Learning algorithms for perceptrons from statistical physics
Mirta Gordon, Pierre Peretto, Dominique Berchier

To cite this version:
Learning algorithms for perceptrons from statistical physics

Mirta B. Gordon(1)(*), Pierre Peretto(2) and Dominique Berchier(1)(**) 

(1) DRFMC/SPSMS, CEN Grenoble, 85X, 38041 Grenoble Cedex, France 
(2) DRFMC/SP2M, CEN Grenoble, 85X, 38041 Grenoble Cedex, France 

(Received 22 May 1992, accepted in final form 23 July 1992) 

Résumé. — Nous déduisons des algorithmes d’apprentissage pour des perceptrons à partir de considérations de mécanique statistique. Des quantités thermodynamiques sont considérées comme des fonctions de coût, dont on obtient, par une dynamique de gradient, les efficacités synaptiques qui apprennent l’ensemble d’apprentissage. Les règles ainsi obtenues sont classées en deux catégories suivant les statistiques, de Boltzmann ou de Fermi, utilisées pour dériver les fonctions de coût. Dans les limites de températures nulle ou infinie, la plupart des règles trouvées tendent vers les algorithmes connus, mais à température finie on trouve des stratégies nouvelles, qui minimisent le nombre d’erreurs dans l’ensemble d’apprentissage.

Abstract. — Learning algorithms for perceptrons are deduced from statistical mechanics. Thermodynamical quantities are used as cost functions which may be extremalized by gradient dynamics to find the synaptic efficacies that store the learning set of patterns. The learning rules so obtained are classified in two categories, following the statistics used to derive the cost functions, namely, Boltzmann statistics, and Fermi statistics. In the limits of zero or infinite temperatures some of the rules behave like already known algorithms, but new strategies for learning are obtained at finite temperatures, which minimize the number of errors on the training set.

J’ai rencontré Rammal pour la première fois dans le cadre du DEA “Matière et Rayonnement” de l’Université de Grenoble. Je l’ai revu bien souvent ensuite et je lui dois, en partie, de m’être intéressé aux réseaux de neurones. A la fin de 1982 en effet, R. Maynard recevait de P.W. Anderson deux preprints, l’un d’Anderson lui-même sur la dynamique de la soupe prébiotique et l’autre de Hopfield sur le modèle Hebbien de mémoire dans les réseaux neuronaux. Il me les communiquait et après avoir programmé la dynamique de la soupe de Anderson je décidais de m’impliquer plus avant dans la théorie des réseaux de neurones. Nous avons donc pendant l’année 1983 formé R. Maynard, R. Rammal et moi-même un petit groupe qui s’est

(*) Also with C.N.R.S. 
(**) Supported by Sunburst-Fonds, Board of the Swiss Federal Institutes of Technology, Zürich.
1. Introduction.

One of the present interests on neural networks arises from the fact that they are endowed with generalization capabilities, that is, they are able to extract, from a set of examples (the training set of patterns), the underlying rule that generates them. If good generalization is achieved, the network should give a correct output to an unfamiliar input, an input not belonging to the training set.

The interactions $J_{ij}$ between neurons $j$ and $i$, $1 \leq i, j < N$, and the thresholds $J_0$, fully characterize a neural network of $N$ neurons. Learning consists in modifying these interactions so as to make the response of the network to the patterns of the learning set as close as possible to the corresponding expected outputs. As usual, we shall write $\xi_i^\mu$ for the state of neuron $i$, and $\mu$ for the labels of the patterns in the training set, $1 \leq \mu \leq P$.

A number of learning algorithms has already been put forward for neural nets [1], most of them stemming from the seminal idea of Hebb, who pointed out the role of correlations of activities in the learning dynamics. However, the original Hebbian law, in which learning a new pattern $\mu$ introduces a modification of the synaptic weights given by [2]:

$$\Delta J_{ij}(\mu) = \epsilon \xi_i^\mu \xi_j^\mu \ ; \ \epsilon > 0 \quad (1.1)$$

is not optimal in the sense that neither does it allow to reach the maximal (theoretical) capacity [3,4] of the network for random patterns nor is it able to learn without errors when $P$, the number of patterns in the learning set, scales with the number of neurons [5]. Attention then focused on the simplest network one can imagine: the perceptron, which consists of $N$ input neurons acting, through a set of interactions $J_i$, onto a unique output unit. A specific algorithm, namely the Perceptron algorithm, was devised for this type of network [6]. It modifies the synaptic efficacies according to (1.1), but taking into account only those patterns the network gives the wrong answer to. The training set is iteratively tested and the $J_i$ modified until all the patterns are well learnt. It has been proved [6] that if the learning set is linearly separable, that is, if a set of interactions $J_i$ exists such as the response

$$\sigma(\mu) = \text{Sign} \left( \sum_{i=0}^{N} J_i \xi_i^\mu \right) \quad (1.2)$$

coincides with the expected response $\xi_i^\mu$ for all the patterns of the learning set, then the algorithm finds these interactions. In (1.2) we have introduced, as usual, a threshold unit $\xi_0^\mu = -1$, associated with $J_0$, the threshold value. The Perceptron algorithm is not very efficient, because it only allows for marginal stabilities, and several new learning rules have been proposed which both speed up the search times and improve the stability of solutions with respect to small perturbations of the input signals and /or interactions [7] [8]. All these algorithms are able to find a solution when it exists, and it is believed that the best generalization rates correspond to the highest stabilities. Recently, an algorithm that finds the best generalization...
rates for a perceptron has been proposed [9], but it needs, in its implementation, a network of higher complexity, namely one with an infinite number of hidden units.

However, problems to be treated by neural networks are in general more complex than linearly separable, and the perceptron architecture is too poor for such performances as good learning and generalization [10]. Therefore, networks with more involved designs are needed. The problem of learning in these complex architectures is difficult and not completely solved yet. Among various attempts made so far, constructivistic algorithms seem most promising. They proceed by adding to the network one new cell at a time, so that learning reduces to learning in the new perceptrons successively created by the construction. The problem of learning remains central. But now one is less interested in finding the solution (which anyway, at least at the stage of the intermediate perceptrons, does not exist) that in finding the best solution, which minimizes the number of errors on the learning set [11, 12]. The relation between minimizing the number of errors on the training set and the generalization performance of such constructivistic algorithms is not clear yet, but it is believed that the smaller the network built up by the learning algorithm, the better its generalization rate [13].

The strategy put forward in this paper is to find a convenient thermodynamical quantity to extremalize, like an energy or any other cost function, from which a learning algorithm may be deduced by gradient dynamics. For this goal to be achieved, we rely upon statistical mechanics. The reason of such a choice is that at finite temperature we have the possibility of accepting solutions which, although not optimal, are nevertheless acceptable. Several approaches which optimize the storing capacities of perceptrons even in the regime with errors, are possible. We classify them in two categories, following the statistics used to derive the cost functions, namely, Boltzmann statistics, which we consider in section 2, and Fermi statistics, discussed in section 3. In an interest of clarity, the learning rules are first deduced without normalizing the synaptic efficacies, the introduction of normalization being postponed to section 4. In section 5 we discuss the different learning rules obtained.

2. Cost functions generated with Boltzmann statistics.

2.1 Canonical free energy maximization. — We consider a perceptron network of \( N \) input units which act on a single output via the synaptic efficacies \( \{ J_i \} \), \( i = 0, 1, \ldots, N \). The units can only be in one of two states, +1 or -1. Each example \( \{ \xi_1^\mu, \xi_2^\mu, \ldots, \xi_N^\mu \} \) of the training set consists in the input \( \{ \xi_1^\mu, \xi_2^\mu, \ldots, \xi_N^\mu \} \), which can be viewed as one of the corners of a hypercube in dimension \( N \), and its expected output \( \xi^\mu \). The learning set is stored by the network, defined by its interactions \( \{ J_i \} \), if the stabilities \( \gamma^\mu \) satisfy:

\[
\gamma^\mu = \xi^\mu \sum_{i=0}^{N} J_i \xi_i^\mu > 0
\]  

(2.1)

where \( \mu = 1, 2, \ldots, P \), that is if all the stabilities are positive quantities. The larger \( \gamma^\mu \) the higher the stability of pattern \( \mu \).

To derive learning algorithms from Boltzmann statistics we consider each example as one possible state of the learning process. The energy associated with state \( \mu \) is

\[
E^\mu = \gamma^\mu (\{ J_i \})
\]  

(2.2)

the stability of the network for that pattern. This definition of energy may look strange because it is lower for the less stable patterns. However, the central idea in building learning dynamics
is that if we succeed in stabilizing the least stable pattern, then we are guaranteed that all the other patterns are stable as well.

We suppose that only the states of the training set are accessible to the network during learning. The probability that the network is in state \( \mu \) at temperature \( T = 1/\beta \) is given by the Boltzmann factor \( \frac{e^{-\beta E^\mu}}{Z} \) where

\[
Z = \sum_{\mu=1}^{p} e^{-\beta E^\mu}
\]  

(2.3)

is the partition function. The network spends more time in those states which have lower stabilities and seem more difficult to learn. The free energy \( F(\{J_i\}, T) \) of the system at temperature \( T \) is given by:

\[
F = -T \log Z
\]  

(2.4)

In order to make the least stable pattern as stable as possible, the synaptic efficacies are modified so as to maximize the free energy. This may be achieved with a gradient dynamics:

\[
\frac{\partial J_i}{\partial t} = \frac{\partial F}{\partial J_i} = \sum_{\mu=1}^{p} \frac{e^{-\beta E^\mu}}{Z} \xi_i^\mu \xi^\mu
\]  

(2.5)

which may be written in a discrete time approximation as:

\[
\begin{cases}
J_i(t + 1) = J_i(t) + \Delta J_i(t) \\
\Delta J_i(t) = \epsilon \sum_{\mu=1}^{p} \frac{e^{-\beta E^\mu}}{Z} \xi_i^\mu \xi^\mu; \quad \epsilon > 0
\end{cases}
\]  

(2.6)

with \( \epsilon \) a small time step. This formula has been proposed in [14]. Each hebbian term \( \xi_i^\mu \xi^\mu \) appears with a weight equal to the probability of the learning state, which is higher the more unstable the pattern within the learning set (Fig. 1). Rule (2.6) interpolates between the usual Hebbian rule in the limit of very high temperatures, for which \( \frac{e^{-\beta E^\mu}}{Z} \to \frac{1}{Z} \), and the Minover algorithm with zero stability [7] in the limit of zero temperature, for in that limit (2.6) writes:

\[
\Delta J_i(t) = \epsilon \xi_i^{\mu_{\text{min}}} \xi_{\mu_{\text{min}}}
\]  

(2.7)

where \( \mu_{\text{min}} \) is the least stable pattern of the training set at time step \( t \). Thus, \( \mu_{\text{min}} \) is the pattern, and the only one, that is learnt at time \( t \).

2.2 VARIATIONS ON THE BOLTZMANN STATISTICS THEME. — Learning rule (2.6) has been deduced from the maximization of the free energy. However, in order to increase the stabilities of the patterns of the training set, one may try to maximize the mean energy of the system:

\[
\langle E \rangle = -\frac{\partial \ln Z}{\partial \beta} = \sum_{\mu} E^\mu \frac{e^{-\beta E^\mu}}{Z}
\]  

(2.8)

If this quantity is positive at zero \( T \), it is clear that all the patterns of the training set are stable, because if \( T = 0 \) then \( \langle E \rangle \) is equal to the stability of the least stable pattern.
Maximizing $\langle E \rangle$ gives the following learning rule:

$$\frac{\partial J_i}{\partial t} = \frac{\partial \langle E \rangle}{\partial J_i} = \sum_{\mu=1}^{P} \frac{[1 - \beta (E^\mu - \langle E \rangle) e^{-\beta E^\mu}]}{Z} \xi^\mu \xi^\mu$$  \hspace{1cm} (2.9)

This rule interpolates between Hebb rule at high $T$ and Minover at $T = 0$, exactly in the same way as (2.6), but at finite temperatures the weights of the most stable patterns are negative: in order to learn the still not stable patterns with this rule, very stable patterns should be unlearned (Fig. 1). This strategy is similar to that of Adaline algorithm [15, 16], in which the stabilities are forced to be all equal to 1. Adaline rule modifies the synaptic strengths so as to minimize the following cost function:

$$H = \frac{1}{2} \sum_{\mu} (E^\mu - 1)^2$$  \hspace{1cm} (2.10)

thus unlearning patterns with stabilities higher than 1 and learning those with lower stabilities.

3. Cost functions generated with Fermi statistics.

3.1 Grand potential maximization. --- This second approach is quite different. Now we consider the $P$ learning patterns as states of energies $E^\mu$ that can be occupied or empty, like the energy states of a system of non interacting spinless fermions. We define the occupation numbers $n^\mu=1$ if state (pattern) $\mu$ is occupied and $n^\mu = 0$ if not. One may view $n^\mu$ as a pointer which tells us whether the response of the network to input $\mu$ is erroneous ($n^\mu = 1$) or not ($n^\mu = 0$). The energy of the system is now:

$$E = \sum_{\mu=1}^{P} E^\mu \ n^\mu$$  \hspace{1cm} (3.1)

with $E^\mu$ given, as before, by (2.2). At zero temperature, the energy (3.1) is minimized if all the not learned patterns $\mu$, which have $E^\mu < 0$, are occupied, i.e. have $n^\mu = 1$ and the learned
patterns, having $E^\mu > 0$ are empty. The sum $\sum_{\mu=1}^{P} n^\mu$ is therefore the total number of errors done by the network on the training set.

Consider now the system in equilibrium with a reservoir of chemical potential $\kappa$ and temperature $T = 1/\beta$. The partition function in the grand canonical ensemble is:

$$\Xi = \sum_{\{n^\mu = 0 \text{ or } 1\}} e^{-\beta \left[ E - \kappa \sum_{\mu} n^\mu \right]} = \prod_{\mu=1}^{P} \left( 1 + e^{-\beta(E^\mu - \kappa)} \right)$$

The grand potential is:

$$\phi = -T \sum_{\mu=1}^{P} \ln \left( 1 + e^{-\beta(E^\mu - \kappa)} \right)$$

(3.3)

As before, one may look for the set of interactions that maximize the potential (3.3), because indirectly this will increase the pattern stabilities thus decreasing their population, which is the number of errors. With expression (2.2) for the energy, the resulting learning dynamics is:

$$\frac{\partial J_i}{\partial t} = \frac{\partial \phi}{\partial J_i} = \sum_{\mu=1}^{P} \left( 1 - \tanh \frac{\beta(E^\mu - \kappa)}{2} \right) \xi_i^\mu \xi^\mu$$

(3.4)

The hyperbolic tangent vanishes in the limit $T \to \infty$, and for $T \to 0$ we have $\frac{1}{2} \left( 1 - \tanh \frac{\beta(E^\mu - \kappa)}{2} \right) \to \frac{1}{2} \left( 1 - \text{sign}(E^\mu - \kappa) \right) \to \frac{1}{2} \left( 1 - \text{sign}(E^\mu - \kappa) \right)$. Therefore, this algorithm (Figs. 2 and 3) interpolates between the Hebbian rule, in the limit of high $T$, and the Perceptron algorithm with stability $\kappa$ in the limit of zero temperature, where it gives:

$$\frac{\partial J_i}{\partial t} = \frac{\partial \phi}{\partial J_i} = \sum_{\mu=1}^{P} \theta \left( \kappa - E^\mu \right) \xi_i^\mu \xi^\mu$$

(3.5)

Therefore, if a solution to the problem of learning exists, the chemical potential enables to impose a given stability.
Learning rule (3.4) maximizes the probability, at temperature $T$, that the network gives the correct output to all the patterns of the learning set. This probability writes:

$$P = \prod_{\mu=1}^{P} \mathcal{P}(n^\mu = 0) = \prod_{\mu=1}^{P} \frac{1}{1 + e^{-\beta(E^\mu - \kappa)}}$$

(3.6)

It is straightforward to verify that the maximization of the grand potential (3.3) is equivalent to maximize the logarithm of the probability (3.6).

3.2 OTHER LEARNING RULES. — Within the fermionic formulation, one can try to minimize the mean number of errors,

$$\langle n \rangle = -\frac{\partial \phi}{\partial \kappa} = \sum_{\mu=1}^{P} \frac{1}{2} \left( 1 - \tanh \frac{\beta(E^\mu - \kappa)}{2} \right)$$

(3.7)

It follows the learning dynamics:

$$\frac{\partial J_i}{\partial t} = -\frac{\partial \langle n \rangle}{\partial J_i} = \frac{\beta}{4} \sum_{\mu=1}^{P} \frac{1}{\cosh^2 \left( \frac{\beta(E^\mu - \kappa)}{2} \right)} \xi_i^\mu \xi^\mu$$

(3.8)

Here again, we find an algorithm that reduces to Hebb's rule at high temperature, but at finite $T$ it prescribes a learning dynamics that gives a higher weight at each time step to those patterns whose stabilities are closer to $\kappa$ (Figs. 2 and 3). This prescription, with $\kappa = 0$, is very similar to that of the $\Delta$-rule [15-17], based upon the minimization of a cost function of the form:

$$H = \frac{1}{2} \sum_{\mu} \left( \xi^\mu - \sigma(\xi^\mu E^\mu) \right)^2$$

(3.9)

where $\sigma$ is a sigmoid function. This rule is nothing else than the popular Backpropagation algorithm [18] applied to a perceptron architecture without hidden units. Taking into account
that we are dealing with binary units, which is not the usual case in networks trained with rule (3.9), and choosing \( \sigma (\xi^\mu E^\mu) = \tanh \left( \frac{\beta}{2} \xi^\mu E^\mu \right) = \xi^\mu \tanh \left( \frac{\beta}{2} E^\mu \right) \), yields a cost function that counts the square of the mean number of errors on each pattern at \( \kappa = 0 \):

\[
H = \frac{1}{2} \sum_{\mu} \left( 1 - \tanh \left( \frac{\beta}{2} E^\mu \right) \right)^2 = 2 \sum_{\mu} (n^\mu)^2
\]  

(3.10)

Minimization of (3.10) gives

\[
\frac{\partial J_i}{\partial t} = - \frac{\partial H}{\partial J_i} = \frac{\beta}{2} \sum_{\mu=1}^P 1 - \tanh \left( \frac{\beta E^\mu}{2} \right) \cosh^2 \left( \frac{\beta E^\mu}{2} \right) \xi_i^\mu \xi^\mu
\]  

(3.11)

Here, the weight with which the patterns are learnt is peaked around slightly negative stabilities.

Finally, as in Boltzmann's formulation, let us consider the rule derived from the maximization of the mean energy of the system:

\[
\langle E \rangle = \sum_{\mu=1}^P E^\mu \langle n^\mu \rangle
\]  

(3.12)

thus trying to increase the stabilities mainly of patterns with \( \langle n^\mu \rangle \approx 1 \) which are those with lower stabilities. Maximization of \( \langle E \rangle \) leads to the following learning rule:

\[
\frac{\partial J_i}{\partial t} = \frac{\partial \langle E \rangle}{\partial J_i} = \sum_{\mu=1}^P \frac{1}{2} \left[ 1 - \tanh \frac{\beta}{2} (E^\mu - \kappa) \right] \left\{ 1 - \frac{\beta E^\mu}{2} \left[ 1 + \tanh \frac{\beta}{2} (E^\mu - \kappa) \right] \right\} \xi_i^\mu \xi^\mu
\]  

(3.13)

With this rule patterns with stabilities around \(-\kappa\) are given higher weights, and unlearning of patterns with large positive stabilities takes place (Figs. 2 and 3), like in Adaline type of rules.

4. Normalizing the synaptic weights.

Up to now, no attention has been paid to the range of values of the \( J_i \). However, we know that it is useful to divide the stabilities by the norm of the interactions, \( |J| \), to avoid their increase by a trivial overall scaling of the \( J_i \) [19]. Two conventions are possible: either the full vector \( \mathbf{J} = (J_0, J_1, \ldots, J_N) \)

\[
|\mathbf{J}| = \left( \sum_{\mu=0}^N J_i^2 \right)^{1/2}
\]

(4.1A)

is considered (convention A), or the threshold is excluded but is constrained to lie inside the hypercube of dimension \( N \) whose corners represent all the possible input states of the network (convention B):

\[
\left\{ \begin{array}{l}
|\mathbf{J}^d| = \left( \sum_{\mu=1}^N J_i^2 \right)^{1/2} \\
-\sqrt{N} \leq J_0 \leq \sqrt{N}
\end{array} \right.
\]

(4.1B)
With convention A, \( J_i/|J| \) are the director cosines of a hyperplane through the center of the hypercube in \( N + 1 \) dimensions, the threshold unit providing for the supplementary dimension. With convention B, \( J_0 \) is the distance from the hyperplane of directors \( J_i/|J|^d \) to the center of the hypercube in dimension \( N \). The definition (2.2) of the energies should be modified:

\[
E^\mu = \sum_{i=0}^{N} \frac{J_i}{|J|} \xi_i^\mu \xi^\mu \tag{4.2.A}
\]

if convention A is adopted, or

\[
E^\mu = \sum_{i=1}^{N} \frac{J_i}{|J|^d} \xi_i^\mu \xi^\mu + J_0 \xi_0^\mu \xi^\mu \tag{4.2.B}
\]

with convention B. The calculation proceeds exactly as above, leading to:

\[
\frac{\partial J_i}{\partial t} = \sum_{\mu=1}^{P} \frac{e^{-\beta E^\mu}}{Z|J|} \left( \xi_i^\mu \xi^\mu - \frac{J_i}{|J|^d} E^\mu \right) \tag{4.3.A}
\]

for rule (2.6) with convention A, or to

\[
\begin{align*}
\frac{\partial J_i}{\partial t} &= \sum_{\mu=1}^{P} \frac{e^{-\beta E^\mu}}{Z|J|^d} \left( \xi_i^\mu \xi^\mu - \frac{J_i}{|J|^d} \left( E^\mu - J_0 \xi_0^\mu \xi^\mu \right) \right), \quad i > 0 \\
\frac{\partial J_0}{\partial t} &= \sum_{\mu=1}^{P} \frac{e^{-\beta E^\mu}}{Z} \xi_0^\mu \xi^\mu - \eta J_0^3, \quad \eta > 0
\end{align*}
\tag{4.3.B}
\]

with convention B. In (4.3.B), a stabilizing term \(-\eta J_0^3\) has been added to make sure that the hyperplane will not escape the hypercube. In general then, the effect of normalizing the stabilities is simply to replace the Hebbian factor, in the learning rule without normalization, by:

\[
\xi_i^\mu \xi^\mu \rightarrow \frac{1}{|J|} \left( \xi_i^\mu \xi^\mu - \frac{J_i}{|J|^d} E^\mu \right) \tag{4.4.A}
\]

if convention A is adopted, or:

\[
\xi_i^\mu \xi^\mu \rightarrow \frac{1}{|J|^d} \left( \xi_i^\mu \xi^\mu - \frac{J_i}{|J|^d} \left( E^\mu - J_0 \xi_0^\mu \xi^\mu \right) \right), \quad i > 0 \tag{4.4.B}
\]

and adding the confining term \(-\eta J_0^3\) in the threshold dynamics, in the case of convention B.

5. Discussion and conclusions.

In the preceding sections we have showed that statistical mechanics provides an interesting framework wherein most of known perceptrons learning rules may find their place. All the learning algorithms take the general form:

\[
\Delta J_i(t) \approx \sum_{\mu} \omega^\mu \xi_i^\mu \xi^\mu \tag{5.1}
\]
where $\omega^\mu$, the weight with which each pattern is learned, depends on the specific learning rule. Now the problem is to determine which rule is the most efficient.

Let us assume that there is a solution: a set of interactions $\{J_i\}$ exists such as the stabilities $E^\mu$ are all positive. In this case, Boltzmann and Fermi statistics give rules whose weights behave in one of two possible ways, either always positive like in Perceptron type of rules,

$$\omega^\mu \approx e^{-k \beta E^\mu} \quad (5.2)$$

with $k = 1/2, 1$ or 2, or like in Adaline types of rules, in which weights are positive for low stabilities and negative for high stabilities:

$$\omega^\mu \approx [1 - f(\beta E^\mu)] e^{-k \beta E^\mu} \quad (5.3)$$

where $f(\beta E^\mu) = 1 - \beta(E^\mu - \langle E \rangle)$ or $f(\beta E^\mu) = 1 - \beta E^\mu$ depending on whether we consider Boltzmann or Fermi statistics. Determining the speed of convergence is another question. As learning starts, in general from tabula rasa or from random synaptic efficacies, some patterns are ill-classified. Convergence might be accelerated if those ill-classified patterns exert a strong force on the hyperplane determined by the $\{J_i\}$ so as to place themselves in the correct side with respect to it. In other words, the weights $\omega^\mu$ should be larger the more negative the stabilities. This is achieved by rules (2.5), (2.9), (3.4) and (3.8) or their normalized version like (4.3).

However, in general, there is no solution to the learning problem within the perceptron architecture. Then the best we can do is to devise a rule that minimizes the number of errors. Suppose that such a solution $\{J_i\}$ has been found. Then there necessarily exist ill-classified patterns. Let us assume on the other hand that the temperature is so low that the action of patterns with positive stabilities vanishes. If the weights $\omega^\mu$ do not vanish for negative $E^\mu$, as may be the case in all the rules but (3.8), there will always exist a non vanishing force exerted on the hyperplane, whatever its position, and the learning procedure will never end. The conclusion is that solutions that minimize the number of errors can only be reached by using weights which vanish for large positive and large negative stabilities, like in algorithm (3.8), or in backpropagation (3.11). However, it may take an infinitely long time to reach a solution this way, since the forces on the hyperplane exerted by patterns with strong negative $E^\mu$ is vanishingly small. A sort of compromise is then necessary that combines this type of algorithms and those of perceptron type to speed up the pace of the convergence. Several ways of tackling this problem are possible:

1) The most simple idea is to linearly combine both algorithms, which in the case of $\kappa = 0$ writes:

$$\Delta J_i = \epsilon \sum_{\mu} \left[ \frac{\beta}{4 \cosh^2(\beta E^\mu/2)} + \lambda \frac{e^{-\beta E^\mu}}{Z} \right] \xi_i^\mu \xi^\mu \quad (5.4)$$

where the parameter $\lambda$ has to be large enough to actually decrease the convergence times. Learning dynamics (5.4) looks rather involved, but it is the price to pay when the space of solutions is no more convex.

2) Another way to increase the weight of ill-classified patterns is to use the rule (3.8) and make it asymmetrical by choosing two different temperatures, according to the sign of the stability:

$$\Delta J_i = \epsilon \sum_{\mu} \frac{\beta^\mu}{4 \cosh^2(\beta^\mu E^\mu/2)} \xi_i^\mu \xi^\mu \quad (5.5)$$
with $\beta^u = \beta^+$ if $E^u > 0$ and $\beta^u = \beta^-$ if $E^u < 0$, $\beta^+ > \beta^-$. The determination of the parameters $\beta, \kappa$ and $\lambda$ and the results of simulations carried out using prescriptions (5.4) and (5.5) are beyond the scope of this article, and will be reported elsewhere.

3) Finally, it may be a good strategy to use all the three perceptron type of learning algorithms in turn, starting with Minover or any other rule that ensures high stabilities, which will swiftly give the solution to the problem if the solution exists, and then finishing the learning session with a combination such as (5.4) or (5.5) if the solution without errors was not found. The problem is to find the criteria that will tell when to swap from a type of rule to another. This may prove to be a difficult problem.

References