Learning and generalization in multineuron interacting feed-forward neural networks

1995 J. Phys. A: Math. Gen. 28 1879

(http://iopscience.iop.org/0305-4470/28/7/011)

J. Phys. A: Math. Gen. 28 (1995) 1879–1887. Printed in the UK

Learning and generalization in multineuron interacting feed-forward neural networks

E Bøtelho†, R M C de Almeida and J R Iglesias

Instituto de Física-Universidade Federal do Rio Grande do Sul, Caixa Postal 15051-91501-970 Porto Alegre, RS, Brazil

Received 4 November 1994

Abstract. We consider learning and generalization in the multi-interacting feed-forward network model recently proposed by H-O Carmesin. With an *a priori* definition of the net architecture, based on symmetries presented by the function to be learnt, we define a generalized Hebb rule, extend the maximum stability learning algorithm to multi-interactions, and obtain training and generalization curves. For rules where different orders of synapses are not correlated the results obtained for the simple perceptron concerning the Hebb rule and through replica calculations in the space of couplings may be straightforwardly adapted to multi-interactions through a simple renormalization of the total number of independent couplings. Analytical and numerical simulation results are compared and show excellent agreement.

1. Introduction

Feed-forward neural networks are made of (usually) binary units—the neurons—disposed in ordered layers that can only act on the next layers, with a well defined direction (for a recent review see [1]). The idea is to use such systems as intelligent devices that may learn a rule from examples; a well known archetype of such devices is the perceptron [2, 3].

The perceptron consists of two layers of neurons, the input one, containing N neurons $S_i = \pm 1$ $i = 1, 2, \ldots, N$ interacting on a single output neuron S_0 through the synaptic intensities J_i. The function—or rule—performed by the perceptron is given by

$$S_0 = f\left(\sum_{i=1}^{N} J_i S_i\right). \tag{1}$$

When S_0 is also taken as a binary variable, f is a Boolean function and may be chosen such that

$$S_0 = \operatorname{sign}(J \cdot S) \tag{2}$$

where J and S are vectors in an N-dimensional space. Equation (2) defines the binary perceptron. When S_0 is a real variable, other perceptron models are realized depending on the choice of $f(x)$; $f(x) = x$ in (1) defines the linear perceptron model [4] while $f(x) = \tanh(\beta x)$, with β being a parameter, defines the analogue (stochastic units) perceptron [4], for instance.

The game now is to teach the network a given rule through examples by presenting to the net a set of P pairs (S_0, S) that satisfy the rule and then modifying the synapses J_i

† E-mail:botelho@if1.ufrgs.br, rita@if1.ufrgs.br,iglesias@if1.ufrgs.br

0305-4470/95/071879+09$19.50

following some dynamics (learning algorithm). The idea is that after the learning session the perceptron is able to give correct answers S_0 to inputs S both belonging or not to the set of previously presented examples. It is not always possible to reach a situation where the perceptron is able to perfectly perform a task or rule. This may either be due to limitations in the learning algorithm or to the intrinsic limitation of perceptrons, that may exactly solve exclusively rules that divide the input space by a plane perpendicular to J into two regions of different answers S_0, that is, linearly separable rules. The performance of a perceptron and a learning algorithm is measured by the training and generalization errors as functions of the number P of examples that are taught to the net. These curves, defined as the average fraction number of wrong answers to inputs respectively in and out of the set of examples, are obtained analytically using statistical mechanics techniques and through numerical simulations (see [1] and references therein).

Very recently Carmesin [5] proposed a multineuron interacting feed-forward network model where the interactions may involve more than one input neuron. This assumption allows feed-forward neural networks to solve rules that are not linearly separable; generalization and learning from examples in these nets are then pertinent items to be investigated. As we shall see in what follows, the analytical results available in the literature concerning the perceptron may be adapted to the multineuron interacting model in some situations. In the following section we analyse the model from a novel point of view and propose learning prescriptions for multi-interacting nets. In section 3 we show that the resulting model may be solved by conveniently adapting the available results for binary perceptrons to an extended configuration space and we present general analytical results for the case where the different orders of couplings are uncorrelated. In section 4 we discuss the numerical simulations and compare them with analytical outcomes. Finally, in section 5 we conclude.

2. The model

Consider a delta function defined as

$$\delta_{\xi,S} = \frac{1}{2^N}\prod_{i=1}^{N}(1 + S_i\xi_i) \tag{3}$$

that returns the value 1 if $S = \xi$ and zero otherwise. Consider also a real function F of binary vectors $S = (S_1, S_2, \ldots, S_N)$, with $S_i = \pm 1$. We define a feed-forward neural network by

$$S_0 = f[F(S)] \tag{4}$$

where $f(x) = \text{sign}(x)$, $f(x) = x$, and $f(x) = \tanh(\beta x)$, with β being a parameter, are, respectively, the binary, linear and analogue versions of feed-forward networks. Using equation (3), any function $F(S)$ may be written as

$$F(S) = \sum_{\{\xi\}} F(\xi)\delta_{S,\xi} \tag{5}$$

where the sum over $\{\xi\}$ stands for the sum over all 2^N input configurations. Should the sum be restricted to a smaller set of input states, equation (5) would return zero for every state outside the summing set. As remarked by Carmesin, this expansion does not compress information, that is, after summing over P states, the function cannot 'guess' the right value for a $(P + 1)$th state, regardless of how large P is.

However, when multilinear perceptrons are being considered, a convenient kind of information compression can be provided to the net in the form of a previously defined architecture, as we shall see in what follows. We first rewrite equation (5) as

$$F(S) = \frac{1}{2^N}\sum_{\{\xi\}} F(\xi) + \frac{1}{2^N}\sum_{i=1}^{N}\left[\sum_{\{\xi\}} F(\xi)\xi_i\right]S_i + \frac{1}{2^N}\sum_{i<j}^{N}\left[\sum_{\{\xi\}} F(\xi)\xi_i\xi_j\right]S_iS_j + \cdots + \frac{1}{2^N}\left[\sum_{\{\xi\}} F(\xi)\xi_1\xi_2\ldots\xi_N\right]S_1S_2\ldots S_N. \tag{6}$$

From the above equation, a threshold B_0 may be defined as

$$B_0 = \frac{1}{2^N}\sum_{\{\xi\}} F(\xi) \tag{7}$$

while multilinear synapses of order λ may be written as

$$B^{(\lambda)}_{i_1 i_2 \ldots i_\lambda} = \frac{1}{2^N}\sum_{\{\xi\}} F(\xi)\xi_{i_1}\xi_{i_2}\ldots\xi_{i_\lambda}. \tag{8}$$

The above equations are equivalent to equation (17) of [5] and represent averages of the product of $F(S)$ with some multilinear form over all configurations.

On the other hand, due to symmetries in $F(S)$ some of these multilinear synapses may be zero. For example, odd functions $F(S)$ yield all even-order synapses zero or linearly separable rules without threshold have all synapses given by equations (8) equal to zero, except for the first-order one.

Here we propose some learning rules and algorithms where only the non-vanishing synapses are considered, that is we choose some orders of multilinear forms to be realized by the multi-interacting perceptron. This is equivalent to telling the perceptron how to compress the information that is being provided through examples, that is it is analogueous to deciding between a straight line, spline or any other method to fit a set of points obtained in an experiment or numerical simulation. When the function $f(x)$ is invertible (which is not the case for the signal function), the expressions for all order synapses are given by averages of $f^{-1}(S_0) = F(\xi)$ over all configuration space. In principle, a representative (hopefully smaller) set should be enough to yield the correct averages, and all synapses could be obtained through averages over these representative sets of examples. In the case of non-invertible functions f, such as the signal function in the binary case, some learning algorithm may be implemented as back propagation [6, 7] or the Hebb learning rule, for example. Observe that $J_i^{(1)} = B_i^{(1)}$ given by equation (8) corresponds to the Hebb learning rule only for the linear perceptron case, for which the transmission function f is the identity.

We shall restrict ourselves to the case where the transmission function $f(x)$ is the signal function. We show how to implement other rules that are not linearly separable—that will display multi-interacting synapses—and, more importantly, how to deal with them, which we do in the next sections.

Finally, we remark that multineuron interactions have been extensively considered showing some success in enhancing the performance of attractor neural network models [8, 9] and a recent review may be found in [10]. For feed-forward networks, previous multineuron interaction models besides the recent work by Carmesin are the parity machine [11, 12] and the sigma–pi units proposed by Rumelhart *et al* [7].

3. Analytical calculations

When considering multi-interacting perceptrons the correlation among different orders of couplings determines the possible analytical approaches. Reversed wedge or many teacher problems, for instance, are easily described by multilinear forms [13], but couplings of different orders are correlated. In this paper we focus on multilinear rules with non-correlated interactions. We also only consider functions with vanishing thresholds B_0, that is the average of $F(\xi)$ over all input space is zero.

Consider then a multilinear rule, containing non-vanishing multilinear forms of orders belonging to the set $\Lambda = (\lambda_1, \lambda_2, ...)$. The number M_{λ_i} of different λ_i interactions is

$$M_{\lambda_i} = \frac{N!}{\lambda_i!(N-\lambda_i)!} \tag{9}$$

such that the total number M of couplings is given by the sum $M = M_{\lambda_1} + M_{\lambda_2} + \cdots$. When all orders are present, then $M = 2^N$.

Consider an input configuration $S = (S_1, S_2, \ldots, S_N)$ and the associated multilinear form S_Λ, written in a vector notation:

$$S_\Lambda = (S_1, S_2, \ldots, S_N, S_1S_2, S_1S_3, \ldots, S_1S_2 \ldots S_N) \tag{10}$$

where all and only orders belonging to the set Λ are present. (In the above expression, we explicit the first, second and Nth orders for clarity reasons, but they could be absent). The couplings of a multi-interacting net may also be written as Λ-vectors:

$$J_\Lambda = \left(J_1^{(1)}, J_2^{(1)}, \ldots, J_N^{(1)}, J_{12}^{(2)}, J_{13}^{(2)}, \ldots, J_{12\ldots N}^{(N)}\right) \tag{11}$$

and a Λ-scalar product may be written as

$$\langle J_\Lambda | S_\Lambda \rangle = \sum_{\lambda \in \Lambda} \sum_{i_1 < i_2 < \ldots < i_\lambda}^{N} J_{i_1 i_2 \ldots i_\lambda}^{(\lambda)} S_{i_1} S_{i_2} \ldots S_{i_\lambda} \tag{12}$$

where only the orders in Λ are considered. The multi-interacting perceptron containing couplings of the orders in Λ may now be defined as

$$S_0 = \text{sign}(\langle J_\Lambda | S_\Lambda \rangle). \tag{13}$$

We shall now consider learning and generalization in the multi-interacting perceptron defined above. As the simple perceptron, that only solves exactly linearly separable rules, the multi-interacting perceptron only solves exactly rules for which the multilinear expansion contains exclusively orders belonging to the set Λ, that is rules that may be written as

$$\sigma_0 = \text{sign}(\langle B_\Lambda | S_\Lambda \rangle) \tag{14}$$

where B_Λ represents a teacher Λ-vector.

Following Opper *et al* [14] we remark that the generalization rate may be obtained as a function of the average Λ-scalar product of the net J_Λ and the teacher B_Λ. We define

$$x = \langle J_\Lambda | S_\Lambda \rangle \tag{15}$$

$$y = \langle B_\Lambda | S_\Lambda \rangle \tag{16}$$

with $\langle J_\Lambda | J_\Lambda \rangle = \langle B_\Lambda | B_\Lambda \rangle = 1$. As in the simple perceptron case, for random inputs S, x and y are correlated Gaussian variables with zero mean, unit variance and covariance $\langle xy \rangle = R$, where R is the Λ-scalar product between the net and the teacher:

$$R = \langle J_\Lambda | B_\Lambda \rangle. \tag{17}$$

We remark at this point that the non-correlation between different orders of the teacher Λ-vector is a necessary condition to obtain x and y as Gaussian correlated variables. The generalization error ϵ_g is then given by the probability that $xy < 0$; the calculations are the same as for the simple perceptron [14], that is

$$\epsilon_g = \frac{1}{\pi} \cos^{-1} R. \tag{18}$$

To obtain R analytically we must consider explicitly the learning strategy; we first present the results for the optimal perceptron, obtained through replica calculations over the coupling space. As the couplings are assumed to be non-correlated, the number of degrees of freedom for the synapses is equal to the total number M of multilinear forms. Thus the volume occupied by all J_Λ that produce correct answers $S_0 = \sigma_0$ for all P inputs belonging to the set of taught examples $\{S^\mu\}$, $\mu = 1, 2, \ldots, P$ is given by

$$Z = \int \mathrm{d}J_\Lambda \delta(\langle J_\Lambda | J_\Lambda \rangle - 1) \prod_\mu \Theta\left[\langle J_\Lambda | S^\mu_\Lambda \rangle \mathrm{sign}(\langle B_\Lambda | S^\mu_\Lambda \rangle) - \kappa\right] \tag{19}$$

where $\kappa > 0$ guarantees a finite basin of attraction, as in the simple perceptron case. As usual, an average over different sets of P examples must be performed using the replica method: the average of $\ln Z$ over all sets $\{S^\mu\}$ of P examples is obtained from the average of Z^n, in the $n \to 0$ limit, that is

$$\langle \langle \ln Z \rangle \rangle_{\{S^\mu\}} = \lim_{n \to 0} \frac{\langle \langle Z^n \rangle \rangle_{\{S^\mu\}} - 1}{n} \tag{20}$$

where $\langle \langle Z^n \rangle \rangle_{\{S^\mu\}}$ may be written as

$$\begin{aligned} \langle \langle Z^n \rangle \rangle_{\{S^\mu\}} = & \int \left(\prod_a \mathrm{d}J^a_\Lambda \delta(\langle J^a_\Lambda | J^a_\Lambda \rangle - 1) \right) \int_\kappa^\infty \left(\prod_{a\mu} \mathrm{d}z^{a\mu} \frac{\mathrm{d}x^{a\mu}}{2\pi} \right) \\ & \times \int \left(\prod_\mu \mathrm{d}y^\mu \frac{\mathrm{d}\sigma^\mu}{2\pi} \right) \exp\left(\mathrm{i} \sum_\mu \sigma^\mu y^\mu + \sum_{a\mu} x^{a\mu} z^{a\mu} \right) \\ & \times \left\langle \left\langle \prod_\mu \exp\left(-\mathrm{i} \sum_a x^{\mu a} \mathrm{sign}(y^\mu) \langle J^a_\Lambda | S^\mu_\Lambda \rangle - \mathrm{i}\sigma^\mu \langle B_\Lambda | S^\mu_\Lambda \rangle \right) \right\rangle \right\rangle_{\{S^\mu\}} \end{aligned} \tag{21}$$

where a runs from 1 to the number of replicas n. This expression is similar to the simple perceptron with an important difference: in the Λ-scalar products there appears terms of higher orders that, in principle, are coupled. However, when synapses of different orders are not correlated the averages for each term in the different multilinear forms may be performed independently and the result is similar to those concerning binary perceptrons, in the same way as pointed out by Kohring [15] for many-neuron interacting neural networks: the calculations are the same. After performing the average over the sets of examples and integrating over σ^μ, we have

$$\begin{aligned} \langle \langle Z^n \rangle \rangle_{\{S^\mu\}} = & \int \left(\prod_a \mathrm{d}J^a_\Lambda \delta(\langle J^a_\Lambda | J^a_\Lambda \rangle - 1) \right) \\ & \times \left\{ \int_\kappa^\infty \left(\prod_a \mathrm{d}z^a \frac{\mathrm{d}x^a}{2\pi} \right) \int \frac{\mathrm{d}y^\mu}{\sqrt{2\pi}} \exp\left(-\frac{y^2}{2} \right) \right. \\ & \times \exp\left[\mathrm{i} \sum_a x^a (z^a - R_a |y|) - \frac{1}{2} \sum_a (x^a)^2 (1 - R_a)^2 \right. \\ & \left. \left. - \sum_{ab} x^a x^b (q_{ab} - R_a R_b) \right] \right\}^{\alpha M} \end{aligned} \tag{22}$$

where

$$q_{ab} = \langle J_\Lambda^a | J_\Lambda^b \rangle \tag{23}$$

$$R_a = \langle J_\Lambda^a | B_\Lambda \rangle. \tag{24}$$

From now on the problem is completely equivalent to the simple perceptron. We can then obtain R as a function of α in the replica symmetric ansatz and $q \to 1$ from saddle-point equations as in Opper *et al* [14]; the only difference lies in the definition of α that must now take into account that the number of terms in the Λ-scalar products is M, that is

$$\alpha = \frac{P}{M}. \tag{25}$$

Figure 1 shows the analytical generalization error ϵ_g against $\alpha = P/M$, given by equation (18) together with data from numerical simulations using an extension to multi-interactions of the maximum stability algorithm, as we shall discuss in the next section.

We have also considered an extended Hebb learning rule. As we have already stated, equations (8) may not be taken as a learning algorithm due to the non-invertibility of the signal function. However, this is also true in the simple perceptron model, and a Hebb rule may still be assumed where the argument $F(S^\mu)$ is replaced by $\mathrm{sign}(\langle B_\Lambda | S_\Lambda^\mu \rangle)$. For every order of interaction λ_i present in Λ, we consider the following:

$$J_{j_1,\ldots,j_{\lambda_i}}^{(\lambda_i)} = \frac{1}{P} \sum_\mu \mathrm{sign}\left(\langle B_\Lambda | S_\Lambda^\mu \rangle\right) S_{j_1}^\mu \ldots S_{j_{\lambda_i}}^\mu. \tag{26}$$

To obtain the generalization and training error curves we follow the prescription by Vallet [16]. First, note that a multilinear perceptron with multi-interactions given by J_Λ returns the right answer to an input S if

$$\mathrm{sign}(\langle B_\Lambda | S_\Lambda \rangle) \langle J_\Lambda | S_\Lambda \rangle \geqslant 0 \tag{27}$$

where B_Λ is the teacher Λ-vector, as before. Using equation (26), the above condition may be expressed as

$$\sum_\mu^P z^\mu(S) = \frac{1}{P} \sum_\mu^P \mathrm{sign}\left(\langle B_\Lambda | S_\Lambda \rangle\right) \mathrm{sign}\left(\langle B_\Lambda | S_\Lambda^\mu \rangle\right) \langle S_\Lambda | S_\Lambda^\mu \rangle \geqslant 0 \tag{28}$$

where S_Λ^μ are the multilinear forms associated with each example S^μ, $\mu = 1, \ldots, P$ used to build up the connections J_Λ. Then, the probability that a wrong answer is given to a randomly chosen input S, that is the generalization error rate, is given by

$$\mathrm{Probability}\left(\sum_\mu^P z^\mu \leqslant 0\right) = 1 - \tfrac{1}{2}\mathrm{erfc}\left(-\frac{P\langle z \rangle}{\sigma\sqrt{2P}}\right) \tag{29}$$

where $\langle z \rangle$ and σ are, respectively, the average value and the standard deviation of z^μ among all possible sets of P examples. Again, the presence of different order terms, in principle, implies correlations that should be handled with care. However, the non-correlation between different order synapses (that is between components of B_Λ of different orders) allows the averages to be taken as in Vallet [16] for the simple perceptron with the important difference that α must be renormalized to the multi-synaptic case, as given by equation (25). After averaging over all input S, the generalization error reads

$$\epsilon_g = \frac{1}{\pi} \tan^{-1}\left(\sqrt{\frac{\pi}{2\alpha}}\right). \tag{30}$$

Similarly, the training error rate, that is the relative number of wrong answers to questions drawn from the set of examples, may also be obtained from the simple perceptron results by Vallet [16]:

$$\epsilon_t = \int_0^\infty \frac{du}{\sqrt{\pi}} \exp(-u^2) \operatorname{erfc}\left(-u\sqrt{\frac{2\alpha}{\pi}} - \frac{1}{\sqrt{2\alpha}}\right). \tag{31}$$

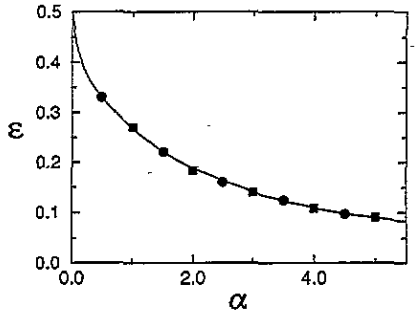

Figure 1. Generalization error curve obtained through the replica method in the space of couplings (full curve) and numerical simulation data using the maximum stability algorithm for $N = 16$ input units, $\Lambda = \{1, 3\}$ (squares) and $\Lambda = \{1, 2, 3\}$ (circles).

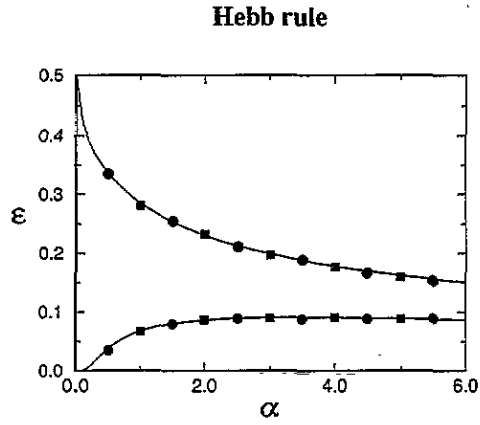

Figure 2. Generalization and training error curves obtained analytically (full curve) and numerical simulation data using the extended Hebb learning rule for $N = 20$ input units, $\Lambda = \{1, 3\}$ (squares) and $\Lambda = \{1, 2, 3\}$ (circles).

Figure 2 shows the theoretical generalization and training error curves for the generalized Hebb rule together with the results for numerical simulations, presenting an excellent agreement.

4. Simulations

For comparison with the generalization error curve with $\alpha = P/M$ for the optimal perceptron we performed numerical simulations with an extension of the maximum stability algorithm [17] to multi-interactions. In fact, the extension to multi-interactions is quite straightforward: in each learning step, the Λ multilinear form of the worst learnt example is subtracted from the Λ-vector J_Λ. We performed simulations with $N = 16$ for perceptrons with multi-interactions of first and third orders ($\Lambda = \{1, 3\}$) and for first, second and third orders ($\Lambda = \{1, 2, 3\}$). We considered averages over 10 different samples, that is 10 different sets of examples, for each load parameter $\alpha = P/M$. After the synaptic matrix J_Λ was built we measured the relative number of correct answers over 1000 randomly chosen inputs S. The simulations were performed on a CRAY Y-MP2E from the National Supercomputing Centre of Universidade Federal do Rio Grande do Sul. The results for the generalization error curve are given in figure 1. Clearly, the good agreement with the analytical data shows that larger nets are not required and demonstrates the absence of correlation among the synapses.

The Hebb rule simulations were performed using equation (26) to build up the synaptic strength from the examples. The simulation conditions considered were: input layers

containing 20 units, $\Lambda = \{1, 3\}$ and $\{1, 2, 3\}$, 20 samples with 1000 questions each, for different load parameters α. In this case we measured the generalization and the training errors. The agreement with theoretical data is evident and is shown in figure 2.

5. Conclusion

With the help of multi-interacting feed-forward neural networks, the problem of devising a machine capable of learning a Boolean function from examples may be stated as follows. Any Boolean function S_0 of N Boolean variables S can be expanded in multilinear forms. The different multilinear terms may be mapped onto multi-interaction couplings between one or more input units and the output unit of a multi-interacting perceptron. Depending on the symmetries present in the function to be learnt, many multilinear terms are zero and this information should be translated in an *a priori* assumption about the orders of interaction present in the architecture of the multiinteracting perceptron. A given architecture of the net is equivalent to a definite way of compressing the relevant information contained in the data (examples) supplied to the net. As an example, the information contained in examples of a linearly separable rule are conveniently compressed by the simple perceptron architecture because every other order in the multilinear expansion of such rules is zero.

Hence the rules should be classified by the set Λ containing the non-vanishing orders of the multilinear expansion. When then is no correlation for any coupling between the input and output units, regardless of their order, the results concerning learning and generalization obtained for the simple perceptron may be adapted to the multi-interacting perceptron through a simple renormalization of the total number of independent synapses, that is replacing N for M.

However, when different orders of interactions are correlated, the number of independent couplings may be drastically reduced. In these cases the renormalization of the simple perceptron results does not apply and specific calculations both in the space of couplings for the optimal perceptron, as well as other learning prescriptions such as the Hebb rule, must be performed. These results, together with numerical simulations for diverse multilinear, correlated rules such as the reversed wedge or many-teacher problem, will be published elsewhere.

Acknowledgments

We acknowledge discussions and careful reading by J J Arenzon and N Lemke. The simulations were performed thanks to a Cray time grant provided by the Centro Nacional de Supercomputação da Universidade Federal do Rio Grande do Sul. Work partially supported by Brazilian agencies Conselho Nacional de Pesquisa Científica e Tecnológica (CNPq), Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul (FAPERGS) and Financiadora de Estudos e Projetos (FINEP).

References

[1] Watkin T L and Rau A 1993 *Rev. Mod. Phys.* 65 499
[2] Rosenblatt F 1962 *Principles of Neurodynamics* (New York: Spartan)
[3] Minsky M L and Papert P A 1969 *Perceptrons* (Cambridge: MIT)
[4] Herz J A, Krogh A and Thorbergsson G I 1989 *J. Phys. A: Math. Gen.* 22 2133
[5] Carmesin H-O 1994 *Phys. Lett.* **188A** 27

[6] Werbos P 1974 Beyond regression: new tools for prediction and analysis in behavioral sciences *PhD thesis* Harvard University
[7] Rumelhart D E, Mclelland J L and the PDP research group 1986 *Parallel Distributed Processing* vol 1 (Cambridge: MIT)
[8] Gardner E 1987 *J. Phys. A: Math. Gen.* **20** 3453
[9] de Almeida R M C and Iglesias J R 1990 *Phys. Lett.* **146A** 239
[10] de Almeida R M C, Penna T J P and de Oliveira P M C 1994 Multineuron interaction effects *Annual Reviews of Computational Physics* (Scientific: World Scientific) to appear
[11] Mitchison G J and Durbin R M 1989 *Biol. Cybern.* **60** 345
[12] Hansel D, Mato G and Meunier C 1992 *Europhys. Lett.* **20** 471
[13] Botelho E and de Almeida R M C in preparation
[14] Opper M, Kinzel W, Kleinz J and Nehl R 1990 *J. Phys. A: Math. Gen.* **23** L581
[15] Kohring G A 1990 *J. Physique* **51** 145
[16] Vallet F 1989 *Europhys. Lett.* **8** 747
[17] Krauth W and Mezard M 1987 *J. Phys. A: Math. Gen.* **20** L745