A Neural Network Approach to MAP in Belief Networks

Yun Peng, Miao Jin, and Kaihua Chen
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County
Baltimore, MD 21250
ypeng@csee.umbc.edu

Abstract

In this paper we suggest a neural network approach to probabilistic inference in Bayesian belief networks (BBN). This is demonstrated by solving maximum *a posteriori* probability problems (MAP), which are known to be NP-hard. In this approach, a belief network is treated as a neural network without any structural changes, and the node activation functions are derived based on the probabilistic calculus of the BBN. Three models are proposed and their convergence analyzed. Computer experiments with two non-trivial example BBN show that this approach may lead to effective approximation methods for MAP.

1. Introduction

It has been well established that probabilistic inference in Bayesian belief networks is NP-hard [1]. Exact solutions for such inference thus are practically intractable with large belief networks. To circumvent this problem, researchers have attempted to develop methods for good *approximate* solutions with sub-exponential complexity. One approach has been to use Gibbs sampleing, (e.g., [6]). In this paper we take another, namely the neural network approach. There are several immediate advantages with this approach. First, a large collection of efficient approximation techniques has been developed in neural network studies. Secondly, since both BBN and neural networks are networks of nodes and weighted links, adopting methods from neural networks to the belief network setting is natural and appealing. Moreover, the localized computation in neural networks makes further speed-up possible through parallel computation.

The rest of this paper is organized as follows. Section 2 describes a particular type of probabilistic inference, namely the MAP problem, which is the focus of this work. Section 3 presents a discrete neural network model for MAP. Sections 4 and 5 extend the discrete model to simulated annealing and continuous models, respectively. Section 6 describes the computer experiments with the proposed methods. Section 7 concludes with future research directions.

2. The MAP Problems

Let $V = \{V_1, V_2, \cdots, V_n\}$ denote the set of all variables in a BBN, and $< V_i, V_j >$ the arc from V_i to V_j (representing the direct causal or other influence from V_i to V_j). An instantiation of V is denoted $v = \{v_1, v_2, \cdots, v_n\}$. This paper only considers BBN of binary variables, thus $v_i \in [0, 1]$. The conditional probability distribution for each V_i is denoted $P(v_i|\pi_i)$ where π_i is an instantiation of the parents of V_i. The joint probability distribution of v can be computed from the conditional distributions as [7]

$$P(v) = \prod_{I=1}^{n} P(v_i|\pi_i). \tag{1}$$

For a type of simpler BBN, known as noisy-or network, each arc $<V_i, V_j>$ is associated with a causal strength c_{ij}, a probability value representing how strongly variable $v_i = 1$ alone can cause $v_j = 1$. It can be shown [8] that, for a noisy-or network,

$$P(v_i = 1|\pi_i) = 1 - \prod_{v_k \in \pi_i} (1 - c_{ki} v_k). \tag{2}$$

The MAP (*maximum a posteriori probability*) problem is to find the most probable instantiation (or a joint assignment) to the set of all uninstantiated variables, given a set of instantiated variables. Formally, a MAP problem can be stated as follows.

> Let X and Y Partition V. Then a MAP problem is to find an instantiation x^* of X such that $P(x^*|y) = \max_x P(x|y)$, where y is a given instantiation of Y. Since y is fixed, x^* also maximizes $P(x, y)$.

MAP problems in singly-connected networks are known to have polynomial-time algorithms [7]. It has been established, however, that MAP is NP-hard if the network is multi-connected [9].

Next, we discuss how to treat a belief network as a neural network for solving MAP. Each node V_i now becomes a processing unit with $v_i(t)$ as its activation at time t. Activations of nodes in Y are fixed (clamped) to their given instantiations during the entire computation, while activations of nodes in X are free to change. In the discrete and simulated annealing models (to be developed in Sections 3 and 4), node activations can take only binary values 0 and 1. Therefore, different activations represent different instantiations for a given node. In the continuous model (Section 5), however, activations take real values between 0 and 1, $x_i(t)$ thus can be interpreted as a partial instantiation of X_i. A node is only connected to its direct causal predecessors (parents) and successors (children), as specified by the arcs in a BBN. Its activation therefore influences and is influenced by the activations of these neighboring nodes. Moreover, as can be seen shortly, activations from parent nodes and child nodes play different roles in determining the activation of a given node.

3. The Discrete Model

For a given instantiation y of Y, the discrete model starts with an arbitrary instantiation $x(0)$ of X. The initial state of the system is thus $v(0) = x(0) \cup y$. At each step, a *single* node X_i is randomly selected for updating its activation. To develop the node activation function, we first define the function

$$E(v(t)) = -\ln P(v(t)). \tag{3}$$

To maximize $P(v)$ is to minimize $E(v)$. Then

$$\begin{aligned}\Delta E_i(t) &= E(v_1(t), \cdots, x_i = 1, \cdots, v_n(t)) \\ &\quad - E(v_1(t), \cdots, x_i = 0, \cdots, v_n(t)) \\ &= -\ln P(v_1(t), \cdots, x_i = 1, \cdots, v_n(t)) \\ &\quad + \ln P(v_1(t), \cdots, x_i = 0, \cdots, v_n(t))\end{aligned}$$

is the energy difference for X_i set to 1 and 0. Let

$$r_i(t) = \frac{P(v_1(t), \ldots, x_i = 1, \ldots, v_n(t))}{P(v_1(t), \ldots, x_i = 0, \ldots, v_n(t))}, \tag{4}$$

then

$$\Delta E_i(t) = -\ln r_i(t). \tag{5}$$

Since $P(v) = \prod_{k=1}^{n} P(v_k|\pi_k)$, $r_i(t)$ can be represented as a fraction of two products of conditional probabilities. Also note that all variables, except x_i, have the same instantiations in both products. Therefore we can simplify Eq. 4 by canceling all conditional probabilities in the two products except those which involve x_i. These terms include $P(x_i = 1|\pi_i(t))$ and $P(x_i = 0|\pi_i(t))$, as well as conditional probabilities of the children of X_i. If V_j is a child of X_i, it would not be right to denote the conditional probability as $P(v_j(t)|\pi_j(t))$ because one parent of V_j, namely X_i, is forced to instantiate to 1 in the numerator and 0 in the denominator in Eq. 4, regardless of the actual value of $x_i(t)$. Therefore, we have

$$\begin{aligned}&r_i(t) \\ &= \frac{P(x_i = 1|\pi_{X_i}(t))}{P(x_i = 0|\pi_{X_i}(t))} \prod_{j \in Child(X_i)} \\ &\quad \frac{P(v_j(t)|\pi_j(t) - \{x_i(t)\} \cup \{x_i = 1\})}{P(v_j(t)|\pi_j(t) - \{x_i(t)\} \cup \{x_i = 0\})},\end{aligned} \tag{6}$$

By Eq. 6, the computation of $r_i(t)$ is completely local, involving only the parents and children of X_i, which are connected by causal arcs to X_i in the network. The node activation function is then defined as follows. When X_i is selected to update at time t,

$$x_i(t+1) = \begin{cases} 1 & \text{if } r_i(t) > 1 \\ x_i(t) & \text{if } r_i(t) = 1. \\ 0 & \text{if } r_i(t) < 1 \end{cases} \tag{7}$$

It is clear from Eqs. 5 and 6 that $x_i(t+1)$ is set to 1 only if $r_i(t) > 1$,which implies $\Delta E_i(t) < 0$, and thus E decreases. Similarly, E decreases when $x_i(t+1)$ is set to 0 when $r_i(t) < 1$. Therefore, $E(t)$ is monotonically decrease. Since $E(v(t)) \geq 0$ is bounded below, the system is guaranteed to converge to a stable state with local minimum of $E(v(t))$ (and thus maximum of $P(v)$).

4. The Simulated Annealing Model

Simulated annealing (SA), a well-known stochastic method, offers an elegant way for a dynamic system to escape from a stable state and continue searching for other, potentially better, solutions [5]. Our discrete model can be easily converted to a SA model. To do so, we only need to change the way of using $r_i(t)$ from the deterministic to the stochastic fashion.

When X_i is selected to update at time t, its activation is assigned to 1 with the probability

$$\begin{aligned}P(X_i \leftarrow 1) &= (1 + e^{\Delta E_i(t)/T(t)})^{-1} \\ &= (1 + r_i^{-1/T(t)})^{-1}\end{aligned} \tag{8}$$

where $T(t)$ is the artificial temperature at t, and $r_i(t)$ is computed by Eq. 6. Therefore, when $r_i(t) > 1$ ($\Delta E_i(t) < 0$), X_i is not set to 1, but has a probability of $(1 + (r_i(t))^{-1/T(t)})^{-1} > 0.5$ being set to 1.

5. The Continuous Model

Simulated annealing may significantly improve the solution accuracy, but the annealing process often takes a long time to complete. Taking cue from Hopfield's continuous model [4], we develop our continuous model for MAP. This model differs from the SA model in several aspects. First, *all* variables X_i change their activations simultaneously in *each* iteration. Secondly, the activations take real values and they change by a *small* amount in each update. As a result, the system moves in the interior of the unit hypercube, instead of jumping from corner to corner in the hypercube. Finally, it is a deterministic rather than stochastic model. The energy difference is used to decide the activation changes rather than the probability of such changes. Moreover, no attempt is made to let the system reach equilibrium at each temperature. Instead, the temperature is reduced with each iteration.

Since the activations for free nodes are continuous, representing partial instantiations of these variables, we can not directly use probability formulas such as $P(v)$ because they are defined for fully instantiated (i.e., binary) variables. Therefore, it is necessary to extend these formulas from binary to the real-valued variables. We denote the extended formulas as $\tilde{P}(.)$. The key here is to extend the conditional probability distribution $P(v_i|\pi_i)$ from binary to continuous variables. The extension is done for the noisy-or BBN as follows. First, the formula $P(v_i = 1|\pi_i)$ can be easily converted for continuous π_i by simply allowing v_k to take real values between 0 and 1 in Eq. 2,

$$\tilde{P}(v_i = 1|\pi_i) = 1 - \prod_{v_k \in \pi_i} (1 - c_{ki} v_k) \qquad (9)$$

It is more difficult to convert $P(v_i|\pi_i)$ in which v_i is also partially instantiated. We handle this by a simple linear interpolation between $\tilde{P}(v_i = 1|\pi_i)$ and $\tilde{P}(v_i = 0|\pi_i)$,

$$\tilde{P}(v_i|\pi_i) = \tilde{P}(v_i = 1|\pi_i)v_i + \tilde{P}(v_i = 0|\pi_i)(1 - v_i). \qquad (10)$$

Now, with Eqs. 9 and 10, we can define

$$\tilde{P}(v(t)) = \prod_{i=1}^{n} \tilde{P}(v_i(t)|\pi_i(t)). \qquad (11)$$

It is clear that $\tilde{P}(v) = P(v)$ when v is fully instantiated. To produce an optimal solution to a given MAP problem, the continuous model computation should move the system to a stable state v^* which 1) is a full instantiation of all variables, and 2) maximizes $\tilde{P}(v)$.

Replacing $P(v)$ by $\tilde{P}(v)$ in Eq. 6, we define

$$\tilde{r_i}(t) = \frac{\tilde{P}(v_1(t), \cdots, x_i = 1, \cdots, v_n(t))}{\tilde{P}(v_1(t), \cdots, x_i = 0, \cdots, v_n(t)} \qquad (12)$$

The continuous model uses ratio $\tilde{r_i}(t)$ differently from either the discrete model or the SA model, as given in the following activation function.

$$x_i(t+1) = (1 + \tilde{r_i}(t)^{-1/T(t)})^{-1}. \qquad (13)$$

Here $T(t)$ is treated in the same way as the artificial temperature in the SA model. It starts with a large initial value and then gradually decreases. When T approaches zero, x_i's approach zero or one, thus becoming fully instantiated. The algorithm for the continuous model then goes as follows.

begin

1. initialization:
   each y_j is set to the given instantiation
   each x_i is set to a small random positive value
   $T \leftarrow T_0$
   $t \leftarrow 0$

2. repeat until $|\Delta x_i|$ is very close to 0 for all x_i
   (a) for every X_i, compute $\tilde{r_i}$ by Eq. 13
   (b) for every X_i, set $x_i = (1 + \tilde{r_i}^{-1/T})^{-1}$
   (c) $T = T * \alpha$ // $\alpha < 1$ is the cooling factor //

end

6. The Computer Experiments

A series of computer simulation experiments were conducted over two examples of noisy-or BBN to test the validity, accuracy, and efficiency of the three proposed models. Example 1 is a two-layer network of 10 roots and 10 leaves with 40 randomly generated arcs between them. To test the performance of the models with networks of complex structure, the Alarm belief network of 37 nodes (11 of which are leaves), 46 arcs, and depth 10 [2] was used in Example 2. The probabilities (prior probabilities for all roots and causal strengths for all arcs) of both example BBN were randomly generated.

In the experiments, Y was comprised of all leaves of a network, and X the rest of the nodes. Each of the $2^{|Y|}$ possible instantiations of Y was used to define a problem instance. Therefore, 1024 problem instances for Example 1 and 2048 for Example 2 were tested. For each problem instance, the solution instantiation x obtained from each of the three models was compared with the first, second, and third globally best instantiations of X obtained by some exact method.

The performance of these models depends on the network initialization and the selections of the parameters such as the initial temperature $T(0)$ and cooling factor α. After some trial-and-error attempts, these parameters were chosen as follows. For the discrete model and SA model, the networks were initialized with $x_i = 1$. For the continuous model, x_i were initialized to small random numbers. For the SA model, we chose $T(0) = 1.0$ for Example 1 and $T(0) = 10.0$ for Example 2. In both examples, $\alpha = 0.99$, and at each temperature the activations of 100 randomly selected nodes were updated, approximating the arrival at equilibrium at that temperature. For the continuous model, $T(0) = 3.0$ and $\alpha = 0.95$ were used for Example 1, and $T(0) = 10.0$ and $\alpha = 0.9$ for Example 2. The experiment results for the two examples, together with their average computing time, were given in Tables 1 and 2, respectively.

Table 1: Results for Example 1 network.

	1st	2nd	3rd	others	time
Discrete	370	58	39	557	6ms
SA	780	61	19	164	1014ms
Continuous	888	56	7	73	31ms

Table 2: Results for Example 2 network

	1st	2nd	3rd	others	time
Discrete	256	51	49	1682	7ms
SA	1101	91	94	762	2511ms
Continuous	1155	39	8	846	49ms

It is clear from the experiments that the discrete model was the fastest, but it performed poorly in terms of the accuracy the solutions it produced. The accuracy of the continuous model was comparable to the SA model, but was more than an order of magnitude faster.

7. The Conclusions

In this paper we presented three neural network-based models for MAP problems. Computer experiments suggest that the continuous model may be the most promising one as it is both fast and accurate. One thing in common for all three models is that their neural style, local computation is directly carried out on the given belief network without requiring the construction of an additional neural network.

The continuous model presented in this paper can be viewed as a mean field approximation of the SA model. In fact, if we assume $<\ln \tilde{P}(v)> = \ln <\tilde{P}(v)>$, then Eq. 13 becomes the same as the mean field equation derived by Haft *et al* for inference in general probabilistic setting [3]. It can be shown that $<\ln \tilde{P}(v)> \approx ln <\tilde{P}(v)>$. Our contribution here is the development of an operational procedure to realize the mean field style approximation which can be carried out by local computation.

One direction of future research is to investigate the scalability of the proposed neural network models. It is conceivable that the computation time for these models may be sub-exponential. However, the accuracy seems to deteriorate as the network size increases. Another promising direction of future research is to extend some of the proposed models or develop new neural style models for other types of probabilistic inference.

Acknowledgement

This work was supported in part by NSF Grant IRI-9309136 to Yun Peng.

References

[1] G. F. Cooper, "The Computational Complexity of Probabilistic Inference Using Bayesian Belief Network", *Artif. Intell.* **42** (2-3), 1990, 393-405.

[2] G. Cooper and E. Herskovits, "A Bayesian Method for the Induction of Probabilistic Networks from Data", *Machine Learning*, **9**, 1992, 309-347.

[3] M. Haft, R. Hofmann, and V. Tresp, "Mean Field Inference in a General Probabilistic Setting", *Proc. The 7th Intl. Workshop on Artificial. Intelligence And Statistics*, Ft. Lauderdale, Jan. 1999.

[4] J. J. Hopfield, "Neurons with Graded Responses Have Collective Computational Properties Like Those of Two-States Neurons", *Proceedings of the National Science Academy, USA*, **81**, 1994, 3088-3092.

[5] S. Kirkpatrick, C. Gelatt, and M. Vecchi, "Optimization by Simulated Annealing", *Science*, **220**, 1983, 671-680.

[6] J. Pearl, "Evidential Reasoning Using Stochastic Simulation of Casal models", *Artif. Intell.* **32**, 1987, 245-257.

[7] J. Pearl, *Probabilistic Reasoning in Intelligent S ystems: Networks of Plausible Inference*, Calif:Morgan Kauffman Publishers, 1988.

[8] Y. Peng and J. Reggia, *Abductive Inference Models for Diagnostic Problem-Solving*, New York: Springer-Verlag, 1990.

[9] S. Shimony, "Finding MAPs for belief networks is NP-hard", *Artif. Intell.*, *68*, 1994, 399-410