Bayesian Adaptation of Hidden Layers in Boolean Feedforward Neural Networks

Wolfgang Utschick and Josef A. Nossek
Lehrstuhl für Netzwerktheorie und Schaltungstechnik
Technische Universität München
Email: wout@nws.e-technik.tu-muenchen.de

Abstract

In this paper a statistical point of view of feedforwared neural networks is presented. The hidden layer of a multilayer perceptron neural network is identified of representing the mapping of random vectors. Utilizing hard limiter activation functions, the second and all further layers of the multilayer perceptron, including the output layer, represent the mapping of a boolean function. Boolean type of neural networks are naturally appropriate for categorization of input data. Training is exclusively carried out on the first layer of the neural network, whereas the definition of the boolean function generally remains a matter of experience or due to considerations of symmetry. In this work a method is introduced, how to adapt the boolean function of the network, utilizing statistical knowledge of the internal representation of input data. Applied to the classification problem of greylevel bitmaps of handwritten characters the misclassification rate of the neural network is approximately reduced by 20%.

Keywords: *Multilayer Perceptron, Indicator Function, Discrete Probability Space, Internal Representation, Boolean Function, Bayesian Adaptation, Classification, Handwritten Characters.*

1. Introduction

Neural Networks consisting of at least two or more layers of neurons are called multilayer perceptrons [3, 19, 10, 8]. Figure 1 shows a multilayer perceptron consisting of L layers of neurons. Each neuron of a network layer performs a nonlinear mapping of the output values of the preceding layer. The nonlinear mapping of a neuron is given by a weighted sum $l : \mathcal{R}^n \rightarrow \mathcal{R},\ \mathbf{x} \mapsto \sum_{j=1}^{n} w_j x_j$ of it inputs and a nonlinear activation function a. Defining a constant input line $x_n \equiv -1$, the additional weight parameter w_n plays the important role of a threshold of the neuron. Applying a hard limiter activation function a to the local fields of a neuron, $p = l \circ a$:

$$\mathcal{R}^n \rightarrow \{0,1\}, \mathbf{x} \mapsto \begin{cases} 1 & , \text{ if } \sum_{j=1}^{n} w_j x_j \geq 0 \\ 0 & , \text{ else} \end{cases},$$

the neuron is called a perceptron [11]. The first layer of the neural network is called a perceptron layer. The neurons of the perceptron layer are called hidden neurons. Because of the binary outputs of the perceptrons, the second layer and all further layers of the architecture represent a mapping L2o L3o$\cdots$oL$L\ b : \{0,1\}^h \rightarrow \{0,1\}^M$, from the outputs of the first layer to the outputs of layer L OUT, whereas h is the number of hidden neurons and M is the number of outputs. The function b is obviously a boolean function [12]. Figure 1 displays a boolean type of neural network, based on a hard limiter activation function, consisting of a single layer perceptron and a second layer of any boolean function. A single layer perceptron performs a mapping of the input space onto an internal representation of the neural network. Hence, the first layer defines a partitioning and discretization of the input space. An element of a partition is called a cell and is defined by $x_{10\ldots1} = \{\mathbf{x} \in \mathcal{R}^n; p_1(\mathbf{x}) = 1, p_2(\mathbf{x}) = 0, \cdots, p_h(\mathbf{x}) = 1\}$. The maximum number of cells is given by 2^h. For the boolean function of the system the crucial property of the internal representation of input data $\mathbf{x}$ is its being an element of a particular cell $x_{p_1 p_2 \cdots p_h}$. Each cell is labeled by an assignment of the boolean function. The choice of the boolean function provides the network with a particular intersection of decision boundaries within the input space [9]. The training of the perceptron layer may be interpreted as a matching of the labeled partitioning of input space with the given data structure. Due to the binary properties of the boolean layer, there is no feasible gradient information and backpropagation like training algorithms [1] are not applicable. Appropriate training algorithms are generally based on perceptron learning algorithms and a particular strategy how to adapt the weights of the first layer in order to produce a correct input of the boolean layer for

1015-4651/96 $5.00 © 1996 IEEE
Proceedings of ICPR '96

its desired output [19, 20, 14, 13]. Training is exclusively carried out on the perceptron layer of the neural network, whereas the definition of the boolean function generally remains a matter of experience or due to considerations of symmetry. As far as known, during the training phase of all algorithms [19, 20, 14, 13] no explicit adaptation of the boolean function of the neural system is enabled. In this paper a method is presented how to adapt the boolean function of the neural network during the training phase.

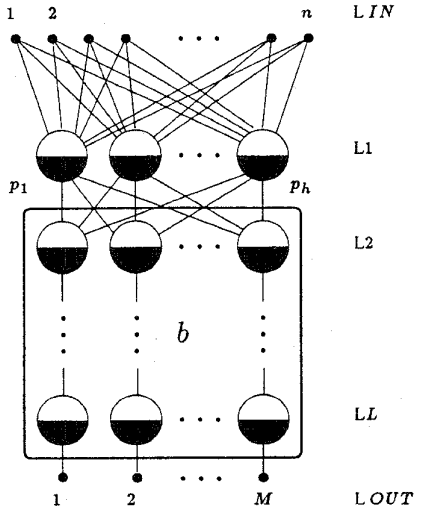


Figure 1. Utilizing hard limiter activation functions the architecture of a multilayer perceptron consisting of an input layer L IN, output layer L OUT and L hidden layers of neurons represents a boolean type of multilayer perceptron consisting of a single layer perceptron SLP = L1 and a boolean function b.

2. Random Network Variables

Following the notation of Anthony and Biggs in [2] it is supposed that in classification problems a probability space is given. A probability space is a tripel $(\mathcal{X}, \Sigma, P)$, where $\mathcal{X}$ is the space of the input pattern, Σ is a system of events called a σ-algebra in $\mathcal{X}$ and P is a probability measure. In probability theory $\mathcal{X}$ consists of all possible results or outcomes of an experiment. An element $\mathbf{x}$ of $\mathcal{X}$ is called an input pattern of the underlying classification problem and a subset of $\mathcal{X}$ is called an event. In pattern recognition $\mathcal{X}$ equals $\mathcal{R}^n$ and an adequate σ-algebra is given by the Borel algebra on $\mathcal{R}^n$. Given two measurable spaces $(\mathcal{X}, \Sigma)$ and $(\mathcal{X}', \Sigma')$, a mapping $l : \mathcal{X} \to \mathcal{X}'$ is called Σ-Σ' measurable if $l^{-1}(x') := \{\mathbf{x} \in \mathcal{X};\ l(\mathbf{x}) \in x'\} = x \in \Sigma$ for all $x' \in \Sigma'$. A measure P' for $(\mathcal{X}', \Sigma')$ is derived by a measure on l, defined by $P'(x') := P(x) = P\left(l^{-1}(x')\right)$ for all $x' \in \Sigma'$, then P' is called a distribution of l [4]. According to this stochastical point of view the mapping of a perceptron layer of a neural network may be interpreted as a random vector consisting of random variables, each of them corresponding to the mapping of a single perceptron.

3. Discretization of the Probability Space

A measurable mapping $i : \mathcal{X} \to \mathcal{R}$, where $\mathcal{X}$ is finite, is called indicator function if

$$i_x : \mathcal{X} \to \mathcal{R},\ \mathbf{x} \mapsto \begin{cases} 1 & , \text{ if } \mathbf{x} \in x \\ 0 & , \text{ else} \end{cases} .$$

A function i_x indicates, whether $\mathbf{x}$ is an element of a set $x \in \Sigma$. The performance of a single perceptron $p_i = l_i \circ a : \mathcal{X} \to \{0,1\}, \mathbf{x} \mapsto x^{p_i}$ is considered to be an indicator function i. Whereas often the geometrical properties and separability capabilities of perceptrons are emphasized [3, 19, 9, 13, 5, 6], this work is based on the statistical properties of the network architecture, i.e. actually on the discretization of the introduced probability space $(\mathcal{X}, \Sigma, P)$ by the perceptron layer [17, 18]. The discretization $(\mathcal{X}^d, P^d)$ is described by the introduction of a finite set $\mathcal{X}^d$, consisting of all sets $x_{p_1 p_2 \cdots p_h}$, element of Σ, generated by $x_{p_1 p_2 \cdots p_h} := \{\mathbf{x} \in \mathcal{X};\ p_1(\mathbf{x}) = x^{p_1},\ p_2(\mathbf{x}) = x^{p_2},\ \ldots,\ p_h(\mathbf{x}) = x^{p_h}\} \in \Sigma$ for all $x^{p_i} \in \{0,1\}, i \in \{1,2,\ldots,h\}$. For the subsequel it is important to point out here, that the events $x_{p_1 p_2 \cdots p_h}$, element of $\mathcal{X}^d$, correspond to elements $\mathbf{x}^p$ of the domain $\mathcal{X}^p = \{0,1\}^h$ of the boolean function $b : \{0,1\}^h \to \{0,1\}$, $\mathbf{x}^p \mapsto x^b$. Because of the geometrical properties of linear inequalities [5], the number of $|\mathcal{X}^p| = 2^h$ elements is only an upper bound of $|\mathcal{X}^d|$. By the set $\Sigma^d = \mathcal{P}(\mathcal{X}^d)$ of all subsets of $\mathcal{X}^d$ an algebra Σ^d on $\mathcal{X}^d$ is defined. The set of all possible mappings $\{0,1\}^h \to \{0,1\}$, representable by the boolean function b of the neural network $n = \mathbf{p} \circ b$: $\mathcal{X} \to \{0,1\}$, whereby $\mathbf{p}$ or $p_1 p_2 \cdots p_h$ is a short for $p_1 \times p_2 \times \cdots \times p_h$, addresses events of Σ^d, by $x^d := \{\mathbf{x} \in \mathcal{X};\ \mathbf{p} \circ b\,(\mathbf{x}) = x^b\} \in \Sigma^d$ for all b and the output variables of the neural network $x^b \in \{0,1\}$. Here, the output of the boolean function b is restricted to $M = 1$. Thus the perceptron layer of the neural network generates a particular σ-algebra $\Sigma^d \subset \Sigma$. A random variable $E^d[f \mid \Sigma^d]$ is called conditional expected value according to Σ^d of the random variable f [4], which is Σ-$\mathcal{R}$ measurable, if $E^d[f]$ is Σ^d-$\mathcal{R}$ measurable and satisfies $\int_{x^d} E^d[f \mid \Sigma^d]\, dP = \int_{x^d} f\, dP$, for all $x^d \in \Sigma^d$. If f is given by the nonlinear mapping of the network $\mathbf{p} \circ b$ or its complement, the conditional expected value $E^d[\mathbf{p} \circ b \mid \Sigma^d]$ coincides with the conditional probability $P[\mathbf{x}^p \mid x^d \in \Sigma^d]$ or $P^d[\mathbf{x}^p]$, because of the nonlinear activation function of the random variables p_i. Actually the perceptron layer $\mathbf{p}$ of the neural network forms a distribution P^d according to the discrete nature of the probability space. A distribution of the random variables

of the neural network is fundamentally important for any statistical reasoning within the hidden layer of a boolean type neural network.

4. Bayesian Adaptation of the Boolean Function

Analyzing disjoint categories of input data, two or more random experiments have to be considered, providing different classes of random results. This work is restricted to two–class problems. The probabilistic model for the discrimination of two differently categorized types of input data is given by a measurable space $(\mathcal{X}, \Sigma)$ and two conditional probability measures P_+ and P_- according to the sets of the positive and negative sample points, X_+ and X_-, within the training set. At any time, including $t = 0$ before the start of the network training, during the training phase, two different distributions $P_+^d[\mathbf{x}^p]$ and $P_-^d[\mathbf{x}^p]$ of the random network vector $\mathbf{x}^p$ of the perceptron layer of the neural network are measurable, with the discrete probability space $(\mathcal{X}^d, P^d)$, and P^d denoting the discrete property of the measure. In the following a method is presented, how to adapt the boolean function of the neural network iteratively, during the training phase of the perceptron layer. The training of both layers of the network is performed consecutively in a "ping-pong" like regime. Therefore the perceptron training of the first layer of the neural network is splitted into a finite number j^* of training epochs including a constant number of training steps Δs. After each epoch an adaptation of the boolean function is performed. The adaptation of the boolean function is based on the discrete probability space $(\mathcal{X}^d, P^d)$, corresponding with the domain of b. This already motivates the adaptation theorem.

Theorem 4.1 (Bayesian Adaptation) *Given a boolean type of multilayer perceptron* $n^{(j)} = p^{(j)} \circ b^{(j)} : \mathcal{X} \rightarrow \{0,1\}$ *after the* j*-th epoch of the perceptron training of the first layer, and the discrete probability distributions* P_+^d, P_-^d *and* P^d *of the positively, negatively and not categorized training samples, as well as the probability measure* P_+, P_- *of the probability space* $(\mathcal{X}, \Sigma, P)$, *an adaptation rule of the boolean function* $b^{(j+1)} : \{0,1\}^h \rightarrow \{0,1\}$ *is given by*

$$\mathbf{x}^{p^{(j+1)}} \mapsto \begin{cases} x^{p^{(j)}} & , \text{ if } P^d[\mathbf{x}^{p^{(j)}}] = 0 \\ 1 & , \text{ if } P_+^d[\mathbf{x}^{p^{(j)}}]P_+ > P_-^d[\mathbf{x}^{p^{(j)}}]P_- \\ 0 & , \text{ else} \end{cases} .$$

With regard to Bayes decision rule for minimum error, the introduced adaptation method is optimal.

PROOF. The Bayes decision rule for two–class problems is based on conditional probabilities $P_+^d[\mathbf{x}^p]$, $P_-^d[\mathbf{x}^p]$ and a priori probabilities P_+, P_-, which are assumed to be known, i.e. the Bayes decision rule is identical with the assignment in Theorem 4.1. In [7, 21] the decision rule is proven to be optimal for the minimization of error. □

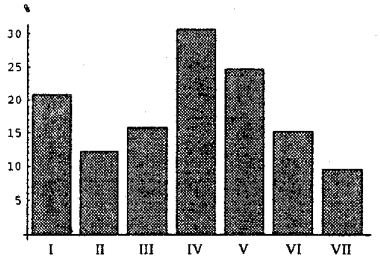
A simple calculation provides an additional motivation of Theorem 4.1. Under the assumption $\frac{m_+}{m_-} = \frac{P_+}{P_-}$ of a balanced number of positively and negatively categorized training samples $m_\pm = |X_\pm|$ within the training set, the Bayesian adaptation in Theorem 4.1 is simply reduced to a comparison of sample counts, i.e. $P_+^d[\mathbf{x}^{p^{(j)}}]P_+ > P_-^d[\mathbf{x}^{p^{(j)}}]P_-$ is simplified to $m_+(x_{p_1p_2\cdots p_h}^{(j)}) > m_-(x_{p_1p_2\cdots p_h}^{(j)})$, whereas the sample count for the positively categorized training samples of each element of $\mathcal{X}^d$ is given by e.g. $m_+(x_{p_1p_2\cdots p_h}^{(j)}) := \sum_{i=1,\mathbf{x}_i\in X_+}^{m_+} i_{x_{p_1p_2\cdots p_h}^{(j)}}(\mathbf{x}_i)$. This rule may be interpreted as follows. An adaptation of the entry in the lookup–table for an element $x_{p_1p_2\cdots p_h}^{(j)} \in \mathcal{X}^d$ is performed if, and only if, the transition from $0 \rightarrow 1$ or $1 \rightarrow 0$ immediately reduces the observed error within the training set. After the j-th adaptation step the number of correctly classified sample points is increased by $\Delta\epsilon_i^X$, with $2\Delta\epsilon_i^X = \sum_{p_1p_2\cdots p_h\in\{0,1\}^h}^{|\mathcal{X}^d|} \left(m_+(x_{p_1p_2\cdots p_h}^{(j+1)}) - m_+(x_{p_1p_2\cdots p_h}^{(j)})\right) - \sum_{p_1p_2\cdots p_h\in\{0,1\}^h}^{|\mathcal{X}^d|} \left(m_-(x_{p_1p_2\cdots p_h}^{(j+1)}) - m_-(x_{p_1p_2\cdots p_h}^{(j)})\right)$.

There may arise the question of computational complexity of the presented method, because if the number h of hidden neurons of the perceptron layer is smaller than the dimension n of the sample space $\mathcal{X}$, the number of elements of the discrete space $\mathcal{X}^d$ equals the upper bound of 2^h [5]. The most important argument for feasibility is the limitation of existing data bases for real applications. For each adaptation step the number of assignments by Theorem 4.1 is bounded by the number m of training samples, because the number of elements $x_{p_1p_2\cdots p_h}$ of $\mathcal{X}^d$ for which $m(x_{p_1p_2\cdots p_h}) \neq 0$, i.e. $P^d[p_1p_2\cdots p_h] \neq 0$, cannot exceed m. For an implementation of the algorithm the computational complexity is transformable into an addressing problem.

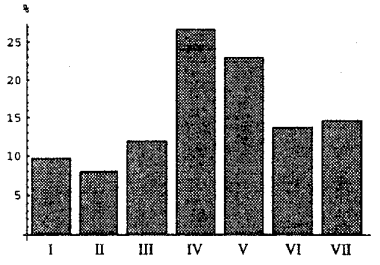
5. Results of Bayesian Adaptation

In this Section the results of Bayesian adaptation applied to a boolean type of neural network are presented. The perceptron layer of the network consists of $h = 5$ hidden neurons and the default function of the boolean layer is defined by the majority function. The results are based on a training set of 10×1000 greylevel bitmaps of handwritten characters randomly drawn from the database of NIST [15], whereby the database consists of $M = 10$ disjoint classes $X_i \cap X_j = 0$, $i \neq j$. The employed neural network is part of a complex system for classificaton of multiple classes, i.e. a single boolean multilayer perceptron just performs a classification of dichotomy of classes. The training of the perceptron layer of the neural networks is carried out by an extension of Widrow's MADALINE rules [19, 20], the

MADATRON algorithm in [13]. After each epoch of the training phase an update of the boolean function is performed by the rules of Bayesian adaptation. After the complete training of the network the error rates within the training (testing) data are displayed in Figure 2a (2b). The charts of all relevant two–class problems (I–VII) present the averaged reduction of the error rates compared with results using a constant default boolean function for the second layer of the network.



a) reduction of ϵ^X within training data in %



b) reduction of $\epsilon^{\mathcal{X}}$ within test data in %

Figure 2. The reduction of (a) the training error ϵ^X and (b) the generalization error $\epsilon^{\mathcal{X}}$ by the introduced "ping-pong" like algorithm compared with a network training based on a constant committee–machine architecture, i.e. when using the majority function as the constant boolean function b. The total number of $2 \cdot 10^5$ training steps has been divided into $j^* = 200$ epochs. The reduction rates have been recorded for a neural network of $h = 5$ hidden neurons applied to all classification problems I–VII. For example, problem VII represents a dichotomy of all classes of handwritten characters into $\{X_0, X_1, X_2, X_3, X_8, X_9\}$ and $\{X_4, X_5, X_6, X_7\}$

Both for a fixed boolean function and for an adaptation of the boolean function of the neural network, Figure 3 shows the evolution of the error rates ϵ^X during 10×2 independent training runs for classification problem VII. The error rates ϵ^X are recorded along the number of $j^* = 200$ training epochs, whereby a reduction of ϵ^X is only displayed when an adaptation of the boolean function has been occured. Finally Figure 4 shows the probability distribution of the discrete elements of the input space after the final adaptation $b_{\mathrm{VII}}^{(j^*-1)} \mapsto b_{\mathrm{VII}}^{(j^*)}$ of the boolean function of the neural network. An alternative presentation of the boolean function is used. The black and white pixels are enumerated by the binary index of the cells, represented by the pixels, whereas black pixels correspond to a "1" and white pixels correspond to a "0" assignment of a cell. All pixels together entirely display the definition of the boolean function on $\mathcal{X}^d$.

6. Considerations of Symmetry

Apart from a trivial class of identical solutions by scaling the weight parameters of the perceptron layer, any solution of the training algorithm represents a total set of $2^h \cdot h!$ identical realizations of the network mapping, i.e. of an unique partitioning of the input space. $2^h \cdot h!$ is given by the number of all consistent transformations $\Phi \circ \Psi$ of the perceptron layer of the neural network. A transformation is called consistent, if its influence is neutralized by permutations and inhibitions of variables x^{p_i} of the boolean function b. Hence, the transformation $\Phi \circ \Psi$ is given by all permutations of neurons of the perceptron layer and all substitutions of weights by their antiparallel equivalent. Permutations Φ and substitutions Ψ are given by $\Phi : (p_1, p_2, \ldots, p_h) \mapsto (p_{i_1}, p_{i_2}, \ldots, p_{i_h})$ and $\Psi : (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_h) \mapsto ((-1)^{s_1}\mathbf{w}_1, (-1)^{s_2}\mathbf{w}_2, \ldots, (-1)^{s_h}\mathbf{w}_h)$, for all $i_j \neq i_k$ and $s_i \in \{0, 1\}$, with $i, i_j, i_k \in \{1, 2, \ldots, h\}$.

7. Conclusion

Neural networks utilizing hard limiter activation functions inherently consist of a boolean function within their description. Boolean neural networks has been shown to be a basic concept even to be competitive with backpropagation trained neural networks [16]. In this paper an approach has been presented, how to adapt the boolean part of multilayer perceptrons using statistics. First of all the perceptron layer of the neural network has been identified to establish a discretization of the input space. The Bayesian adaptation rule has been proven to be optimal within a single training epoch. A "ping–pong" like regime of the training of both parts of the neural network is performed during the training. The results of Bayesian adaptation are based on the classification problem of greylevel bitmaps of handwritten characters. In all examples the misclassification rate of the neural network has been reduced about $10\% - 30\%$ compared with a network training based on a constant boolean function. A reason for this is the higher expressive power of the neural network given by the additional variety of the boolean layer. An additional interpretation of the higher efficiency of the presented algorithm refers to the large ambiguity of possible

solutions. The flexibility of the boolean function multiplies the number of equivalent solutions and the "basins of attraction" of an efficient training of the neural network by $2^h \cdot h!$. Therefore the transition of the network configuration into an adequate solution is considerably much easier.

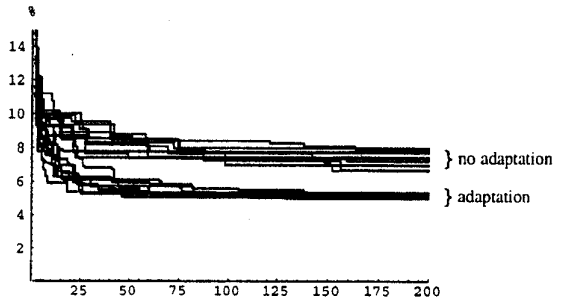


Figure 3. The error rates ϵ^X within the training set of classification problem VII during the training phase of independent training runs of the neural network, whereby $\Delta s = 1000$ and $j^* = 200$. The set of training curves with worse error rates are related to a training without Bayesian Adaptation, whereas for using the adaptation rule the rates are considerably better.

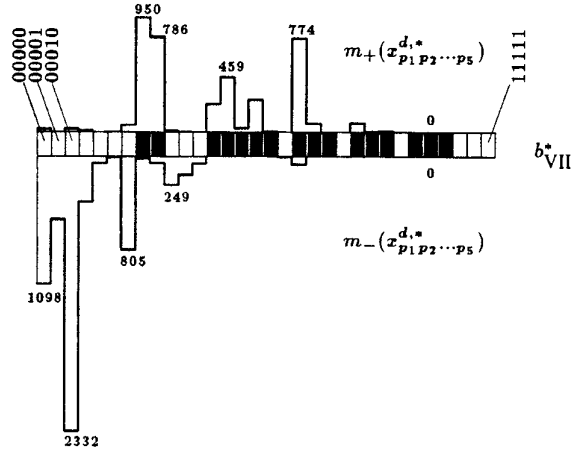


Figure 4. The discrete probability distribution of the two–class problem VII is displayed by the sample counts $m_+(x_{p_1 p_2 \cdots p_h})$ and $m_-(x_{p_1 p_2 \cdots p_h})$ of the discrete elements of the probability space $(\mathcal{X}, P^d)$ within the training set $X = X_+^{\mathrm{VII}} \cup X_-^{\mathrm{VII}}$.

References

[1] Learning internal representations by error propagation,. In D. Rumelhart and J. McClelland, editors, *Parallel Distributed Processing*, volume 1, chapter 8. MIT Press, Cambridge, MA, 1986.

[2] M. Anthony and N. Biggs. Computational Learning Theory for Artificial Neural Networks. In J. G. Taylor, editor, *Mathematical Approaches to Neural Networks*, pages 25–62. Elsevier Science Publishers B.V., 1993.

[3] E. B. Baum. On the Capabilities of Multilayer Perceptrons. *J. of Complexity*, 4:193–215, 1988.

[4] P. Billingsley. *Probability and Measure*. Wiley, 1979.

[5] T. M. Cover. Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition. *IEEE Trans. on Elect. Comp.*, 14:326–334, 1965.

[6] R. Eigenmann and J. Nossek. Constructive and Robust Combination of Perceptrons. 13th International Conference on Pattern Recognition, Vienna, 1996.

[7] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, 1972.

[8] J. Ghosh and K. Tumer. Structural Adaptation and Generalization in Supervised Feed-Forward Networks. Technical report, The University of Texas at Austin, Department of Electrical and Computer Engineering, 1995.

[9] G. Gibson and C. Cowan. On the Decision Regions of Multilayer Perceptrons. *IEEE Proc.*, 78(10):1590–1594, 1990.

[10] D. Hush and B. Horne. Progress in Supervised Neural Networks. *IEEE Signal Processing Magazine*, pages 8–39, January 1993.

[11] M. Minsky and S. Papert. *Perceptrons*. MIT Press, Cambridge, 1969.

[12] S. Muroga. *Threshold Logic and Its Applications*. Wiley-Interscience, 1971.

[13] P. Nachbar. *Entwurf robuster neuronaler Netze*. PhD thesis, Technical University Munich, Institute for Network Theory and Circuit Design, 1994.

[14] P. Nachbar, J. Strobl, and J. Nossek. The generalized adatron algorithm. In *International Symposium on Circuits and Systems*, volume 4, pages 2152–2156. IEEE, 1993.

[15] The state-of-the-art in OCR is subject of NIST conference. *Intelligence*, 9(6):1–5, 1992.

[16] J. Nossek, P. Nachbar, and A. Schuler. Comparison of Algorithms for Feedforward Multilayer Neural Nets. ISCAS, Atlanta, to be published, 1996.

[17] W. Utschick. How To Improve Multi Layer Perceptrons Using Statistics. Technical Report TUM-LNS-TR-95-4, Technical University Munich, 1995.

[18] H. Veit. Automatischer Design von Codes für die Mustererkennung von Mehrfachklassen mit neuronalen Netzen, to be published. Master's thesis, Technical University Munich, Institute for Network Theory and Circuit Design, 1996.

[19] B. Widrow and M. Lehr. 30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation. *IEEE Proc.*, 78(9):1415–1441, 1990.

[20] B. Widrow, R. Winter, and R. Baxter. Layered Neural Nets for Pattern Recognition. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 36(7):1109–1118, 1988.

[21] T. Young and T. Calvert. *Classification, Estimation And Pattern Recognition*. American Elsevier Publishing Co., Inc., 1974.