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

Vinay Deolalikar
Information Theory Research Group
HP Laboratories Palo Alto
HPL-2001-64 (R.1)
July 30th , 2001*

E-mail: vinayd@exch.hpl.hp.com

Binary Neural Networks, Boolean Function Mapping, 1-layer Networks, 2-layer Networks

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 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 1-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 2-layer network are, in some sense, minimal.

* Internal Accession Date Only Approved for External Publication

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

VINAY DEOLALIKAR
Information Theory Research Group
Hewlett-Packard Laboratories
1501 Page Mill Road
Palo Alto, CA 94304
Email: vinayd@exch.hpl.hp.com
Tel: (650) 857 8605 Fax: (650) 852 3791

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 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 1-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 2-layer network are, in some sense, minimal.

Keywords

Binary Neural Networks, Boolean Function Mapping, 1-layer Networks, 2-layer Networks.

1 Preliminaries

The classical *feedforward* neural network architecture comprises layers of neurons. Each neuron has a pair $(\mathbf{w}, t)$ associated to it, 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 $sgn(\mathbf{w} \cdot \mathbf{x} - t)$ 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. Firstly, it is a matter of theoretical curiosity whether networks comprising neurons with such restrictions on their weights have reasonable function mapping capabilities. Secondly, as is noted in [1, 2, 4, 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 1 and 2-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 1 and 2-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 1-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 l^{th} 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 l^{th} 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 L^{th} layer. There are no interconnections between neurons within a layer. The weight vector of the i^{th} neuron, $1 \leq i \leq m_l$, in the l^{th} layer is denoted $\mathbf{w}_i^l$.

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

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

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 i^{th} neuron of the l^{th} layer by y_{ki}^l and the collective outputs of the l^{th} layer neurons by $\mathbf{y}_k^l = (y_{k1}^l \ldots y_{km_l}^l)$. For the network to map

the k^{th} 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 1.1 *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

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

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 1.2 *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

$$\begin{array}{rccc} i : \{-1,1\}^n & \hookrightarrow & \{-1,1\}^{n+1} \\ (\mathbf{v}) & \mapsto & (\mathbf{v}, 1) \end{array}$$

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 Prop. 1.1. Practically, this means we might have to add dummy inputs to the network.

Definition 1.3 *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*

$$\mathbf{Z}_{ki}^{1} = Y_{ki}\mathbf{X}_k, \tag{1.1}$$

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

$$\mathbf{Z}_{ki}^{l} = (sgn(\mathbf{w}_1^{l-1} \cdot \mathbf{Z}_{ki}^{l-1}) \dots sgn(\mathbf{w}_{m_{l-1}}^{l-1} \cdot \mathbf{Z}_{ki}^{l-1})). \tag{1.2}$$

Proposition 1.4 *For a network with fixed weights, the following are equivalent:*

(i). The network performs the mappings $\mathbf{X}_k \mapsto \mathbf{Y}_k, 1 \leq k \leq K$.

(ii). The network performs each of the mappings $\mathbf{Z}_{ki}^{1} \mapsto (* \dots 1 \dots *), 1 \leq k \leq K, 1 \leq i \leq m_L$, *where a "* $*$ *" denotes a 'don't care' and the* 1 *is in the* i^{th} *position. By this we mean that if* $\mathbf{Z}_{ki}^{1}$ *is supplied as the input to the network, the* i^{th} *neuron of the* L^{th} *layer will output a '1.'*

(iii). $\mathbf{Z}_{ki}^{L+1} = (* \dots 1 \dots *), 1 \leq k \leq K, 1 \leq i \leq m_L$, *where again the* 1 *is in the* i^{th} *position.*

Proof. $(i) \Rightarrow (ii)$ For the first layer neurons, we have

$$y_{ki}^{1} = sgn(\mathbf{w}_i^{1} \cdot \mathbf{X}_k), 1 \leq k \leq K, 1 \leq i \leq m_1. \tag{1.3}$$

For the second layer onwards,

$$y_{ki}^{l} = sgn(\mathbf{w}_i^{l} \cdot \mathbf{y}_k^{l-1}), 2 \leq l \leq L, 1 \leq k \leq K, 1 \leq i \leq m_l. \tag{1.4}$$

In particular, for the L^{th} layer, we have

$$y_{ki}^{L} = sgn(\mathbf{w}_i^{L} \cdot \mathbf{y}_k^{L-1}), 1 \leq k \leq K, 1 \leq i \leq m_L. \tag{1.5}$$

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

$$y_{ki}^{L} y_{ki}^{L} = y_{ki}^{L} sgn(\mathbf{w}_i^{L} \cdot \mathbf{y}_k^{L-1}), 1 \leq k \leq K, 1 \leq i \leq m_L, \tag{1.6}$$

or,

$$1 = sgn(\mathbf{w}_i^{L} \cdot (y_{ki}^{L} \mathbf{y}_k^{L-1})), 1 \leq k \leq K, 1 \leq i \leq m_L. \tag{1.7}$$

Since $\mathbf{y}_k^{L-1}$ is, in turn, an odd function of $\mathbf{y}_k^{L-2}$ and so on, we can use Equation 1.4 for $L-1 \geq l \geq 2$ to obtain, for the indices $L-1 \geq l \geq 2, 1 \leq k \leq K, 1 \leq i \leq m_L, 1 \leq j \leq m_l$,

$$y_{ki}^{L} y_{kj}^{l} = sgn(\mathbf{w}_j^{l} \cdot (y_{ki}^{L} \mathbf{y}_k^{l-1})), \tag{1.8}$$

and using Equation 1.3, we obtain for the indices $1 \leq k \leq K, 1 \leq i \leq m_L, 1 \leq j \leq m_1$,

$$y_{ki}^{L} y_{kj}^{1} = sgn(\mathbf{w}_j^{1} \cdot (y_{ki}^{L} \mathbf{X}_k)). \tag{1.9}$$

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

$(ii) \Rightarrow (iii)$ This is immediate from the statement of (ii) and Equation 1.2.

$(iii) \Rightarrow (i)$ We have $\mathbf{Z}_{ki}^{L+1} = (* \dots 1 \dots *), 1 \leq k \leq K, 1 \leq i \leq m_L$. But using Equation 1.2, 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

$$sgn(\mathbf{w}_i^L \cdot Y_{ki} \mathbf{Z}_{ki}^L) = Y_{ki}, 1 \leq k \leq K, 1 \leq i \leq m_L. \tag{1.10}$$

Now using Equation 1.2 iteratively to go down $L - 1 \geq l \geq 2$, we get, for the indices $L - 1 \geq l \geq 2, 1 \leq k \leq K, 1 \leq i \leq m_L$,

$$Y_{ki} \mathbf{Z}_{ki}^l = (sgn(\mathbf{w}_1^{l-1} \cdot Y_{ki} \mathbf{Z}_{ki}^{l-1}) \dots sgn(\mathbf{w}_{m_l}^{l-1} \cdot Y_{ki} \mathbf{Z}_{ki}^{l-1})). \tag{1.11}$$

In particular,

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

since $Y_{ki} \mathbf{Z}_{ki}^1 = Y_{ki} Y_{ki} \mathbf{X}_k = \mathbf{X}_k$. But this means that that output of the first layer when the input is $\mathbf{X}_k$ is $Y_{ki} \mathbf{Z}_{ki}^2$. Plugging this into Equation 1.2 and iterating for $2 \leq l \leq L$, we see that the output of the $(L-1)^{st}$ layer is $Y_{ki} \mathbf{Z}_{ki}^L$. Now, using Equation 1.10, we get the desired result. □

Some explanation is in order here. Proposition 1.4 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 $(* \dots 1 \dots *)$. In other words, if we provide the network with an input $\mathbf{Z}_{ki}^1$, the i^{th} neuron of the L^{th} layer will output a '1,' and this statement holds true for $1 \leq k \leq K$ and $1 \leq i \leq m_L$. 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: $\mathbf{Z}_{ki}^l, 2 \leq l \leq L$ 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 1.5 *For $n \geq 1$ 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 $S_{\mathbf{v}_1} \bigcap S_{\mathbf{v}_2} \neq \emptyset$, $\mathbf{v}_1 \neq -\mathbf{v}_2$.*

2 Mapping Criteria

In this section, we state two theorems that provide necessary and, in the case of a 1-layer binary network, sufficient conditions for a set of samples to be mapped by a binary network. We will focus on 1 and 2-layer binary networks. To understand the difference in mapping capabilities of 1 and 2-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 1-layer binary network is given below.

Theorem 2.1 *A 1-layer binary network can map K samples $\{\mathbf{X}_k, \mathbf{Y}_k\}_{1 \leq k \leq K}$ of a Boolean function iff for every $i, 1 \leq i \leq m_1, \exists\, \mathbf{w}_i^1$ such that $\{\mathbf{Z}_{ki}^1\}_{1 \leq k \leq K} \subseteq S_{\mathbf{w}_i^1}$.*

Proof. From (ii), Proposition 1.4, we know that mapping the i^{th} component of the output for the k^{th} sample correctly is equivalent to performing the mapping $\mathbf{Z}_{ki}^1 \mapsto (* \ldots 1 \ldots *)$. To correctly map the i^{th} 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 i^{th} neuron that satisfies $\mathbf{w}_i^1 \cdot \mathbf{Z}_{ki}^1 > 0, 1 \leq k \leq K$. In other words, $\{\mathbf{Z}_{ki}^1\}_{1 \leq k \leq K} \subseteq S_{\mathbf{w}_i^1}$. To correctly map all the K samples, we would need the above to be true for all the m_1 neurons. □

Notice that Theorem 2.1 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 2-layer network, which is stated below.

Theorem 2.2 *A 2-layer binary network can map K samples $\{\mathbf{X}_k, \mathbf{Y}_k\}_{1 \leq k \leq K}$ of a Boolean function only if for all $i, 1 \leq i \leq m_2$ and every pair k_1, k_2 where $1 \leq k_1, k_2 \leq K$, the points $\mathbf{Z}_{k_1 i}^1, \mathbf{Z}_{k_2 i}^1$ satisfy*

$$\mathbf{Z}_{k_1 i}^1 \neq -\mathbf{Z}_{k_2 i}^1. \tag{2.1}$$

Proof. From (ii), Proposition 1.4, we know that for the samples to be mapped, we need

$$\mathbf{w}_i^2 \cdot \mathbf{Z}_{ki}^2 > 0, 1 \leq k \leq K, 1 \leq i \leq m_2. \tag{2.2}$$

Since $\mathbf{w}_i^2 = (w_{i1}^2 \ldots w_{im_1}^2)$, and $\mathbf{Z}_{ki}^2 = (sgn(\mathbf{w}_1^1 \cdot \mathbf{Z}_{ki}^1) \ldots sgn(\mathbf{w}_{m_1}^1 \cdot \mathbf{Z}_{ki}^1))$, we can rewrite Equation 2.2 as

$$w_{i1}^2 sgn(\mathbf{w}_1^1 \cdot \mathbf{Z}_{ki}^1) + \ldots + w_{im_1}^2 sgn(\mathbf{w}_{m_1}^1 \cdot \mathbf{Z}_{ki}^1) > 0, \tag{2.3}$$

for the indices $1 \leq k \leq K, 1 \leq i \leq m_2$.

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 indices as below

$$sgn(\mathbf{w'}_1^1 \cdot \mathbf{Z}_{ki}^1) + \ldots + sgn(\mathbf{w'}_{m_1}^1 \cdot \mathbf{Z}_{ki}^1) > 0. \tag{2.4}$$

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 $\{\mathbf{w'}_j^1 \cdot \mathbf{Z}_{ki}^1\}_{1 \leq j \leq m_1}$ to be positive for $1 \leq k \leq K, 1 \leq i \leq m_2$. Thus, at least $\lceil m_1/2 \rceil$ out of $\{\mathbf{w'}_j^1\}_{1 \leq j \leq m_1}$ lie in $S_{\mathbf{Z}_{ki}^1}$ for $1 \leq k \leq K, 1 \leq i \leq m_2$. Invoking the "pigeonhole principle" of combinatorics now tells us that any two sets $S_{\mathbf{Z}_{k_1 i}^1}, S_{\mathbf{Z}_{k_2 i}^1}, 1 \leq k_1, k_2 \leq K, 1 \leq i \leq m_2$ have a non-empty intersection. Now Proposition 1.5 gives us the result. □

Proposition 2.3 *Let $(\mathbf{X}_k, \mathbf{Y}_k)_{1 \leq k \leq K}$ be samples of an odd function f (*i.e.*, $f(-\mathbf{X}) = -f(\mathbf{X})$). Then for every pair k_1, k_2, where $1 \leq k_1, k_2 \leq K$, we have that $\mathbf{Z}_{k_1 i}^1 \neq -\mathbf{Z}_{k_2 i}^1$.*

Proof. We have constructed the points $\mathbf{Z}^1_{ki}$ by the rule $\mathbf{Z}^1_{ki} = Y_{ki} \cdot \mathbf{X}_k$, where $Y_{ki} = \pm 1$. Thus for two distinct inputs $\mathbf{X}_{k_1}$ and $\mathbf{X}_{k_2}$, $\mathbf{Z}^1_{ik_1} = -\mathbf{Z}^1_{ik_2}$ 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 2.2 and Proposition 2.3 indicate that the constraints placed upon a sample set by a 2-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 $L \geq 3$ 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 2-layer binary network, so that in that case the constraint of Theorem 2.2 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 2.4 Let $n = 3, m = 1$ and consider the following function on $\{-1, 1\}^3$:

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

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

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

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

The $\mathbf{Z}^1_{k1}$ do not satisfy the conditions of Theorem 2.1. To see this, note that the $\mathbf{Z}^1_{k1}$'s are not linearly separable from their complement in $\{-1, 1\}^3$. This rules out a 1-layer implementation of the Boolean function.

However, the $\mathbf{Z}^1_{k1}$ do satisfy the condition of Theorem 2.2, *i.e.*, they do not contain any set of antipodal elements. Thus, a 2-layer implementation is plausible. Using the symmetry of the $\mathbf{Z}^1_{k1}$, we determine that the mapping can be performed using a 2-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) -

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

There is a geometric interpretation for the placement of the points $\mathbf{Z}^1_{ki}$ for fixed i. For the samples to be mapped by a 1-layer network, the $\mathbf{Z}^1_{ki}$ must be linearly separable from their complement on the n-cube. For the 2-layer case, the above is the constraint on the $\mathbf{Z}^2_{ki}$, 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).

3 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.

Acknowledgements

The author thanks P. G. Poonacha for several useful comments and Lisa Fleming for her help in preparing this manuscript.

References

[1] V. Deolalikar, *New approaches to learning and classification in feedforward neural networks*, M. Tech. Dissertation, 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. Info. Theory*, vol. IT-44, no. 1, pp. 256-268, Jan. 1998.

[3] C.-L.J Hu, "A novel, Geometric, Supervised learning scheme using Linear Programming approach," *IEEE INNS Int. Joint Conf. on Neural Networks*, vol. 3, pp. 305-308, June 1990.

[4] E. Mayoraz, and F. Aviolat, "Constructive training methods for feedforward neural networks with binary weights," *Int. J. of Neural Systems*, vol.7, no.2, pp. 149-66, 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. TNN-4, no. 2, pp. 343-347, March 1993.

[6] H. Suyari, and I. Matsuba, "New information theoretical approach to the storage capacity of neural networks with binary weights," *Proc.* 9^{th} *Int. Conf. on Artificial Neural Networks: ICANN '99*, vol. 1, pp. 431-436, Sept. 1999.

[7] A. Weiland and R. Leighton, "Geometric analysis of network capabilities," *IEEE Int. Conf. on Neural Networks*, vol. 3, pp. 385-392, June 1987.