

Upper Bounds on the Number of Hidden Neurons in Feedforward Networks with Arbitrary Bounded Nonlinear Activation Functions

Guang-Bin Huang and Haroon A. Babri

Abstract—It is well known that standard single-hidden layer feedforward networks (SLFN's) with at most N hidden neurons (including biases) can learn N distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$ with zero error, and the weights connecting the input neurons and the hidden neurons can be chosen "almost" arbitrarily. However, these results have been obtained for the case when the activation function for the hidden neurons is the signum function. This paper rigorously proves that standard single-hidden layer feedforward networks (SLFN's) with at most N hidden neurons and with any bounded nonlinear activation function which has a limit at one infinity can learn N distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$ with zero error. The previous method of arbitrarily choosing weights is not feasible for any SLFN. The proof of our result is constructive and thus gives a method to directly find the weights of the standard SLFN's with any such bounded nonlinear activation function as opposed to iterative training algorithms in the literature.

Index Terms—Activation functions, feedforward networks, hidden neurons, upper bounds.

I. INTRODUCTION

The widespread popularity of neural networks in many fields is mainly due to their ability to approximate complex nonlinear mappings directly from the input samples. Neural networks can provide models for a large class of natural and artificial phenomena that are difficult to handle using classical parametric techniques. Out of many kinds of neural networks multilayer feedforward neural networks have been investigated more thoroughly.

From a mathematical point of view, research on the approximation capabilities of multilayer feedforward neural networks has focused on two aspects: universal approximation in \mathbf{R}^n or one compact set of \mathbf{R}^n , i.e., $[a, b]^n$, and approximation in a finite set $\{(\mathbf{x}_i, \mathbf{t}_i) | \mathbf{x}_i \in \mathbf{R}^n, \mathbf{t}_i \in \mathbf{R}^m, i = 1, 2, \dots, N\}$. Many researchers [4]–[23] have explored the universal approximation capabilities of standard multilayer feedforward neural networks. Hornik [16] proved that if the activation function is continuous, bounded and nonconstant, then continuous mappings can be approximated in measure by neural networks over compact input sets. Leshno [17] improved the results of Hornik [16], and proved that feedforward networks with a nonpolynomial activation function can approximate (in measure) continuous functions. Ito [14] proved the uniform approximation capability of feedforward networks in $C(\overline{\mathbf{R}^n})$, where the activation function was assumed to be the monotonic sigmoidal function. In a recent paper [23] we proved that standard single-hidden layer feedforward networks (SLFN's) with arbitrary bounded nonlinear (continuous or noncontinuous) activation functions that have two unequal limits at infinities can uniformly approximate arbitrary continuous mappings in $C(\overline{\mathbf{R}^n})$ with any precision, and the boundedness on the activation function is sufficient, but not necessary.

In applications neural networks are trained using finite input samples. It is known that N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$ can be learned precisely by standard SLFN's with N hidden neurons

(including biases)¹ and the signum activation function. The bounds on the number of the hidden neurons were derived in [2] by finding particular hyperplanes that separate the input samples and then using the equations describing these hyperplanes to choose the weights for the hidden layer. However, it is not easy to find such hyperplanes, especially for nonregular activation functions. Sartori and Antsaklis [3] observed that particular hyperplanes separating input samples need not be found, and the weights for the hidden layer can be chosen "almost" arbitrarily. *These results were proved for the case where the activation function of the hidden neurons is the signum function.* It was further pointed out [3] that the nonlinearities for the hidden layer neurons are not restricted to be the signum function. Although Sartori and Antsaklis's method is efficient for activation functions like the signum and sigmoidal functions, it is not feasible for all cases. The success of the method depends on the activation function and the distribution of the input samples because for some activation function such "almost" arbitrarily chosen weights may cause the inputs of hidden neurons to lie within a linear subinterval of the nonlinear activation function (see the Appendix).

What possible nonlinear functions can be used for the hidden layer neurons such that an SLFN with at most N hidden neurons can approximate any N arbitrary distinct samples with zero error has not been answered yet. In this paper we show that an SLFN with at most N hidden neurons and with any arbitrary bounded nonlinear activation function which has a limit at one infinity can approximate any N arbitrary distinct samples with zero error. In fact, the proof of our result is constructive and gives a method to directly find the weights of such standard SLFN's without iterative learning. The proof also shows that from the theoretical point of view such weight combinations are numerous. There exists a reference point $x_0 \in \mathbf{R}$ so that the required weights can be directly determined by any point $x \geq x_0$ if there exists $\lim_{x \rightarrow +\infty} g(x)$ ($x \leq x_0$ if there exists $\lim_{x \rightarrow -\infty} g(x)$).

II. PRELIMINARIES

A. N Distinct Samples Approximation Problem

For N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$, where $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{in}]^T \in \mathbf{R}^n$ and $\mathbf{t}_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T \in \mathbf{R}^m$, standard SLFN's with N hidden neurons and activation function $g(x)$ are mathematically modeled as

$$\sum_{i=1}^N \beta_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \mathbf{o}_j, \quad j = 1, \dots, N \quad (1)$$

where $\mathbf{w}_i = [w_{i1}, w_{i2}, \dots, w_{in}]^T$ is the weight vector connecting the i th hidden neuron and the input neurons, $\beta_i = [\beta_{i1}, \beta_{i2}, \dots, \beta_{im}]^T$ is the weight vector connecting the i th hidden neuron and the output neurons, and b_i is the threshold of the i th hidden neuron. $\mathbf{w}_i \cdot \mathbf{x}_j$ denotes the inner product of \mathbf{w}_i and \mathbf{x}_j . The output neurons are chosen linear in this paper. That standard SLFN's with N hidden neurons with activation function $g(x)$ can approximate these N samples with zero error means that $\sum_{j=1}^N \|\mathbf{o}_j - \mathbf{t}_j\| = 0$, i.e., there exist β_i, \mathbf{w}_i ,

¹In [2] and [3], the authors show that when using the signum functions $N - 1$ hidden units are sufficient. However, their schemes have N hidden units with the input connection of the last unit set to zero. This accounts for the figure $N - 1$ hidden units in their work. It is similar to our construction (N hidden units with the last unit input connection set to zero) [see (12) and (13) or (17) and (18) in this letter].

Manuscript received February 8, 1997; revised October 19, 1997.

The authors are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798.

Publisher Item Identifier S 1045-9227(98)01077-7.

and b_i such that

$$\sum_{i=1}^N \beta_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \mathbf{t}_j, \quad j = 1, \dots, N. \quad (2)$$

The above N equations can be written compactly as

$$\mathbf{H}\beta = \mathbf{T} \quad (3)$$

where

$$\mathbf{H}(\mathbf{w}_1, \dots, \mathbf{w}_N, b_1, \dots, b_N, \mathbf{x}_1, \dots, \mathbf{x}_N) = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \dots & g(\mathbf{w}_N \cdot \mathbf{x}_1 + b_N) \\ \vdots & \dots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \dots & g(\mathbf{w}_N \cdot \mathbf{x}_N + b_N) \end{bmatrix}_{N \times N}. \quad (4)$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_N^T \end{bmatrix}_{N \times m} \quad \text{and} \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix}_{N \times m}. \quad (5)$$

We call matrix \mathbf{H} the hidden layer output matrix of the neural network; the i th column of \mathbf{H} is the i th hidden neuron output with respect to inputs $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$.

We show that for any bounded nonlinear activation $g(x)$ which has a limit at one infinity we can choose \mathbf{w}_i, β_i and $b_i, i = 1, \dots, N$, such that $\text{rank } \mathbf{H} = N$ and $\beta = \mathbf{H}^{-1}\mathbf{T}$, so that $\mathbf{H}\beta = \mathbf{T}$.

B. Necessary Lemmas

Lemma 2.1: Let $M(x) = [m_{ij}(x)]$ be an $n \times n$ matrix with respect to x and all of its elements bounded for all $x \in \mathbf{R}$. If $\lim_{x \rightarrow +\infty} m_{ii}(x) = c_i (\lim_{x \rightarrow -\infty} m_{ii}(x) = c_i)$ for $1 \leq i \leq n$ and $\lim_{x \rightarrow +\infty} m_{ij}(x) = 0 (\lim_{x \rightarrow -\infty} m_{ij}(x) = 0)$ for $n \geq i > j \geq 1$, where $c_i (i = 1, \dots, n)$ are nonzero constants, then there exists a point x_0 such that $M(x)$ is invertible for $x \geq x_0 (x \leq x_0)$.

The lemma states that for $n \times n$ matrix $M(x)$ with bounded elements if all the main diagonal elements converge to nonzero constants and all the lower triangular elements converge to zero as $x \rightarrow +\infty (x \rightarrow -\infty)$ then there exists a point x_0 such that $\text{rank } M(x) = n$ for $x \geq x_0 (x \leq x_0)$.

Proof: The determinant of $M(x)$ can be defined [24] as

$$\det M(x) = \sum_{(j_1, \dots, j_n)} s(j_1, \dots, j_n) m_{1j_1}(x) m_{2j_2}(x) \dots m_{nj_n}(x). \quad (6)$$

The summation extends over all $n!$ permutations j_1, \dots, j_n of $1, \dots, n$. $s(j_1, \dots, j_n)$ is the sign of the permutation j_1, \dots, j_n which is $+1$ or -1 [24].

Because $\lim_{x \rightarrow +\infty} m_{ii}(x) = c_i (\lim_{x \rightarrow -\infty} m_{ii}(x) = c_i)$ for $1 \leq i \leq n$ and $\lim_{x \rightarrow +\infty} m_{ij}(x) = 0 (\lim_{x \rightarrow -\infty} m_{ij}(x) = 0)$, for $n \geq i > j \geq 1$, out of the $n!$ terms of (6) only $m_{11}(x) \dots m_{nn}(x)$ converges to $c_1 \dots c_n$ (a nonzero constant) and others to zero as $x \rightarrow +\infty (x \rightarrow -\infty)$. That is

$$\lim_{x \rightarrow +\infty} \det M(x) = c_1 \dots c_n \neq 0 \quad (7)$$

$$\left(\lim_{x \rightarrow -\infty} \det M(x) = c_1 \dots c_n \neq 0 \right). \quad (8)$$

Thus, there exists a point x_0 such that $M(x)$ is invertible for $x \geq x_0 (x \leq x_0)$. \square

Lemma 2.2: Let $M(x) = [m_{ij}(x)]$ be an $n \times n$ matrix with respect to x and all of its elements bounded for all $x \in \mathbf{R}$. If $\lim_{x \rightarrow +\infty} m_{i,i+1}(x) = c_1 (\lim_{x \rightarrow -\infty} m_{i,i+1}(x) = c_1)$ for $1 \leq i \leq n-1$ and $\lim_{x \rightarrow +\infty} m_{i,j}(x) = c_2 (\lim_{x \rightarrow -\infty} m_{i,j}(x) = c_2)$ for $n \geq i \geq j \geq 1$, where c_1 and c_2 are nonzero constants, and $c_1 \neq c_2$, then there exists a point x_0 such that $M(x)$ is invertible for $x \geq x_0 (x \leq x_0)$.

Proof: We prove the lemma for the case when $\lim_{x \rightarrow +\infty} m_{i,i+1}(x) = c_1$ for $1 \leq i \leq n-1$ and $\lim_{x \rightarrow +\infty} m_{i,j}(x) = c_2$ for $n \geq i \geq j \geq 1$. Matrix $M(x)$ can be transformed into matrix $EM(x) = [em_{ij}(x)]_{n \times n}$ using the elementary operations: the $(n-1)$ th row, multiplied by -1 , is added to the n th row; then the $(n-2)$ th row, multiplied by -1 , is added to the $(n-1)$ th row, and so on. Finally, the first row, multiplied by -1 , is added to the second row. Thus, $em_{ij}(x)$ is bounded for all i and j , and we have

$$em_{ij}(x) = \begin{cases} m_{1j}(x), & \text{if } i = 1 \\ m_{ij}(x) - m_{i-1,j}(x), & \text{if } i \neq 1. \end{cases} \quad (9)$$

Because $\lim_{x \rightarrow +\infty} m_{i,i+1}(x) = c_1$ for $1 \leq i \leq n-1$ and $\lim_{x \rightarrow +\infty} m_{i,j}(x) = c_2$ for $i \geq j$, we get

$$\lim_{x \rightarrow +\infty} em_{ij}(x) = \begin{cases} c_2, & \text{if } i = j = 1 \\ c_2 - c_1 \neq 0, & \text{if } i = j \neq 1 \\ 0, & \text{if } i > j. \end{cases} \quad (10)$$

According to Lemma 2.1 there exists a point x_0 such that $\text{rank } EM(x) = n$ for $x \geq x_0$. We know $\text{rank } M(x) = \text{rank } EM(x)$, thus, $M(x)$ is invertible for $x \geq x_0$.

The case when $\lim_{x \rightarrow +\infty} m_{i,i+1}(x) = c_1$ for $1 \leq i \leq n-1$ and $\lim_{x \rightarrow +\infty} m_{i,j}(x) = c_2$ for $n \geq i \geq j \geq 1$ can be dealt with similarly. \square

Lemma 2.3: For any N distinct vectors $\mathbf{x}_i \in \mathbf{R}^n, i = 1, \dots, N$, there exists a vector \mathbf{w} such that the N inner products $\mathbf{w} \cdot \mathbf{x}_1, \dots, \mathbf{w} \cdot \mathbf{x}_N$ are different from each other.

Proof: Let

$$V_{ij} = \{\mathbf{w} | \mathbf{w} \cdot (\mathbf{x}_i - \mathbf{x}_j) = 0, \mathbf{w} \in \mathbf{R}^n\} \quad (11)$$

V_{ij} is a hyperplane in \mathbf{R}^n . Obviously, $V_{ij} = V_{ji}$, and $\mathbf{w} \cdot (\mathbf{x}_i - \mathbf{x}_j) = 0$ iff $\mathbf{w} \in V_{ij}$. Thus, for any $\mathbf{w} \in \mathbf{R}^n - \cup_{i,j} V_{ij}$ we have $\mathbf{w} \cdot (\mathbf{x}_i - \mathbf{x}_j) \neq 0$ for $1 \leq i \neq j \leq N$, i.e., there exists a vector \mathbf{w} such that $\mathbf{w} \cdot \mathbf{x}_i \neq \mathbf{w} \cdot \mathbf{x}_j$ for any $1 \leq i \neq j \leq N$. \square

It is noted that only the vectors which lie within two of the $(N+1) \times N/2$ hyperplanes V_{ij} can make at least two of the N inner products $\mathbf{w} \cdot \mathbf{x}_1, \dots, \mathbf{w} \cdot \mathbf{x}_N$ equal to each other. However, over all \mathbf{R}^n this case seldom happens, so that the vector \mathbf{w} which satisfies the above lemma can be chosen randomly.

III. UPPER BOUND ON THE NUMBER OF HIDDEN NEURONS

In this section we prove that an SLFN with at most N hidden neurons and with any bounded nonlinear activation function which has a limit at one infinity can approximate any N arbitrary different input samples with zero error. Such activation functions include the signum, ramp, and sigmoidal functions (shown in Fig. 1), as well as the radial basis [5], "cosine squasher" [11], generalized sigmoidal [20], and most nonregular functions as shown in Fig. 2.

Theorem 3.1: Given a bounded nonlinear activation function g in \mathbf{R} for which there exists $\lim_{x \rightarrow +\infty} g(x)$ or $\lim_{x \rightarrow -\infty} g(x)$, then for any N arbitrary distinct input samples $\{(\mathbf{x}_i, \mathbf{t}_i) | \mathbf{x}_i \in \mathbf{R}^n, \mathbf{t}_i \in \mathbf{R}^m, i = 1, \dots, N\}$, there exist \mathbf{w}_i and $b_i, i = 1, \dots, N$, such that hidden layer output matrix \mathbf{H} is invertible.

Proof: Because all \mathbf{x}_i 's are different, according to Lemma 2.3 there exists a (arbitrarily chosen) \mathbf{w} such that $\mathbf{w} \cdot \mathbf{x}_1, \dots, \mathbf{w} \cdot \mathbf{x}_N$ are all different. Without loss of generality, assume $\mathbf{w} \cdot \mathbf{x}_1 < \mathbf{w} \cdot \mathbf{x}_2 < \dots < \mathbf{w} \cdot \mathbf{x}_N$ since one can always change the sequence of

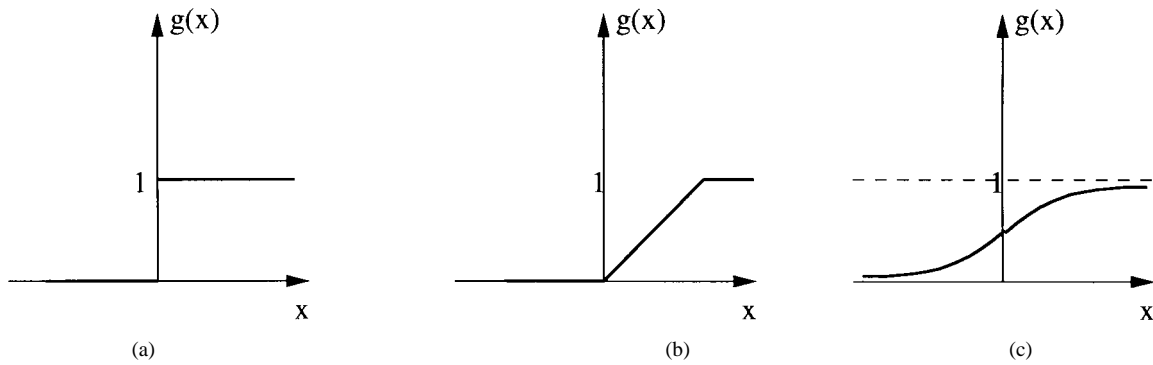


Fig. 1. Three classical activation functions: (a) signum function, (b) ramp function, and (c) sigmoidal function.

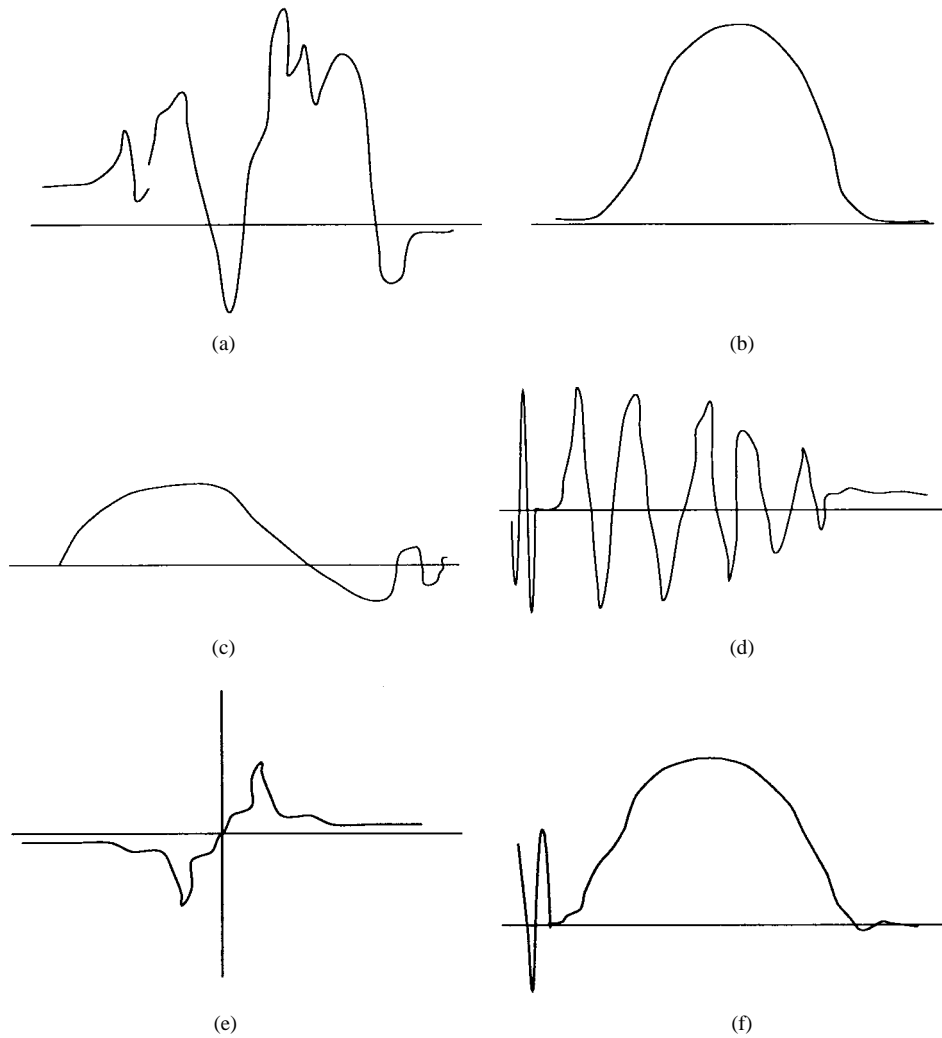


Fig. 2. Six nonregular activation functions, all of them have a limit at one infinity.

samples \$(x_i, t_i)\$ without changing the rank of \$\mathbf{H}\$. With respect to inputs \$\mathbf{x}_1, \dots, \mathbf{x}_n\$, the hidden layer output matrix \$\mathbf{H}\$ is

$$\mathbf{H} = [h_{ij}]_{N \times N} = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \dots & g(\mathbf{w}_N \cdot \mathbf{x}_1 + b_N) \\ g(\mathbf{w}_1 \cdot \mathbf{x}_2 + b_1) & \dots & g(\mathbf{w}_N \cdot \mathbf{x}_2 + b_N) \\ \vdots & \dots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \dots & g(\mathbf{w}_N \cdot \mathbf{x}_N + b_N) \end{bmatrix}$$

where \$\mathbf{w}_i\$ is the weight vector connecting the \$i\$th hidden neuron and the input neurons and \$b_i\$ the threshold of the \$i\$th hidden neuron. We

now show that we can choose \$\mathbf{w}_i\$ and \$b_i\$ such that \$\mathbf{H}\$ is invertible.

Case 1: \$\lim_{x \rightarrow +\infty} g(x) = 0\$ (\$\lim_{x \rightarrow -\infty} g(x) = 0\$) and \$g(x)\$ is nonlinear. There exists a point \$x_{01}\$ such that \$g(x_{01}) \neq 0\$. For the point \$x_{01}\$ and any other point \$x_{02}\$ we can choose \$\mathbf{w}_i\$ and \$b_i\$ (\$1 \le i \le N\$) such that \$\mathbf{w}_i \cdot \mathbf{x}_i + b_i = x_{01}\$, \$1 \le i \le N\$, and \$\mathbf{w}_i \cdot \mathbf{x}_{i+1} + b_i = x_{02}\$, \$1 \le i \le N - 1\$. Indeed, we can choose

$$\mathbf{w}_i = \begin{cases} \frac{x_{02} - x_{01}}{\mathbf{w} \cdot \mathbf{x}_{i+1} - \mathbf{w} \cdot \mathbf{x}_i} \mathbf{w}, & \text{if } 1 \leq i \leq N - 1 \\ \mathbf{0}, & \text{if } i = N \end{cases} \quad (12)$$

and

$$b_i = \begin{cases} \frac{x_{01}\mathbf{w} \cdot \mathbf{x}_{i+1} - x_{02}\mathbf{w} \cdot \mathbf{x}_i}{\mathbf{w} \cdot \mathbf{x}_{i+1} - \mathbf{w} \cdot \mathbf{x}_i}, & \text{if } 1 \leq i \leq N-1 \\ x_{01}, & \text{if } i = N. \end{cases} \quad (13)$$

For $1 \leq i \leq N-1$ we get (14), shown at the bottom of the page. Since $\mathbf{w} \cdot \mathbf{x}_{i+1} - \mathbf{w} \cdot \mathbf{x}_i > 0$ and $\mathbf{w} \cdot \mathbf{x}_j - \mathbf{w} \cdot \mathbf{x}_i > 0$ for $j > i$, we have

$$\lim_{x_{02} \rightarrow +\infty} h_{ji} = \lim_{x_{02} \rightarrow +\infty} g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = 0, \quad \text{if } j > i \quad (15)$$

$$\left(\lim_{x_{02} \rightarrow -\infty} h_{ji} = \lim_{x_{02} \rightarrow -\infty} g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = 0, \quad \text{if } j > i \right). \quad (16)$$

We know $h_{ii} = g(\mathbf{w}_i \cdot \mathbf{x}_i + b_i) = g(x_{01}) \neq 0$ for $1 \leq i \leq N$. Thus, according to Lemma 2.1 there exists x_{02} , and also the corresponding \mathbf{w}_i and b_i [given by (12) and (13)] such that hidden layer output matrix \mathbf{H} is invertible. (In fact, according to Lemma 2.1 there exists x_0 such that \mathbf{H} is invertible for any $x_{02} \geq x_0 (x_{02} \leq x_0)$, and also the corresponding \mathbf{w}_i and b_i given by (12) and (13).)

Case 2: $\lim_{x \rightarrow +\infty} g(x) = A \neq 0$ ($\lim_{x \rightarrow -\infty} g(x) = A \neq 0$) and $g(x)$ is nonlinear. There exists a point x_{01} such that $g(x_{01}) \neq A$. For the point x_{01} and any other point x_{02} we can choose \mathbf{w}_i and b_i ($1 \leq i \leq N$) such that $\mathbf{w}_i \cdot \mathbf{x}_i + b_i = x_{02}$, $1 \leq i \leq N$, and $\mathbf{w}_{i+1} \cdot \mathbf{x}_i + b_{i+1} = x_{01}$, $1 \leq i \leq N-1$. Indeed, we can choose

$$\mathbf{w}_i = \begin{cases} \mathbf{0}, & \text{if } i = 1 \\ \frac{x_{02} - x_{01}}{\mathbf{w} \cdot \mathbf{x}_i - \mathbf{w} \cdot \mathbf{x}_{i-1}} \mathbf{w}, & \text{if } 2 \leq i \leq N \end{cases} \quad (17)$$

and

$$b_i = \begin{cases} x_{02}, & \text{if } i = 1. \\ \frac{x_{01}\mathbf{w} \cdot \mathbf{x}_i - x_{02}\mathbf{w} \cdot \mathbf{x}_{i-1}}{\mathbf{w} \cdot \mathbf{x}_i - \mathbf{w} \cdot \mathbf{x}_{i-1}}, & \text{if } 2 \leq i \leq N. \end{cases} \quad (18)$$

For $2 \leq i \leq N$ we get (19), shown at the bottom of the page.

Since $\mathbf{w} \cdot \mathbf{x}_i > \mathbf{w} \cdot \mathbf{x}_{i-1}$ and $\mathbf{w} \cdot \mathbf{x}_j > \mathbf{w} \cdot \mathbf{x}_{i-1}$ for $j \leq i$, we have

$$\lim_{x_{02} \rightarrow +\infty} h_{ji} = \lim_{x_{02} \rightarrow +\infty} g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = A, \quad \text{if } j \geq i \quad (20)$$

$$\left(\lim_{x_{02} \rightarrow -\infty} h_{ji} = \lim_{x_{02} \rightarrow -\infty} g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = A, \quad \text{if } j \geq i \right). \quad (21)$$

We know $h_{i,i+1} = g(\mathbf{w}_{i+1} \cdot \mathbf{x}_i + b_{i+1}) = g(x_{01}) \neq A$ for $1 \leq i \leq N-1$. Thus, according to Lemma 2.2 there exists x_{02} , and also the corresponding \mathbf{w}_i and b_i [given by (17) and (18)] such that hidden layer output matrix \mathbf{H} is invertible. (In fact, according to Lemma 2.2 there exists x_0 such that \mathbf{H} is invertible for any

$x_{02} \geq x_0 (x_{02} \leq x_0)$, and also the corresponding \mathbf{w}_i and b_i given by (17) and (18).)

This completes the proof of the theorem. \square

Theorem 3.2: Given a bounded nonlinear activation function g in \mathbf{R} for which there exists $\lim_{x \rightarrow +\infty} g(x)$ or $\lim_{x \rightarrow -\infty} g(x)$, then for any N arbitrary distinct samples $\{(\mathbf{x}_i, \mathbf{t}_i) | \mathbf{x}_i \in \mathbf{R}^n, \mathbf{t}_i \in \mathbf{R}^m, i = 1, \dots, N\}$, there exist \mathbf{w}_i, b_i and $\beta_i, i = 1, \dots, N$, such that

$$\sum_{i=1}^N \beta_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \mathbf{t}_j, \quad j = 1, \dots, N. \quad (22)$$

Proof: From Theorem 3.1 we know that for N different samples $(\mathbf{x}_i, \mathbf{t}_i)$, there exist \mathbf{w}_i and b_i such that hidden layer output matrix \mathbf{H} is invertible. Let β and \mathbf{T} be defined as in (5). Then choose $\beta = \mathbf{H}^{-1}\mathbf{T}$ which implies $\mathbf{H}\beta = \mathbf{T}$. This completes the proof of the theorem. \square

IV. DISCUSSION

A fundamental question that is often raised in the applications of neural networks is ‘‘how large is the network required to perform a desired task?’’ This paper gives an upper bound on the number of hidden neurons required. Such network may have redundancy in some cases, especially in applications. The redundancy may be wasted due to two reasons: 1) in trying to obtain zero-error precision and 2) the correlation between the activation function and the given samples. These two aspects give rise to the problem of optimum network construction. In most applications, the error can be larger than zero and the number of hidden neurons can be less than the upper bound. On the other hand, if the activation function and the given samples are correlated, the number of hidden neurons can again be less than our upper bound. However, generally speaking, in applications there does not exist the least upper bound (LUB) which is available for general cases. For example, for N distinct samples $(\mathbf{x}_i, g(\mathbf{w} \cdot \mathbf{x}_i))$ and the activation function $g(x)$, where \mathbf{w} is one constant vector, only one hidden neuron is enough.

Our constructive method is feasible for many activation functions used in applications. This paper shows that there exists a reference point $x_0 \in \mathbf{R}$ so that the required weights can be directly determined by any point $x \geq x_0$ if there exists $\lim_{x \rightarrow +\infty} g(x)$ ($x \leq x_0$ if there exists $\lim_{x \rightarrow -\infty} g(x)$). This means that x_0 is overestimated in the proof of the result of this paper and in applications the required weights may be directly determined by some points $x < x_0$ ($x > x_0$). The choice of the reference point x_0 depends on the specific activation function used. In practical applications, one would use some regular functions such as the well-known classical functions. For these functions, the reference point x_0 can be chosen so that the absolute value of x_0 is not large and thus the absolute values of elements of

$$\begin{aligned} g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) &= g\left(\frac{x_{02} - x_{01}}{\mathbf{w} \cdot \mathbf{x}_{i+1} - \mathbf{w} \cdot \mathbf{x}_i} \mathbf{w} \cdot \mathbf{x}_j + \frac{x_{01}\mathbf{w} \cdot \mathbf{x}_{i+1} - x_{02}\mathbf{w} \cdot \mathbf{x}_i}{\mathbf{w} \cdot \mathbf{x}_{i+1} - \mathbf{w} \cdot \mathbf{x}_i}\right) \\ &= g\left(\frac{x_{02}(\mathbf{w} \cdot \mathbf{x}_j - \mathbf{w} \cdot \mathbf{x}_i) - x_{01}(\mathbf{w} \cdot \mathbf{x}_j - \mathbf{w} \cdot \mathbf{x}_{i+1})}{\mathbf{w} \cdot \mathbf{x}_{i+1} - \mathbf{w} \cdot \mathbf{x}_i}\right) \end{aligned} \quad (14)$$

$$\begin{aligned} g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) &= g\left(\frac{x_{02} - x_{01}}{\mathbf{w} \cdot \mathbf{x}_i - \mathbf{w} \cdot \mathbf{x}_{i-1}} \mathbf{w} \cdot \mathbf{x}_j + \frac{x_{01}\mathbf{w} \cdot \mathbf{x}_i - x_{02}\mathbf{w} \cdot \mathbf{x}_{i-1}}{\mathbf{w} \cdot \mathbf{x}_i - \mathbf{w} \cdot \mathbf{x}_{i-1}}\right) \\ &= g\left(\frac{x_{02}(\mathbf{w} \cdot \mathbf{x}_j - \mathbf{w} \cdot \mathbf{x}_{i-1}) - x_{01}(\mathbf{w} \cdot \mathbf{x}_j - \mathbf{w} \cdot \mathbf{x}_i)}{\mathbf{w} \cdot \mathbf{x}_i - \mathbf{w} \cdot \mathbf{x}_{i-1}}\right) \end{aligned} \quad (19)$$

w_i and b_i are not very large. For example, for the ramp function $g(x) = x \cdot 1_{0 \leq x \leq 1} + 1_{x \geq 1}$, x_0 can be chosen as one.

This paper provides a sufficient condition for the activation functions by which SLFN's with N hidden neurons can precisely represent N distinct samples. These functions include nearly all the activation functions which are often used in the applications. Some functions, especially from the theoretical viewpoint, may be also sufficient but are not included in our results. Intuitively speaking, it can be conjectured that "the sufficient and necessary condition for activation functions by which SLFN's with N hidden neurons can precisely represent N distinct samples is that these activation functions are nonlinear." We found that for many specific nonlinear activation functions it is easy to prove its correctness. However, other than through constructive methods, it seems much more difficult and complicated to prove the conjecture for general nonlinear functions.

V. CONCLUSION

In this paper it has been rigorously proved that for N arbitrary distinct samples $\{(x_i, t_i) | x_i \in \mathbf{R}^n, t_i \in \mathbf{R}^m, i = 1, \dots, N\}$, SLFN's with at most N hidden neurons and with *any bounded nonlinear activation function which has a limit at one infinity* can learn these N distinct samples with zero error. In most applications, especially in hardware implementations, the above condition is satisfied since the bounded nonlinear functions are often considered in one interval of \mathbf{R} with the values outside the interval considered constant.

Sartori and Antsaklis [3] pointed out "Actually, the signum function does not need to be used as the nonlinearity of the neurons, and in fact almost any arbitrary nonlinearity will suffice to satisfy (that the hidden layer output matrix is invertible)." We hope that we have rigorously proved the conjecture and realized what the "almost any arbitrary nonlinearity" means in a satisfactory way.

APPENDIX

In [3], the weights for hidden neurons are chosen "almost" arbitrarily. This method is not feasible for all cases. The success of the method depends on the activation function and the distribution of the input samples because for some activation functions such "almost" arbitrarily chosen weights may cause the inputs of hidden neurons to lie within a linear subinterval of the nonlinear activation function. In the following we give an example in the one-dimensional case.

Example: Consider N different samples $(x_i, t_i), i = 1, \dots, N$ where $x_i \in (0, 1)$ and the ramp activation function

$$g(x) = \begin{cases} 0, & \text{if } x \leq 0 \\ x, & \text{if } 0 < x < 1 \\ 1, & \text{if } x \geq 1. \end{cases}$$

U^1 and $W^1 = [w_1, \dots, w_{N-1}]$ are defined as in [3]. As in [3], choose W^1 at random in the interval $[0, 1]$ with uniform distribution. Thus, there exist w_i and w_j so that $0 < w_i \neq w_j < 1$. So, $0 < w_i x_l < 1$ and $0 < w_j x_l < 1$ for $l = 1, \dots, N$. We have

$$\Phi(U^1{}^T W^1) = \begin{bmatrix} g(w_1 x_1) & \cdots & g(w_{N-1} x_1) \\ \vdots & \cdots & \vdots \\ g(w_1 x_N) & \cdots & g(w_{N-1} x_N) \end{bmatrix}.$$

The i th and j th columns of $\Phi(U^1{}^T W^1)$ are $[w_i x_1, \dots, w_i x_N]^T$ and $[w_j x_1, \dots, w_j x_N]^T$, respectively. They are proportional to each other, $\text{rank}[\Phi(U^1{}^T W^1)] < N - 1$, and thus, $\text{rank}[\Phi(U^1{}^T W^1)\mathbf{1}] < N$. In fact, any $N - 1$ weights arbitrarily chosen in $[0, 1]$ will cause $\text{rank}[\Phi(U^1{}^T W^1)\mathbf{1}] < N$. Thus, the method fails for the ramp activation function and these input samples.

Actually, in applications the activation function and the distribution of input samples are not known in advance. So the method is not feasible for general cases.

ACKNOWLEDGMENT

The authors wish to thank the reviewers and the editor for their constructive suggestions and comments. The authors also would like to thank Dr. M. Palaniswami, Department of Electrical and Electronic Engineering, the University of Melbourne, Australia, Prof. H.-T. Li, Prof. J.-R. Liu, and Prof. G.-R. Chang, Department of Computer Science and Engineering, Northeastern University, P. R. China, for their valuable discussions.

REFERENCES

- [1] E. Baum, "On the capabilities of multilayer perceptrons," *J. Complexity*, vol. 4, pp. 193–215, 1988.
- [2] S.-C. Huang and Y.-F. Huang, "Bounds on the number of hidden neurons in multilayer perceptrons," *IEEE Trans. Neural Networks*, vol. 2, pp. 47–55, 1991.
- [3] M. A. Sartori and P. J. Antsaklis, "A simple method to derive bounds on the size and to train multilayer neural networks," *IEEE Trans. Neural Networks*, vol. 2, pp. 467–471, 1991.
- [4] T. Poggio and F. Girosi, "A theory of networks for approximation and learning," Artificial Intell. Lab., Massachusetts Inst. Technol., A.I. Memo 1140, 1989.
- [5] F. Girosi and T. Poggio, "Networks and the best approximation property," Artificial Intell. Lab., Massachusetts Inst. Technol., A.I. Memo no. 1164, 1989.
- [6] V. Kůrková, "Kolmogorov's theorem and multilayer neural networks," *Neural Networks*, vol. 5, pp. 501–506, 1992.
- [7] V. Y. Kreinovich, "Arbitrary nonlinearity is sufficient to represent all functions by neural networks: A theorem," *Neural Networks*, vol. 4, pp. 381–383, 1991.
- [8] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989.
- [9] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Contr., Signals, Syst.*, vol. 2, no. 4, pp. 303–314, 1989.
- [10] K. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol. 2, pp. 183–192, 1989.
- [11] A. Gallant and H. White, "There exists a neural network that does not make avoidable mistakes," in *Artificial Neural Networks: Approximation and Learning Theory*, H. White, Ed. Oxford, U.K.: Blackwell, 1992, pp. 5–11.
- [12] M. Stinchcombe and H. White, "Universal approximation using feedforward networks with nonsigmoid hidden layer activation functions," in *Artificial Neural Networks: Approximation and Learning Theory*, H. White, Ed. Oxford, U.K.: Blackwell, 1992, pp. 29–40.
- [13] C.-H. Choi and J. Y. Choi, "Constructive neural networks with piecewise interpolation capabilities for function approximations," *IEEE Trans. Neural Networks*, vol. 5, pp. 936–944, 1994.
- [14] Y. Ito, "Approximation of continuous functions on \mathbf{R}^d by linear combinations of shifted rotations of a sigmoid function with and without scaling," *Neural Networks*, vol. 5, pp. 105–115, 1992.
- [15] —, "Approximation of functions on a compact set by finite sums of a sigmoid function without scaling," *Neural Networks*, vol. 4, pp. 817–826, 1991.
- [16] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251–257, 1991.
- [17] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural Networks*, vol. 6, pp. 861–867, 1993.
- [18] J. Wray and G. G. Green, "Neural networks, approximation theory and finite precision computation," *Neural Networks*, vol. 8, no. 1, pp. 31–37, 1995.
- [19] R. Hecht-Nielsen, "Kolmogorov's mapping neural network existence theorem," in *Proc. Int. Conf. Neural Networks*, 1987, pp. 11–14.
- [20] T. Chen, H. Chen, and R.-W. Liu, "Approximation capability in $C(\overline{\mathbf{R}}^n)$ by multilayer feedforward networks and related problems," *IEEE Trans. Neural Networks*, vol. 6, pp. 25–30, 1995.
- [21] T. Chen and H. Chen, "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application

to dynamical systems," *IEEE Trans. Neural Networks*, vol. 6, pp. 911-917, 1995.

[22] —, "Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks," *IEEE Trans. Neural Networks*, vol. 6, pp. 904-910, 1995.

[23] G.-B. Huang and H. A. Babri, "General approximation theorem on feed-forward networks," in *1997 IEEE Int. Conf. Neural Networks (ICNN'97)*, Houston, TX, 1997.

[24] J. N. Franklin, "Determinants," in *Matrix Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1968, pp. 1-25.

Doubly Stochastic Poisson Processes in Artificial Neural Learning

Howard C. Card

Abstract—This paper investigates neuron activation statistics in artificial neural networks employing stochastic arithmetic. It is shown that a doubly stochastic Poisson process is an appropriate model for the signals in these circuits.

Index Terms—Poisson processes, stochastic arithmetic.

I. INTRODUCTION

It is of considerable practical importance to reduce both the power dissipation and the silicon area in digital implementations of artificial neural networks (ANN's), particularly for portable computing applications. Stochastic arithmetic [1] is one method of achieving these goals which has been successfully employed in several digital implementations of ANN's such as [2]-[4]. We have also recently used stochastic arithmetic in the learning computations of backpropagation networks [5]. In our work we have also employed cellular automata as parallel pseudorandom number generators [6], which substantially improves the hardware efficiency of stochastic arithmetic.

In stochastic arithmetic circuits, it is well known that accuracy in both learning and recall operations is adversely affected by the fluctuations in the arrival rates of signals represented by the stochastic pulse streams. This paper formulates an appropriate model for these processes, in order to better estimate the effects of the statistical fluctuations.

Fig. 1 illustrates an individual neuron in an ANN which has inputs from m other neurons. Let us assume an average pulse arrival rate of λ from a single one of these fan-in neurons. If the pulses are collected over a time interval T , then the expected number of counts per interval is given by

$$\langle n \rangle = \lambda T. \tag{1}$$

The arrival of pulses is assumed to be governed by a Poisson process, so that the actual probability of n counts in an interval T may be written [7]

$$p(n) = \frac{(\lambda T)^n \exp(-\lambda T)}{n!}. \tag{2}$$

Manuscript received April 17, 1997. This work was supported by NSERC and CMC.

The author is with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, Manitoba, Canada R3T 5V6.

Publisher Item Identifier S 1045-9227(98)00602-X.

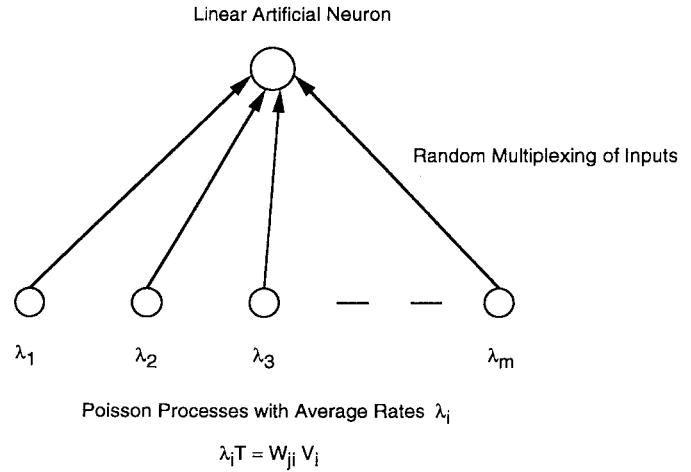


Fig. 1. An artificial neuron driven by m sources with average pulse rates $\lambda_1, \lambda_2, \dots, \lambda_m$.

A plot of $p(n)$ versus N is called a Poisson distribution, and represents a random process with no correlations among successive pulses. Equation (2) can actually represent a source of arbitrary statistics, provided that its coherence time is much longer than the measurement time.

For the neuron of Fig. 1, each of the fan-in neurons will in general have a different average rate parameter λ . We may call these rates $\lambda_1, \lambda_2, \dots, \lambda_m$, respectively, and write for the average rate

$$\lambda_{av} = \frac{1}{m} \sum_{i=1}^m \lambda_i. \tag{3}$$

If we sample all inputs equally in a deterministic fashion, the average count per interval T is given by (2) with $\lambda = \lambda_{av}$. However, it is actually necessary that the neuron in Fig. 1 instead randomly select exactly one of its input neurons at any given time. Many different input neurons are therefore sampled uniformly over the integration period T [8]. In this way the magnitude of the total input does not grow with the fan-in, but rather scales with m . This may be readily accomplished in circuitry by using a multiplexer with a pseudorandom number generator. Therefore, in the actual case, the count distribution is given by the summation

$$p(n) = \frac{1}{m} \sum_{i=1}^m \frac{(\lambda_i T)^n \exp(-\lambda_i T)}{n!}. \tag{4}$$

In the limit as m becomes large, we may approximate the summation by an integral, given by

$$p(n) = \int \frac{(\lambda T)^n \exp(-\lambda T)}{n!} p(\lambda T) d(\lambda T) \tag{5}$$

where it has been assumed that the average rates from individual neurons vary according to the probability distribution $p(\lambda T)$. In the special case of a uniform distribution over the region $\lambda(1 - k)$ to $\lambda(1 + k)$

$$p(n) = \frac{1}{2k\lambda T} \int_{\lambda T(1-k)}^{\lambda T(1+k)} \frac{(\lambda T)^n \exp(-\lambda T)}{n!} d(\lambda T). \tag{6}$$

This is referred to as a doubly stochastic Poisson distribution. The results of (6) are shown in Fig. 2 for various values of the modulation parameter k from zero to one. The increased spread with k in the