From this, noting (15), it follows that

$$\|z(t)\| \le \sqrt{\delta_2/\delta_1}\|z^0\| \exp\left(-\frac{1}{2\delta_2}\, t\right), \quad \forall\, z^0 \in \Re^n, \quad \forall\, t \ge 0. \tag{16}$$

Thus, the unique equilibrium $z = 0$ of system (10) is globally exponentially stable.

Let $u(t;\, u^0)$ for $t \ge 0$ be the solution of the autonomous system (5), which is uniquely determined by the initial condition of $u(0;\, u^0) = u^0 \in \Re^n$ since the right-hand side of (5), as a mapping of $u \in \Re^n$, is globally Lipschitz continuous [8]. If we set $z^0 = u^0 - u^*$, by the uniqueness of solutions for system (10), we have $z(t;\, z^0) = u(t;\, u^0) - u^*$ for $t \ge 0$. Therefore, it follows from (16) that

$$\|u(t;\, u^0) - u^*\| \le \sqrt{\delta_2/\delta_1}\|u^0 - u^*\| \exp\left(-\frac{1}{2\delta_2}\, t\right), \quad \forall\, u^0 \in \Re^n, \quad \forall\, t \ge 0. \tag{17}$$

That is, the network model (5) is globally exponentially stable with degree of $1/(2\delta_2)$, where $\delta_2 > 0$ and is defined in (14). □

Remark 1: It is acknowledged here that one of the reviewers alerts the author that in a more recent paper [9] an improved condition over (6) for global exponential stability of the neural-network model (5) was presented. Under the above special condition, a better lower bound for the rate of global exponential convergence of the network model (5) than the lower bound in (17) can be obtained (see [9] for details). However, the lower bound in (17) for the rate of global exponential convergence of the network model (5) remains hold even if the above imposed conditions are not satisfied, without resorting to any additional condition.

Example 2: Consider the bound constrained quadratic minimization problem (1), where $Q = \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}$, $y = (1,\, 3)^\top$, $\mu = (-1,\, -1)^\top$, and $\nu = (1,\, 1)^\top$. From Example 1, it follows that Q does *not* satisfy the convergence condition (6) in Proposition 1. According to Theorem 1, for the given Q, the network (5) for solving the above minimization problem is actually globally exponentially stable for any given positive diagonal preconditioner B of Q. In the following, we take $B = \text{diag}\{1.0,\, 0.2\}$ such that the matrix QB has ones as diagonal entries. Note that $\text{cond}(QB) = 30.78 < 33.97 = \text{cond}(Q)$, where $\text{cond}(\cdot)$ represents the condition number of the argument matrix. Let $u^0 = (6,\, -5)^\top$ be an initial point. Its corresponding solution trajectory of the network (5) is convergent to the unique equilibrium point $u^* = (-1,\, 5)^\top$, as shown in Fig. 2. It is clear from Fig. 2 that the unique bound constrained minimum of (1) is $x^* = Bf(u^*) = (-1,\, 1)^\top$ with minimal value $J(x^*) = -1$. We then choose randomly 100 uniformly distributed points in the set $[-10,\, 10] \times [-10,\, 10]$ as the initial state points of the solution trajectories of the network (5). It is clear from Fig. 3 that all the network solution trajectories corresponding to the above 100 random initial points are convergent to the unique equilibrium point u^*. Finally, we choose 20 points randomly from the above set as the initial states of the solution trajectories of the network. It is seen from Fig. 4 that the norms of $\|u(t) - u^*\|$ converge to zero exponentially. □

III. CONCLUSION

In this letter we have provided a complete proof of the global exponential convergence of the existing neural-network models for bound constrained quadratic optimization. The global exponential convergence behavior of the network model is illustrated through a numerical simulation example for solving the bound constrained quadratic optimization problems.

ACKNOWLEDGMENT

The author would like to appreciate the anonymous reviewers for their helpful comments.

REFERENCES

[1] S. I. Sudharsanan and M. K. Sundareshan, "Exponential stability and a systematic synthesis of a neural network for quadratic minimization," *Neural Networks*, vol. 4, no. 5, pp. 599–613, 1991.

[2] B. Davis and T. R. Pattison, "Error in proof of exponential convergence," *Neural Networks*, vol. 5, no. 6, p. 869, 1992.

[3] A. Bouzerdoum and T. R. Pattison, "Neural network for quadratic optimization with bound constraints," *IEEE Trans. Neural Networks*, vol. 4, no. 2, pp. 293–303, 1993.

[4] D. W. Tank and J. J. Hopfield, "Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Trans. Circuits Syst.*, vol. 33, no. 5, pp. 533–541, 1986.

[5] M. P. Kennedy and L. O. Chua, "Neural networks for nonlinear programming," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 554–562, 1988.

[6] J. J. Moŕe and G. Toraldo, "On the solution of large quadratic programming problems with bound constraints," *SIAM J. Optimization*, vol. 1, no. 1, pp. 93–113, 1991.

[7] D. G. Kelly, "Stability in contractive nonlinear neural networks," *IEEE Trans. Biomed. Eng.*, vol. 37, pp. 231–242, 1990.

[8] M. Vidyasagar, *Nonlinear Systems Analysis*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[9] A. Bouzerdoum and T. R. Pattison, "Exponential stability of a neural network for bound-constrained quadratic optimization," in *Proc. IEEE Int. Conf. Neural Networks*, vol. II, 1994, pp. 918–923.

Mapping Boolean Functions with Neural Networks having Binary Weights and Zero Thresholds

Vinay Deolalikar

***Abstract*—In this paper, the ability of a binary neural-network comprising only neurons with zero thresholds and binary weights to map given samples of a Boolean function is studied. A mathematical model describing a network with such restrictions is developed. It is shown that this model is quite amenable to algebraic manipulation. A key feature of the model is that it replaces the two input and output variables with a single "normalized" variable. The model is then used to provide *a priori* criteria, stated in terms of the new variable, that a given Boolean function must satisfy in order to be mapped by a network having one or two layers. These criteria provide necessary, and in the case of a one-layer network, sufficient conditions for samples of a Boolean function to be mapped by a binary neural network with zero thresholds. It is shown that the necessary conditions imposed by the two-layer network are, in some sense, minimal.**

***Index Terms*—Binary neural networks, Boolean function mapping, one-layer networks, two-layer networks.**

I. PRELIMINARIES

The classical *feedforward* neural-network architecture comprises layers of neurons. Each neuron has a pair $(\mathbf{w},\, t)$ associated to it,

Manuscript received November 10, 2000; revised February 8, 2001. This work was supported in part by a grant from the Signal Processing and Artificial Neural Networks Lab at I.I.T. Bombay and in part by a Government of India scholarship. This work was done as part of the author's Master's thesis at the Indian Institute of Technology, Bombay.

The author is with Hewlett-Packard Laboratories, Palo Alto, CA 94304 USA (e-mail: vinayd@hpl.hp.com).

Publisher Item Identifier S 1045-9227(01)04160-1.

where $\mathbf{w}$ and t are called its weight vector and threshold, respectively. The output of the neuron for an input $\mathbf{x}$ is given by $\text{sgn}(\mathbf{w} \cdot \mathbf{x} - t)$ which is $+1$ when $\mathbf{w} \cdot \mathbf{x} > t$ and -1 otherwise. The quantity $\mathbf{w} \cdot \mathbf{x}$ is called the *activation* of the neuron.

In this paper, we will examine the particular case of a feedforward architecture where the threshold, t, associated with every neuron is zero, and the weight vectors are binary, with each component chosen from $\{-1, 1\}$. We will study the performance of such networks in mapping Boolean functions $f: \{-1, 1\}^n \rightarrow \{-1, 1\}^m$, where n and m are integers greater than or equal to one. Networks with binary weights have been studied in literature [1], [2], [4], [6], for obvious reasons. First, it is a matter of theoretical curiosity whether networks comprising neurons with such restrictions on their weights have reasonable function mapping capabilities. Second, as is noted in [1], [2], [4], and [6], such networks would have considerably simpler implementations, especially in hardware, and thus would be implementations of choice in cases where their performance proves satisfactory.

Our added restriction—that the thresholds all be zero—in some sense reduces these to the "simplest" class of networks. It is therefore of interest to know what performance they can offer. In this paper, we demonstrate that these restrictions on the weights and thresholds can be exploited fully to develop a model for the network that is very amenable to algebraic manipulation. The key feature of the model that enables comparisons between one- and two-layer networks is the replacement of the two input and output variables with a single "normalized" variable. We then derive constraints placed upon this normalized variable by one- and two-layer networks, thereby comparing their function mapping capabilities. We then provide apriori criteria, stated in terms of the new variable, that a given Boolean function must satisfy in order to be mapped by a network having one or two layers. These criteria provide necessary, and in the case of a one-layer network, sufficient conditions for samples of a Boolean function to be mapped by a binary neural network. We stress that our model relies crucially on the thresholds being zero. Thus this added restriction pays off well in facilitating analysis.

In general, the network will have L layers, with the lth layer, $1 \leq l \leq L$, comprising of m_l neurons. We will assume, for reasons explained below, that for $1 \leq l \leq L-1$, m_l is odd. The network input is fed to its first layer neurons. For $2 \leq l \leq L-1$, each neuron in the lth layer receives as input the outputs of all the neurons in the $(l-1)$st layer and it feeds its output to every neuron in the $(l+1)$st layer, and so on. The network output is taken from the Lth layer. There are no interconnections between neurons within a layer. The weight vector of the ith neuron, $1 \leq i \leq m_l$, in the lth layer is denoted $\mathbf{w}_i^l$.

Let there be K samples of the Boolean function that are to be mapped, given by

$$\mathbf{X}_k = (X_{k1} \cdots X_{kn}) \mapsto \mathbf{Y}_k = (Y_{k1} \cdots Y_{km}), \quad 1 \leq k \leq K,$$

where $\mathbf{X}_k \in \{-1, 1\}^n$ and $\mathbf{Y}_k \in \{-1, 1\}^m$. When the input to the network is $\mathbf{X}_k$, denote the output of the ith neuron of the lth layer by y_{ki}^l and the collective outputs of the lth layer neurons by $\mathbf{y}_k^l = (y_{k1}^l \cdots y_{km_l}^l)$. For the network to map the kth sample correctly, we must have that $\mathbf{y}_k^L = \mathbf{Y}_k$. This, of course, implies that $m_L = m$.

A brief explanation about the notation above and throughout the paper. We will use boldfaced letters to denote vector quantities. The scalar components of these vector quantities will be denoted using the same letters in plain script, with an extra subscript. Superscripts on these quantities will always denote the layer of the network that the quantity pertains to. The first subscript will index the sample of the Boolean function, and the second will index the neuron under consideration within a particular layer. The exception is weight vectors, which have no subscript indexing the sample since we assume them fixed for all samples.

Proposition I.1: A feedforward neural network with each neuron using a sgn transfer function, operating on a domain of odd dimension, and threshold set to zero, can map only odd functions, i.e., functions f that satisfy $f(-\mathbf{X}) = -f(\mathbf{X})$.

Proof: Since sgn itself is an odd function, and the thresholds are all zero, it follows that the output of an individual neuron in the first layer is an odd function of its input since

$$\text{sgn}(\mathbf{w} \cdot -\mathbf{X}) = -\text{sgn}(\mathbf{w} \cdot \mathbf{X}).$$

The case $\mathbf{w} \cdot \mathbf{X} = 0$ is precluded by the condition that the dimension of the space be odd. Furthermore, m_l is odd for $1 \leq l \leq L-1$ precluding zero activation to neurons in higher layers. Now, since the composition of odd functions remains odd, the network as a whole will have an odd transfer function. □

The conditions of the above proposition should not be considered very constraining if we are willing to work with extended functions as explained below.

Lemma I.2: Any function f defined on $\{-1, 1\}^n$ can be uniquely extended to an odd function, f_o, defined on a domain of odd dimension such that the restriction of f_o to $\{-1, 1\}^n$ agrees with f.

Proof: If n is even, we can embed $\{-1, 1\}^n$ into $\{-1, 1\}^{n+1}$ by the following relation:

$$\begin{aligned} i\colon \{-1, 1\}^n &\hookrightarrow \{-1, 1\}^{n+1} \\ (\mathbf{v}) &\mapsto (\mathbf{v}, 1). \end{aligned}$$

Now we can extend f to f_o as follows. If $\mathbf{v} \in i(\{-1, 1\}^n)$, set $f_o(\mathbf{v}) = f(i^{-1}(\mathbf{v}))$. Else, $i^{-1}(-\mathbf{v})$ is defined and set $f_o(\mathbf{v}) = -f(i^{-1}(-\mathbf{v}))$. Then f_o is odd and its restriction to $\{-1, 1\}^n$, compatible with the map i, agrees with f.

If n is odd (and f is not an odd function), we can repeat the above procedure twice to define f_o on $\{-1, 1\}^{n+2}$. □

Henceforth, we will assume that we are working with functions (extended if necessary) that satisfy the hypothesis of Proposition I.1. Practically, this means we might have to add dummy inputs to the network.

Definition I.3: Define the variables $\mathbf{Z}_{ki}^l$, $1 \leq l \leq L+1$, $1 \leq k \leq K$, $1 \leq i \leq m_L$ by the following relations:

$$\mathbf{Z}_{ki}^1 = Y_{ki}\mathbf{X}_k \tag{I.1}$$

and for $2 \leq l \leq L+1$,

$$\mathbf{Z}_{ki}^l = \left(\text{sgn}(\mathbf{w}_1^{l-1} \cdot \mathbf{Z}_{ki}^{l-1}) \cdots \text{sgn}(\mathbf{w}_{m_{l-1}}^{l-1} \cdot \mathbf{Z}_{ki}^{l-1})\right). \tag{I.2}$$

Proposition I.4: For a network with fixed weights, the following are equivalent:

1) The network performs the mappings $\mathbf{X}_k \mapsto \mathbf{Y}_k$, $1 \leq k \leq K$.
2) The network performs each of the mappings $\mathbf{Z}_{ki}^1 \mapsto (* \cdots 1 \cdots *)$, $1 \leq k \leq K$, $1 \leq i \leq m_L$, where a "$*$" denotes a "don't care" and the one is in the ith position. By this we mean that if $\mathbf{Z}_{ki}^1$ is supplied as the input to the network, the ith neuron of the Lth layer will output a "1."
3) $\mathbf{Z}_{ki}^{L+1} = (* \cdots 1 \cdots *)$, $1 \leq k \leq K$, $1 \leq i \leq m_L$, where again the 1 is in the ith position.

Proof: 1) ⇒ 2) For the first layer neurons, we have

$$y_{ki}^1 = \text{sgn}(\mathbf{w}_i^1 \cdot \mathbf{X}_k), \qquad 1 \leq k \leq K, 1 \leq i \leq m_1. \tag{I.3}$$

For the second layer onwards

$$y_{ki}^l = \text{sgn}\left(\mathbf{w}_i^l \cdot \mathbf{y}_k^{l-1}\right), \quad 2 \leq l \leq L, 1 \leq k \leq K, 1 \leq i \leq m_l. \tag{I.4}$$

In particular, for the Lth layer, we have

$$y_{ki}^L = \text{sgn}\left(\mathbf{w}_i^L \cdot \mathbf{y}_k^{L-1}\right), \qquad 1 \leq k \leq K, 1 \leq i \leq m_L. \tag{I.5}$$

But since the function we are mapping is Boolean, $y_{ki}^L = \pm 1$. Furthermore, sgn is an odd function. Therefore, we can multiply both sides of the above equation by y_{ik}^L to obtain

$$y_{ki}^L y_{ki}^L = y_{ki}^L \text{sgn}\left(\mathbf{w}_i^L \cdot \mathbf{y}_k^{L-1}\right), \qquad 1 \le k \le K,\, 1 \le i \le m_L \tag{I.6}$$

or

$$1 = \text{sgn}\left(\mathbf{w}_i^L \cdot (y_{ki}^L \mathbf{y}_k^{L-1})\right), \qquad 1 \le k \le K,\, 1 \le i \le m_L. \tag{I.7}$$

Since $\mathbf{y}_k^{L-1}$ is, in turn, an odd function of $\mathbf{y}_k^{L-2}$ and so on, we can use (I.4) for $L-1 \ge l \ge 2$ to obtain, for the indexes $L-1 \ge l \ge 2$, $1 \le k \le K$, $1 \le i \le m_L$, $1 \le j \le m_l$

$$y_{ki}^L y_{kj}^l = \text{sgn}\left(\mathbf{w}_j^l \cdot (y_{ki}^L \mathbf{y}_k^{l-1})\right) \tag{I.8}$$

and using (I.3), we obtain for the indexes $1 \le k \le K$, $1 \le i \le m_L$, $1 \le j \le m_1$

$$y_{ki}^L y_{kj}^1 = \text{sgn}\left(\mathbf{w}_j^1 \cdot (y_{ki}^L \mathbf{X}_k)\right). \tag{I.9}$$

The desired result now follows by noting that $(y_{ki}^L \mathbf{X}_k) = \mathbf{Z}_{ki}^1$.

2) ⇒ 3) This is immediate from the statement of *2)* and (I.2).

3) ⇒ 1) We have $\mathbf{Z}_{ki}^{L+1} = (* \cdots 1 \cdots *)$, $1 \le k \le K$, $1 \le i \le m_L$. But using (I.2), this implies that sgn$(\mathbf{w}_i^L \cdot \mathbf{Z}_{ki}^L) = 1$. Multiplying both sides of the above equation by Y_{ki}, we get

$$\text{sgn}\left(\mathbf{w}_i^L \cdot Y_{ki} \mathbf{Z}_{ki}^L\right) = Y_{ki}, \qquad 1 \le k \le K,\, 1 \le i \le m_L. \tag{I.10}$$

Now using (I.2) iteratively to go down $L-1 \ge l \ge 2$, we get, for the indexes $L-1 \ge l \ge 2$, $1 \le k \le K$, $1 \le i \le m_L$

$$Y_{ki} \mathbf{Z}_{ki}^l = \left(\text{sgn}(\mathbf{w}_1^{l-1} \cdot Y_{ki} \mathbf{Z}_{ki}^{l-1}) \cdots \text{sgn}(\mathbf{w}_{m_l}^{l-1} \cdot Y_{ki} \mathbf{Z}_{ki}^{l-1})\right). \tag{I.11}$$

In particular

$$\begin{aligned} Y_{ki} \mathbf{Z}_{ki}^2 &= \left(\text{sgn}(\mathbf{w}_1^1 \cdot Y_{ki} \mathbf{Z}_{ki}^1) \cdots \text{sgn}(\mathbf{w}_{m_l}^1 \cdot Y_{ki} \mathbf{Z}_{ki}^1)\right) \\ &= \left(\text{sgn}(\mathbf{w}_1^1 \cdot \mathbf{X}_k) \cdots \text{sgn}(\mathbf{w}_{m_l}^1 \cdot \mathbf{X}_k)\right) \end{aligned}$$

since $Y_{ki} \mathbf{Z}_{ki}^1 = Y_{ki} Y_{ki} \mathbf{X}_k = \mathbf{X}_k$. But this means that output of the first layer when the input is $\mathbf{X}_k$ is $Y_{ki} \mathbf{Z}_{ki}^2$. Plugging this into (I.2) and iterating for $2 \le l \le L$, we see that the output of the $(L-1)$st layer is $Y_{ki} \mathbf{Z}_{ki}^L$. Now, using (I.10), we get the desired result. □

Some explanation is in order here. Proposition I.4 basically says that in a binary network with zero thresholds, we can replace the two variables $\mathbf{X}$ and $\mathbf{Y}$, for the input and output respectively, by a single "normalized" variable $\mathbf{Z}$. The network maps $\mathbf{X}_k$ to $\mathbf{Y}_k$ iff it maps $\mathbf{Z}_{ki}^1$ to $(* \cdots 1 \cdots *)$. In other words, if we provide the network with an input $\mathbf{Z}_{ki}^1$, the ith neuron of the Lth layer will output a "1," and this statement holds true for $1 \le k \le K$ and $1 \le i \le m_L$. Thus, we can normalize the input–output samples such that the output is always normalized to $+1$ and we are left with only a normalized input $\mathbf{Z}$. More is true: $\mathbf{Z}_{ki}^l$, $2 \le l \le L$ will form the output of the $(l-1)$st layer when the input to the first layer is $\mathbf{Z}_{ki}^1$.

We end this section with a proposition, stated without proof, that we will need later.

Proposition I.5: For $n \ge 1$ and $\mathbf{v} \in \{-1, 1\}^n$, define $S_\mathbf{v} = \{\mathbf{x} \in \{-1, 1\}^n \mid \mathbf{x} \cdot \mathbf{v} > 0\}$. Let $\mathbf{v}_1, \mathbf{v}_2 \in \{-1, 1\}^n$. Clearly, if $S_{\mathbf{v}_1} \bigcap S_{\mathbf{v}_2} \ne \emptyset$, $\mathbf{v}_1 \ne -\mathbf{v}_2$.

II. Mapping Criteria

In this section, we state two theorems that provide necessary and, in the case of a one-layer binary network, sufficient conditions for a set of samples to be mapped by a binary network. We will focus on one– and two-layer binary networks. To understand the difference in mapping capabilities of one– and two-layer binary networks, we shall study their performance over the same set of samples.

The theorem that completely characterizes the Boolean functions that can be mapped by a one-layer binary network is given below.

Theorem II.1: A 1-layer binary network can map K samples $\{\mathbf{X}_k, \mathbf{Y}_k\}_{1 \le k \le K}$ of a Boolean function iff for every i, $1 \le i \le m_1$, $\exists\, \mathbf{w}_i^1$ such that $\{\mathbf{Z}_{ki}^1\}_{1 \le k \le K} \subseteq S_{\mathbf{w}_i^1}$.

Proof: From *2)*, Proposition I.4, we know that mapping the ith component of the output for the kth sample correctly is equivalent to performing the mapping $\mathbf{Z}_{ki}^1 \mapsto (* \cdots 1 \cdots *)$. To correctly map the ith components of the outputs of each of the K samples, we would then need to find a binary weight vector $\mathbf{w}_i^1$ for the ith neuron that satisfies $\mathbf{w}_i^1 \cdot \mathbf{Z}_{ki}^1 > 0$, $1 \le k \le K$. In other words, $\{\mathbf{Z}_{ki}^1\}_{1 \le k \le K} \subseteq S_{\mathbf{w}_i^1}$. To correctly map all the K samples, we would need the above to be true for all the m_1 neurons. □

Notice that Theorem II.1 states the constraint in terms of the single variable $\mathbf{Z}$, instead of two separate input and output variables. The same is true for the theorem that states the constraints placed on a Boolean function by a two-layer network, which is stated below.

Theorem II.2: A two-layer binary network can map K samples $\{\mathbf{X}_k, \mathbf{Y}_k\}_{1 \le k \le K}$ of a Boolean function only if for all i, $1 \le i \le m_2$ and every pair k_1, k_2 where $1 \le k_1, k_2 \le K$, the points $\mathbf{Z}_{k_1 i}^1$, $\mathbf{Z}_{k_2 i}^1$ satisfy

$$\mathbf{Z}_{k_1 i}^1 \ne -\mathbf{Z}_{k_2 i}^1. \tag{II.1}$$

Proof: From *2)*, Proposition I.4, we know that for the samples to be mapped, we need

$$\mathbf{w}_i^2 \cdot \mathbf{Z}_{ki}^2 > 0, \qquad 1 \le k \le K,\, 1 \le i \le m_2. \tag{II.2}$$

Since $\mathbf{w}_i^2 = (w_{i1}^2 \cdots w_{im_1}^2)$, and $\mathbf{Z}_{ki}^2 = (\text{sgn}(\mathbf{w}_1^1 \cdot \mathbf{Z}_{ki}^1) \cdots \text{sgn}(\mathbf{w}_{m_1}^1 \cdot \mathbf{Z}_{ki}^1))$, we can rewrite (II.2) as

$$w_{i1}^2 \text{sgn}(\mathbf{w}_1^1 \cdot \mathbf{Z}_{ki}^1) + \cdots + w_{im_1}^2 \text{sgn}(\mathbf{w}_{m_1}^1 \cdot \mathbf{Z}_{ki}^1) > 0 \tag{II.3}$$

for the indexes $1 \le k \le K$, $1 \le i \le m_2$.

But since $\mathbf{w}_i^2 \in \{-1, 1\}^{m_1}$, and sgn is an odd function, we can rewrite the inequalities for the same indexes as below

$$\text{sgn}\left(\mathbf{w'}_1^1 \cdot \mathbf{Z}_{ki}^1\right) + \cdots + \text{sgn}\left(\mathbf{w'}_{m_1}^1 \cdot \mathbf{Z}_{ki}^1\right) > 0. \tag{II.4}$$

where $\mathbf{w'}_j^1 = -\mathbf{w}_j^1$ if $w_{ij}^2 = -1$ and $\mathbf{w'}_j^1 = \mathbf{w}_j^1$ if $w_{ij}^2 = +1$.

Since each term in the above inequalities is equal to ± 1, at least $\lceil m_1/2 \rceil$ terms must be equal to $+1$ for the inequalities to hold. This forces at least $\lceil m_1/2 \rceil$ dot-products out of $\{\mathbf{w'}_j^1 \cdot \mathbf{Z}_{ki}^1\}_{1 \le j \le m_1}$ to be positive for $1 \le k \le K$, $1 \le i \le m_2$. Thus, at least $\lceil m_1/2 \rceil$ out of $\{\mathbf{w'}_j^1\}_{1 \le j \le m_1}$ lie in $S_{\mathbf{Z}_{ki}^1}$ for $1 \le k \le K$, $1 \le i \le m_2$. Invoking the "pigeonhole principle" of combinatorics now tells us that any two sets $S_{\mathbf{Z}_{k_1 i}^1}$, $S_{\mathbf{Z}_{k_2 i}^1}$, $1 \le k_1, k_2 \le K$, $1 \le i \le m_2$ have a nonempty intersection. Now Proposition I.5 gives us the result. □

Proposition II.3: Let $(\mathbf{X}_k, \mathbf{Y}_k)_{1 \le k \le K}$ be samples of an odd function f [i.e. , $f(-\mathbf{X}) = -f(\mathbf{X})$]. Then for every pair k_1, k_2, where $1 \le k_1, k_2 \le K$, we have that $\mathbf{Z}_{k_1 i}^1 \ne -\mathbf{Z}_{k_2 i}^1$.

Proof: We have constructed the points $\mathbf{Z}_{ki}^1$ by the rule $\mathbf{Z}_{ki}^1 = Y_{ki} \cdot \mathbf{X}_k$, where $Y_{ki} = \pm 1$. Thus for two distinct inputs $\mathbf{X}_{k_1}$ and $\mathbf{X}_{k_2}$, $\mathbf{Z}_{ik_1}^1 = -\mathbf{Z}_{ik_2}^1$ if either $\mathbf{X}_{k_1} = \mathbf{X}_{k_2}$ and $Y_{ik_1} = -Y_{ik_2}$ or $\mathbf{X}_{k_1} = -\mathbf{X}_{k_2}$ and $Y_{ik_1} = Y_{ik_2}$. Clearly the first case is a degenerate one and the only possibility is the second. But that would contradict the hypothesis that f is odd. □

Thus, Theorem II.2 and Proposition II.3 indicate that the constraints placed upon a sample set by a two-layer binary network and deducible using our model are, in some sense, minimal. More precisely, the analysis that we have carried out cannot possibly give us a weaker constraint for implementation by a L-layer network for $L \ge 3$ than it did for $L = 2$, since the latter is already the equivalent of insisting that the function be odd. Furthermore, if $n = 3$, it can be verified that all odd functions can be mapped by a two-layer binary network, so that in that case the constraint of Theorem II.2 is indeed the best possible we can arrive at by any means.

We conclude this section with an example demonstrating the use of the criteria developed in this paper.

Example II.4: Let $n = 3,\ m = 1$ and consider the following function on $\{-1, 1\}^3$:

$$\begin{aligned} \mathbf{X}_1 &= (-1, 1, 1) & \mathbf{X}_2 &= (1, -1, 1) \\ \mathbf{X}_3 &= (1, 1, -1) & \mathbf{X}_4 &= (-1, -1, -1) \\ \mathbf{X}_5 &= (1, -1, -1) & \mathbf{X}_6 &= (-1, 1, -1) \\ \mathbf{X}_7 &= (-1, -1, 1) & \mathbf{X}_8 &= (1, 1, 1) \end{aligned}$$

$$\begin{aligned} \mathbf{Y}_1 = \mathbf{Y}_2 = \mathbf{Y}_3 = \mathbf{Y}_4 &= +1, \\ \mathbf{Y}_5 = \mathbf{Y}_6 = \mathbf{Y}_7 = \mathbf{Y}_8 &= -1. \end{aligned}$$

The values of the $\{\mathbf{Z}_{k1}^1\}_{1 \le k \le 8}$ are

$$\begin{aligned} \mathbf{Z}_{11}^1 = \mathbf{Z}_{51}^1 &= (-1, 1, 1), & \mathbf{Z}_{21}^1 = \mathbf{Z}_{61}^1 &= (1, -1, 1), \\ \mathbf{Z}_{31}^1 = \mathbf{Z}_{71}^1 &= (1, 1, -1), & \mathbf{Z}_{41}^1 = \mathbf{Z}_{81}^1 &= (-1, -1, -1). \end{aligned}$$

The $\mathbf{Z}_{k1}^1$ do not satisfy the conditions of Theorem II.1. To see this, note that the $\mathbf{Z}_{k1}^1$s are not linearly separable from their complement in $\{-1, 1\}^3$. This rules out a one-layer implementation of the Boolean function.

However, the $\mathbf{Z}_{k1}^1$ do satisfy the condition of Theorem II.2, i.e. , they do not contain any set of antipodal elements. Thus, a two-layer implementation is plausible. Using the symmetry of the $\mathbf{Z}_{k1}^1$, we determine that the mapping can be performed using a two-layer binary network with three first-layer neurons and a single second layer neuron with the two following possible sets of weights (and appropriate reorderings thereof)

$$\begin{aligned} \mathbf{w}_1^1 &= (-1, -1, 1) & \mathbf{w}_2^1 &= (1, -1, -1) \\ \mathbf{w}_3^1 &= (-1, 1, -1) & & \\ \mathbf{w}_1^2 &= (1, 1, 1) & \mathbf{w}_1^1 &= (1, 1, -1) \\ \mathbf{w}_2^1 &= (-1, 1, 1) & & \\ \mathbf{w}_3^1 &= (1, -1, 1) & \mathbf{w}_1^2 &= (-1, -1, -1). \end{aligned}$$

There is a geometric interpretation for the placement of the points $\mathbf{Z}_{ki}^1$ for fixed i. For the samples to be mapped by a one-layer network, the $\mathbf{Z}_{ki}^1$ must be linearly separable from their complement on the n-cube. For the two-layer case, the above is the constraint on the $\mathbf{Z}_{ki}^2$, which when translated into a constraint on the $\mathbf{Z}_{ki}^1$ says it is enough that they not have any antipodal elements (a much weaker constraint and one that is satisfied automatically if they satisfy linear separability).

III. Conclusion

In this work, we have shown that networks with weights in $\{-1, 1\}$ and zero thresholds provide the tangible advantage of enabling us to work with a single variable replacing the two variables corresponding to the sample inputs and outputs. This single variable then provides a good comparison of the function mapping capabilities of networks having different number of layers. Using the constraints that a network places upon this single variable, we have clearly dilineated the advantage of adding a second layer over the first. The constraints placed on the variable have a geometric interpretation in terms of dichotomies of the hypercube, which is noted. Furthermore, it is shown that within the model developed, the effect of adding a third layer could not possibly weaken the constraints that were placed by the second layer.

Acknowledgment

The author thanks P. G. Poonacha for several useful comments.

References

[1] V. Deolalikar, "New approaches to learning and classification in feedforward neural networks," M.Tech. thesis, Indian Institute of Technology, Bombay, July 1994.
[2] C. Ji and D. Psaltis, "Capacity of two-layer feedforward neural networks with binary weights," *IEEE Trans. Inform. Theory*, vol. 44, pp. 256–268, Jan. 1998.
[3] C.-L. J. Hu, "A novel, geometric, supervised learning scheme using linear programming approach," in *IEEE INNS Int. Joint Conf. Neural Networks*, vol. 3, June 1990, pp. 305–308.
[4] E. Mayoraz and F. Aviolat, "Constructive training methods for feedforward neural networks with binary weights," *Int. J. Neural Syst.*, vol. 7, no. 2, pp. 149–166, May 1996.
[5] R. Shonkwiler, "Separating the vertices of N-cubes by hyperplanes and its application to artificial neural networks," *IEEE Trans. Neural Networks*, vol. 4, pp. 343–347, Mar. 1993.
[6] H. Suyari and I. Matsuba, "New information theoretical approach to the storage capacity of neural networks with binary weights," in *Proc. 9th Int. Conf. Artificial Neural Networks: ICANN'99*, vol. 1, Sept. 1999, pp. 431–436.
[7] A. Weiland and R. Leighton, "Geometric analysis of network capabilities," in *IEEE Int. Conf. Neural Networks*, vol. 3, June 1987, pp. 385–392.

Comments on "Classification Ability of Single Hidden Layer Feedforward Neural Networks"

Irwin W. Sandberg

***Abstract*—A recent paper addresses a certain classification problem, and concludes that classification can be achieved using a single hidden layer neural network. We note here that conclusions along similar lines in a more general setting were reached in an earlier paper.**

The problem of classifying signals arises in several application areas. Typically we are given a finite number m of pairwise disjoint subsets $C_1, \ldots, C_m$ of some set S of signals, and we would like to synthesize a system that maps the elements of each C_j into a real number a_j, such that the numbers $a_1, \ldots, a_m$ are distinct. A recent paper[1] addresses this interesting problem in the setting of $S = \mathbb{R}^n$, and concludes that under certain assumptions this classification can be achieved using a single hidden layer neural network.

However, conclusions along similar lines in a more general setting were reached in the earlier paper [1],[2] where S is an arbitrary real normed linear space. It was found that classification can always be achieved using a bank of linear functionals feeding a static network terminated in a quantizer, under just the assumption that the disjoint C_j are compact. For the case in which $S = \mathbb{R}^n$, some information

Manuscript received November 1, 2000; revised January 10, 2001.
The author is with the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712 USA(e-mail: sandberg@ece.utexas.edu).
Publisher Item Identifier S 1045-9227(01)03352-5.

[1] G. Huang, Y. Chen, and H. A. Babri, "Classification ability of single hidden layer feedforward neural networks," *IEEE Trans. Neural Networks*, vol. 11, pp. 799–801, May 2000.

[2] A slightly updated version of this material appears in [2, Ch. 2].