Yuuji Ichisugi and Naoto Takahashi,
An Efficient Recognition Algorithm for Restricted Bayesian Networks,
In Proc. of 2015 International Joint Conference on Neural Networks (IJCNN 2015).
http://ieeexplore.ieee.org/document/7280330/

# An Efficient Recognition Algorithm for Restricted Bayesian Networks

Yuuji Ichisugi
and Naoto Takahashi
National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba Central 2, Tsukuba, Ibaraki 305-8568, Japan
e-mail: y-ichisugi@aist.go.jp

***Abstract*—We propose an optimization method for belief propagation. First we mathematically show that the belief propagation algorithm can be optimized by imposing a reasonable restriction on the conditional probability tables in a Bayesian network. Then we demonstrate the efficiency of the proposed algorithm with experiments. Compared to the previously derived approximate algorithm, the proposed algorithm has the following features: 1. the proposed algorithm calculates more accurate maximum posterior marginal values, 2. similar to the approximate algorithm, its execution time grows only linearly against the number of edges, and 3. the proposed algorithm is slower than the approximate algorithm, but the difference between their execution time is less than twice.**

## I. Introduction

According to the recent studies in computational neuroscience, Bayesian network [1] can be the underlying mechanism of the cerebral cortex. Various neuroscientific phenomena are successfully reproduced with models based on Bayesian networks. The cerebral cortex shares many aspects, not only in functions but also in structures, with Bayesian networks [2][3][4][5][6][7].

On the other hand, the deep learning technology has been receiving attention because of its high performance in recognition tasks that are easy for human but difficult for machines. Systems that are based on deep learning have been breaking records in many competitions. It is worth pointing out that the deep learning technology is a descendant of Neocognitron [8], which is a neural circuit model that emulates the structure of the ventral stream in the visual cortex.

Considering the above two facts, it is expected that we obtain more human-like machine learning algorithms by combining Bayesian networks and the deep learning technology. Such machine learning algorithms would show better performance and higher functionality compared to the current machine learning algorithms used in recognition tasks.

There are also technical reasons for which we can expect superior man-chine learning algorithms in the combination of Bayesian networks and the deep learning technology:

- Bayesian network is a knowledge representation model that expresses causal relationship among multiple events, using probabilities. Under certain conditions, Bayesian networks can concisely express causal relationships between the signal source and the observed data; various inferences can be performed efficiently in these cases.
- Bayesian networks use not only bottom-up input information but also top-down context information to perform robust inference [2]. Which means Bayesian networks can be applied to a wider range of tasks compared to ordinary feed-forward neural networks.
- Bayesian networks can straightforwardly represent generative models with multiple layers. Therefore it is easy to design a Bayesian network that reflects prior knowledge of the target domain. It is expected that learning performance improves by designing a network so that its internal structure and the prior distribution of parameters reflect the prior knowledge. Prior knowledge can be obtained both from engineering analysis of the target domain and from neuroscientific studies, e.g. how multiple areas of the cerebral cortex are connected, etc.

There are, however, two potential problems in using Bayesian networks with the deep learning technology: 1) execution time and required memory grow rapidly as the size of the network becomes bigger, and 2) learning would converge to a local minimum or overfitting would occur.

To overcome the first problem above (i.e. complexities), we propose an algorithm that greatly accelerates the belief propagation of a Bayesian network by imposing a restriction on the conditional probability tables. We then demonstrate its positive effect with experimental results.

Although we do not discuss the second problem (i.e. local minimum/overfitting) in this paper, it can be mitigated by specifying the prior distribution of parameters.

## II. Proposed Recognition Algorithm

The main reason of the potential problems mentioned above is that the size of a conditional probability table increases exponentially against the number of parent nodes. We impose the following linear-sum restriction on the conditional probabilities to reduce the number of parameters of the Bayesian network:

$$P(x|u_1,\cdots,u_m) = \frac{1}{m}\sum_{k=1}^{m} w(x,u_k) \tag{1}$$

where $u_k$ is the value of node $X$'s parent node $U_k$, and $w$ is an arbitrary function. One of the simplest form of $w$ is

$$w(x,u_k) = P(x|u_k). \tag{2}$$

$$
\begin{aligned}
BEL(x) &= \alpha\lambda(x)\pi(x) \\
\pi(x) &= \sum_{u_1,\cdots,u_m} P(x|u_1,\cdots,u_m)\prod_k \pi_X(u_k) \\
\lambda(x) &= \prod_l \lambda_{Y_l}(x) \\
\pi_{Y_l}(x) &= \beta_1\pi(x)\prod_{j\neq l}\lambda_{Y_j}(x) \\
\lambda_X(u_k) &= \beta_2\sum_x\Big[\lambda(x) \\
&\quad \times \sum_{u_1,\cdots,u_m/u_k} P(x|u_1,\cdots,u_m)\prod_{i\neq k}\pi_X(u_i)\Big]
\end{aligned}
$$

Fig. 1. Pearl's original belief propagation algorithm. Here, $\alpha$, $\beta_1$ and $\beta_2$ are normalizing constants.

When the conditional probability tables are restricted as in Equation 1, we can transform Pearl's original belief propagation algorithm [1], which is reproduced in Fig. 1, as follows.

First, we assume that the messages from the parent nodes are normalized as

$$\sum_{u_k}\pi_X(u_k) = 1 \tag{3}$$

and therefore the equation

$$\sum_{u_1,\cdots,u_m}\prod_{i=1}^{m}\pi_X(u_i) = 1 \tag{4}$$

holds.

Next, we define $\kappa_{U_k}(x)$ as follows:

$$\kappa_{U_k}(x) \stackrel{\text{def}}{=} \sum_{u_k} w(x,u_k)\pi_X(u_k) \tag{5}$$

Now the equation of $\pi(x)$ in Fig. 1 is transformed as follows. (Note that $1/m$ is a constant.)

$$
\begin{aligned}
&\pi(x) = \sum_{u_1,\cdots,u_m} P(x|u_1,\cdots,u_m)\prod_i \pi_X(u_i) \\
&= \sum_{u_1,\cdots,u_m}(\frac{1}{m}\sum_k w(x,u_k))\prod_i \pi_X(u_i) \\
&= \frac{1}{m}\sum_{u_1,\cdots,u_m}\sum_k w(x,u_k)\pi_X(u_k)\prod_{i\neq k}\pi_X(u_i) \\
&= \frac{1}{m}\Big[\sum_{u_1,\cdots,u_m} w(x,u_1)\pi_X(u_1)\prod_{i\neq 1}\pi_X(u_i) \\
&\quad +\cdots+\sum_{u_1,\cdots,u_m} w(x,u_m)\pi_X(u_m)\prod_{i\neq m}\pi_X(u_i)\Big] \\
&= \frac{1}{m}\Big[\sum_{u_1} w(x,u_1)\pi_X(u_1)\sum_{u_1,\cdots,u_m/u_1}\prod_{i\neq 1}\pi_X(u_i) \\
&\quad +\cdots+\sum_{u_m} w(x,u_m)\pi_X(u_m)\sum_{u_1,\cdots,u_m/u_m}\prod_{i\neq m}\pi_X(u_i)\Big] \\
&= \frac{1}{m}\sum_k\sum_{u_k} w(x,u_k)\pi_X(u_k)\sum_{u_1,\cdots,u_m/u_k}\prod_{i\neq k}\pi_X(u_i) \\
&= \frac{1}{m}\sum_k\sum_{u_k} w(x,u_k)\pi_X(u_k) \\
&= \frac{1}{m}\sum_k \kappa_{U_k}(x)
\end{aligned}
$$

With the definition of $\rho(x) \stackrel{\text{def}}{=} \pi(x)\lambda(x)$, the equation of $\pi_{Y_l}(x)$ in Fig. 1 is transformed as follows.

$$
\begin{aligned}
\pi_{Y_l}(x) &= \beta_1\pi(x)\prod_{j\neq l}\lambda_{Y_j}(x) \\
&= \beta_1(\pi(x)\prod_j \lambda_{Y_j}(x))/\lambda_{Y_l}(x) \\
&= \beta_1\rho(x)/\lambda_{Y_l}(x)
\end{aligned}
$$

The term $\sum_{u_1,\cdots,u_m/u_k} P(x|u_1,\cdots,u_m)\prod_{i\neq k}\pi_X(u_i)$ in the equation of $\lambda_X(u_k)$ in Fig. 1 is transformed as follows. Here we use the same transformation technique that we use to transform $\pi(x)$.

$$
\begin{aligned}
&\sum_{u_1,\cdots,u_m/u_k} P(x|u_1,\cdots,u_m)\prod_{i\neq k}\pi_X(u_i) \\
&= \sum_{u_1,\cdots,u_m/u_k}\frac{1}{m}(\sum_{j\neq k} w(x,u_j)+w(x,u_k))\prod_{i\neq k}\pi_X(u_i) \\
&= \frac{1}{m}\Big[\sum_{u_1,\cdots,u_m/u_k}\sum_{j\neq k} w(x,u_j)\prod_{i\neq k}\pi_X(u_i) \\
&\quad +w(x,u_k)\sum_{u_1,\cdots,u_m/u_k}\prod_{i\neq k}\pi_X(u_i)\Big] \\
&= \frac{1}{m}(m\pi(x)-\sum_{u_k} w(x,u_k)\pi_X(u_k)+w(x,u_k)) \\
&= \frac{1}{m}(m\pi(x)-\kappa_{U_k}(x)+w(x,u_k))
\end{aligned}
$$

Then we can calculate $\lambda_X(u_k)$ as follows.

$$
\begin{aligned}
&\lambda_X(u_k) \\
&= \beta_2\sum_x\lambda(x)\sum_{u_1,\cdots,u_m/u_k} P(x|u_1,\cdots,u_m)\prod_{i\neq k}\pi_X(u_i) \\
&= \frac{\beta_2}{m}\sum_x\lambda(x)(m\pi(x)-\kappa_{U_k}(x)+w(x,u_k))
\end{aligned}
$$

Replacing $m\pi(x)$ with $\pi(x)$ (as $\pi(x)$ is rescalable), combining all the results above, reassigning normalizing constants and adding the indices $t$ and $t+1$ so that the equations are applicable to loopy networks, we get the following algorithm, which we call Optimized Original Belief Propagation (OOBP).

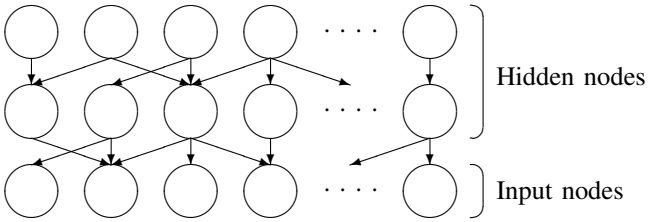



Fig. 2. The generalized structure of the Bayesian networks used for the evaluation. All layers contain the same number of nodes. Edges exist only between two adjacent layers. All nodes have two states. The conditional probability tables are randomly set before each recognition trial.

$$
\begin{aligned}
\lambda_{Y_l}^{t+1}(x) &= \beta_2 \sum_{y_l} \lambda^t(y_l)(\pi^t(y_l) - \kappa_X^t(y_l) + w(y_l, x)) \\
\lambda^{t+1}(x) &= \prod_{l=1}^{n} \lambda_{Y_l}^{t+1}(x) \\
\pi_{Y_l}^{t+1}(x) &= \beta_1 \rho^{t+1}(x) / \lambda_{Y_l}^{t+1}(x) \\
\kappa_{U_k}^{t+1}(x) &= \sum_{u_k} w(x, u_k) \pi_X^t(u_k) \\
\pi^{t+1}(x) &= \sum_{k=1}^{m} \kappa_{U_k}^{t+1}(x) \\
\rho^{t+1}(x) &= \lambda^{t+1}(x) \pi^{t+1}(x) \\
BEL^{t+1}(x) &= \alpha \rho^{t+1}(x)
\end{aligned} \tag{6}
$$

Note that, unlike done in [3], we do not use any approximation here.

## III. Evaluation of the Proposed Algorithm

### A. Evaluation Method

We evaluated the algorithm proposed in this article by comparing it with the approximate algorithm derived in [3] and with the original loopy belief propagation. All algorithms were implemented in Java [1] and executed on a laptop computer with an Intel Core 2 Duo P8600 (2.4GHz) CPU.

The Bayesian networks used for the evaluation have a layered structure as shown in Fig. 2. This structure is an implementation of a cerebral cortex model called BESOM (BidirEctional Self-Organizing Map). [6]

We used three measured values for evaluation: *correct rate*, *convergence rate* and *mean iteration steps*.

The *correct rate* is defined as:

$$\textit{correct rate} = \frac{h_{cor}}{h_{all}} \tag{7}$$

where $h_{cor}$ is the number of hidden nodes that converged to the correct maximum posterior marginal (MPM) values [2] and $h_{all}$ is the total number of hidden nodes.

The *convergence rate* is defined as:

$$\textit{convergence rate} = \frac{t_{con}}{t_{all}} \tag{8}$$

where $t_{con}$ is the number of recognition trials in which all hidden nodes converged within a predetermined iteration steps (set to 50 for this article) and $t_{all}$ is the total number of recognition trials.

The *mean iteration steps* is defined as:

$$\textit{mean iteration steps} = \frac{1}{t_{con}} \sum_{i=1}^{t_{con}} s_i \tag{9}$$

where $s_i$ is the iteration steps at which the network converged in the $i$-th converged recognition trial. Recognition trials that did not converge are excluded. This measure is meaningful only for network configurations that achieve high convergence rates.

### B. Accuracy for Small Networks

First we show the accuracy of the proposed algorithm. Fig. 3 and Fig. 4 plot the mean measured values of 100 recognition trials. Only networks of small sizes were used for evaluation because calculation of strict MPM values requires long execution time. (See footnote 2.)

The proposed algorithm gives better *correct rates* compared to the previously derived algorithm that uses approximation, thanks to its more accurate calculation. Since the *correct rate* of the proposed algorithm and that of the original loopy belief propagation are always almost the same, we can confirm that the proposed algorithm and the original loopy belief propagation are mathematically equivalent. (Fig. 3 A and Fig. 4 D)

All the *convergence rates* are close to 1.0, which means both proposed and approximate algorithms have stable behaviour and seldom oscillate. When the number of layers increases, however, the *convergence rate* of the approximate algorithm decreases slightly. (Fig. 3 B and Fig. 4 E)

With shallow networks, the proposed algorithm requires slightly greater *mean iteration steps* than the approximate algorithm. The difference becomes clearer as the number of nodes increases. (Fig. 3 C)

With narrow networks, both proposed and approximate algorithms require almost the same *mean iteration steps* if the number of layers is less than or equal to 4. When the number of layers reaches to 5, the approximate algorithm requires more *mean iteration steps* compared to the proposed algorithm. The reason for this phenomenon remains to be elucidated. (Fig. 4 F)

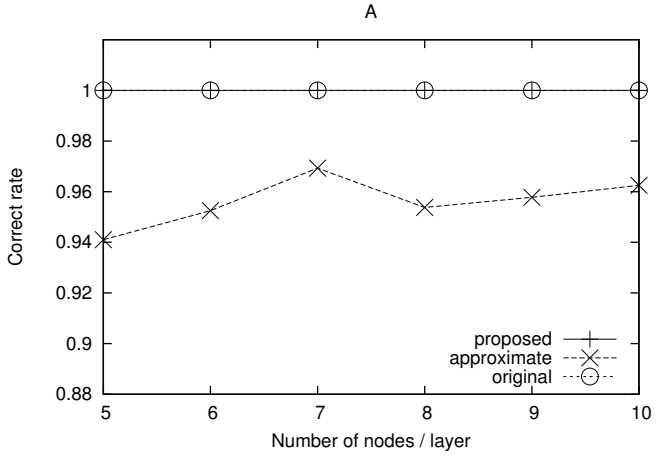
### C. Scalability

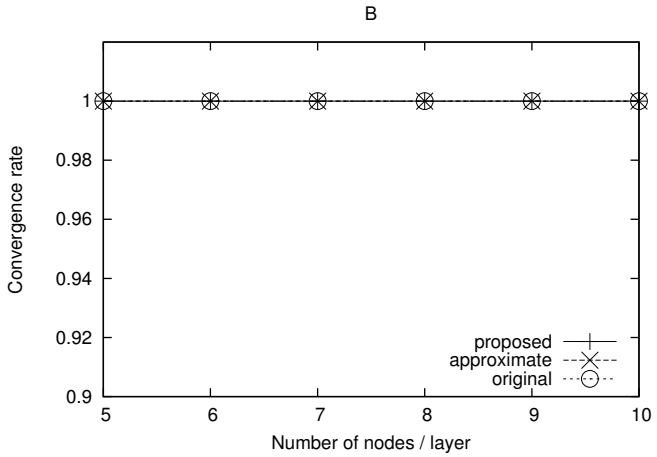
Next, we show the scalability of the proposed algorithm.

Fig. 5 plots execution time for networks that have 3 layers. The number of nodes in a layer varies between 100 and 900. Both algorithms show execution time that grows almost linearly against the number of nodes in a layer. Compared to the approximate algorithm, the proposed algorithm requires

[1] The source code is available at:
https://staff.aist.go.jp/y-ichisugi/besom/download.html

[2] We implemented yet another algorithm to find the "correct" MPM values. This algorithm calculates the strict MPM values based on naive full search and requires long calculation time, which increases exponentially to the number of nodes. This naive algorithm was used solely for strict MPM calculation and not a target for the current evaluation.



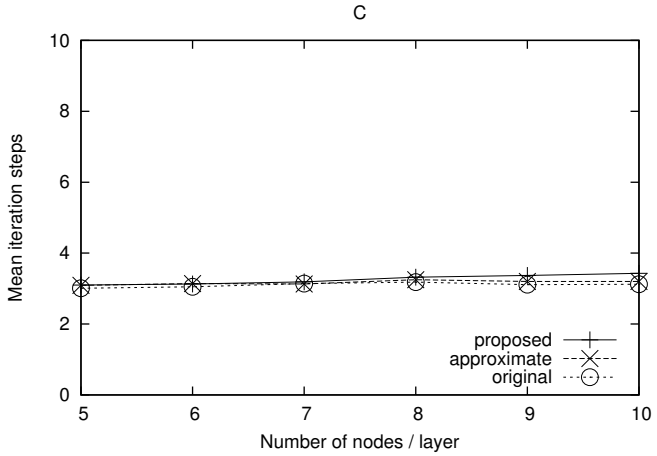
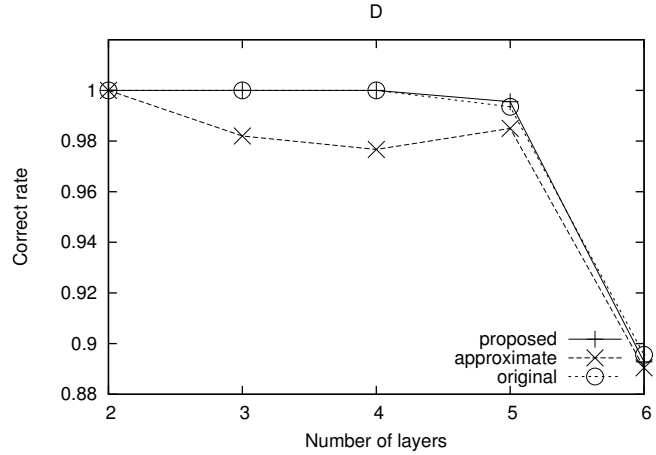
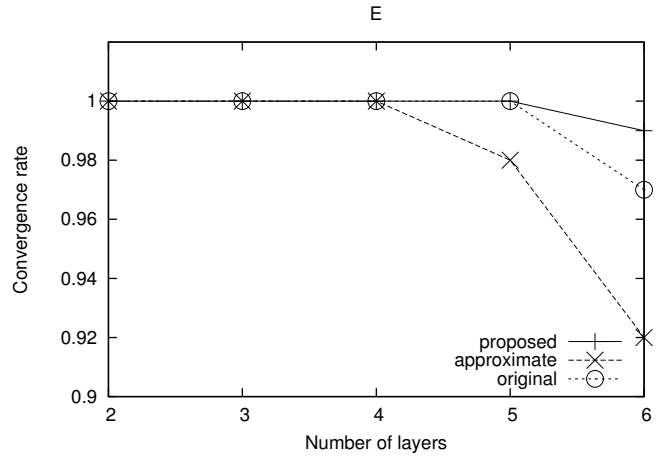




Fig. 3. Recognition results of shallow networks: (A) correct rates, (B) convergence rates and (C) mean iteration steps. The number of layers is fixed to 3. The number of nodes in a layer varies between 5 and 10. Each hidden node is connected to randomly selected 5 child nodes allowing duplication.



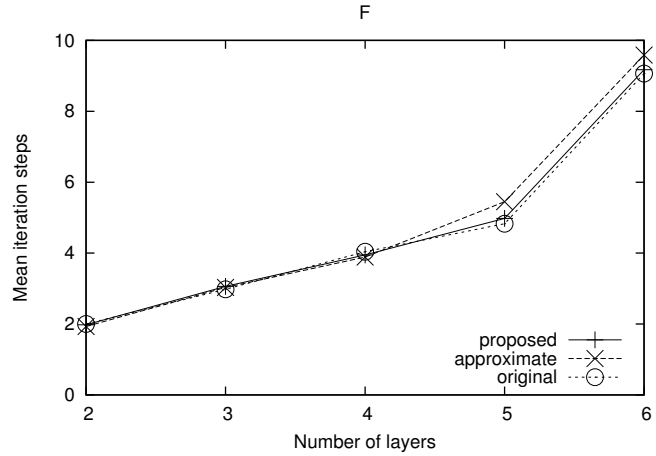




Fig. 4. Recognition results of narrow networks: (D) correct rates, (E) convergence rates and (F) mean iteration steps. The number of nodes in a layer is fixed to 5. The number of layers varies between 2 and 6. Each hidden node is connected to every node in the layer below.

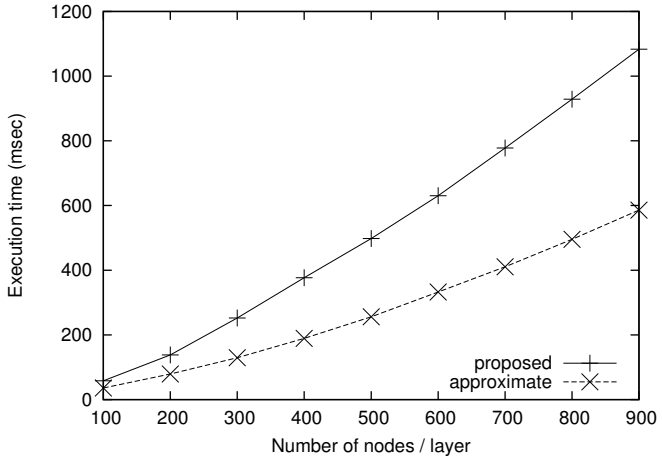



Fig. 5. Execution time of the proposed and the approximate algorithms for shallow and wide networks. The number of layers is fixed to 3. The number of nodes in a layer varies between 100 and 900. Each hidden node is connected to randomly selected 20 child nodes allowing duplication.

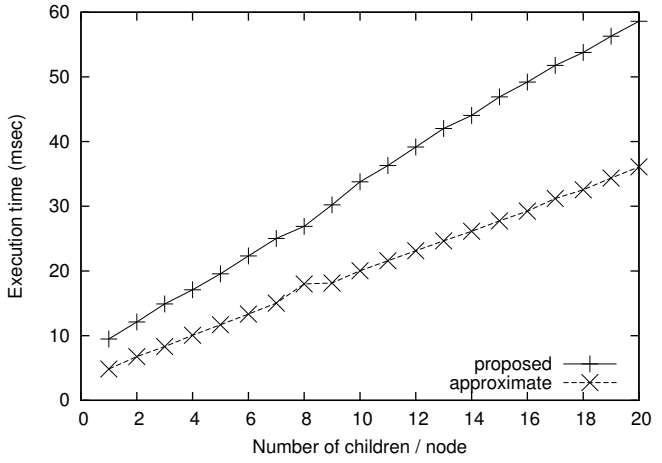



Fig. 6. Execution time of the proposed and the approximate algorithms for shallow networks. The number of layers and the number of nodes in a layer are fixed to 3 and 100, respectively. Each hidden node is connected to randomly selected $n$ child nodes allowing duplication, where $n$ varies between 1 and 20.

longer execution time, but the difference is suppressed less than twice.

Fig. 6 is another graph that indicates the scalability of the proposed algorithm. For this time, the number of layers and the number of nodes in a layer are fixed, but the number of child nodes varies. Again, execution time of the proposed algorithm grows almost linearly against the number of child nodes.

### *D. Digit Recognition*

We also performed a preliminary experiment of handwritten digit recognition using MNIST [9] to estimate applicability of the proposed algorithm.

Fig. 7 shows the structure of the Bayesian network used for this purpose. The network has a layered structure as the networks used in the evaluation above, but the number of nodes and the number of states differ with layers. The input layer

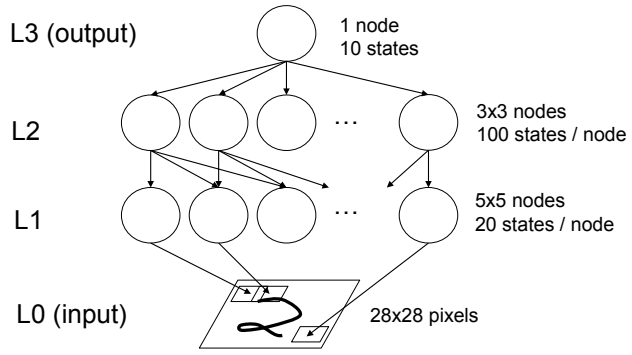



Fig. 7. The structure of the Bayesian network used for handwritten digit recognition. The input layer consists of 28 by 28 pixels. The first hidden layer contains 25 nodes with 20 states. The second hidden layer contains 9 nodes with 100 states.

consists of 28 by 28 pixels. The first hidden layer consists of 5 by 5 nodes and the second 3 by 3. Each node has 20 states in the first hidden layer and 100 in the second. Each hidden node is connected to nodes in the layer below to have a local receptive field as in general deep learning networks. The output layer has only one node of 10 states; each state corresponds to a label, i.e. one of the ten digits.

We first trained the network using the online version of the EM algorithm, giving the label to the output node. Then we tested its performance using the proposed algorithm. The label that had the largest posterior probability in the output node was selected as the output.

Its recognition accuracy was about 91% without pre-training. Although this result is not satisfactory as a practical machine learning algorithm, we should be able to increase the accuracy by improving the model of conditional probability tables and tuning hyper parameters.

## IV. Related Work

Researchers often impose restrictions on the conditional probability tables and/or the network structure of a Bayesian network for the purpose of decreasing the number of parameters and increasing the accuracy of recognition.

The noisy-or model [1] not only concisely represents causal relationships among binary variables but also reduces computational cost by optimizing the algorithm. This model straightforwardly represents the situation in which one of the causal events should have occurred when an event is observed. Situations of this kind are common in the real world.

George and Hawkins' cerebral cortex model [2] restricts the network to tree structure to avoid explosion of memory size and computational cost. However, tree structure looks insufficient as a model of the cerebral cortex; the actual cerebral cortex is definitely not of the form of tree structure.

Hosoya [4] successfully reproduces multiple phenomena found in the visual cortex by using conditional probability tables based on softmax. The used learning method is Markov chain Monte Carlo (MCMC) and not the belief propagation algorithm.

Dura-Bernal et al. [5] develop a visual perception model using two different optimization techniques. The first one is restricting conditional probability tables, which is similar to the one proposed in this article. They optimize message calculation, which is also similar to our method, but their optimization is done only partially. Their second optimization technique is introducing approximation in messages calculation. This second one is independent of the method proposed in this article, therefore it should be possible to merge this second technique with our method to accelerate calculation.

## V. Conclusion

We first mathematically showed that the belief propagation algorithm can be optimized by imposing a restriction on the conditional probability tables. Then we demonstrated the efficiency of the proposed algorithm with experiments. Compared to the previously derived approximate algorithm, the proposed algorithm has the following features:

1) The proposed algorithm calculates more accurate maximum posterior marginal values.
2) Similar to the approximate algorithm, its execution time grows only linearly against the number of edges.
3) The proposed algorithm is slower than the approximate algorithm, but the difference between their execution time is less than twice.

As already reported [6], our earlier version of cerebral cortex model, which satisfies the same restriction on the conditional probability tables as in this article, resulted in having variables that correspond to the six-layers and columnar-structure in the cerebral cortex and relation among variables that corresponds to connections between areas of the cortex. Their resemblance is remarkable and unlikely to be a mere coincidence. Therefore we believe that Equation 1 is a fairly reasonable restriction.

However, it is also clear that the restriction limits the ability of the Bayesian network to which it applies. We are now investigating its theoretical limits and trying to find a looser restriction for better performance.

## References


[1] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, 1988.

[2] D. George and J. Hawkins, A hierarchical Bayesian model of invariant pattern recognition in the visual cortex, In Proc. of International Joint Conference on Neural Networks (IJCNN2005), Vol.3, pp.1812–1817, 2005.

[3] Y. Ichisugi, Recognition Model of Cerebral Cortex based on Approximate Belief Revision Algorithm, In Proc. of International Joint Conference on Neural Networks (IJCNN2011), pp.386–391, 2011.

[4] H. Hosoya, Multinomial Bayesian Learning for Modeling Classical and Nonclassical Receptive Field Properties, Neural Computation, Vol.24, No.8, pp.2119–2150, 2012.

[5] S. Dura-Bernal, T. Wennekers and S.L. Denham, Top-Down Feedback in an HMAX-Like Cortical Model of Object Perception Based on Hierarchical Bayesian Networks and Belief Propagation, PLOS ONE, Vol.7, No.11, 2012.

[6] Y. Ichisugi, The cerebral cortex model that self-organizes conditional probability tables and executes belief propagation, In Proc. of International Joint Conference on Neural Networks (IJCNN2007), pp.1065–1077, 2007.

[7] F. Roehrbein, J. Eggert and E. Koemer, Bayesian Columnar Networks for Grounded Cognitive System, In Proc. of the 30th Annual Conference of the Cognitive Science Society, pp.1423–1428, 2008.

[8] K. Fukushima, Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, Biological Cybernetics, Vol.36, pp.193–202, 1980.

[9] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, Gradient-Based Learning Applied to Document Recognition, In Proc. of the IEEE, 86(11), 2278–2324 1998.