JOURNAL OF COMPUTER AND SYSTEM SCIENCES 48, 347–356 (1994)

On the Perceptron Learning Algorithm on Data with High Precision

KAI-YEUNG SIU*

Department of Electrical & Computer Engineering, University of California, Irvine, California 92717

AMIR DEMBO

Department of Mathematics, Stanford University, Stanford, California 94305

AND

THOMAS KAILATH

Information Systems Laboratory, Stanford University, Stanford, California 94305

Received October 1991

We investigate the convergence rate of the perceptron algorithm when the patterns are given with high precision. In particular, using the result of A. Dembo (*Quart. Appl. Math.* **47** (1989), 185–195), we show that when the n pattern vectors are independent and uniformly distributed over $\{+1, -1\}^{n \log n}$, as $n \to \infty$, with high probability, the patterns can be classified into all 2^n possible ways using perceptron algorithm with $O(n \log n)$ iteration. Further, the storage of parameters requires only $O(n \log^2 n)$ bits. We also indicate some interesting mathematical connections with the theory of random matrices.

1. INTRODUCTION

It is well known that the perceptron algorithm can be used to find the appropriate parameters in a linear threshold device for pattern classification, provided that the pattern vectors are linearly separable. Since the number of parameters in a perceptron is significantly fewer than that needed to store the whole data set, it is tempting to conclude that when the patterns are linearly separable, the perceptron can achieve a reduction in storage complexity. However, Minsky

* This work was supported in part by the Joint Services Program at Stanford University (U.S. Army, U.S. Navy, U.S. Air Force) under Contract DAAL03-88-C-0011, and NASA Headquarters, Center for Aeronautics and Space Information Sciences (CASIS), under Grant NAGW-419-S5.

0022-0000/94 $6.00

and Papert [9] have shown an example in which both the learning time and the magnitude of the parameters increase exponentially, so the use of the perceptron would result in storage larger than that required by a table consisting of the whole list of patterns.

Ways around such examples can be explored by noting that analysis that assumes real arithmetic and disregards finite precision aspects might yield misleading results. For example, we present below a simple system with one real-valued weight that can simulate all possible classifications of n real-valued patterns into k classes, when unlimited accuracy and continuous distribution of the patterns are assumed. For simplicity, let us assume that the patterns are real numbers in $[0, 1]$. Consider the following sequence $\{x_{i,j}\}$ generated by each pattern x_i for $i = 1, ..., n$,

$$x_{i,1} = k \cdot x_i \bmod k$$

$$x_{i,j} = k \cdot x_{i,j-1} \bmod k \qquad \text{for} \quad j > 1$$

$$\sigma(x_i, j) = [x_{i,j}],$$

where $[\ ]$ denotes the integer part.

Let $f: \{x_1, ..., x_n\} \to \{0, ..., k-1\}$ denote the desired classification of the patterns. It is easy to see that when $x_1, ..., x_n$ are n random variables independent identically distributed according to any continuous distribution on $[0, 1]$, there exists with probability 1 an integer j such that $\sigma(x_i, j) = f(x_i)$. So, the network $y = \sigma(x, w)$ may simulate any classification with $w = j$ determined from the desired classification as shown above.

Therefore, we emphasize here the finite precision computational aspects of pattern classification problems and provide partial answers to the following questions:

- How many bits of information are required to store the parameters in a perceptron for "typical" linearly separable patterns?
- Does the "learning" time of a perceptron become too long to be practical most of the time even when the patterns are assumed to be linearly separable?
- How are the convergence results in comparison with those obtained by solving a system of linear inequalities via a linear programming approach?

Although we do not assume that the patterns are given with infinite precision, we still consider the data to be given with high precision. We attempt to answer the above questions by using a probabilistic approach. A preliminary version of this paper appeared in [11].

In the following analysis, the phrase "with high probability" means the probability of the underlying event goes to 1 as the number of patterns goes to infinity.

2. The Perceptron

A *perceptron* is a linear threshold device that computes a weighted sum of the coordinates of the pattern vector, compares the value with a threshold, and outputs $+1$ (or -1) if the value is larger (or smaller) than the threshold. More formally, a perceptron computes a function

$$\operatorname{sign}\{\langle \mathbf{w}, \mathbf{x} \rangle - \theta\} = \operatorname{sign}\left\{\sum_{i=1}^{d} x_i \cdot w_i - \theta\right\},$$

where $\mathbf{x} = (x_1, ..., x_d) \in R^d$ is the input pattern vector, $\mathbf{w} = (w_1, ..., w_d) \in R^d$ is the *weight* vector, $\theta \in R$ is the threshold, and $\operatorname{sign}(y) = 1$ if $y \geqslant 0$ and -1 otherwise.

Given m patterns $\mathbf{x}_1, ..., \mathbf{x}_m$ in R^d, there are 2^m possible ways of classifying each of the patterns to ± 1. Each such classification is called a *dichotomy* of the patterns. When there exist some $\mathbf{w}$ and θ such that the perceptron achieves the desired classification with no errors, the patterns are said to be *linearly separable.* Geometrically, this happens when there is a *separating hyperplane* such that patterns that are classified into $+1$ lie on the side of the hyperplane opposite to those patterns classified into -1. Rosenblatt [10] showed that if the patterns are linearly separable, then there is a "learning" algorithm which he called the *perceptron learning algorithm* that finds the appropriate parameters $\mathbf{w}$ and θ. We can always append each pattern vector $\mathbf{x}_i$ an extra coordinate with value $= -1$ and consider the threshold θ as part of the weight vector $\mathbf{w}$. We assume that this has been done before running the perceptron algorithm. Let $\sigma_i = \pm 1$ be the desired classification of the pattern $\mathbf{x}_i$. Also, let $\mathbf{y}_i = \sigma_i \cdot \mathbf{x}_i$. We would like to find a separating weight vector $\mathbf{w}$ such that $\mathbf{w} \cdot \mathbf{y}_i > 0$ for $i = 1, ..., n$. The perceptron learning algorithm runs as follows:

1. Set $k = 1$, Choose $\mathbf{w}(k) = \mathbf{y}_i$ for some $i \in \{1, ..., n\}$. (In fact, any initial value of $\mathbf{w}(k)$ will work.)
2. Select an $i \in \{1, ..., n\}$ set $\mathbf{y}(k) = \mathbf{y}_i$.
3. If $\mathbf{w}(k) \cdot \mathbf{y}(k) > 0$, goto 2. Else
4. Set $\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{y}(k)$, $k \to k+1$, goto 2.

The algorithm terminates when step 3 is true for all $\mathbf{y}_i$, and $\mathbf{w}(k)$ is then the desired separating vector $\mathbf{w}$. Rosenblatt [10] showed that if the patterns are linearly separable, then the above perceptron algorithm is guaranteed to converge in finitely many iterations; i.e., step 4 would be reached only finitely often, provided that every pattern appears infinitely often in the training sequence.

3. New Look at the Perceptron

Cover [2] showed that the number of binary classifications of n pattern vectors in R^d that can be achieved by a perceptron is very small compared with the 2^n possibilities, when $d \ll n$. Instead of considering the patterns to be real pattern

vectors, let us consider the binary expansion of each coordinate and view the real pattern vector as a binary vector, but in a much higher dimensional space. The intuition behind this is that we are now making use of every bit of information in the pattern. Let us assume that each pattern vector has dimension d and that each coordinate is given with m bits of accuracy, which grows with the number of patterns n in such a way that $d \cdot m = n \log n$. By considering the binary expansion, we can treat the patterns as binary vectors; i.e., each vector belongs to $\{+1, -1\}^{dm}$. If we want to classify the patterns into k classes, we can use $\log k$ number of binary classifiers, each classifying the patterns into the corresponding bit of the binary encoding of the k classes. So without loss of generality, we assume that the number of classes equals 2.

To see how we can upper bound the running time of the algorithm, we first state the general result due to Rosenblatt.

THEOREM 1 (Rosenblatt [10]). *Suppose the n pattern vectors* $\mathbf{y}_i$, $i = 1, \ldots, ,$ *are linearly separable and let* $\mathbf{w}^*$ *be a unit vector such that* $\mathbf{w}^* \cdot \mathbf{y}_i > \delta$ *for some* $\delta > 0$. *If the input pattern vectors are presented in a cyclic order to the perceptron, the number of iterations of the algorithm, m, is bounded as follows*:

$$m < M^2/\delta^2,$$

where

$$M = \sqrt{(1/n) \sum_{j=1}^{n} \|\mathbf{y}(j)\|^2}.$$

In other words, the upper bound on the running time of the perceptron depends on the average length squared M^2 of the input pattern vectors, and the minimum distance δ of the pattern vectors to a separating hyperplane. In the following analysis, our probabilistic assumption guarantees the pattern vectors to be linearly independent with high probability. If we put the pattern vectors as columns in a matrix, say A, then linear independence of the pattern vectors implies that there exists a weight vector $\mathbf{w}$ such that

$$\mathbf{w}^{\mathrm{T}} A = [\pm 1 \cdots \pm 1]$$

for any choice of the $\pm$ sign in each coordinate of the vector on the right-hand side above. Therefore, given any desired dichotomy of the pattern vectors, the vector $\mathbf{w}$ can serve as the weight vector in the perceptron. Thus our probabilistic assumption also guarantees the pattern vectors to be linearly separable with high probability.

We now provide some partial answers to the questions raised at the beginning:

THEOREM 2. *Assume that the n pattern vectors are independent and uniformly distributed over* $\{+1, -1\}^N$, *where* $N = n \log n$. *Then as* $n \to \infty$, *with high probability, the patterns can be classified into all* 2^n *possible ways using the perceptron algorithm.*

Further, the number of iterations is $O(n \log n)$ if the patterns are cyclically presented, and the storage of parameters requires only $O(n \log^2 n)$ bits.

Proof. Since the patterns are binary vectors, in every iteration of the algorithm, each of the "weights" in the perceptron increases or decreases by 1. If the number of iterations is $O(n \log n)$ (in fact, any polynomial in n), then it takes at most $O(\log n)$ bits to specify each of the weights. Because there are only $N = n \log n$ weights, at most $O(n \log^2 n)$ bits are needed to specify all the weights. By Theorem 1, it suffices to show that the minimum distance of the pattern vectors to a separating hyperplane is bounded below by a constant $\delta > 0$, with high probability. If we put the pattern vectors as columns in a matrix, say A_n, then A_n is a $N \times n$ matrix of i.i.d. entries.

Without loss of generality, we can assume the patterns are all classified into $+1$. With high probability, A_n is of full rank and thus $A_n^{\mathrm{T}} A_n$ is invertible. Now let

$$\mathbf{v} = A_n (A_n^{\mathrm{T}} A_n)^{-1} \mathbf{1} \qquad \text{and} \qquad \mathbf{w}^* = \mathbf{v}/\|\mathbf{v}\|,$$

where $\mathbf{1}$ is the n-dimensional vector with all coordinates equal to $+1$. Therefore,

$$A_n^{\mathrm{T}} \cdot \mathbf{w}^* = \frac{\mathbf{1}}{\|\mathbf{v}\|} \geqslant \delta \mathbf{1} \qquad \text{iff} \quad \|\mathbf{v}\|^2 \leqslant \frac{1}{\delta^2}.$$

Now $\|\mathbf{v}\|^2 = \mathbf{1}^{\mathrm{T}} (A_n^{\mathrm{T}} A_n)^{-1} \mathbf{1}$. We show that

$$\Pr\left\{ \mathbf{1}^{\mathrm{T}} (A_n^{\mathrm{T}} A_n)^{-1} \mathbf{1} > \frac{1}{\delta^2} \right\} \to 0 \qquad \text{as} \quad n \to \infty \text{ for some constant } \delta > 0.$$

Let $|A_n^{\mathrm{T}} A_n|$ and $\operatorname{adj}(A_n^{\mathrm{T}} A_n)$ denote the determinant and the adjoint of $A_n^{\mathrm{T}} A_n$, respectively. Also let

$$X_\delta = \delta^2 \mathbf{1}^{\mathrm{T}} \operatorname{adj}(A_n^{\mathrm{T}} A_n) \mathbf{1} - |A_n^{\mathrm{T}} A_n|.$$

Then

$$\Pr\left\{ \mathbf{1}^{\mathrm{T}} (A_n^{\mathrm{T}} A_n)^{-1} \mathbf{1} > \frac{1}{\delta^2} \right\} = \Pr\{X_\delta > 0\} \leqslant \frac{\operatorname{var}(X_\delta)}{E(X_\delta)^2} = \frac{E(X_\delta^2)}{E(X_\delta)^2} - 1 \tag{1}$$

by Chebyshev's inequality, provided $E(X_\delta) < 0$. Note that $n/N \to 0$. It follows from the results in [3] that

$$E(X_\delta) = \frac{N!}{(N-n)!} \left(\delta^2 \frac{n}{N-n+1} - 1 \right) < 0 \qquad \text{as} \quad n \to \infty$$

and

$$\begin{aligned} E(X_\delta)^2 = {} & h(n) - 2\delta^2 n (N-n+1)\, h(n-1) + \delta^4 n h(n-1) \\ & + \delta^4 n (n-1)(N^2 - (n-2)^2)\, h(n-2), \end{aligned}$$

where

$$h(n) = E(|A_n^T A_n|^2) = \frac{N!}{(N-n)!} \sum_{j=0}^{n} \binom{n}{j} (-2)^j \frac{(N+2-j)!}{(N+2-n)!}.$$

It can be shown that

$$\frac{h(n)}{\left(\frac{N!}{(N-n)!}\right)^2} \to 1 \quad \text{as } n \to \infty \qquad \text{and} \qquad \frac{h(n-1)}{h(n)} = O\left(\frac{1}{N^2}\right);$$

thus

$$\frac{E(X_\delta^2)}{E(X_\delta)^2} \to 1.$$

Since the proof in [3] is very cumbersome, we do not repeat the calculations here. Therefore, we have the right-hand side in (1) above $\to 0$ as $n \to \infty$. ∎

Remark 1. In fact, the first part of Theorem 2 easily follows from the result in [8] and was proved by Füredi [5] in a much stronger form. He showed that for randomly chosen vertices in N-space, $2N$ is a threshold function for the attribute that a randomly specified dichotomy of n points is linearly separable. Our contribution is to show that the perceptron algorithm converges rapidly when the assumption in the theorem is satisfied.

Now let us consider pattern vectors where each coordinate of the vectors is drawn from some distributions other than the $\{1, -1\}$ distribution. In particular, we consider a wide class of distributions on the pattern vector $\mathbf{x}$ that satisfy the following:

MPD Condition. There exists a positive constant K such that $\Pr\{|\mathbf{v}^T\mathbf{x}| \leqslant \varepsilon\} \leqslant K\varepsilon$ for any unit vector $\mathbf{v}$ and positive constant ε.

LEMMA 1. *Let n be the number of pattern vectors each in R^m, where $m = \alpha n$ and α is any constant > 1. Assume that the pattern vectors have entries that are non-degenerate i.i.d. random variables with zero mean and bounded fourth moment and that they satisfy the MPD condition. Then with probability $\to 1$ as $n \to \infty$, there exists a separating hyperplane and a constant $\delta^* > 0$ such that each pattern vector is at a distance of at least δ^* from it.*

The proof of the above lemma makes use of a result from the theory of random matrices [4] which gives a lower bound on the minimum singular value of a random matrix. The following result states that the minimum singular value of a αn by n random matrix with $\alpha > 1$, grows as $\sqrt{n}$ almost surely (see [4] for a proof).

PROPOSITION 1. *Let A_n be a $\alpha n \times n$ random matrix with $\alpha > 1$, whose entries are nondegenerate i.i.d. entries with zero mean and bounded fourth moment, and the columns of the matrix satisfy the MPD condition. Let $\sigma(\cdot)$ denote the minimum singular value of a matrix. Then there exists a $\gamma > 0$ such that*

$$\liminf_{n \to \infty} \sigma\left(\frac{A_n}{\sqrt{n}}\right) > \gamma$$

with probability 1.

Proof of Lemma 1. Let $\mathbf{P} = [\pm 1 \cdots \pm 1]$ be an n-dimensional vector such that the ith coordinate of $\mathbf{P}$ equals the desired binary classification of the ith pattern vector. We have n pattern vectors, each of which has αn coordinates. So if we put the pattern vectors as columns in a matrix form, we obtain a matrix A_n as in Proposition 1. With high probability, A_n will be of full rank. Thus there exists a vector $\mathbf{w} \in R^{\alpha n}$ such that

$$A_n^{\mathrm{T}} \cdot \mathbf{w} = \mathbf{P}.$$

Consider the *singular value decomposition* of A_n^{T} [6],

$$A_n^{\mathrm{T}} = \sigma_1 \mathbf{v}_1 \mathbf{u}_1^{\mathrm{T}} + \sigma_2 \mathbf{v}_2 \mathbf{u}_2^{\mathrm{T}} + \cdots + \sigma_n \mathbf{v}_n \mathbf{u}_n^{\mathrm{T}},$$

where σ_i's are the non-zero singular values of A_n and $\mathbf{u}_i$'s are orthonormal vectors in $R^{\alpha n}$. We can write $\mathbf{w}$ as

$$\mathbf{w} = \lambda_1 \mathbf{u}_1 + \lambda_2 \mathbf{u}_2 + \cdots + \lambda_{\alpha n} \mathbf{u}_{\alpha n},$$

where now $\{\mathbf{u}_1, \ldots, \mathbf{u}_{\alpha n}\}$ forms an orthonormal basis in $R^{\alpha n}$. Observe that

$$A_n^{\mathrm{T}} \cdot \mathbf{w} = \lambda_1 \sigma_1 \mathbf{v}_1 + \lambda_2 \sigma_2 \mathbf{v}_2 + \cdots + \lambda_n \sigma_n \mathbf{v}_n = \mathbf{P}. \tag{2}$$

Take γ as in Proposition 1 and let

$$\mathbf{w}^* = \gamma(\lambda_1 \mathbf{u}_1 + \lambda_2 \mathbf{u}_2 + \cdots + \lambda_n \mathbf{u}_n)$$

$$\Rightarrow A_n^{\mathrm{T}} \cdot \mathbf{w}^* = \gamma \mathbf{P}. \tag{3}$$

Note that $\|\mathbf{w}^*\| = \gamma \cdot \sqrt{\lambda_1^2 + \lambda_2^2 + \cdots + \lambda_n^2}$. By Proposition 1, we have with high probability that $\sigma_i > \gamma \sqrt{n}$ for $i = 1, \ldots, n$. Therefore, by considering the Euclidean norm on both sides of Eq. (2), we obtain

$$\sqrt{\lambda_1^2 + \lambda_2^2 + \cdots + \lambda_n^2} \cdot \gamma \sqrt{n} < \sqrt{n}$$

$$\Rightarrow \|\mathbf{w}^*\| < 1.$$

It follows from (3) with high probability that there exists a separating hyperplane such that every pattern vector is at a distance at least $\delta^* = \gamma$ from it. Note that the minimum distance δ^* from the pattern vectors to the separating hyperplane holds uniformly for all binary classifications of the patterns. ∎

Examples that satisfy the MPD condition include the Gaussian distribution and the Uniform distribution on a bounded interval $[-a, a]$. However, the MPD condition is not satisfied in the case of a ± 1 binary random vector. In general, when the dimension of the pattern vectors is larger than and increases linearly with the number of patterns, Lemma 1 applies, provided that the patterns are given with high enough precision that the underlying distribution is a sufficiently good model for analysis. The above proposition without the MPD condition was a famous conjecture in the theory of random matrices (see [4]). We note here that after we have obtained the results in this paper, this conjecture was recently proved and shown in [12]. Thus Theorem 3 (to be proved later) would also apply to binary random vectors, yielding a stronger result than that stated in Theorem 2.

Under our probabilistic assumptions, the entries of the pattern vectors $\mathbf{x}_i$ are i.i.d. random variables, $E(\|\mathbf{x}_i\|^2) = nm_2$, where m_2 is the second moment of each entry. A trivial application of the Law of Large Numbers yields the following:

LEMMA 2. *Assuming the same conditions on the pattern vectors as in Lemma* 1, *then there exists a constant* $c > 0$ *such that* $\Pr\{M^2 > cn\} \to 0$ *as* $n \to \infty$.

In other words, the average length-squared of the pattern vectors will be bounded by $O(n)$ with high probability. Note that Lemma 2 holds without the additional assumption of the MPD condition.

THEOREM 3. *Suppose that the patterns satisfy the probabilistic assumptions stated in Lemma* 1; *then when the patterns are presented in a cyclic fashion, with high probability, the perceptron takes* $O(n^2)$ *arithmetic operations to terminate.*

Proof. Recall that number of iterations of the perceptron learning algorithm is at most M^2/δ^2, where M^2 is the average length-squared of the pattern vectors and δ is the minimum distance of the pattern vectors to a separating hyperplane. Now by Lemma 1, with high probability, δ is a constant independent of n. There exists a constant $c > 0$ such that $M^2 \leqslant cn$ with high probability by Lemma 2. It follows that the number of iterations is bounded by $O(n)$. Since it takes $O(n)$ arithmetic operations in each iteration, the total number of operations will be bounded by $O(n^2)$, with high probability. ∎

Remark 2. We would like to point out a related result in [1]. It was shown that the perceptron algorithm can learn an arbitrary half-space in time $O(n^2/\varepsilon^3)$ if the n patterns are taken uniformly over the unit sphere in R^n, within the context of Valiant's protocol for learning. However, this result in [1] still allows the

perceptron to misclassify a pattern with a probability of error $\leqslant \varepsilon$, and the algorithm might not terminate with a probability smaller than a fixed constant $\delta > 0$ which is hidden under the $O(\cdot)$ notation.

Another way of finding a separating hyperplane is to solve a system of linear inequalities via a linear programming approach, which requires $O(n^{3.5})$ arithmetic operations [7]. Under our probabilistic assumptions, the patterns are linearly independent with high probability, so that we can actually solve a system of linear equations. However, this still requires $O(n^3)$ arithmetic operations. Further, these methods require batch processing in the sense that typically all patterns must be stored in advance in order to find the desired parameters, in contrast to the sequential "learning" nature of the perceptron algorithm. So for training this pattern classifier, the perceptron algorithm seems to be preferable.

4. Conclusion

In this paper, we show that the perceptron, in contrast to common belief, can be quite efficient as a pattern classifier, provided that the patterns are given with high enough precision. Using a probabilistic approach, we show that the perceptron algorithm can even out-perform linear programming under certain conditions. During the course of this work, we also discovered some interesting connections with the theory of random matrices.

Our results on the efficiency of the perceptron algorithm depend on the assumption that the patterns are given with high enough precision, comparable (in the number of bits) to the total number of patterns. Suppose that the number of patterns is polynomial in the total number of bits representing each pattern; say that we have n^c patterns with $c > 1$ and that each pattern is represented by n bits. We may first extend each vector $\mathbf{x} = [x_1, ..., x_n]$ to another vector $\tilde{\mathbf{x}}$ with dimension αn^c, where $\alpha > 1$. Then assuming Theorem 3 still holds for this new set of pattern vectors, we can apply the perceptron algorithm to compress the storage of parameters. This would require $O(n^c \log n)$ bits of storage in the perceptron in comparison with the complete storage of $O(n^{c+1})$ bits of the patterns. One way of adding these extra bits is to form products of the coordinates within each pattern. For example, $\tilde{\mathbf{x}} = [x_1, ..., x_n, x_1x_2, ..., x_{n-1}x_n]$ would have dimension $O(n^2)$. Note that by doing so, the coordinates of each pattern are pairwise independent. We conjecture that Theorem 3 still applies, implying even more reduction in the storage requirements. Simulation results strongly support our conjecture.

References

1. E. Baum, The perceptron algorithm is fast for non-malicious distributions, *in* "Advances in Neural Information Processing Systems 2" (S. Touretzky, Ed.), pp. 676–685, Morgan Kaufmann, San Mateo, CA, 1990.

2. T. M. COVER, Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition, *IEEE Trans. Electron. Comput.* **EC-14** (1965), 326–334.
3. A. DEMBO, On random determinants, *Quart. Appl. Math.* **47** (1989), 185–195.
4. J. COHEN, H. KESTEN, AND C. NEWMAN (Eds.), "Random Matrices and Their Applications," Contemp. Math., Vol. 50, Amer. Math. Soc., Providence, RI, 1986.
5. Z. FÜREDI, Random polytopes in the d-dimensional cube, *Discrete Comput. Geom.* **1** (1986), 315–319.
6. G. GOLUB AND C. VAN LOAN, "Matrix Computations," The Johns Hopkins Univ. Press, Baltimore, MD, 1983.
7. N. KARMARKAR, A new polynomial-time algorithm for linear programming, *Combinatorica* **4** (1984), 373–395.
8. KOMLÓS, On the determinant of (0, 1) matrices, *Studia Sci. Math. Hungar.* **2** (1967), 7–12.
9. M. MINSKY AND S. PAPERT, "Perceptrons," expanded edition, MIT Press, Cambridge, MA, 1988.
10. F. ROSENBLATT, "Principles of Neurodynamics," Spartan Books, New York, 1962.
11. A. DEMBO, K.-Y. SIU, AND T. KAILATH, Complexity of finite precision neural network classifier, *in* "Advances in Neural Information Processing Systems 2" (S. Touretzky, Ed.), pp. 668–675, Morgan Kaufmann, San Mateo, CA, 1990.
12. Z. D. BAI AND Y. Q. YIN, Limit of the smallest eigenvalue of a large dimensional sample covariance matrix, *Ann. Probab.* **21**, No. 3. (1993), 1275–1294.