

Learning Algorithms for extended models of Boltzmann machines.

Laurent Younes

CMLA-DIAM,

Ecole Normale Supérieure de Cachan,

61 avenue du Président Wilson, 94 235 Cachan CEDEX, France.

abstract

We present and study learning rules for specific models of Boltzmann machines which extend classical sequential and reversible synchronous models.

1 Introduction.

Boltzmann machines are stochastic neural networks which associate to each input a probability distribution over the output. The neuronal interpretation comes from the fact that this probability distribution arises from random interactions between elementary units or formal neurons. To be more specific, we give ourselves three disjoint sets of such units, which we classically denote by I , H and R , respectively input, hidden and response neurons, and put $S = I \cup H \cup R$ for the set of all neurons.

To each $s \in S$ is associated a random variable, $x_s = 0$ or 1 , describing its state. In that way, one can encode a given input into the set of bits $(x_s, s \in I)$, and similarly for the output.

Denoting by Ω_C the set of all configurations $(x_s, s \in C)$, where C is any subset of S , we shall therefore associate to each input configuration x_I a probability distribution $\psi(\cdot; x_I)$ on Ω_R . This distribution, ψ , will be defined as the marginal distribution of a law $\pi(\cdot; x_I)$ on $\Omega_H \times \Omega_R$, ie.

$$\psi(x_R; x_I) = \sum_{x_H \in \Omega_H} \pi(x_H, x_R; x_I), \quad (1)$$

and π will in turn be defined as the invariant distribution of a random dynamics over $\Omega_H \times \Omega_R$, controlled by x_I , and involving elementary interactions between neurons. This dynamics is specified by a family of transition probability matrices which is built as follows. We assume that at discrete time n , the state of s depends of the states of the whole network at former times $n-1, \dots, n-q$, and that the probability of x_s , given that these previous states are $x(1), \dots, x(q) \in \Omega_S$ is

$$q_s[x(q), \dots, x(1); x_s].$$

The transitions q_s thus describe the way a single neuron s updates its state given former states of the whole network (including input neurons, so that the

dynamics is, as announced, controlled by the clamped configuration x_I). To describe the stochastic evolution of the network, it remains to decide in which order these updatings must be done. We can discriminate two main types of such ordering: *sequential updating*, for which only one neuron is updated at a given time, and *synchronous updating*, for which all neurons are updated at a given time. In both cases, the input x_I is clamped. These two kinds of ordering lead to different equilibrium distributions, $\pi(\cdot; x_I)$ on $\Omega_H \times \Omega_R$, and to different learning rules. When $q = 1$, the first case corresponds to classic sequential Boltzmann machines, introduced in [AHS] (see also [AZ2]), and the second case to synchronous Boltzmann machines described in [AZ1], [AZ2], [LAC]. Between these two cases, one can consider "partially synchronous" evolutions, in which only some subset of $H \cup R$ is updated at each time chunk which are not addressed here. Moreover, we shall only consider the following particular cases:

Sequential case :

We always assume that $q = 1$, and that the evolution is obtained from an interaction potential. This means that we are given a family $(w_C, C \in S)$ of numbers, and that, if $x' \in \Omega_S$, $q_s(x; x')$ is given by

$$q_s(x; x') = \frac{\exp\left(x'_s \sum_{C, s \in C} w_C \prod_{t \in C \setminus \{s\}} x_t\right)}{1 + \exp\left(\sum_{C, s \in C} w_C \prod_{t \in C \setminus \{s\}} x_t\right)}. \quad (2)$$

In this case, define the distribution $\bar{\pi}$ on Ω_S by

$$\bar{\pi}(x) = \exp\left(\sum_C w_C \cdot \prod_{s \in C} x_s\right) / Z, \quad (3)$$

where Z is a normalizing constant. Then, the equilibrium distribution, $\pi(\cdot; x_I)$ is simply the *conditional distribution* for $\bar{\pi}$, $\bar{\pi}(\cdot | x_I)$; (see [AZ2]).

Synchronous case :

Let $p = q + 1$. The construction we make corresponds to p -periodic synchronous random fields, as introduced in [YOU1]. We add to S a new site, denoted 0 , ($0 \notin S$) and consider a family $(w_c, c \in (S \cup \{0\})^p)$ of numbers, subject to the following periodicity condition :

$$\text{for all } s_q, \dots, s_0 \in S \cup \{0\}, w_{s_q \dots s_0} = w_{s_{q-1} \dots s_0 s_q}.$$

We then define

$$q_s[x(q), \dots, x(1); x_s(0)] = \frac{\exp\left(x_s(0) \sum_{c=(s_0, \dots, s_1)} w_c \prod_{1 \leq k \leq q, s_k \neq 0} x_{s_k}(k)\right)}{Z(x(q), \dots, x(1))} \quad (4)$$

where $Z(x(q), \dots, x(1))$ is a suitable normalizing constant.

To define the equilibrium distribution, put

$$\mu(x(q), \dots, x(0)) = \frac{\exp\left[\sum_{c=(s_0, s_1, \dots, s_q)} w_c \prod_{0 \leq k \leq q, s_k \neq 0} x_{s_k}(k)\right]}{Z}, \quad (5)$$

which is a probability distribution on $(\Omega_S)^p$. One can show that $\pi(\cdot; x_I)$ is one of the (identical) marginals over $\Omega_H \times \Omega_R$ of the conditional distribution for μ given that $x_s(k) = x_s$ for all $k = 0, \dots, q$ and $s \in I$, which we shall denote by $\mu(\cdot; x_I)$. It is important to note that $\mu(\cdot; x_I)$ is the distribution of p consecutive configuration obtained during the synchronous evolution with clamped input to x_I .

When $q = 1$, one can check that the last case coincides with reversible synchronous Boltzmann machines as defined in [AZ1].

According to the common terminology, the parameters w_C and w_c above will be referred to in the following as *interaction weights*, or simply weights.

Boltzmann machine models other than the ones we consider have been studied and used ([AZ2], [LAC]), especially in the synchronous case. They all assume $q = 1$, but use complex clique interactions, asymmetric links. However, in these cases, the equilibrium distribution $\pi(\cdot; x_I)$ is unknown, and the associated learning algorithms are more complicated (they involve sampling reversed process of non reversible Markov chains).

We now present learning algorithms for the models we have defined.

2 Learning algorithms.

2.1 Introduction

Learning algorithms compute parameters w_C or w_c above in order that the network represents a given (but generally unknown) function f from Ω_I to Ω_R , which associates to a configuration x_I in input its associated "correct answer" $f(x_I)$ in response. Since the response of a Boltzmann machine is random, and follows the distribution $\psi(\cdot; x_I)$, we shall consider (following [AHS], [AZ1]) learning algorithms which minimize the Kullback information distance between $\psi(\cdot; x_I)$ and the Dirac measure $\delta_{f(x_I)}(\cdot)$ (or slight modifications of them). We shall also assume that there exists an unknown probability of occurrence of each input x_I , denoted $\nu(x_I)$. The input-response configurations of the network should ideally follow the distribution $\nu(x_I)\delta_{f(x_I)}(x_R)$ whereas its true distribution is $\nu(x_I)\psi(x_R; x_I)$. Up to an additive constant, the

Kullback distance between these two distributions is a function of the parameters w and given by $-K(w)$ where $K(w) = \sum_{x_I \in \Omega_I} \nu(x_I) \log \psi(f(x_I); x_I)$.

However, since the distribution ν and function f can only be observed through a finite sample (x_I^1, \dots, x_I^n) of training input configurations together with the associated correct values $(f(x_I^1), \dots, f(x_I^n))$, one uses an empirical version of $K(w)$ given by

$$\hat{K}(w) = \frac{1}{n} \sum_{k=1}^n \log \psi(f(x_I^k); x_I^k).$$

The learning rules which are currently used implement steepest descent maximization algorithms for \hat{K} . They are given by the iterative weight updating rules (if w^t is the current value of the weights at time t of the learning process) :

$$w^{t+1} - w^t = \alpha_t \cdot \text{grad}_{w^t} \hat{K}. \quad (6)$$

where α_t is a (generally decreasing) gain.

Before describing these algorithms, we mention the fact that the models we are studying may represent arbitrary functions from Ω_I to Ω_R (provided their dimension is large enough). More precisely, this is true in the following cases (define the range of a sequential model as the cardinality of the largest C with $w_C \neq 0$) :

- Sequential models :
 - Models with ranges $|S|$, and $H = \emptyset$. This is a direct consequence of the Hammersley Clifford theorem ([BES]).
 - Models of range 2 if the number of hidden neurons is large enough ([SUS]).
- Synchronous models:
 - Models with $q = |S| - 1$ and $H = \emptyset$ ([YOU3]).
 - Models with $q = 1$ if the number of hidden neurons is large enough ([YOU2]).

2.2 Sequential case ([C1]).

Learning rules in this case are provided in [AHS], [AZ2]. One has :

$$\frac{\partial \hat{K}}{\partial w_C} = \frac{1}{n} \sum_{k=1}^n \left\{ E_{\bar{\pi}} \left[\prod_{s \in C} x_s \mid x_I = x_I^k, x_R = f(x_I^k) \right] - E_{\bar{\pi}} \left[\prod_{s \in C} x_s \mid x_I = x_I^k \right] \right\} \quad (7)$$

where $E_{\bar{\pi}}$ is the (conditional) expectation for the distribution $\bar{\pi}$ given in (3).

2.3 Synchronous case ([C2]).

In this case, we adopt a slightly different approach, still based on the Kullback information distance. Instead of constraining the equilibrium distribution of a single configuration, namely $\pi(\cdot; x_I)$ to have a response almost equal to $f(x_I)$, we consider the p -step distribution with clamped input $\mu(\cdot; x_I)$ which we have introduced in [C2]. It is defined on $(\Omega_H \times \Omega_R)^p$ and (setting $V = H \cup R$) $\mu(x_V(0), \dots, x_V(q); x_I)$ is the probability of observing the sequence $(x_V(0), \dots, x_V(q))$ during the synchronous evolution with clamped input to x_I at equilibrium. Let $\bar{\mu}(\cdot; x_I)$ be the marginal distribution of $\mu(\cdot; x_I)$ over $(\Omega_R)^p$. The ideal behaviour of the network, given the fact that the input is x_I , still should give an (almost) constant output, equal to $f(x_I)$. Since we now are thinking in terms of p -step distribution, a natural candidate for the ideal network distribution over these p -steps (note that x_I is clamped during running) is

$$\nu(x_I)\delta_{f(x_I)}[x_R(q)] \dots \delta_{f(x_I)}[x_R(0)],$$

which has to be compared with the true p -step distribution which is $\nu(x_I)\bar{\mu}[x_R(q), \dots, x_R(0); x_I]$.

Writing the Kullback information distance between these distributions, and replacing it with the empirical version as before, one obtains the following expression

$$\hat{K}(w) = \frac{1}{n} \sum_{k=1}^n \log \bar{\mu}[f(x_I^k), \dots, f(x_I^k); x_I^k]$$

which has to be maximized in w .

The derivation of the learning rule is analogous to the sequential one, and one obtains, for $c = (s_q, \dots, s_0)$, denoting, for short $\mu^k(\cdot) = \mu(\cdot; x_I^k)$,

$$\begin{aligned} \frac{\partial \hat{K}}{\partial w_c} = & \frac{1}{n} \sum_{k=1}^n \left\{ E_{\mu^k} \left[\prod_{r, s_r \neq 0} x_{s_r}(r) \mid x_R(0) = \dots = f(x_I^k) \right] \right. \\ & \left. - E_{\mu^k} \left[\prod_{r, s_r \neq 0} x_{s_r}(r) \right] \right\}, \end{aligned} \quad (8)$$

2.4 Practical implementation.

It must be noted that the expectations in (7) and (8) can only be evaluated by moving averages over outputs of Monte Carlo simulation of the relevant distribution. This implies that, in order to update the weights (equation (6)), it is necessary to run a whole series of sampling procedures (two for each example) to compute the gradient of the Kullback distance. At this point, one can easily measure the impact of synchronous sampling on the acceleration of learning procedures (provided, of course, that a parallel hardware is available).

Two kinds of sampling algorithms with clamped input must be defined, namely one for each expectation in the right hand term of equations (7) and (8). The

second expectation is with respect to the equilibrium distribution of the network (with clamped inputs, cf. section 1.) so that it suffices to run the network (with clamped inputs) to simulate it. The first one is a conditional version of the second given that the values in response are correct, and the important fact is that, in both sequential and synchronous cases, it can be computed by making the network run with inputs *and* responses clamped to the correct values. That one obtains the conditional distribution by clamping the concerned part of the network heavily relies on the assumptions [C1] and [C2] which are made, because they relate to Gibbs field formulation in which this property is typically true. Moreover, this would also be false in the synchronous case without the choice we made to use p -step distributions, instead of retaining the same formulation as the sequential case.

References

- [AHS] D.H. Ackley, G. Hinton, T. Sejnowski (1985) : A learning algorithm for Boltzmann machines. *Cognitive Sci.* 9 147-169.
- [AZ1] R. Azencott (1990) : Synchronous Boltzmann Machines and Gibbs Fields in *Neurocomputing edit.* Folgerman-Hierault, NATO ASI series, vol.F 68, Springer Verlag.
- [AZ2] R. Azencott (1991) : High-Order Interactions and Synchronous Learning. *Proceedings of "stochastic models, statistical methods and algorithms in image analysis"*, Ed. P. Barone and A. Frigessi, Lecture Notes in Statistics, Springer.
- [BES] J. Besag (1974) : Spatial Interaction and the Statistical Analysis of Lattice Systems. *J. of Roy. Stat. Soc.* B-36 pp 192-236.
- [ADY] R. Azencott, A. Doutriaux, L. Younes (1992) : Synchronous Boltzmann machines and curve identification tasks. *Network* 4 (1993) 461-480.
- [LAC] J. Lacaille (1991) : Thesis at University Paris XI.
- [SUS] H.J. Sussmann (1988) : On the convergence of learning algorithms for Boltzmann Machines (preprint Rutgers University).
- [YOU1] L. Younes (1993) : Synchronous Random Fields and Image restoration. (preprint CMLA-DIAM).
- [YOU2] L. Younes (1993) : Synchronous Boltzmann machines can be universal estimators (preprint CMLA-DIAM).
- [YOU3] L. Younes (1994) : Representation of Gibbs fields with synchronous random fields (preprint CMLA-DIAM).