Bericht 97-5 des Fachbereichs Informatik der Technischen Universität Berlin

Efficient Inference and Learning in Decimatable Boltzmann Machines

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

February 7, 1997

Abstract

Inference and learning in general Boltzmann machines are NP-hard problems. Several restricted Boltzmann machines can be handled efficiently with the decimation technique known from statistical mechanics. A set of thus decimatable Boltzmann machines is defined. We show that the Fourier-Stieltjes transformation of the probability distribution given by a Boltzmann machine is closely related to its partition sum. Decimation allows for the efficient calculation of the partition sum with the help of an adjoint deterministic feedforward network. We exploit this structure in order to specify powerful algorithms for exact inference and learning. The original stochastic Boltzmann machine may be disregarded, since all relevant probability computations can be performed exactly in the corresponding adjoint networks.

1 Introduction

Boltzmann machines (Hinton and Sejnowski 1983) were the first explicitly stochastic networks for which a learning rule (Ackley, Hinton, and Sejnowski 1985) 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 (Jordan and Bishop 1997), Boltzmann machines are even more interesting. They can be implemented massively parallelly and have a strong theoretical embedding in thermodynamics.

This deserves further comment: thermodynamics, from which statistical mechanics has emerged, remained essentially unaffected by all the revolutions in physics of this century that radically changed our view of the composition and interaction of elementary constituents of matter (with one exception: the third axiom of thermodynamics which became a theorem through the assumption of the validity of quantum mechanics). It seems as if the thermodynamical description has little to do with the inner structure of matter and the corresponding interaction, but instead abstracts basic properties of matter that are not directly coupled to its physical realisation: the informational properties and the organisational structure.

It was not until the end of the twenties, when Leo Szilard sharpened Ludwig Boltzmann's long before given interpretation of entropy as a measure of disorder by identifying entropy loss with information gain. Since then, information has been considered as a physical object (one bit being roughly $9.57 \cdot 10^{-24}$ Joule/Kelvin, which immediately follows from Boltzmann's celebrated formula "entropy equals k times log(phase space volume)", where k denotes a universal constant now known as Boltzmann's constant). From this point of view it seems more than natural that thermodynamics, taken as a phenomenological subject, together with the theory of statistical mechanics supplies tools to examine small, cooperating, and massively parallel information processing units.

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. These 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 for a better approximation. Stochastic relations that are given solely by examples can be stored in the parameter vector, a. k. a. weight vector, w of a Boltzmann machine. The process of recalling a stochastic relation is called inference.

Section 3 describes the relevance of the weight dependence $w \mapsto Z_w$ of the partition sum Z_w of the Boltzmann machine for both inference and learning. This mapping comprises the Boltzmann-Gibbs distribution and thus the complete behaviour of the system. The moments, the correlation functions, and all the relevant terms for the learning rule can be calculated from partial derivatives of $w \mapsto \log(Z_w)$. The obvious advantage is that differentiation is much simpler than the integration or summation that is otherwise needed to calculate moments.

Traditionally, the relevant quantities for inference and learning are estimated in simulations using ergodic theory (Gibbs sampling). Many efforts, such as mean-field approximation (Peterson and Anderson 1987), have been made since then to overcome the — even for modern computers — comparatively slow Gibbs sampling of a Boltzmann machine. One especially interesting technique, which has been transferred from statistical mechanics by (Saul and Jordan 1994), is decimation: 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 works in tree-like structures, such as Boltzmann trees and chains.

In Section 4, we will take up this idea and complete it by developing all possible decimation rules. This leads to a definition of a set of decimatable Boltzmann machines, which contains tree-like structures as a special case. We will show that exploiting the process of decimation allows us to compute the partition sum of the system efficiently: linear in the number of nodes.

The rest of this article shows how this approach leads to simple and powerful learning rules. The principal idea is to formulate the calculation of $\log(Z_w)$ as a feedforward network, where the weight vector w is the input vector. Generalising the backpropagation algorithm, a simple forward-backward pass in this network suffices to compute $\nabla \log(Z_w)$, thereby calculating all relevant terms for learning together. Another benefit of the partition-sum approach is that the cost function itself can be computed, which in turn allows us to apply any acceleration method for backpropagation learning, such as quasi-Newton, conjugate gradient, and learning rate adaptation.

2 The Boltzmann Machine

The underlying structure of a Boltzmann machine is given by a set $N = \{0, \ldots, n\}$ of nodes and a set of edges $E \subset \{(i,j) \in N \times N | i < j\}$ together with a weight vector $w \in \mathbb{R}^E$. The component w_{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, output nodes, and the bias node. 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

$$P_w(s) := \frac{\exp(-h(w,s)/T)}{Z_w} \quad \text{with} \quad h(w,s) := -\sum_{(i,j) \in E} s_i w_{ij} s_j, \tag{1}$$

where $h(w,s)$ is the energy of activation s, and $s_o := 1$ is the constant activation of the bias node 0. The partition sum Z_w is defined to be the normalisation term, and the temperature $T \in \mathbb{R}^+$ characterises the variability of the distribution.

Clamping values ± 1 to input nodes (and sometimes 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 w_{oj} by $s_i w_{ij}$ of all those nodes with $(i,j) \in E$ or $(j,i) \in E$. At the same time, the node i together with all edges connected with 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 will denote all quantities of the new Boltzmann machine with the Greek superscript ξ: $N^\xi := N \setminus X$, $P^\xi_{w^\xi}(s^\xi) := P_w(s|\xi)$, 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_w from P_w — 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

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

is a suitable error measure for the deviation of r with respect to p_w; the information gain is nonnegative and vanishes if p_w coincides with r. Note that setting $I = \emptyset$ reduces the information gain to the Kullback-Leibler distance (Kullback 1959) of r and p_w. In general, the difference of the Kullback-Leibler distance and the information gain is the Kullback-Leibler distance of the input pattern distributions q and $\alpha \mapsto \sum_\gamma p_w(\alpha\gamma)$. Gradient descent for (2) results in the learning rule

$$\Delta w_{ij} = -\eta \nabla(w \mapsto \mathrm{IG}(r, p_w))_{ij} = \frac{\eta}{T}\left(\overline{\langle s_i s_j \rangle_{\alpha\gamma}}^{r} - \overline{\langle s_i s_j \rangle_{\alpha}}^{q}\right), \tag{3}$$

where η is the learning rate. The brackets $\langle \cdot \rangle_\xi$ denote the expectation value w. r. t. the conditional Boltzmann-Gibbs distribution P^ξ_w, and $\overline{\cdot}^p$ denotes the expectation value w. r. t. p or, if p is given by samples, the average over the samples.

Inference means the computation of the conditional probability

$$p_w(\gamma|\alpha) = \sum_\beta P_w(\beta\gamma|\alpha) \tag{4}$$

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 $X \supset I$ comes 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 in the beginning.

3 The Partition Sum

The partition sum was defined in Section 2 as a normalisation factor, and the specific value seems to be irrelevant. However, the dependence of the partition sum on the parameters plays an important role in statistical mechanics. This is the case for Boltzmann machines as well, and the rest of this section contains a mathematical justification for this. Less mathematically inclined readers may skip the theorems and the the proofs in this section.

Let S be a $\mathbb{R}^n$ valued random variable with probability measure P_S. Then the *characteristic function* $\phi_S : \mathbb{R}^n \to \mathbb{C}$ of S is defined as the expectation value

$$x \mapsto \Big\langle \exp(\imath x S) \Big\rangle .$$

ϕ_S is known to have the following properties, cf. (Feller 1971):

(i) If all moments $\langle \prod_k S_k^{m_k} \rangle$ exist for all $m_1, m_2, \ldots, m_n \in \mathbb{N}_o$, then all partial derivatives of ϕ_S exist, and it holds for the r-th moment ($r = m_1 + \ldots + m_n$) that

$$\left\langle \prod_k S_k^{m_k} \right\rangle = (-\imath)^r \frac{\partial^r \phi_S(x)}{\partial x_1^{m_1} \ldots \partial x_n^{m_n}} \Big|_{x=0} .$$

(ii) The *Fourier-Stieltjes transformation* $P_S \mapsto \phi_S$ is injective; hence the name characteristic function.

(iii) The definition of the logarithm is ambiguous in the complex plane. However, there is exactly one continuous mapping $\Phi_S : \mathbb{R}^n \to \mathbb{C}$ associated with each characteristic function ϕ_S such that $\phi_S = \exp \circ \Phi_S$ and $\Phi_S(0) = 0$. In this sense, the *cumulant generating function* Φ_S is defined as the logarithm of ϕ_S. If all moments of S exist, then both ϕ_S and Φ_S are expansible into a Taylor series about 0:

$$\Phi_S(x) = \sum_{r=1}^{\infty} \imath^r \sum_{\substack{m_1,\ldots,m_n \\ \sum m_k = r}} \mathrm{C}_m^r \prod_{k=1}^{n} \frac{x_k^{m_k}}{m_k!} \quad \text{with} \quad \mathrm{C}_m^r = (-\imath)^r \frac{\partial^r \Phi_S(x)}{\partial x_1^{m_1} \ldots \partial x_n^{m_n}} \Big|_{x=0}$$

Every factor C_m^r is a *cumulant* of S of the order $r = \sum m_k$ with $m = (m_1, \ldots, m_n)$. Cumulants are functions of the moments (as is easily seen by expanding the logarithm into a Taylor series and comparing the coefficients). Hence, we will also use the more symbolic notation $\mathrm{C}(S_1^{m_1} \dots S_n^{m_n})$ instead of C_m^r. Cumulants involving more than just one component of S are also known as *correlation functions.* They are the main reason for using cumulants instead of moments, since correlation functions are able to describe the higher order correlations of the components of S in the following sense:

(iv) C_m^1 are the expectation values of the components of S, i.e., $\mathrm{C}(S_k) = \langle S_k \rangle$. The second-order cumulants C_m^2 are elements of the variance-covariance matrix of S, i.e., $\mathrm{C}(S_k S_l) = \langle S_k S_l \rangle - \langle S_k \rangle \langle S_l \rangle$. If the $S_1, \dots, S_n$ are independent, the correlation functions vanish. The components of S are said to be correlated with order r, if r is the smallest integer such that all correlation functions of an order exceeding r vanish.

The following theorem establishes a connection between the partition sum of a Boltzmann machine and the above well-known mathematics.

Theorem 3.1 (partition sum and cumulants). *Let E, the set of edges of a Boltzmann machine N, contain all edges of the form $(0, a)$, $a \in N_*$. Let S denote the random variable associated with the activation vector $s \in \{-1, 1\}^{N_*}$. Then, the cumulant C_m^r of S with $r > 0$ is given by*

$$\mathrm{C}_m^r = T^r \frac{\partial^r (w \mapsto \log(Z_w))}{\partial w_{01}^{m_1} \dots \partial w_{0n}^{m_n}}. \qquad \square$$

Proof. The probability measure of S is defined on the finite set of possible activations. Hence, not only the cumulant generating function exists, but also all moments and all cumulants exist. Let the weight vector w be divided into a subvector $w^o := (w_{01}, \dots, w_{0n})$ and another subvector w^1 containing all other components of w. The cumulant generating function Φ_S was defined as

$$\begin{aligned} \Phi_S(x) &= \log\left(\sum_s \frac{\exp\left(\sum w_{ab} s_a s_b / T\right)}{Z_w} \exp(\imath x s) \right) \\ &= \log\left(\sum_s \exp\left(\sum (w_{0a}^o / T + \imath x_a) s_a + \sum w_{ab}^1 s_a s_b / T \right) \right) - \log(Z_w). \end{aligned}$$

Since $r > 0$, calculating C_m^r involves at least one partial differentiation w.r.t. a component of x. The derivative of the second term vanishes, and only the first term has to be regarded, which in turn may be considered as the mapping $x \mapsto \log(Z_{w^o + \imath T x, w^1})$. Hence,

$$\begin{aligned} \mathrm{C}_m^r &= (-\imath)^r \frac{\partial^r (x \mapsto \log(Z_{w^o + \imath T x, w^1}))}{\partial x_1^{m_1} \dots \partial x_n^{m_n}} \Big|_{x=0} \\ &= T^r \frac{\partial^r (w^o \mapsto \log(Z_{w^o, w^1}))}{\partial w_{01}^{m_1} \dots \partial w_{0n}^{m_n}}. \qquad \blacksquare \end{aligned}$$

Corollary 3.2 of the proof. *The characteristic function of a random variable following a Boltzmann-Gibbs distribution, whose partition sum is Z_w, is given by the mapping $x \mapsto Z_{w^o+\imath Tx,w^1}/Z_w$ from $\mathbb{R}^n \to \mathbb{C}$.* $\square$

Remark 3.3. Although the terms $\langle s_i s_j \rangle_\xi$ of the learning rule (3) can be calculated with the help of the above theorem,

$$\langle s_i s_j \rangle_\xi = T \frac{\partial \log(Z^\xi_{w^\xi})}{\partial w^\xi_{ij}} \tag{5}$$

is a similarly related possibility calculating $\langle s_i s_j \rangle_\xi$. If one of the nodes, say i, is clamped, then $\langle s_i s_j \rangle_\xi = s_i \langle s_j \rangle_\xi = s_i T \partial \log(Z^\xi_{w^\xi})/\partial w^\xi_{oj}$. $\square$

Fourier transformations may even help to reduce the number of elementary operations needed for the inference computation (4) when few output nodes exist:

Theorem 3.4 (inference 1). *Let $\mathrm{B}\Gamma$ be the random variable that is induced by the activation vector $\beta\gamma$ in the Boltzmann machine $N^\alpha = H \cup O$, where $\beta \in \{-1,1\}^H$ and $\gamma \in \{-1,1\}^O$. Let $\Pi_O \colon \mathbb{R}^{H\cup O} \to \mathbb{R}^O$ denote the projection $\beta\gamma \mapsto \gamma$, and let Π'_O be the transposition of the linear mapping Π_O. Then,*

$$p_w(\gamma|\alpha) = \frac{1}{2^{|O|}} \sum_{x \in \{0,\pi/2\}^O} \exp(-\imath\gamma x)\, (\phi^\alpha_{\mathrm{B}\Gamma} \circ \Pi'_O)\,(x). \qquad \square$$

Proof. The principal idea is that the characteristic function ϕ^α_Γ of $\Gamma = \Pi_O \circ \mathrm{B}\Gamma$ is given by

$$\phi^\alpha_\Gamma = \phi^\alpha_{\mathrm{B}\Gamma} \circ \Pi'_O,$$

which immediately follows from the definition of the characteristic function. An inverse Fourier transformation yields the desired probabilities (4) as is indicated by the following commuting diagram:

$$\begin{tikzcd}
\beta\gamma \mapsto P^\alpha_{w^\alpha}(\beta\gamma) \arrow[r, "\text{Fourier-Stieltjes}", "\text{transformation}"'] \arrow[d, "\sum_\beta"'] & \phi^\alpha_{\mathrm{B}\Gamma} \colon \mathbb{R}^{H\cup O} \to \mathbb{C} \arrow[d, "\circ\, \Pi'_O"] \\
\gamma \mapsto p^\alpha_{w^\alpha}(\gamma) & \phi^\alpha_\Gamma \colon \mathbb{R}^O \to \mathbb{C} \arrow[l, "\text{inverse Fourier}"', "\text{transformation}"]
\end{tikzcd}$$

$\blacksquare$

Remark 3.5 (inference 2). The combination of Corollary 3.2 and Theorem 3.4 reduces the computation of (4) to the calculation of the partition sum. Another, somewhat more direct, way of computation is given as follows: inserting the definition of conditional probabilities, it holds that for every $\beta \in \{-1,1\}^H$

$$\begin{aligned} p_w(\gamma|\alpha) &= \frac{P_w(\beta\gamma|\alpha)}{P_w(\beta|\alpha\gamma)} = \frac{P^{\alpha}_{w^\alpha}(\beta\gamma)}{P^{\alpha\gamma}_{w^{\alpha\gamma}}(\beta)} \\ &= \frac{\exp(-h^\alpha(w^\alpha,\beta\gamma)/T)}{\exp(-h^{\alpha\gamma}(w^{\alpha\gamma},\beta)/T)} \cdot \frac{Z^{\alpha\gamma}_{w^{\alpha\gamma}}}{Z^{\alpha}_{w^\alpha}} \\ &= \exp\Big(\sum_{c\in O}\gamma_c w_{oc}/T + \sum_{\substack{(c,d)\in E\\ c,d\in O}} w_{cd}\gamma_c\gamma_d/T + \sum_{\substack{(a,c)\in E\\ a\in I, c\in O}} w_{ac}\alpha_a\gamma_c/T\Big)\frac{Z^{\alpha\gamma}_{w^{\alpha\gamma}}}{Z^{\alpha}_{w^\alpha}}. \end{aligned} \quad (6)$$

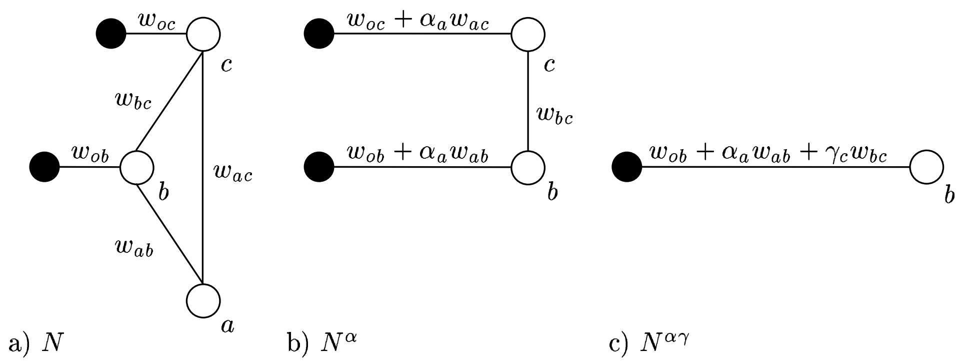


Figure 1: Examples of Boltzmann machines N, N^α, and $N^{\alpha\gamma}$

Figure 1 explains why the energy difference $h^{\alpha\gamma}(w^{\alpha\gamma},\beta) - h^\alpha(w^\alpha,\beta\gamma)$ is independent of β and except for temperature scaling is given by the argument of the exponential function in (6). Consider the full Boltzmann machine of Figure 1a with one input, one hidden, and one output node; first an input pattern α is clamped resulting in a Boltzmann machine N^α (Figure 1b) and then both input and output patterns are clamped (Figure 1c). The resulting energies $h^{\alpha\gamma}$ and h^α can easily be calculated from the weights, their difference being $(w_{oc} + \alpha_a w_{ac})\gamma_c$. The generalisation to more weights is straightforward, thereby yielding the argument of the exponential in (6).

4 Decimatable Boltzmann Machines

Equations (6) and (5) link the important quantities for inference (4) and learning (3) to the partition sum; the transformation is polynomial in $|N|$ in both cases. These and all other results of the preceding sections are valid for general bipolar

Boltzmann machines. In principle, Z_w can be calculated by summing all $2^{|N_*|}$ terms $\exp(-h(w,s)/T)$, which is arguably intractable in $|N|$: (Cooper 1990) 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 toward 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 2 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 $w'_{a_i a_j}$ such that

$$\sum_{s_{a_1} \in \{-1,1\}} P_w^{2a}(s_{a_1}, s_{a_2}, s_{a_3}, \ldots) = P_{w'}^{2b}(s_{a_2}, s_{a_3}, \ldots), \tag{7}$$

where a_2, a_3, and a_4 may be embedded in an arbitrarily complex network. This *ansatz for decimation* leads to the following theorem; here and throughout the following text $v := w/T$ denotes an *effective weight vector.*

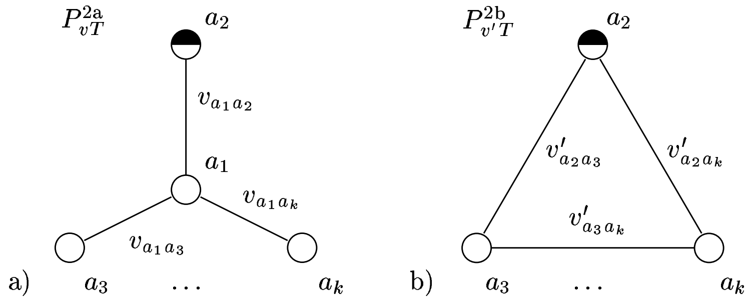


Figure 2: Ansatz for decimation of the node a_1

Theorem 4.1 (decimation). *Let a be an enumeration of nodes of a Boltzmann machine with effective weight vector v. Let $a_1 \neq 0$ be connected with exactly $k-1$ neighbour nodes a_2, …, a_k (one of which may be the bias node).*

(i) Ansatz (7) can be solved for $k = 4$ with

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

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

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

and the partition sum is changed by the factor

$$\frac{Z_w^{2\mathrm{a}}}{Z_{w'}^{2\mathrm{b}}} = 2\left(\prod_{s_{a_3},s_{a_4}} \cosh(v_{a_1a_2}+v_{a_1a_3}s_{a_3}+v_{a_1a_4}s_{a_4})\right)^{1/4}. \quad (11)$$

The cases in which $k \le 3$ are special cases of $k = 4$: $v_{a_1a_4} = 0$ if $k = 3$, $v_{a_1a_3} = v_{a_1a_4} = 0$ if $k = 2$, and $v_{a_1a_2} = v_{a_1a_3} = v_{a_1a_4} = 0$ if $k = 1$.

(ii) In general, ansatz (7) cannot be solved for $k > 4$.

(iii) If necessary, a new effective weight $v'_{a_ia_j}$ arising from (i) must be added to an existing effective weight $v_{a_ia_j}$ (parallel rule). *Otherwise, a new edge (a_i, a_j) has to be inserted.* □

Proof. (iii) The energy (1) is the central defining quantity determining the Boltzmann-Gibbs distribution. The parallel rule is a simple consequence of the linearity of the energy in the effective weights.

(i) Decimating the node a_1 changes the partition sum, which in turn depends on the effective weights. In order to avoid considering this dependence, ansatz (7) may use relative probabilities

$$\frac{\sum_{s_{a_1}} P_{vT}^{2\mathrm{a}}(s_{a_1}, s_{a_2}, s_{a_3}, \ldots)}{\sum_{s_{a_1}} P_{vT}^{2\mathrm{a}}(s_{a_1}, 1, 1, \ldots)} = \frac{P_{v'T}^{2\mathrm{b}}(s_{a_2}, s_{a_3}, \ldots)}{P_{v'T}^{2\mathrm{b}}(1, 1, \ldots)}$$

as well. The respective denominators $P_{w'}^{2\mathrm{b}}(1,1,\ldots) = \sum_{s_{a_1}} P_w^{2\mathrm{a}}(s_{a_1},1,1,\ldots)$ do not vanish if the temperature T is positive and the weights are finite. Using the definition (1) of the Boltzmann-Gibbs distribution and by cancelling all terms interacting between nodes a_2, …, a_n that are independent of s_{a_1} the above equation yields

$$\frac{\cosh\left(\sum_{i=2}^4 v_{a_1a_i}\right)}{\cosh\left(\sum_{i=2}^4 v_{a_1a_i}s_{a_i}\right)} = \exp\Big(\sum_{1<i<j<5} v'_{a_ia_j}(1 - s_{a_i}s_{a_j})\Big). \quad (12)$$

(12) is invariant under the operation $(s_{a_2}, s_{a_3}, s_{a_4}) \leftrightarrow (-s_{a_2}, -s_{a_3}, -s_{a_4})$, which reduces the eight possible combinations of the activations $s_{a_2}, s_{a_3}, s_{a_4} \in \{-1, 1\}$ to four combinations, e. g., $(1,-1,-1)$, $(1,-1,1)$, $(1,1,-1)$, and $(1,1,1)$. One of the three nodes, say a_2, could have been chosen to be the bias node. $s_2 = s_3 = s_4$ trivially fulfills (12) leaving three equations for three variables $v'_{a_2a_3}$, $v'_{a_2a_4}$, and $v'_{a_3a_4}$. Taking the logarithm makes (12) a system of linear equations with the unique solution (8)–(10). Replacing node a_1 with new weights $v'_{a_2a_3}$, $v'_{a_2a_4}$, and $v'_{a_3a_4}$ mediates

between honeycomb and triangle spin lattices in physics (Onsager 1944; Syozi 1972), where this operation is termed star-triangle or Y-Δ transformation. In the context of Boltzmann machines we prefer the name *star decimation.*

The case $k = 3$ can be solved in the same way resulting in one equation with one variable $v'_{a_2a_3}$, whose solution is

$$v'_{a_2a_3} = \log\left(\frac{\cosh(v_{a_1a_2} + v_{a_1a_3})}{\cosh(v_{a_1a_2} - v_{a_1a_3})}\right)/2. \tag{13}$$

The same result is achieved with $k = 4$ and $v_{a_1a_4} = 0$; the additional effective weights $v'_{a_3a_4} = v'_{a_2a_4} = 0$ are irrelevant owing to the parallel rule (iii). In the same way $k = 2$ can be regarded as a special case of $k = 3$ with $v_{a_1a_3} = 0$: there is no need to compensate for the removal of node a_1. The same is true for the trivial case $k = 1$. Using (8)–(10) in ansatz (7) and cancelling all identical terms on both sides of (7) finally results in Equation (11).

(ii) Ansatz (7) leads to a linear system of $2^{k-2} - 1$ equations ($k - 2$ bipolar activations minus one trivially fulfilled equation) in $(k-1)(k-2)/2$ variables (the pairwise weights between the nodes a_2, a_3, ..., a_k). This system of equations is overdetermined in the case $k > 4$ and, in general, cannot be solved unless the weights $v_{a_1a_2}, \ldots, v_{a_1a_k}$ underlie certain restrictions. ∎

Corollary 4.2 (Saul and Jordan 1994). *If node $a_1 \neq 0$ is only connected with node a_2 and node a_3, then node a_1 can be decimated, and the interaction mediated by a_1 is replaced by the insertion of the effective weight $v'_{a_2a_3}$ with* $\tanh(v'_{a_2a_3}) = \tanh(v_{a_1a_2})\tanh(v_{a_1a_3})$. □

Proof. By the definition of tanh, the claim is equivalent to Eq. (13). ∎

Figure 3a shows an example of a Boltzmann machine whose partition sum Z^{a} can be computed by decimating the Boltzmann machine step by step until the trivial Boltzmann machine $\{0\}$ of Figure 3f with partition sum $Z^{\mathrm{f}} = 1$ is reached. The factors $Z^{\mathrm{a}}/Z^{\mathrm{b}}$, $Z^{\mathrm{b}}/Z^{\mathrm{c}}$, ..., $Z^{\mathrm{e}}/1$ are computed according to (11) in the corresponding decimation steps; their product is the partition sum Z^{a} of the initial network.

Here, the partition sum can be factorised into $|N_*|$ factors in a way that each factor depends on one or more weights. Although the last factor Z^{e} depends indirectly on all the original weights, the calculation of the partition sum involves only $|N_*|$ operations, none of which are more complicated than the evaluation of Eqs. (8)–(11). These Boltzmann machines seem to be ideal candidates for a sufficiently interesting class of tractable networks.

Definition 4.3 (decimatable Boltzmann machines). A Boltzmann machine, which — after clamping an arbitrary pattern in $\{-1, 1\}^X$ — can be transformed into the trivial network $\{0\}$ by successive applications of Theorem 4.1, is termed *decimatable* or — more precisely — *decimatable with respect to X.* □

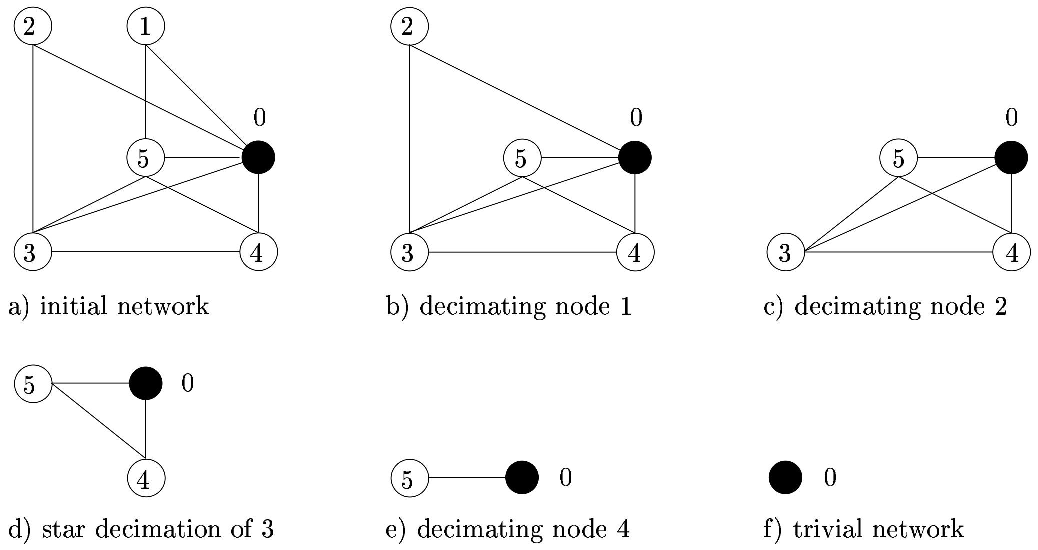
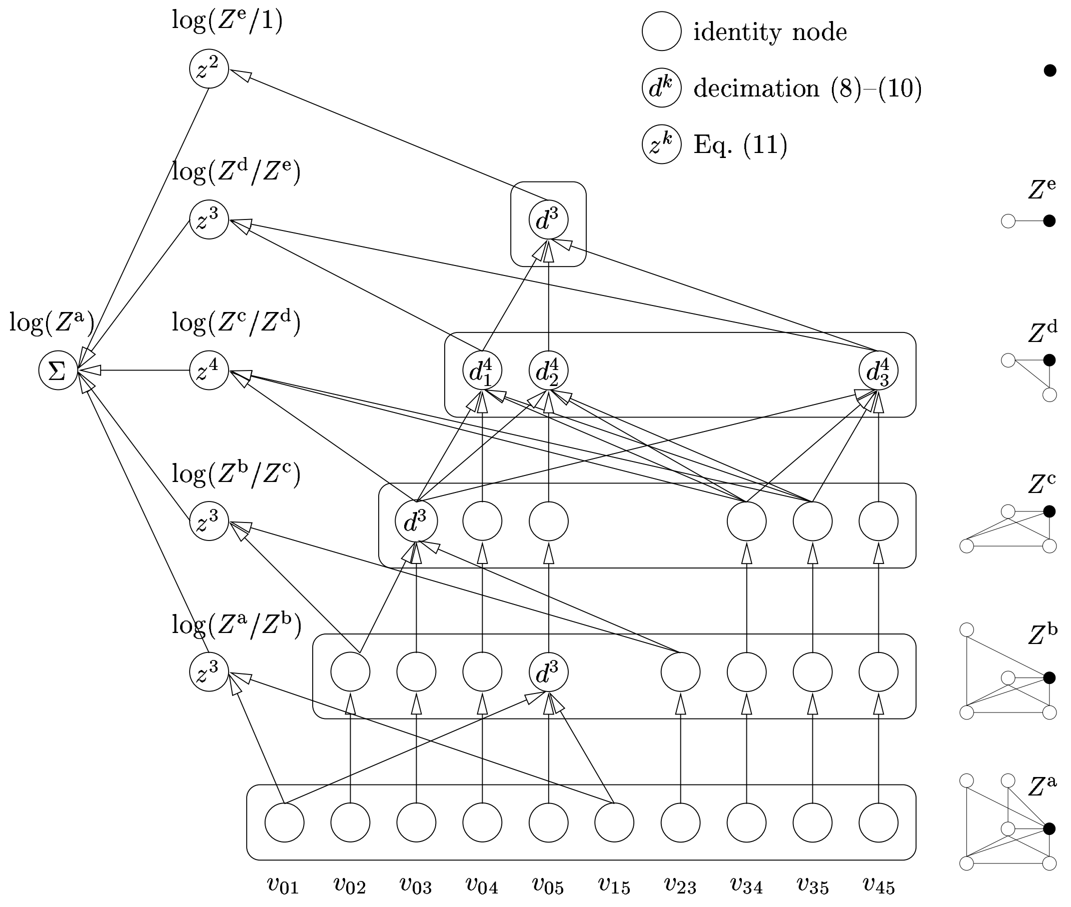


Figure 3: Decimation of a Boltzmann machine

Remark 4.4 (decimation order). It is easy to construct a straightforward algorithm that (in a continuing loop over all remaining nodes of a Boltzmann machine) decimates all non-bias nodes that have fewer than four neighbours until no further nodes can be decimated. If only the bias node is left, then obviously the initial Boltzmann machine was decimatable. The converse, however, is not true: if this algorithm gets stuck with a nontrivial network, the original Boltzmann machine may have been decimated by choosing another order in which the nodes were decimated (when, for example, at a certain step of the algorithm more than one node has fewer than four neighbours). The reason for this is that star decimation may increase the number of neighbours of a remaining node. One pragmatic peace of advice would be to prefer decimating the node among several possibilities whose decimation results in the fewest number of edges. The complexity of the corresponding algorithm does not exceed $O(|N|^2)$.

5 Algorithms for Efficient Inference and Learning

The decimation of a Boltzmann machine factorises the partition sum in 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:



identity node
d^k decimation (8)–(10)
z^k Eq. (11)
$\log(Z^{\mathrm{e}}/1)$
$\log(Z^{\mathrm{d}}/Z^{\mathrm{e}})$
$\log(Z^{\mathrm{a}})$
$\log(Z^{\mathrm{c}}/Z^{\mathrm{d}})$
$\log(Z^{\mathrm{b}}/Z^{\mathrm{c}})$
$\log(Z^{\mathrm{a}}/Z^{\mathrm{b}})$
z^2
z^3
z^4
z^3
z^3
Σ
d^3
d_1^4
d_2^4
d_3^4
d^3
d^3
Z^{e}
Z^{d}
Z^{c}
Z^{b}
Z^{a}
v_{01} v_{02} v_{03} v_{04} v_{05} v_{15} v_{23} v_{34} v_{35} v_{45}

Figure 4: A mapping network for the computation of the log partition sum

Figure 4 shows an example network for the computation of partition sum Z^{a} of the Boltzmann machine of Figure 3a. The input of the network comprises of all effective weights of the initial Boltzmann machine. Decimating node 1 means replacing the weights v_{01} and v_{15} by an additive correction of the weight v_{05} that is given by Corollary 4.2. All other weights remain invariant, which is indicated by identity nodes (which do nothing but copy the input value). Eq. (11) specifies the factor $Z^{\mathrm{a}}/Z^{\mathrm{b}}$ by which the partition sum changes. In the same way all other factors can be computed from the weights. Finally the output node of the mapping network yields the logarithm of the total partition sum by adding the logarithms of all factors. This mapping network is clearly a deterministic feedforward network with $O(|N|)$ nontrivial nodes: the redundant identity nodes in Figure 4 were only inserted for the sake of visualisation. Usually, neural feedforward networks employ sigmoidal activation functions and weights associated with the edges. In this case, however, the mapping network of Figure 4 employs activation functions that arise from Eqs. (8)–(11), and no weights are associated with the edges.

Definition 5.1 (adjoint 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 effective weights of the Boltzmann machine. This mapping network is called *adjoint network.* $\square$

The biggest advantage of adjoint networks is that a generalisation of the back-propagation algorithm allows the computation of the gradient $\nabla(v \mapsto \log(Z_{vT}))$, which is needed for the learning rule (3), in a simple forward-backward pass. This is the subject of the following theorem.

Theorem 5.2 (chain rule for mapping networks). *Let N^ξ be a decimatable Boltzmann machine that arises from clamping a pattern ξ. Let $\tilde{N}^\xi$ and $\tilde{E}^\xi$ be the set of nodes and the set of edges of a corresponding adjoint network. $L_a = \{b \in \tilde{N}^\xi \,|\, (b,a) \in \tilde{E}^\xi\}$ defines the set of predecessor nodes of node a and $R_a = \{b \in \tilde{N}^\xi \,|\, (a,b) \in \tilde{E}^\xi\}$ the corresponding set of successor nodes. Every input node $(i,j) \in \tilde{N}^\xi$ of the adjoint network is characterised by $L_{(i,j)} = \emptyset$ and by $(i,j) \in E^\xi$; the input value is given by the effective weight v^ξ_{ij} of the Boltzmann machine N^ξ. Every non-input node a of the adjoint network computes a certain function value $f_a(h_a)$ of its input vector h_a. Let $b \in L_a$; the b-component h_{ab} of the input vector h_a of node a is inductively defined as output of the predecessor node b:*

$$h_{ab} = \begin{cases} v^\xi_b & \text{if } L_b = \emptyset \\ o_b := f_b(h_b) & \text{otherwise} \end{cases}$$

A forward pass, i. e., the calculation of the values o_b in topological order of the directed acyclic graph $(\tilde{N}^\xi, \tilde{E}^\xi)$, eventually computes the desired output $o_z = f_z(h_z) = \log(Z^\xi_{v^\xi T})$ of the output node z.

A backward pass

$$\delta_b = \begin{cases} 1 & \text{if } R_b = \emptyset, \text{ i. e., if } b = z \\ \displaystyle\sum_{a \in R_b} \delta_a \left.\frac{\partial f_a(h_a)}{\partial h_{ab}}\right|_{h_a|_{v^\xi}} & \text{otherwise} \end{cases}$$

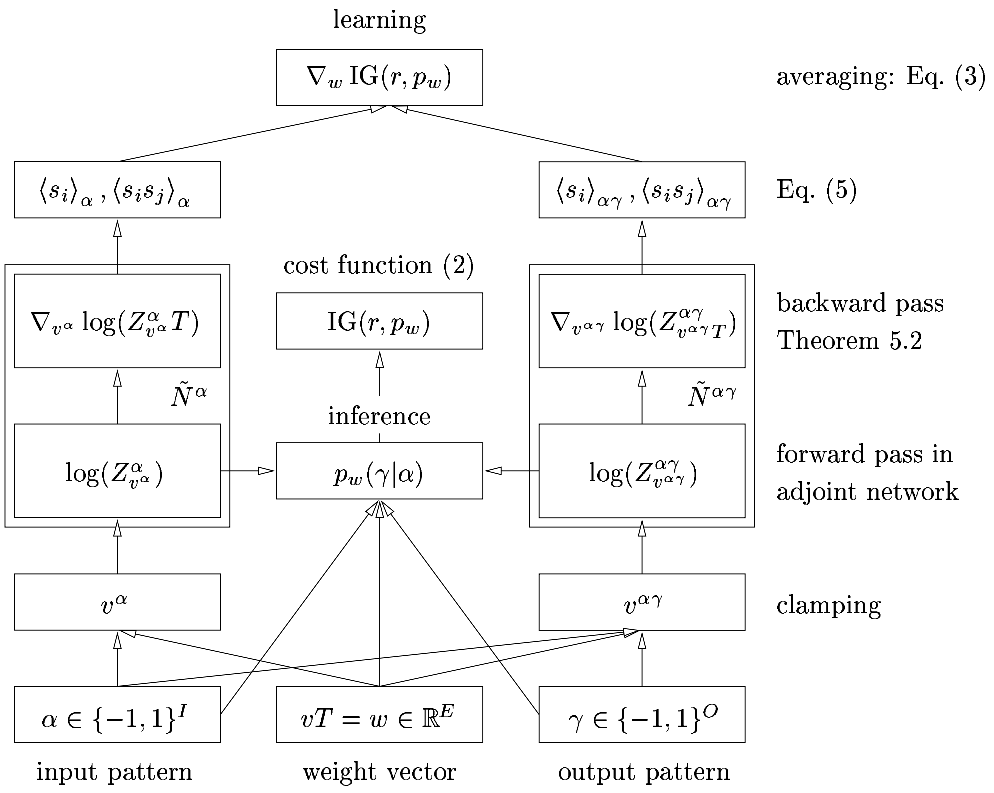
computes all partial derivatives

$$\partial \log(Z^\xi_{v^\xi T}) / \partial v^\xi_{ij} = \delta_{(i,j)}$$

together. $\square$

Proof. (Rüger 1997, lemma 1.1.8) ■

Figure 5 summarises how the gradient of the cost function IG can be computed. The adjoint networks $\tilde{N}^{\alpha}$ and $\tilde{N}^{\alpha\gamma}$, which are constructed at the outset, represent all important quantities of the original Boltzmann machine, which then may be disregarded. Note that the adjoint network $\tilde{N}^{\xi}$ depends on the set X of nodes rather than on the actual $\xi \in \{-1,1\}^X$. Inference — even learning — with respect to arbitrary node sets poses no problem once the corresponding adjoint networks have been constructed.



learning
$\nabla_w \, \mathrm{IG}(r, p_w)$
averaging: Eq. (3)
$\langle s_i \rangle_\alpha, \langle s_i s_j \rangle_\alpha$
$\langle s_i \rangle_{\alpha\gamma}, \langle s_i s_j \rangle_{\alpha\gamma}$
Eq. (5)
cost function (2)
$\nabla_{v^\alpha} \log(Z^\alpha_{v^\alpha T})$
$\mathrm{IG}(r, p_w)$
$\nabla_{v^{\alpha\gamma}} \log(Z^{\alpha\gamma}_{v^{\alpha\gamma} T})$
backward pass
Theorem 5.2
$\tilde{N}^\alpha$
$\tilde{N}^{\alpha\gamma}$
inference
$\log(Z^\alpha_{v^\alpha})$
$p_w(\gamma|\alpha)$
$\log(Z^{\alpha\gamma}_{v^{\alpha\gamma}})$
forward pass in
adjoint network
v^α
$v^{\alpha\gamma}$
clamping
$\alpha \in \{-1,1\}^I$
$vT = w \in \mathbb{R}^E$
$\gamma \in \{-1,1\}^O$
input pattern
weight vector
output pattern

Figure 5: Inference and learning using the adjoint networks $\tilde{N}^{\alpha}$ and $\tilde{N}^{\alpha\gamma}$

The learning rule (3) was given by gradient descent in the cost function (2). It is well-known that pure gradient descent is not optimal for learning. Either certain minima of the cost function cannot be found if the learning rate is too large, or learning takes a long time if η is too small, cf. (Rüger 1996, trade-off-theorem). Virtually every convex-optimisation method utilises cost function values for efficient optimisation.

Usually, the desired probability distribution r is given empirically by m patterns (α^1, γ^1), ..., (α^m, γ^m). Sorting these examples lexicographically, it is easy to determine $q(\alpha)$ as the number of examples whose first component is α divided by m. $r(\gamma|\alpha)$ may then be set to the frequency of examples that coincide with (α, γ) divided by $q(\alpha)$. The evaluation of (2) is then reduced to the summation over αs and γs *that occur in the training data.* This leaves $p_w(\gamma|\alpha)$ to be determined, which can easily be computed using (6) and the forward pass in the adjoint networks.

Since the values of the cost function are available now, all established convex-optimisation methods (Press, Teukolsky, Vetterling, and Flannery 1988) can be used or, e. g., stable dynamic learning-rate adaptation (Rüger 1996) that uses comparatively few but well-chosen evaluations of the cost function.

6 Discussion

We have studied the decimation of bipolar Boltzmann machines exhaustively. The decimation of tree-like structures, which was presented in (Saul and Jordan 1994), has been generalised using the star-triangle transformation from statistical physics. We have shown that no other decimation rule exists. Hence, we have been able to give a definition of a set of Boltzmann machines that can be addressed by decimation.

This article emphasises the role of the partition sum in general Boltzmann machines. As a result, both inference and improved convex-optimisation learning can be expressed in terms of the partition sum. The computation of the partition sum is intractable in general. In decimatable Boltzmann machines, however, the partition sum can be calculated with a complexity that is linear in the number of nodes, once the deterministic adjoint network has been constructed. What is more, the gradient of the log partition sum can be computed in a simple forward-backward fashion. *Decimatable Boltzmann machines have all benefits of recurrent and stochastic networks — and that with the costs of a deterministic feedforward network!*

Small applications (Saul and Jordan 1994; Rüger, Weinberger, and Wittchen 1996) have demonstrated that decimatable Boltzmann machines can be used to approximate probability distributions, to store knowledge about stochastic relations given by examples, and to do inference in such databases even in the presence of missing values. One interesting question is how this approach can be extended to previous work (Saul and Jordan 1996; Nijman and Kappen 1996) exploiting tractable substructures in sparsely connected networks.

Future research includes the application of the presented techniques to multivalued Boltzmann machines. Their decimation rules are somewhat different, and the star decimation can only be applied if one of the neighbour nodes is a bias node. Nevertheless, decimatable multivalued Boltzmann machines are a rich superset of hidden Markov models (Saul and Jordan 1995; MacKay 1996). As hidden Markov models play a big role in speech recognition (Juang and Rabiner 1991), interesting applications of multivalued decimatable Boltzmann machines can be expected. This issue and others remain for further studies.

References

Ackley, D. H., G. E. Hinton, and T. J. Sejnowski (1985). A learning algorithm for Boltzmann machines. *Cognitive Science 9*, 147–169.

Cooper, G. F. (1990). The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence 42*, 393–405.

Feller, W. (1971). *An Introduction to Probability Theory and Its Applications* (2nd ed.), Volume II. New York: Wiley.

Hinton, G. E. and T. J. Sejnowski (1983). Optimal perceptual inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 448–453. IEEE.

Jordan, M. I. and C. M. Bishop (in press). Neural networks. *Computing Surveys*.

Juang, B. H. and L. R. Rabiner (1991). Hidden Markov models for speech recognition. *Technometrics 33*, 251–272.

Kullback, S. (1959). *Information Theory and Statistics*. New York: Wiley.

MacKay, D. J. C. (1996). Equivalence of Boltzmann chains and hidden Markov models. *Neural Computation 8*(1), 178–181.

Nijman, M. J. and H. J. Kappen (1996). Efficient learning in sparsely connected Boltzmann machines. In C. von der Malsburg, W. von Seelen, J. C. Vorbrüggen, and B. Sendhoff (Eds.), *Artificial Neural Networks - ICANN 96*, pp. 41–46. Springer-Verlag.

Onsager, L. (1944). Crystal statistics. I. A two-dimensional model with an order-disorder transition. *Physical Review 65*(3/4), 117–149.

Peterson, C. and J. R. Anderson (1987). A mean field theory learning algorithm for neural networks. *Complex Systems 1*, 995–1019.

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

Rüger, S. M. (1996). Stable dynamic parameter adaptation. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 8*, pp. 225–231. MIT Press.

Rüger, S. M. (1997). *Zur Theorie künstlicher neuronaler Netze.* Frankfurt/Main: Verlag Harri Deutsch. ISBN 3-8171-1542-3.

Rüger, S. M., A. Weinberger, and S. Wittchen (1996). Decimatable Boltzmann machines vs. Gibbs sampling. Technical Report 96-29, Fachbereich Informatik der Technischen Universität Berlin.

Saul, L. K. and M. I. Jordan (1994). Learning in Boltzmann trees. *Neural Computation 6*(6), 1174–1184.

Saul, L. K. and M. I. Jordan (1995). Boltzmann chains and hidden Markov models. In G. Tesauro, D. S. Touretzky, and T. K. Leen (Eds.), *Advances in Neural Information Processing Systems 7*, pp. 435–442. MIT Press.

Saul, L. K. and M. I. Jordan (1996). Exploiting tractable substructures in intractable networks. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 8.* MIT Press.

Syozi, I. (1972). Transformation of Ising models. In C. Domb and M. S. Green (Eds.), *Phase Transitions and Critical Phenomena. Volume 1. Exact results*, London, pp. 269–329. Academic Press.