To appear in proceedings of IMACS World Congress ’97

Decimatable Boltzmann Machines for Diagnosis: Efficient Learning and Inference

Stefan M. Rüger[*]
Technische Universität Berlin, Fachbereich Informatik
Sekr. FR 5-9, Franklinstr. 28/29, D-10 587 Berlin, Germany
e-mail: async@cs.tu-berlin.de

Keywords: Boltzmann machine, decimation, diagnosis, inference, partition sum

ABSTRACT

By using hidden nodes, Boltzmann machines can be employed to approximate probability distributions, store stochastic knowledge extracted from examples (a. k. a. learning) and retrieve the stored statistical information (a. k. a. inference or diagnosis). However, in general, the algorithms for such operations are NP-hard. Efficient algorithms can therefore only be expected to be found in special cases. Such algorithms have recently been specified for a certain set of so-called *decimatable* Boltzmann machines [9].
This approach is demonstrated for a small diagnosis system that was introduced in [5] as a didactic example: it deals with the joint probability of several medical variables. We show that a batch of fully diagnosed examples of medical data is sufficient not only to store the underlying stochastic correlations in a decimatable Boltzmann machine, but also to use this for inference or diagnosis. We argue that models like those obtained by decimatable Boltzmann machines — which learn from given examples only — may also help in building medically relevant models.

1 INTRODUCTION

Boltzmann machines [3] were the first explicitly stochastic networks for which a learning rule [1] was developed. The learning rule has a simple structure and is local in the sense that only items of neighbouring nodes are used. As an important special case of undirected graphical models [4], Boltzmann machines are even more interesting. They can be implemented in a massively parallel manner and have a strong theoretical rooting in thermodynamics.
Once in equilibrium, the random variable of activations associated with the nodes of a Boltzmann machine follows a parameterised Boltzmann-Gibbs distribution, which is already known from the canonical formalism of thermodynamics. By introducing hidden nodes, a marginal distribution arises at the visible nodes. Such marginal distributions can be used to approximate probability distributions. The learning rule utilises examples that describe a desired distribution empirically in order to adapt the parameters of the Boltzmann machine to obtain a better approximation. Stochastic relations that are given solely by examples can be stored in the parameter vector, a. k. a. weight vector, of a Boltzmann machine.
Usually, the set of nodes of a Boltzmann machine is partitioned into input, hidden and output nodes, and the bias node. Learning a weight vector v from input/output examples results in a conditional

[*] as of 1 May 1997: Imperial College, Dept. of Computing, Huxley Building, 180 Queen’s Gate, London SW7 2BZ

probability of the activations of the hidden nodes, given both input and output. Inference means computing the conditional probability of output, given input (and a previously learned and now fixed weight vector v). Diagnosis, on the other hand, corresponds to the conditional probability of input, given an observable output. As the underlying graph of a Boltzmann machine is undirected, the roles of input and output nodes are interchangeable. Thus, diagnosis and inference are basically the same tasks in Boltzmann machines.
Traditionally, the relevant quantities for inference and learning are estimated by simulations using ergodic theory (Gibbs sampling). Many efforts, such as mean-field approximation [6], have since been made to overcome the — even with modern computers — comparatively slow Gibbs sampling of a Boltzmann machine. One especially interesting technique, which has been borrowed from statistical mechanics [10], is decimation: here, the interaction of two nodes can be calculated exactly by reducing the rest of the network in such a way that the interaction is not disturbed. This cannot be done in arbitrarily complex networks, but it works in tree-like structures, such as Boltzmann trees and chains. In [9], this idea has been taken up and was extended by developing all possible decimation rules. This has led not only to a definition of a tractable set of decimatable Boltzmann machines (they contain tree-like structures as a special case) but also to powerful inference and learning rules. Section 2 summarises this approach.

2 DECIMATABLE BOLTZMANN MACHINES

The underlying structure of a Boltzmann machine is given by a set $N = \{0, \dots, n\}$ of nodes and a set of edges $E \subset \{(i,j) \in N \times N | i < j\}$ together with a weight vector $v \in \mathbb{R}^E$. The component v_{ab} is defined as the weight associated with the edge $(\min(a,b), \max(a,b))$. Usually, N is partitioned into four mutually disjoint sets I, H, O and $\{0\}$, denoting the input, hidden and output nodes, and the bias node, respectively. Let $N_* := N \setminus \{0\}$; at every time-step, the Boltzmann machine produces a random bipolar activation vector $s \in \{-1,1\}^{N_*}$, the sequence of which may be viewed as a Markov chain. Once in equilibrium, s obeys a Boltzmann-Gibbs distribution

$$P_v(s) := \frac{\exp(-h(v,s))}{Z_v} \quad \text{with} \quad h(v,s) := -\sum_{(i,j)\in E} s_i v_{ij} s_j \text{ and } Z_v := \sum_s \exp(-h(v,s)), \tag{1}$$

where $s_o := 1$ is the constant activation of the bias node 0. The normalisation term Z_v is also known as the partition sum.
Clamping values ± 1 to input nodes (and sometimes to output nodes) changes the Boltzmann-Gibbs distribution in an obvious way: clamping a value s_i to the node i corresponds to changing the bias weights v_{oj} by $s_i v_{ij}$ of all those nodes j with $(i,j) \in E$ or $(j,i) \in E$. At the same time, the node i, together with all edges connected to i, can be disregarded. If a complete pattern $\xi \in \{-1,1\}^X$ with $X \subset N_*$ is clamped, a new Boltzmann machine is created whose Boltzmann-Gibbs distribution is the conditional Boltzmann-Gibbs distribution of the original Boltzmann machine, given ξ. We denote all quantities of the new Boltzmann machine with Greek superscripts, e. g., $N^\xi := N \setminus X$, $P^\xi_{v^\xi}(s^\xi) := P_v(s|\xi)$, etc.
One possible application of Boltzmann machines is in marginalising the Boltzmann-Gibbs distribution over all possible activations of the hidden nodes — thus obtaining a distribution p_v from P_v — to approximate a desired distribution r of activations in $\{-1,1\}^{I \cup O}$. Let $\alpha \in \{-1,1\}^I$, $\beta \in \{-1,1\}^H$ and $\gamma \in \{-1,1\}^O$ denote subvectors of the activation vector $s = \alpha\beta\gamma$, and let $q(\alpha) = \sum_\gamma r(\alpha\gamma)$ be the probability of an input pattern α. Assuming that the distribution of input patterns is *not* to be learned, the information gain

$$\mathrm{IG}(r, p_v) = \sum_\alpha q(\alpha) \sum_\gamma r(\gamma|\alpha) \log\left(\frac{r(\gamma|\alpha)}{p_v(\gamma|\alpha)}\right) \tag{2}$$

is a commonly used error measure for the deviation of r with respect to p_v. Gradient descent for (2) results in the learning rule

$$\Delta v_{ij} = -\eta \left(\nabla (v \mapsto \mathrm{IG}(r, p_v)) \right)_{ij} = \eta \Big(\overline{\langle s_i s_j \rangle_{\alpha\gamma}}^{r} - \overline{\langle s_i s_j \rangle_{\alpha}}^{q} \Big), \tag{3}$$

where $\eta \in \mathbb{R}^+$ is the learning rate. The brackets $\langle \cdot \rangle_\xi$ denote the expectation value w. r. t. the conditional Boltzmann-Gibbs distribution P_v^ξ, and $\overline{\cdot}^p$ denotes the expectation value w. r. t. p or, if p is given by samples, the average over the samples. The expectation value of $s_i s_j$ can also be expressed by

$$\langle s_i s_j \rangle_\xi = \frac{\partial \log(Z^\xi_{v^\xi})}{\partial v^\xi_{ij}}. \tag{4}$$

Inference means the computation of the conditional probability

$$p_v(\gamma|\alpha) = \sum_\beta P_v(\beta\gamma|\alpha) \tag{5}$$

with $\alpha \in \{-1,1\}^X$, $X \supset I$, $\gamma \in \{-1,1\}^Y$, $Y \subset (I \cup O) \setminus X$ and $\beta \in \{-1,1\}^{N_* \setminus (X \cup Y)}$. The idea is that all knowledge of the activations is clamped, and the probability of the activations of interest, here of nodes in Y, is queried. The condition $X \supset I$ originates from the fact that the information gain (2) does not model the probability of input patterns. This is no loss of generality, since $I = \emptyset$ could have been chosen at the outset.
Inserting the definition of conditional probabilities, it holds that for every $\beta \in \{-1,1\}^H$

$$p_v(\gamma|\alpha) = \frac{P_v(\beta\gamma|\alpha)}{P_v(\beta|\alpha\gamma)} = \exp\Big(\sum_{(0,c)\in E} \gamma_c v_{oc} + \sum_{\substack{(a,c)\in E \\ a\in I, c\in O}} v_{ac}\alpha_a\gamma_c + \sum_{\substack{(c,d)\in E \\ c,d\in O}} v_{cd}\gamma_c\gamma_d \Big) \frac{Z^{\alpha\gamma}_{v^{\alpha\gamma}}}{Z^{\alpha}_{v^{\alpha}}}. \tag{6}$$

Equations (6) and (4) link the important quantities for inference (5) and learning (3) to the partition sum; the transformation is polynomial in $|N|$ in both cases. These and all other results of the preceding paragraphs are valid for general bipolar Boltzmann machines. In principle, Z_v can be calculated by summing all $2^{|N_*|}$ terms $\exp(-h(v,s))$, which is arguably intractable in $|N|$: [2] has proved that inference in the related belief networks is NP-hard, and this proof can be transferred to Boltzmann machines. We conclude that research should be directed away from the search for efficient inference and learning rules in general Boltzmann machines, and towards the design of special-case algorithms. An attractive special case are networks, where one node after the other can be replaced by edges and weights in such a way that in each step the probability distribution of the remaining network's activation vector coincides with the marginal distribution of the original network's activation vector. Figure 1 shows the removal of a node $a_1 \neq 0$ with all corresponding edges and the insertion of all edges (a_i, a_j) between the neighbour nodes of a_1. The aim is to calculate the new weights $v'_{a_i a_j}$ such that

$$\sum_{s_{a_1}\in\{-1,1\}} P_v(s_{a_1}, s_{a_2}, s_{a_3}, \ldots) = P'_{v'}(s_{a_2}, s_{a_3}, \ldots), \tag{7}$$

where a_2, a_3 and a_4 may be embedded in an arbitrarily complex network. One of a_2, a_3 or a_4 may even be the bias node. Ansatz (7) can be solved for $k = 4$ with

$$v'_{a_2a_3} = \log\left(\frac{\cosh(v_{a_1a_2} + v_{a_1a_3} - v_{a_1a_4})\cosh(v_{a_1a_2} + v_{a_1a_3} + v_{a_1a_4})}{\cosh(v_{a_1a_2} - v_{a_1a_3} + v_{a_1a_4})\cosh(v_{a_1a_2} - v_{a_1a_3} - v_{a_1a_4})} \right)/4 \tag{8}$$

$$v'_{a_2a_4} = \log\left(\frac{\cosh(v_{a_1a_2} - v_{a_1a_3} + v_{a_1a_4})\cosh(v_{a_1a_2} + v_{a_1a_3} + v_{a_1a_4})}{\cosh(v_{a_1a_2} + v_{a_1a_3} - v_{a_1a_4})\cosh(v_{a_1a_2} - v_{a_1a_3} - v_{a_1a_4})} \right)/4 \tag{9}$$

$$v'_{a_3a_4} = \log\left(\frac{\cosh(v_{a_1a_2} + v_{a_1a_3} + v_{a_1a_4})\cosh(v_{a_1a_2} - v_{a_1a_3} - v_{a_1a_4})}{\cosh(v_{a_1a_2} + v_{a_1a_3} - v_{a_1a_4})\cosh(v_{a_1a_2} - v_{a_1a_3} + v_{a_1a_4})} \right)/4, \tag{10}$$

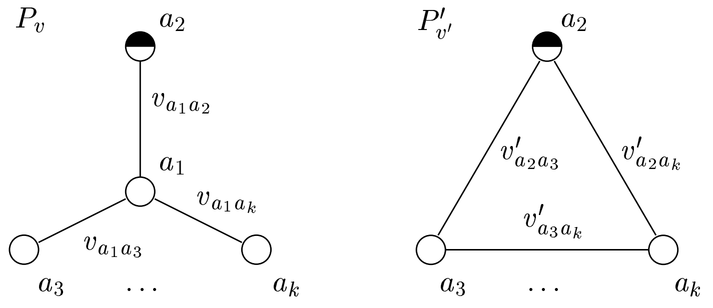


Figure 1: Ansatz for decimation of the node a_1

and the partition sum is changed by the factor

$$\frac{Z_v}{Z'_{v'}} = 2 \left(\prod_{s_{a_3}, s_{a_4}} \cosh(v_{a_1 a_2} + v_{a_1 a_3} s_{a_3} + v_{a_1 a_4} s_{a_4}) \right)^{1/4} . \tag{11}$$

The cases in which $k \leq 3$ are special cases of $k = 4$ with appropriate weights $v_{a_1 a_i}$ set to zero. In general, ansatz (7) cannot be solved for $k > 4$. If necessary, a new weight $v'_{a_i a_j}$ arising from (8)–(10) must be added to an existing weight $v_{a_i a_j}$ *(parallel rule)*. Otherwise, a new edge (a_i, a_j) has to be inserted.

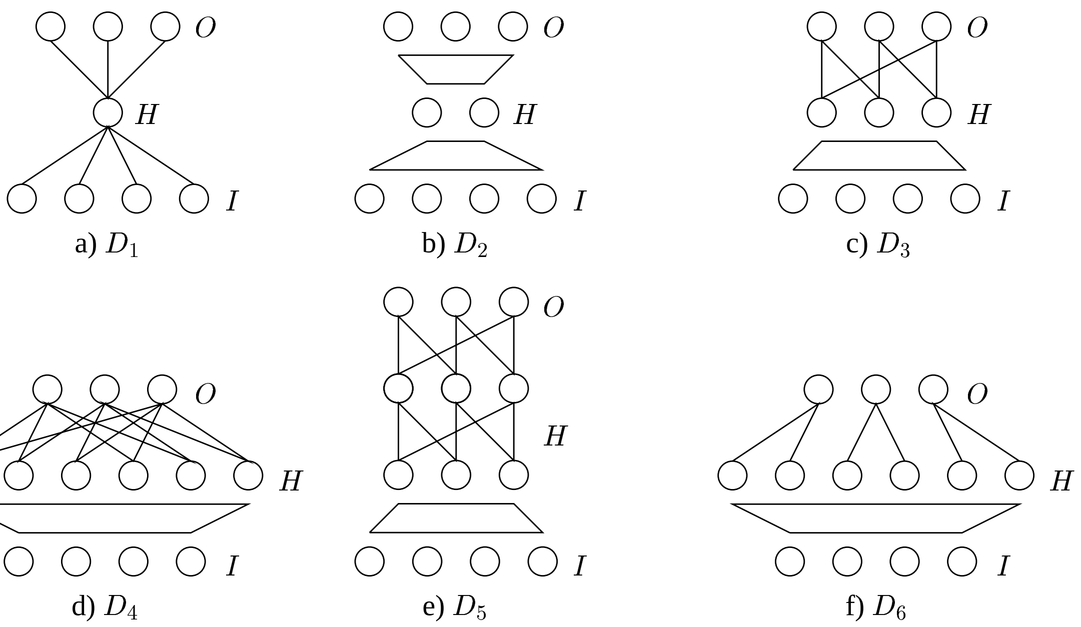


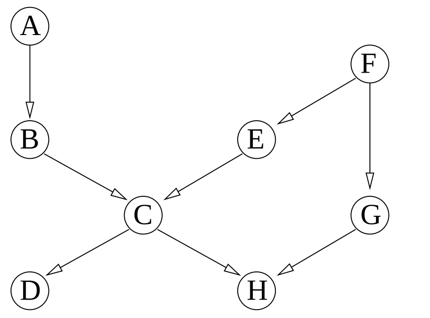
Figure 2: Some decimatable Boltzmann machines (bias nodes not shown)

Figure 2 shows examples of Boltzmann machines whose partition sum can be computed (after clamping input or input and output nodes) by decimating the resulting Boltzmann machine step by step until the trivial Boltzmann machine $\{0\}$ is reached. Such Boltzmann machines seem to be ideal candidates for a sufficiently interesting class of tractable networks — we call them *decimatable*.
The decimation of a Boltzmann machine factorises the partition sum in such a way that in each decimation step one factor can be computed. One natural way to do the bookkeeping for the computation of the total partition sum is to formulate this process as a mapping network. Given a Boltzmann machine N together with an order in which the nodes can be decimated, there is a uniquely defined mapping network $\tilde{N}$ that computes the log partition sum from the weights of the Boltzmann machine using (8)–(11). This mapping network is called *adjoint network*.
An adjoint network has a clamped weight vector v^ξ as input and calculates $\log(Z^\xi_{v^\xi})$. A generalisation of the backpropagation algorithm allows computation of the gradient $\nabla(v^\xi \mapsto \log(Z^\xi_{v^\xi}))$, which is

needed for the learning rule (3), in a simple forward-backward pass [9]. A combination of two such networks (one with $\xi = \alpha$, and one with $\xi = \alpha\gamma$) allows efficient inference (6), which in turn can be used for computing the information gain (2). Hence, all established convex-optimisation methods [7] can be used or, e. g., stable dynamic learning-rate adaptation [8], which uses comparatively few but well-chosen evaluations of the information gain. What is more, the decimation-based algorithms of this section are exact, as opposed to algorithms based on Gibbs sampling.

3 A DIDACTIC EXAMPLE

Figure 3 shows a didactic example of [5]. It represents a fictitious medical model for shortness of breath (dyspnœa). Tuberculosis and lung cancer may be indicated by a positive chest X-ray; both can cause dyspnœa. The same is true of bronchitis. A recent visit to Asia increases the probability of contracting tuberculosis, while smoking is a possible cause of both lung cancer and bronchitis. Endowing the graph shown in Figure 3 with conditional probabilities associated to every node (given its parents) creates a belief network from which the joint probability of all nodes can be computed.



A: visit to Asia
B: tuberculosis
C: E or B
D: positive X-ray
E: lung cancer
F: smoker
G: bronchitis
H: dyspnœa

Figure 3: Belief network for dyspnœa

Usually, a belief network models a joint probability very precisely. However, in many real-world applications, at the outset no good model of the probability distribution exists. Instead, we have a more or less large multi-set of examples. Imitating this, we generated an artificial pattern multi-set R from a specific Bayesian network. Our goal was to use Boltzmann machines to approximate the probability density given by the pattern multi-set. Variables that are subject to a diagnosis (e. g., B, E and G) are modelled as output variables, and those that can easily be observed as input variables (A, D, F and H). The variable C is subsumed under hidden nodes of the Boltzmann machine.
An efficient decimation-based algorithm from the previous section was used to train the architecture D_4 of Figure 2 with the pattern set R that describes the desired distribution r. Figure 4 shows the deviation of the probability distribution p_w with respect to r after a few learning epochs in the form of a histogram. Longer learning would have resulted in a better approximation.

4 DISCUSSION

This article emphasises the importance of decimatable Boltzmann machines. One result is that both inference and improved convex-optimisation learning can be expressed in terms of the partition sum. In decimatable Boltzmann machines the partition sum (which is, in general, intractable) can be calculated with a complexity that is linear in the number of nodes, once the deterministic adjoint network has been constructed. Moreover, the gradient of the log partition sum can be computed in a simple forward-backward fashion. *Decimatable Boltzmann machines offer all the benefits of recurrent and stochastic networks — and that at the cost of a deterministic feedforward network!*
We have demonstrated that decimatable Boltzmann machines can be used to approximate probability distributions, to store knowledge about stochastic relations given by examples, and to carry out inference in such databases, even in the presence of missing values.

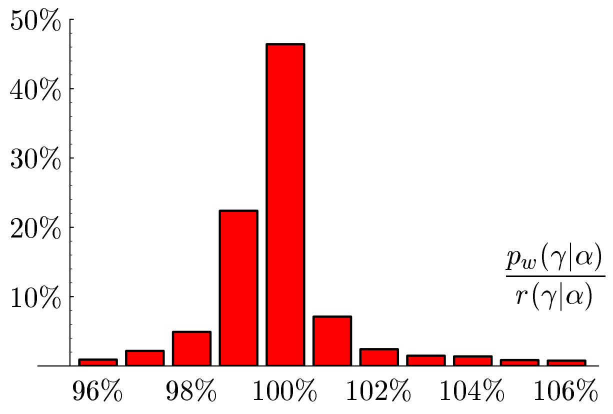


Figure 4: Histogram of the relative error of inference

Any *missing* edge implies a certain stochastic conditional independence in the resulting Boltzmann-Gibbs distribution. One open question is whether one can always find additional sparsely connected hidden nodes, which ensure that the Boltzmann machine remains decimatable and which help in approximating the desired probability distribution. Although the task of finding an *optimal* architecture may be intractable, one would like to have an algorithm that finds a *sufficiently good* decimatable Boltzmann architecture. These and other issues are left for further studies.

REFERENCES

[1] D. H. Ackley, G. E. Hinton and T. J. Sejnowski. *A learning algorithm for Boltzmann machines*. Cognitive Science 9, p. 147 (1985).

[2] G. F. Cooper. *The computational complexity of probabilistic inference using Bayesian belief networks*. Artificial Intelligence 42, p. 393 (1990).

[3] G. E. Hinton and T. J. Sejnowski. *Optimal perceptual inference*. In “Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition”, p. 448. IEEE (1983).

[4] Michael I. Jordan and Christopher M. Bishop. *Neural networks*. Computing Surveys (in press).

[5] Steffen L. Lauritzen and David J. Spiegelhalter. *Local computations with probabilities on graphical structures and their application to expert systems* (with discussion). Journal of the Royal Statistical Society B 50, p. 157 (1988).

[6] C. Peterson and J. R. Anderson. *A mean field theory learning algorithm for neural networks*. Complex Systems 1, p. 995 (1987).

[7] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press (1988).

[8] S. M. Rüger. *Stable dynamic parameter adaptation*. In D. Touretzky, M. Mozer and M. Hasselmo, editors, “Advances in Neural Information Processing Systems 8”, p. 225. MIT Press (1996).

[9] S. M. Rüger. *Efficient inference and learning in decimatable Boltzmann machines*. Technical Report 97-5, Fachbereich Informatik der Technischen Universität Berlin (1997).

[10] L. K. Saul and M. I. Jordan. *Learning in Boltzmann trees*. Neural Computation 6(6), p. 1174 (1994).