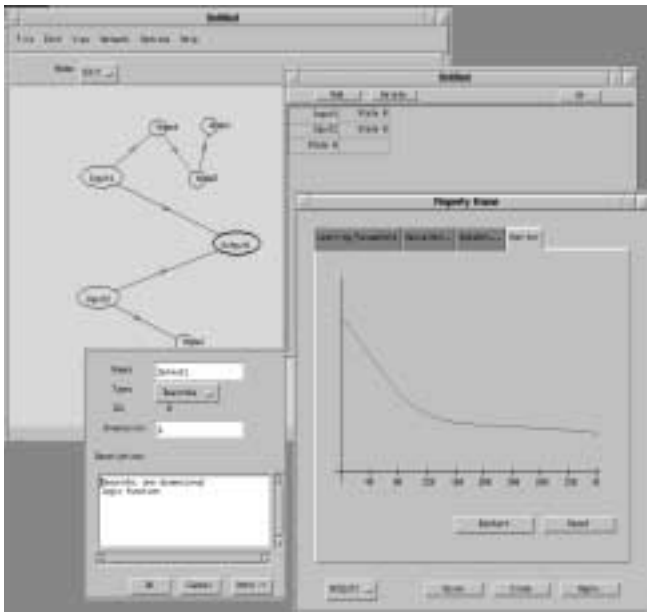

Figure 3: BAYONET

BAYONET is regarded as a probabilistic reasoning server in a network. The server receives requests from a client program that requires probabilistic evaluation about uncertain variables. After probabilistic reasoning, the answer is sent back to the client. For example, a process that observes sensor information sends the sensor values to BAYONET. Then the other processes that need estimation (like the position of a robot) ask BAYONET and get the result of probabilistic reasoning. In the same manner, a node in the BAYONET running on a machine can also connect to the node in another BAYONET running on another machine (Figure 4). We define communication protocol messages as the following; 'setEvidence', 'getMAP', 'getMean', 'getVariance', 'evalValue'. The modules communicate with each other by these messages.

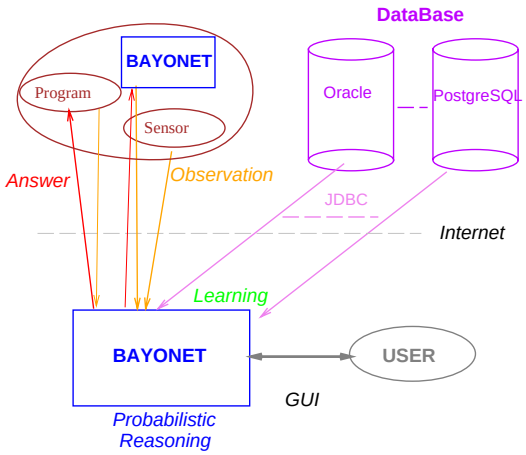

Figure 4: TCP/IP connections

Bayesian networks are essentially parallel processing models. The Monte Carlo method in each node can be also executed independently. For these reasons, a development environment that has a multi-thread mechanism and object-oriented features is desirable for implementing this model. Therefore, our system is implemented in Java that has multi-thread capability.

Java has database connectivity methods called JDBC. Using JDBC in the learning process, BAYONET can get data samples from major SQL database like ORACLE, Postgress and so on. The advantages of using SQL database is the following.

- We can use the huge size of data stored in major databases.
- It is possible to operate huge data with simple SQL command.
- Computation for huge data is optimized in such major databases.

Databases are specified by URL and particular items in the database is assigned to corresponding node by drag and drop operation of GUI (Figure 5).

Moreover, users can make simple operations on the database using GUI of the BAYONET (Figure 6). Repeating such operations, users can understand properties and tendencies of data. This helps to construct a suitable graph structure.

[1]BAYONET implies a powerful tool combined with two different functions

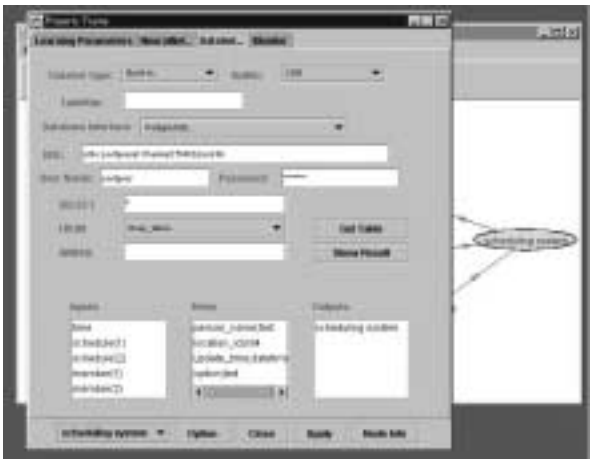

Figure 5: Assign nodes to items in databases

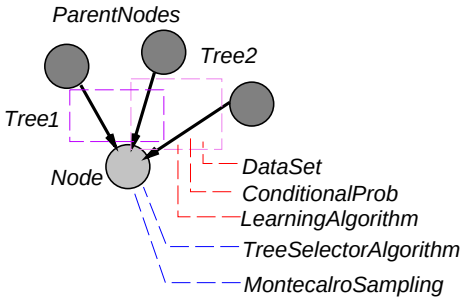

Figure 7: Node object

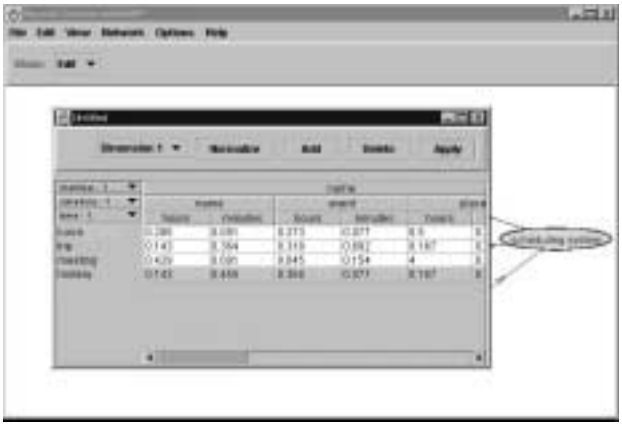

Figure 8: Edit conditional probability tables

Figure 6: GUI operation for databases

In order to realize the tree selection mechanism described in the previous section, we use object oriented programming in Java. Each node is created as an instance of an object, and many other objects, datasets, algorithms can be assigned to the node object(Figure 7).

For conditional probabilities in the system, we have two different kind of representations. One is a conditional probability table style for discrete variables(Figure 8). Connecting to a database, the system can calculate corresponding conditional probability from frequency under particular condition in the database.

The other representation is a neural network style for both continuous and discrete variables. By connecting to a database, the neural network can be trained with a supervised dataset in the database. The system can also monitor the learning curve of the back propagation(Figure 3). If users notice that the learning is converging with small error, then the neural network successfully approximate corresponding conditional probability. Success in learning means the structure is justified by the fitting to statistical data. Finally, we can model the data set including non-linear conditional dependency. It can be regarded as a interactive data analysis.

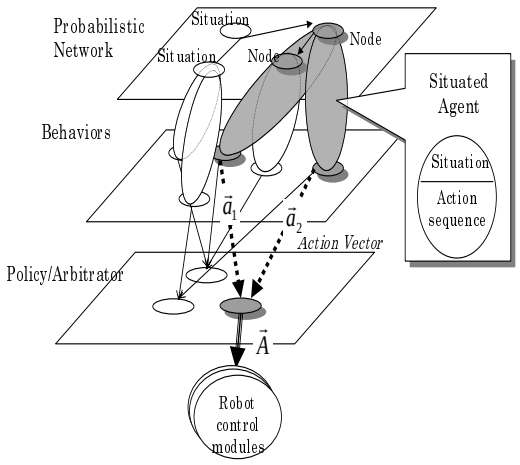

Figure 9: Robot control architecture

4 Application

After constructing appropriate graph structure and acquiring the conditional probabilities of Bayesian network, we can use the model to evaluate unknown random variable given observed variables. Therefore, this model can work as an expert system. There are many real world applications suitable to use Bayesian networks[7]. The authors apply the model for robot controlling in uncertain situations [9]. The architecture to control a robot is shown in Figure 9.

The first layer of the architecture is BAYONET. The second layer consists of independent program modules to control the robot or to observe sensors. These are called *behavior modules*. The third layer is the decision maker to select the final action of the robot. Each module is designed as a situation dependent module. If the robot is in the particular situation, then the corresponding situated behavior module designed for this situation can work well, and the robot get highest utility value. However, when the actual situation can not be determined exactly, the selection of which behavior module to get higher utility is a kind of a decision making problem. In this case, we can evaluate probability of each situation on the BAYONET. Then decision theoretic selection can be made by maximizing expected utility that is intuitively utility multiplied by probability of each situation. By using this architecture, the robot can take the optimal behavior in an uncertain environment.

5 Conclusion

In this paper, we introduced a Bayesian network learning system based on neural networks. The system is written in Java, designed as a network server and can connect to major database system. The system has user friendly GUI for interactive data analysis with the database. Conditional probability can be obtained from examples in the database. The advantage of the system is that neural networks are utilized to represent conditional probabilities of many kinds of variables including non-linear continuous variables. In order to decide the most appropriate structure of Bayesian network, the system can support the local tree selecting mechanism. We can apply many different model selection algorithms to it.

As a feature work, the system will be used for controlling the robot moving in uncertain situations. Thanks to probabilistic evaluation using BAYONET , the robot can take optimal action to maximize expected utility. The GUI and learning properties in our system are useful for modelling state and situation transitions in the robotics domain with many continuous, non-linear uncertain variables.

Acknowledgements

This work is supported by the Real World Computing Program organized by the Ministry of International Trade and Industry of Japan. The authors thank Hideki Asoh and Simon Thompson for his helpful comments.

References

[1] S.Russell and P.Norvig: "Artificial Intelligence: a modern approach", Prentice Hall, 1994.

[2] Cowell,R.G., Dawid A.P., Lauritzen,S.L. and Spiegelhalter,D.J. "Probabilistic Networks and Expert Systems", Springer, 1999.

[3] Castillo,E., Gutierrez,J. and Hadi,A., "Expert Systems and Probabilistic Network Models", Springer, 1997.

[4] Friedman,N., Goldszmidt,M., Heckerman,D. and Russel,S., "Challenge: What is the Impact of Bayesian Networks on Learning",proc. of Int. Joint Conf. on AI'97, (1997).

[5] Cooper,G. and Herskovits,E., "A Bayesian Method for Inductionof Probabilistic Network from Data", Machine Learning, 9,309-347, 1992.

[6] Geiger,D. and Heckerman,D., " Learning Gaussian Networks", Uncertainty AI,10,235-243, 1994.

[7] Haddawy, P., "An Overview of Some Recent Developments in Bayesian Problem Solving Techniques", AI Magazine, Introduction to special issue on Uncertainty in AI, Spring, 1999.

[8] Motomura,Y., Hara,I.,Asoh,H. and Matsui,T. , "Bayesian network that learns conditional probabilities by neural networks", the Progress in Connectionist-Based Information Systems, pp.584-587, Springer, 1997.

[9] Motomura,Y. and Hara,I.: "Probabilistic Network based Multi Agent Model", Technical report of IPSJ SIG-MPS, 2000-MPS-28, pp.17-20, 2000.