Tilinglike learning in the parity machine

Michael Biehl and Manfred Opper
Institut für Theoretische Physik, Justus-Liebig-Universität Giessen, 6300 Giessen, Federal Republic of Germany
(Received 25 April 1991)

An algorithm for the training of multilayered feedforward neural networks is presented. The strategy is very similar to the well-known tiling algorithm, yet the resulting architecture is completely different. New hidden units are added to one layer only in order to correct the errors of the previous ones; standard perceptron learning can be applied. The output of the network is given by the product of these k (± 1) neurons (parity machine). In a special case with two hidden units, the capacity α_c and stability of the network can be derived exactly by means of a replica-symmetric calculation. Correlations between the two sets of couplings vanish exactly. For the case of arbitrary k, estimates of α_c are given. The asymptotic capacity per input neuron of a network trained according to the proposed algorithm is found to be $\alpha_c \sim k \ln k$ for $k \to \infty$ in the estimation. This is in agreement with recent analytic results for the algorithm-independent capacity of a parity machine.

PACS number(s): 87.10.+e, 75.10.Hk, 64.60.Cn, 89.70.+c

I. INTRODUCTION

Feedforward neural networks have proven to be a powerful tool for solving classification problems [1]. In such a device the input is represented by N units or formal neurons and the output is given by the states of, in general, several output units. In the following we exclusively consider the case where only two different classes are assigned to the input, thus one single binary output neuron is sufficient.

The task is to map a given set of p input configurations or patterns $\{\xi^\nu=(\xi_1^\nu,\xi_2^\nu,\ldots,\xi_N^\nu)^T\}_{\nu=1,\ldots p}$ into the corresponding desired output $\{S^\nu=\pm 1\}_{\nu=,\ldots,p}$. The inputs are taken to be binary values ($\xi_j^\nu=\pm 1$) in the following, but some of the results also hold in more general cases.

The simplest realization of such a classifier, the perceptron, has been studied intensively [2–5]. It consists of only the input layer and the output unit S_0, which performs a threshold operation directly on the weighted sum of the inputs:

$$S_0=\mathrm{sgn}\left[\sum_{j=1}^{N} J_j\xi_j - T\right] . \tag{1}$$

The weights (or synaptic couplings) J_j, $j=1,\ldots,N$, and the threshold T are taken to be real numbers in the following. The problem of learning is to find couplings such that Eq. (1) is satisfied with $S_0^\nu=S^\nu$ for all patterns. The perceptron can only realize linearly separable functions of the input: in the N space of configurations the patterns having different output are separated by a single hyperplane. This is, of course, a rather drastic restriction. If we consider, for example, independent random variables $\xi_j^\nu, S^\nu=\pm 1$ with equal probability, sets of only up to $p_c=2N$ patterns can be classified correctly [4,6,7]; the critical storage capacity of the system is $\alpha_c=p_c/N=2$ for such patterns.

Several learning procedures, which are guaranteed to find a solution in case the given problem is indeed linearly separable, have been proposed (e.g., [2,8,9]).

In order to realize an arbitrary input-output relation the structure of the network must be more complicated. One possibility is to introduce one or several layers of hidden units between input and output but still restrict the flow of information such that it is passed on by one layer to the next one only, and finally results in the output. It has been proven that any boolean or continuous function of the input can be realized by a feedforward net with one hidden layer of sufficiently many neurons [10,11].

Two major problems occur in the training of such networks with a given fixed architecture.

(a) Learning itself is hard. Nonlocal procedures, for example, a gradient search for the set of couplings with minimal output error (backpropagation and its modifications [1]), are not guaranteed to converge to a solution.

(b) The complexity of the network (i.e., the number of hidden units) needed to solve the given problem is not known *a priori*. If the structure is too simple, no solution can be found; too many hidden units will result in poor generalization abilities [12].

Recently two strategies (among others) have been investigated which circumvent these difficulties.

(i) A fixed mapping is used to determine the output from the configuration of the first hidden layer. Thus only the couplings between this layer and the input have to be adjusted. Examples are the committee machine and the parity machine, where the so-called least-action-algorithm can be applied [13,14].

(ii) The network is constructed while learning: units (or layers of units) are added to the system until the problem is solved. This is the basic concept of the tiling algorithm [15] and its modifications [16–18] or the more geometrically motivated learning schemes by Rujan and

Marchand [19]. Convergence can be guaranteed and learning is performed locally for the most recently added neuron by use of simple perceptron like algorithms.

In the following we present a learning procedure that combines the latter two concepts: neurons are added to the first hidden layer until the mapping is realized and the output of the network is given by the result of the parity operation of this layer.

In the next section we present the algorithm and give a proof of convergence. In Sec. III we study a special case with only two hidden neurons, for which the network's capacity can be calculated exactly. Section IV gives an estimate of the capacity of a parity machine with k hidden units, trained by use of the introduced learning procedure. In Sec. V conclusions are drawn and perspectives are discussed.

II. THE LEARNING PROCEDURE

A. Architecture and growth

The type of network considered consists of N input neurons $\xi_j = \pm 1$ and a number of hidden units of the perceptron type $S_h = \pm 1$, $h = 1,2,\ldots,k$. The total output of the net for an input configuration ξ^ν is given by

$$S^\nu(k) = \prod_{h=1}^{k} S_h^\nu = \pm 1 ,$$

$$S_h^\nu = \mathrm{sgn}\left(\sum_{j=1}^{N} J_j^{(h)} \xi_j^\nu - T^{(h)} \right) . \quad (2)$$

Learning proceeds such that hidden units are added one by one until the required task is performed (see Fig. 1).

Assume that for a net with k units the output for the input patterns $\nu = 1,\ldots,p$ is

$$S^\nu(k) = \begin{cases} -S^\nu & \text{for } \nu = 1,\ldots e_k \\ +S^\nu & \text{for } \nu = e_k +,\ldots,p \end{cases} \quad (3)$$

where the $\{S^\nu\}$ are the desired values and e_k denotes the number of errors. We then add a neuron S_{k+1} and try to find couplings $J_j^{(k+1)}$ and a threshold $T^{(k+1)}$ such that

$$S_{k+1}^\nu = \begin{cases} -1 & \text{for } \nu = 1,\ldots,e_k \quad \text{(output was wrong)} \\ +1 & \text{for } \nu = e_k+1,\ldots,p \quad \text{(output was right)} . \end{cases} \quad (4)$$

Thus by comparison of the given output with the desired one we define a new classification problem for the same input patterns which we try to realize with a perceptron that serves as the $(k+1)$th hidden unit. This has to be repeated until the latest problem of type (4) is linearly separable. Then, after training the last unit by a perceptron algorithm, the total output will be correct for all patterns.

The strategy is in fact very similar to the tiling algorithm [15], yet the growth of the network is restricted to only one hidden layer, where single neurons are added. Of course, the resulting architecture is completely different, since the network is determined to work as a parity machine in the end.

B. Proof of convergence

The proof is in principle the very same as for the tiling algorithm: Suppose ξ^μ is one of the patterns wrongly classified by a net of k hidden units. If we add a unit S_{k+1} with couplings $J_j^{(k+1)} = -\xi_j^\mu$ and a local threshold $T^{(k+1)} = -(N-1)$ it is easy to see that

$$S_{k+1}^\nu = \mathrm{sgn}\left(\sum_{j=1}^{N} (-\xi_j^\mu)\xi_j^\mu + (N-1) \right)$$

$$= \begin{cases} -1 & \text{for } \nu = \mu \text{ with } \xi_j^\nu = \pm 1 \\ +1 & \text{for } \nu \neq \mu . \end{cases} \quad (5)$$

With this choice S_{k+1} acts as a "grandmother" neuron for pattern μ and its only effect is flipping the total output for this pattern from wrong to right. Thus, by adding a neuron, it is always possible to reduce the number of errors by one. As a consequence, any arbitrary binary mapping of p patterns can be realized by a parity machine of at most $O(p)$ hidden units.

In practice, of course, one will try to do better and reduce the number of errors more effectively in each step. Examples of such learning strategies will be discussed in the following sections.

III. SOLVABLE EXAMPLE

In this section we consider a rather simple but instructive case with only two hidden units. The set of couplings for the first neuron is constructed directly from the patterns as it is in the Hopfield model [20], whereas a perceptron of optimal stability will serve as the second unit. We study the network's ability to map $p = \alpha N$ random patterns ξ^ν into random outputs ($\xi_j^\nu, S^\nu = \pm 1$ with equal probability). The first unit S_1 is connected to the input by weights

$$J_j^{(1)} = \frac{1}{\sqrt{N}} \sum_{\nu=1}^{p} \xi_j^\nu S^\nu , \quad T^{(1)} = 0 . \quad (6)$$

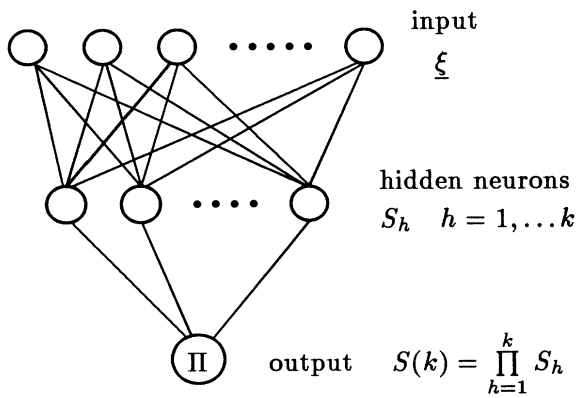

FIG. 1. Architecture of a parity machine with k hidden units.

It is well known that the mapping cannot be realized correctly by such a unit for any $\alpha > 0$. The distribution of the "internal fields"

$$E_1^\nu = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} J_j^{(1)} \xi_j^\nu S^\nu \tag{7}$$

is a Gaussian with $\langle E_1^\nu \rangle = 1$ and width $\sqrt{\alpha}$. Negative values of E_1^ν correspond to wrongly classified patterns and their fraction is given by

$$\frac{e_1}{p} = \Phi\left[-\frac{1}{\sqrt{\alpha}}\right] = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{-1/\sqrt{\alpha}} dz \exp(-\tfrac{1}{2}z^2) \,. \tag{8}$$

The purpose of the second neuron is to correct exactly these errors.

Following Gardner's method [4,22] we calculate the phase-space volume of the couplings $J_j^{(2)}$, which yield the desired total output:

$$V = \left[\int \prod_{j=1}^{N} dJ_j^{(2)} dJ_j^{(1)}\right] \prod_{\nu=1}^{p} \left[\Theta\left[\frac{1}{\sqrt{N}} \sum_{j=1}^{N} J_j^{(1)} \xi_j^\nu S^\nu\right] \Theta\left[\frac{1}{\sqrt{N}} \sum_{j=1}^{N} J_j^{(2)} \xi_j^\nu - T^{(2)} - \kappa\right] + \Theta\left[-\frac{1}{\sqrt{N}} \sum_{j=1}^{N} J_j^{(1)} \xi_j^\nu S^\nu\right] \Theta\left[-\frac{1}{\sqrt{N}} \sum_{j=1}^{N} J_j^{(2)} \xi_j^\nu + T^{(2)} - \kappa\right]\right] \times \delta\left[\sum_{j=1}^{N} (J_j^{(2)})^2 - N\right] \left[\prod_{j=1}^{N} \delta\left[J_j^{(1)} - \frac{1}{\sqrt{N}} \sum_{\mu=1}^{p} \xi_j^\mu S^\mu\right]\right] . \tag{9}$$

The parameter κ is called the stability of the second unit, analogous to the stability of a simple perceptron [4]. Only those sets of couplings are counted which satisfy

$$E_2^\nu = \left[\frac{1}{\sqrt{N}} \sum_{j=1}^{N} J_j^{(2)} \xi_j^\nu - T^{(2)}\right] S_1^\nu S^\nu \geq \kappa \,. \tag{10}$$

V shrinks to zero, where the optimal value of κ is reached. In order to calculate the stability as a function of α, the average $\langle \ln(V) \rangle_{\xi,S}$ over the random patterns and outputs has to be performed. This is done by use of the replica trick; see the Appendix for a more detailed description.

Note again, that only the $J_j^{(2)}$ are treated as free variables; the couplings of the first neuron are fixed according to Eq. (6).

Three types of parameters appear in the calculation.

For the ones which only refer to the Hopfield couplings, e.g., $H = (1/N) \sum_{j=1}^{N} (J_j^{(1)})^2$, the result does not depend on the $J_j^{(2)}$. The properties of the first set of weights are directly determined by the random variables and cannot be altered by the existence of a second unit.

For the second hidden neuron only, the very same variables are introduced as in [4]. The quantity

$$q = \frac{1}{N} \sum_{j=1}^{N} J_j^{(2),\alpha} J_j^{(2),\beta} \quad \text{for } \alpha \neq \beta \tag{11}$$

is the overlap between the solutions in two different replicas. The limit $q \to 1$ refers to the point of optimal stability.

The last group of order parameters describes the correlation between the two sets of couplings. The one with an obvious physical meaning is

$$R = \frac{1}{N} \sum_{j=1}^{N} J_j^{(1)} J_j^{(2)} \,. \tag{12}$$

As it is discussed in the Appendix, one can show that there is a replica-symmetric solution of the problem with vanishing correlations and thus also $R = 0$. We did not rule out the existence of saddle points with nonzero R analytically, but since simulations confirm our result extremely well, $R = 0$ is expected to be the relevant solution.

In the limit $q \to 1$ we obtain, for a given value of $T^{(2)} = T$, the saddle-point equation

$$\frac{1}{\alpha} = \Phi\left[\frac{1}{\sqrt{\alpha}}\right] \int_{-\kappa-T}^{\infty} Dz\,(z + \kappa + T)^2 + \Phi\left[-\frac{1}{\sqrt{\alpha}}\right] \int_{-\kappa+T}^{\infty} Dz\,(z + \kappa - T)^2 \tag{13}$$

where $Dz = dz \exp(-z^2/2)/\sqrt{2\pi}$.

Furthermore the threshold T can be optimized in order to give maximal stability. This yields the equation

$$0 = \Phi\left[\frac{1}{\sqrt{\alpha}}\right] \int_{-\kappa-T}^{\infty} Dz\,(z + \kappa + T) - \Phi\left[-\frac{1}{\sqrt{\alpha}}\right] \int_{-\kappa+T}^{\infty} Dz\,(z + \kappa - T) \tag{14}$$

which has to be satisfied simultaneously.

The critical capacity of the system, where the maximal stability becomes $\kappa = 0$, is given by $\alpha_c \approx 2.373$ with T according to (14).

Note that without a local threshold the second unit is not able to take advantage of the "preprocessing" done by the Hopfield network. For fixed $T = 0$ Eq. (13) become the well-known result of [4] for unbiased patterns and outputs with a capacity $\alpha_c = 2$ only.

In Fig. 2(a) and 2(b) the results for $\kappa(\alpha)$ and $T(\alpha)$ are shown in comparison with simulations for a system of $N = 500$ input neurons. Further studies of the finite-size dependence confirm our result very well, and also $R = 0$

is approved in the simulations.

Our result is remarkable because it can be achieved by a very simple argument as well: According to Eq. (8) the desired S_2^ν are biased variables with a mean value $\langle S_2^\nu \rangle = m_2$ such that

$$\Phi\left[\pm\frac{1}{\sqrt{\alpha}}\right]=\frac{1\pm m_2}{2} . \tag{15}$$

If we neglect dependencies of the S_2^ν on the inputs (which are clearly caused by the $J_j^{(1)}$), we are left with a single perceptron with independent random output which is biased according to (15). A straightforward generalization of Gardner's result [4], averaging over the input variables and output variables independently (as, for example, published in [21]), then yields Eqs. (13) and (14).

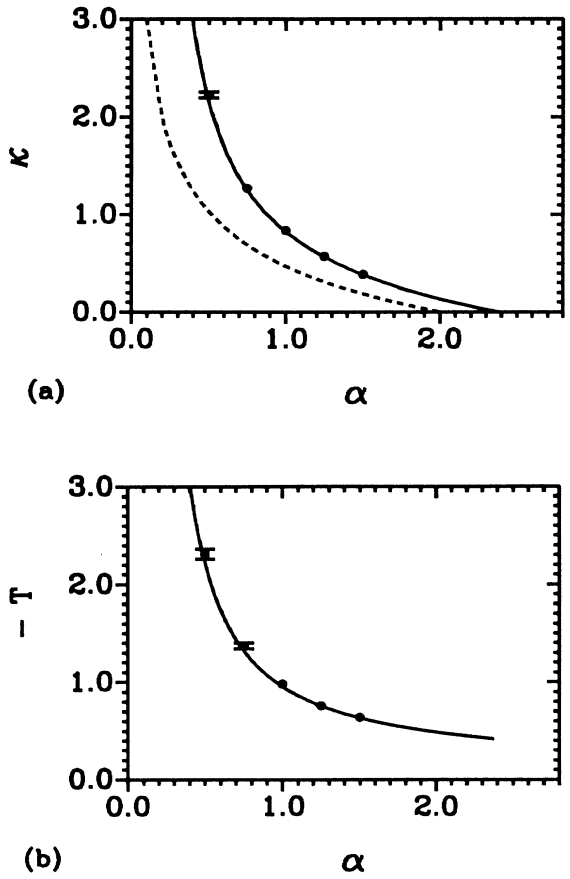

FIG. 2. Solution of Eqs. (13) and (14). The optimal stability κ (a) and the corresponding local threshold T (b) are shown (solid lines) as functions of α [note that in (b) $-T$ is plotted]. The capacity of the system is $\alpha_c \approx 2.373$ and both curves end at this point. In (a) additionally the optimal stability of a single perceptron is shown [4], as it would be recovered for $T=0$ (dashed line). Simulations were done using the AdaTron algorithm [10] for the second unit with a given T and performing a simple ascent search for the optimal threshold. The results were averaged over 50 sets of random patterns/outputs each and the size of the system was $N=500$ input neurons. If not shown, error bars are smaller than the symbols.

Since in our case the correlations of type R vanish exactly this rather simple estimation gives the right result for the optimal stability and capacity. This might not be true in more general cases but we expect that an estimate according to the described method still yields reasonable values for α_c.

IV. ESTIMATION OF CAPACITY

In practice neither the "grandmother" network of Sec. II nor the use of Hopfield couplings will yield reasonably good solutions for a given classification problem. A more promising approach might be to minimize an appropriate cost function for each neuron S_k successively. Possible choices could be [with E_k^ν as defined in Eq. (10) for $k=2$]

$$g_k(x)\sim\sum_{\nu=1}^{p}|1-E_k^\nu|^x , \quad x=1,2,\ldots \tag{16a}$$

or

$$\tilde{g}_k(x)\sim\sum_{\nu=1}^{p}|1-E_k^\nu|^x\Theta(1-E_k^\nu) , \quad x=1,2,\ldots . \tag{16b}$$

Algorithms for these well-behaved optimization problems can be constructed easily; for example, Adaline learning [23,24] refers to $g_k(2)$, whereas the AdaTron algorithm can be used to minimize $\tilde{g}_k(2)$ [25].

A steepest-descent strategy in the context of the proposed learning scheme is to minimize the number of errors

$$e_k=\sum_{\nu=1}^{p}\Theta(-E_k^\nu) . \tag{17}$$

Then, the decrease of e_k is maximal in each step and presumably the resulting network is one with a low number of units.

Minimizing (17) can be done by applying the so-called pocket algorithm [15,26], a modification of simple perceptron learning. Yet this search is time consuming and in practice one assumes that after a reasonable number of learning steps a good yet suboptimal solution is provided [16].

For unbiased random inputs and outputs Gardner and Derrida [22] have calculated the minimal fraction $f_{\min}(\alpha)$ of wrongly mapped patterns that can be achieved by a simple perceptron if the problem is not linearly separable (i.e., $\alpha>2$). We have generalized their result to biased outputs $\langle S^\nu \rangle = m_{\text{out}}$. The replica-symmetric saddle-point equation is for unbiased inputs:

$$\frac{1}{\alpha}=\frac{1+m_{\text{out}}}{2}\int_{T-x}^{T}Dz\,(T-z)^2 + \frac{1-m_{\text{out}}}{2}\int_{-T-x}^{-T}Dz\,(T+z)^2 , \tag{18}$$

the optimal threshold [27] T satisfies

$$0=(1+m_{\text{out}})\int_{T-x}^{T}Dz\,(T-z) + (1-m_{\text{out}})\int_{-T-x}^{T}Dz\,(T+z) , \tag{19}$$

and finally

$$f_{\min}=\frac{1+m_{\text{out}}}{2}\int_{T-x}^{T}Dz+\frac{1-m_{\text{out}}}{2}\int_{-T-x}^{-T}Dz\ . \qquad (20)$$

Although this solution is unstable because replica symmetry is broken [22], it is used in the following to estimate the storage capacity of a parity machine which is trained by the steepest—descent strategy described above. Correlations between the different sets of couplings will be neglected, although we do not expect them to vanish in this case. The scheme is the following.

Assume for a given α the minimal fraction of wrongly mapped patterns is $f_{\min}(h)$ at the hth neuron. This results in an output bias $m_{\text{out}}(h+1)=\langle S^{\nu}_{h+1}\rangle=1-2f_{\min}(h)$ for the next unit to be added. Equations (18)–(20) then give the new $f_{\min}(h+1)<f_{\min}(h)$. The procedure is iterated until $f_{\min}(k)=0$, that is, the problem is solvable with k hidden neurons. In doing so for all values of α, the critical $\alpha_c(k)$ can be determined as a function of the size of the hidden layer. The results are shown in Table I for a few hidden units and in Fig. 3 for values up to $k\approx 130$.

It is very instructive to compare the result with a recent work of Barkai, Hansel, and Kanter [14]. They studied a modification of the parity machine, where correlations between the hidden units vanish by construction. The k units are connected to disjoint sets of N input neurons only [28].

For random classification problems they calculate the fractional volume in the space of the kN couplings, which classify correctly. The physical symmetry of the net is not broken *a priori*, in contrast to our sequential learning procedure. Yet replica symmetry is broken and a one-step replica-symmetry-breaking (RSB) calculation was done. Further RSB effects are expected to only slightly modify the results [29].

Since the total number of couplings is the same as in the fully connected net with only N input neurons, we still define $\alpha_c=p_c/N$.

The estimation as described above still holds for the nonoverlapping architecture; the very same algorithm can be applied. In fact it should be exact, except for RSB corrections of Eqs. (18)–(20), because no correlations have to be taken into account.

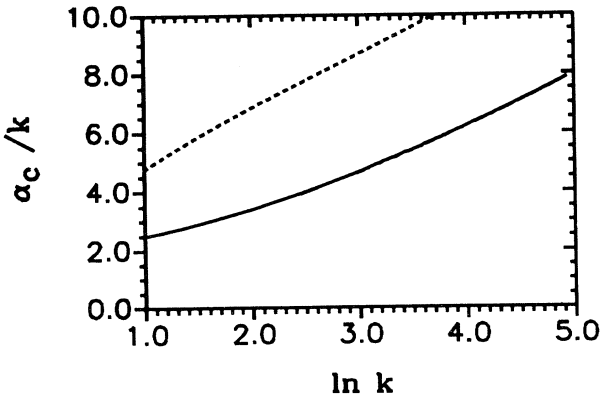

FIG. 3. Result of the estimate for the algorithm-dependent $\alpha_c(k)$ as described in Sec. IV (solid line). For comparison the general result for the nonoverlapping architecture of [15] is shown (dashed line). In both cases the asymptotic behavior is $\alpha_c\sim k\ln k$.

TABLE I. α_c for the tilinglike procedure and as calculated in [14].

	Capacity $\alpha_c=p_c/N$	
k	Tilinglike	As in [14]
1	2.0	2.0
2	4.6	8.1
3	7.7	10
4	11.2	22

The capacity as calculated in [14] does not refer to a specific learning algorithm but states what can be achieved by such a network in principle. Therefore the result should be an upper bound for the capacity of a nonoverlapping network that has learned according to our tilinglike procedure. By training the units one by one, not all of the in principle possible solutions can be found.

For a comparison the results from [14] are also shown in Table I and Fig. 3. As expected, the tiling procedure realizes capacities which are significantly smaller than the general result. Yet the dependence $\alpha_c(k)$ is superlinear as well: $\alpha_c>2k$ holds in both cases. An assembly of k perceptrons in a parity machine can store more information than the same number of independent perceptrons.

The interesting property in this context is the asymptotic behavior of α_c for large numbers of hidden units ($k\to\infty$ but $k/N=0$). In [14] $\alpha_c(k)\sim k\ln k$ is found in leading order, which coincides with a theoretical upper bound [13].

The results of the estimate for the tiling procedure indicate that the asymptotic capacity also is $\alpha_c\sim k\ln k$; see Fig. 3. Thus the asymptotic behavior of the network is not changed by our very special training algorithm.

V. SUMMARY AND OUTLOOK

We have presented an algorithm for the training of a feedforward multilayered neural network. The algorithm combines the strategy of the tiling algorithm with the in principle fixed structure of the parity machine. Only the size of one hidden layer is adapted to the complexity of the given problem. Thus the procedure is rather easy to realize and might be useful for practical classification problems. For a comparison of our algorithm with different learning schemes [15,16,17] numerical tests shall be done for special generalization tasks, such as, for example, the "two or more clumps" problem [15].

In order to investigate the performance of the algorithm, the storage of random patterns has been considered. The study of the analytically solvable case with two hidden units has revealed an interesting result: If Hopfield couplings are used for the first hidden unit, whereas the second set of weights is a perceptron of optimal stability such that the total output is correct, corre-

lations between the two sets of weights exactly vanish. It is an interesting question whether this is true in more general cases or not. It can be shown that in the replica-symmetric approach these correlations also vanish in the most general parity machine [30]. However, for the general model, replica symmetry must be broken in order to get feasible results for the capacity. Further studies of problems similar to that of Sec. III do not involve RSB and might help to clarify under what conditions these correlations can be neglected.

In Sec. IV we have shown that our learning algorithm does not limit the network's abilities too much. The storage capacity $\alpha_c(k)$ turns out to increase like $k \ln k$ with the number of hidden units in our scheme, as it does for the algorithm-independent capacity. Numerical tests shall be done in order to study the differences between the overlapping and the nonoverlapping architecture. The simple estimation of Sec. IV can be generalized easily with respect to the minimization of several cost functions of type (16). It will be interesting to compare the efficiencies, i.e, the numbers of hidden units needed for the different strategies.

Further numerical studies will aim at the generalization properties of a network trained by our tilinglike procedure in comparison with the general parity machine.

ACKNOWLEDGMENTS

The authors would like to thank D. Hansel, W. Kinzel, M. Mezard, J.-P. Nadal, and H. Schwarze for useful and stimulating discussions. This work has been supported by the Stiftung Volkswagenwerk and the Höchstleistungs-Rechenzentrum Jülich.

APPENDIX

We calculate $\langle \ln V \rangle_{\xi,S}$ from Eq. (9) using the replica trick and assuming replica symmetry.

Introducing replicas for the perceptron weights $J^{(2)}_{j,\alpha}$, $\alpha=1,\ldots,n$, the replicated volume reads, with the integral representations of δ and Θ functions,

$$V^n=\left[\int\prod_{j,\alpha}dJ^{(2)}_{j,\alpha}\right]\left[\frac{1}{2\pi}\int\int\prod_{\nu,\alpha}de^\nu_1 dg^\nu_\alpha\frac{1}{2\pi}\left[\int_0^\infty\int_{\kappa+T}^\infty+\int_{-\infty}^0\int_{-\infty}^{-\kappa+T}\right]\prod_{\nu,\alpha}dE^\nu_1 dG^\nu_\alpha\right]$$
$$\times\left[\frac{1}{2\pi}\int\int\prod_j db_j dJ^{(1)}_j\right]\exp\left[i\sum_j b_j\left[J^{(1)}_j-\frac{1}{\sqrt{N}}\xi^\nu_j S^\nu\right]\right]$$
$$\times\exp\left[i\sum_{\nu,\alpha}g^\nu_\alpha\left[\frac{1}{\sqrt{N}}\sum_j J^{(2)}_{j,\alpha}\xi^\nu_j-G^\nu_\alpha\right]\right]\exp\left[i\sum_\nu e^\nu_1\left[\frac{1}{\sqrt{N}}\sum_\nu J^{(1)}_j\xi^\nu_j S^\nu-E^\nu_1\right]\right]\prod_{\alpha=1}^n\delta\left[\sum_j (J^{(2)}_{j,\alpha})^2-N\right]. \tag{A1}$$

After averaging over the inputs ξ^ν_j we introduce the following order parameters:

$$H=\frac{1}{N}\sum_j (J^{(1)}_j)^2\,,\quad P=\frac{1}{N}\sum_j J^{(1)}_j b_j$$

for the first set of couplings,

$$q_{\alpha\beta}=\frac{1}{N}\sum_j J^{(2)}_{j,\alpha}J^{(2)}_{j,\beta}\quad(\alpha<\beta)\quad Q_\alpha=\frac{1}{N}\sum_j (J^{(2)}_{j,\alpha})^2=1$$

for the perceptron,

$$R^\alpha=\frac{1}{N}\sum_j J^{(1)}_j J^{(2)}_{j,\alpha}\,,\quad Z^\alpha=\frac{1}{N}\sum_j b_j J^{(2)}_{j,\alpha}$$

connecting the two sets of weights. These constraints are imposed by δ functions which again are written as integrals over the corresponding conjugate variables $\hat{h},\hat{p},\hat{q}_{\alpha\beta}=if_{\alpha\beta},\hat{Q}_\alpha=iF_\alpha$, $\hat{r}_\alpha$ and $\hat{z}_\alpha$. A saddle-point integration is performed and we obtain in the replica-symmetric scheme the form

$$\frac{1}{N}\ln\langle V^n\rangle_{\xi,S}=\underset{\{H,\hat{h},\ldots\}}{\text{extremum}}[\alpha W_1(H,P,R,Z,q)+W_2(\alpha,\hat{h},\hat{p},\hat{r},\hat{z},f,F)]+[-(\hat{h}H+\hat{p}P)-n(\hat{r}R+\hat{z}Z+F-fq/2)]\,. \tag{A2}$$

The entropylike contribution $\exp(W_2)$ of the integrals over the b_j, $J^{(1)}_j$, and $J^{(2)}_{j,\alpha}$ is given by

$$\exp(W_2)=\left[\frac{2\pi}{f-2F}\right]^{n/2}[(1-i\hat{p})^2-2\alpha\hat{h}]^{-1/2}\exp\left[\frac{1}{2}n\frac{\tilde{W}}{f-2F}\right],\quad \tilde{W}=f+\frac{\alpha\hat{r}^2-2\hat{h}\hat{z}^2+2(i+\hat{p})\hat{z}\hat{r}}{(1-i\hat{p})^2-2\alpha\hat{h}}\,. \tag{A3}$$

For $n=0$ we recover at the saddle point

$$1=[(1-i\hat{p})^2-2\alpha\hat{h}]^{-1/2}\ \text{ for }\hat{p}=\hat{h}=0\,,\ P=i\,,\ H=\alpha\,. \tag{A4}$$

Hence we are left with

$$\exp[W_2(f,F)]=\left[\frac{2\pi}{f-2F}\right]^{n/2}\exp\left[\frac{1}{2}n\frac{f+\alpha\hat{r}^2+2i\hat{z}\hat{r}}{f-2F}\right]. \tag{A5}$$

F, f, $\hat{r}$, and $\hat{z}$ can be eliminated by setting the corresponding derivatives of (A2) to zero. This yields

$$W_2=n\left[\frac{1}{2}\ln(1-q)+\frac{1}{2}\frac{q}{1-q}+\frac{1}{2}\alpha Z^2\frac{1}{1-q}+iRZ\frac{1}{1-q}+\text{const}\right]. \tag{A6}$$

After performing the integrals over e_1^γ, g_α^γ, and G_α^γ we obtain, in the limit $n\to 0$,

$$W_1(\alpha,R,Z,q)=n\frac{1}{2}\sum_{S=\pm1}\int Dx\left\{\frac{1}{\sqrt{2\pi\alpha}}\int_0^\infty dE_1\exp\left[-\frac{1}{2\alpha}(E_1-1)^2\right]\ln\left[\Phi\left[\frac{-\kappa-T+\sqrt{q}\tilde{x}}{\sqrt{1-q}}\right]\right]\right.$$
$$\left.+\frac{1}{\sqrt{2\pi\alpha}}\int_{-\infty}^0 dE_1\exp\left[-\frac{1}{2\alpha}(E_1-1)^2\right]\ln\left[\Phi\left[\frac{-\kappa+T-\sqrt{q}\tilde{x}}{\sqrt{1-q}}\right]\right]\right\},$$
$$\tilde{x}=\left[-\left[q+\frac{R^2}{\alpha}\right]^{1/2}x-\frac{RS}{\alpha}(E_1^\gamma-1)+iZS\right]\Big/\sqrt{q}\ . \tag{A7}$$

Now it can be shown that ($R=Z=0$) satisfies

$$\frac{\partial W_1}{\partial R}=\frac{\partial W_2}{\partial R}=0\ ,\quad \frac{\partial W_1}{\partial Z}=\frac{\partial W_2}{\partial Z}=0$$

and thus this point is a solution of the problem. Inserting this, the E_1 integrations in (A7) can be easily performed and we finally get

$$\frac{1}{N}\langle\ln V\rangle_{\xi,S}=\operatorname*{extremum}_q\alpha\left\{\Phi\left[\frac{1}{\sqrt{\alpha}}\right]\int Dx\ln\left[\Phi\left[\frac{-\kappa-T+\sqrt{q}x}{\sqrt{1-q}}\right]\right]+\Phi\left[-\frac{1}{\sqrt{\alpha}}\right]\int Dx\ln\left[\Phi\left[\frac{\kappa-T-\sqrt{q}x}{\sqrt{1-q}}\right]\right]\right\}. \tag{A8}$$

The calculation proceeds completely analogously to [4] now and yields in the limit $q\to 1$ the saddle-point equation (13).

[1] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing* (Bradford, Cambridge, MA, 1986).
[2] F. Rosenblatt, *Principles of Neurodynamics* (Spartan, New York, 1962).
[3] M. Minsky and S. Papert, *Perceptrons* (MIT Press, Cambridge, MA 1988).
[4] E. Gardner, J. Phys. A **21**, 257 (1988).
[5] W. Kinzel, in *Statistical Mechanics of Neural Networks,* edited by L. Garrido (Springer, Berlin, 1990); W. Kinzel and Manfred Opper, in *Physics of Neural Networks,* edited by J. L. van Hemmen, E. Domany, and K. Schluten (Springer, Berlin, 1991).
[6] R. O. Winder, IRE Trans. Electron. Comput. **EC12**, 561 (1963).
[7] T. M. Cover, IEEE Trans. Electromagn. Compat. **EC14**, 326 (1965).
[8] W. Krauth and M. Mezard, J. Phys. A **20**, L745 (1987).
[9] J. K. Anlauf and M. Biehl, Europhys. Lett. **10**, 687 (1989); M. Biehl, J. K. Anlauf, and W. Kinzel, in *Neurodynamics,* edited by F. Pasemann (World Scientific, Singapore, in press).
[10] J. S. Denker *et al.*, Complex Syst. **1**, 877 (1987).
[11] R. Hecht-Nielsen, in *IEEE First International Conference on Neural Networks,* edited by M. Caudill and C. Butler (IEEE, New York, 1987), Vol. II.
[12] V. V. Anshelevich *et al.*, Biol. Cybern. **61**, 25 (1989).
[13] G. J. Mitchison and R. M. Durbin, Biol. Cybern. **60**, 345 (1989).
[14] E. Barkai, D. Hansel, and I. Kanter, Phys. Rev. Lett. **65**, 2312 (1990).
[15] M. Mezard and J. -P. Nadal, J. Phys. A **22**, 2191 (1989).
[16] J.-P. Nadal, Int. J. Neural Syst. **1**, 55 (1989).
[17] M. Frean, Neural Comput. **2**, 198 (1990).
[18] J. A. Sirat and J. -P. Nadal, Network **1**, 423 (1990).
[19] P. Rujan and M. Marchand, Complex Syst. **3**, 229 (1989); P. Rujan, in *Statistical Mechanics of Neural Networks,* edited by L. Garrido (Springer, Berlin, 1990).
[20] J. J. Hopfield, Proc. Natl. Acad. Sci. U.S.A. **79**, 2554 (1982).
[21] W. Krauth, M. Mezard, and J. -P. Nadal, Complex Syst. **2**, 387 (1988).
[22] E. Gardner and B. Derrida, J. Phys. A **21**, 271 (1988).
[23] G. Widrow and M. E. Hoff, in *WESCON Convention Record 4* (IRE, New York, 1960).
[24] S. Diederich and M. Opper, Phys. Rev. Lett. **58**, 949 (1987).
[25] J. K. Anlauf an M. Biehl, in *Parallel Processing in Neural Systems and Computers,* edited by R. Eckmiller, G. Hartmann, and G. Hauske (Elsevier, North-Holland, Amsterdam, 1990).
[26] S. I. Gallant, in *Eighth International Conference on Pattern Recognition* (IEEE, New York, 1986).
[27] Note that for nonzero $\langle\xi_j^\gamma\rangle=m_{\text{in}}$ T has to be replaced by an order paramter $u=T-(m_{\text{in}}/\sqrt{N})\sum_j J_j$ and (19) becomes the corresponding saddle-point equation.
[28] In [14] the sets are of size N/k originally but still of $O(N)$.
[29] D. Hansel (private communication).
[30] M. Mezard and S. Partanello (unpublished).