A new algorithm for learning representations in Boolean neural networks

N. N. Biswas and R. Kumar
Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore 560 012, India

A new algorithm for learning representations in a multilayer Boolean neural networks is presented. The algorithm depends on a theorem which states that any nonlinearly separable Boolean function can be expressed as a convergent series of linearly separable functions connected by the logical OR (+) and logical INHIBIT (−) operators. The formation of the series is accomplished by the implied minterm structure of linearly separable functions. The algorithm produces the representation much faster than the back propagation, and unlike the latter does not encounter the problem of local minima.

THE back propagation (BP) algorithm developed by Rumelhart *et al.*[1] suggests a new way to explore good representations. It is also the most extensively used learning algorithm for complex, multilayer systems. Among many networks where back propagation has been successfully applied, there is a class of networks where the input and output vectors are strings of binary bits, 0 and 1. In such networks, called Boolean neural networks by us, the representation obtained by the BP algorithm becomes a logic circuit implemented by the familiar threshold gates[2,3] working as hidden and output units. In the terminology of logical design and switching theory of computer science and engineering, the output vector becomes the output Boolean function, and the input state vectors which are binary combinations of input variables are known as fundamental products or minterms. The output Boolean function can, therefore, be depicted in a truth table or may be expressed as a sum of minterms[3,4]. Boolean functions which can be realized by a single threshold gate are linearly separable (LS) functions[2,3]. However, quite often the desired output function is not LS. In such a case the output function can be expressed as a convergent series of LS functions, given by a new theorem stated and proved in this paper. Based on this theorem and the implied minterm structure (IMS)[5] of LS functions, a new algorithm for learning good representations of Boolean neural networks is being presented in this paper. As we have discussed later, the algorithm does not have many shortcomings of the BP, and promises to provide a powerful alternative to BP for the Boolean neural networks.

While expounding the BP algorithm, Rumelhart *et al.*[1] have discussed in their paper seven problems, namely, XOR, parity, encoding, symmetry, addition, negation, and the T–C problem. In all these except the T–C problem, the learning network can be straightaway classified as Boolean. There are many other applications of learning, where the information from the outside world are encoded into binary bits by the input units of the learning network. Hence, Boolean networks, although a subclass of all types of neural networks, form undoutedly the most predominant and significant subclass. Therefore, an algorithm without many shortcomings of the BP although applicable only to Boolean learning networks, has the promise of being a most extensively used algorithm.

Implied minterm structure

In order to introduce the implied minterm structure to the reader, we first define a few basic terms.

Definition 1

Given a Boolean function, the *ON set* is the set of minterms whose output is desired to be 1. Minterms belonging to the ON set will be called *on* minterms.

Definition 2

Given a Boolean function, the *OFF set* is the set of minterms whose output is desired to be 0. Minterms belonging to the OFF set will be called *off* minterms.

Definition 3

The *tabular form* of a given function is the ON set minterms written in their binary form. It may also be considered as a matrix where each row represents a minterm whose output is desired to be 1, and each column is headed by a variable. For example, the tabular form of the 5-variable function

$$F(x_1, x_2, x_3, x_4, x_5) = \Sigma\,(0, 4, 10, 14, 17, 21, 27, 31)$$

can be seen in Table 1*a* of the next section.

Definition 4

The number of 1s in the binary representation of a minterm is called its *weight*.

Definition 5

A Boolean function is called *positive* and *ordered*, if in every column of the tabular form of the given function, the number of 1s is not less than the number of 0s, and the ratios of the number of 1s to that of 0s are such that if r_i and r_j are these ratios of the ith and the jth columns respectively, then $r_i \geq r_j$ when $i<j$.

Any arbitrary Boolean function can be made positive and ordered by suitable permutations and/or complementations of its columns.

In a LS function, it has been shown that if a minterm m_i is realized by a set of weights and threshold, then there are minterms which will also be realized by the same set. If m_j is one such minterm, then m_i is said to imply m_j.

Definition 6

Let m_1 and m_2 be two minterms such that

$$m_1 = b_1 b_2 \ldots 0 \ldots b_n$$
$$m_2 = b_1 b_2 \ldots 1 \ldots b_n,$$

then m_2 is said to have been obtained from m_1 by *positive complementation.*

Such a positive complementation when applied to the least significant bit (b_n in our notation) of the minterm is called *elementary positive complementation.* Note that the minterm m_2, obtained from m_1 by elementary positive complimentation has a weight $w+1$, where w is the weight of m_1.

Definition 7

Let m_1 and m_2 be two minterms such that

$$m_1 = b_1 b_2 \ldots 0 \ldots 1 \ldots b_n \text{ and}$$
$$m_2 = b_1 b_2 \ldots 1 \ldots 0 \ldots b_n.$$

Then m_2 is said to have been obtained from m_1 by *positive permutation.* Note that in positive permutation, a 1 is shifted to the left and a 0 to the right, and the weight of m_2 is the same as that of m_1.

Positive permutation when applied to adjacent bits of a minterm is called a *unit positive permutation.*

It will now be obvious that in a positive and ordered function, if the minterm m_j has been obtained by unit positive permutations on m_i, and m_k has been obtained by elementary positive complementations on m_i, then m_i implies m_j and m_k. Also note that the implication relation is transitive, that is, if m_i implies m_j, and m_j implies m_k, then m_i implies m_k.

The IMS of 4-variable minterms is shown in Figure 1. The IMS is a graphical structure showing implications between the minterms. In this structure if m_i implies m_j by virtue of m_j being obtained from m_i by either elementary positive complementation or unit positive permutation, then m_j appears below or on the right of m_i (or sometimes in an angle in between these two directions).

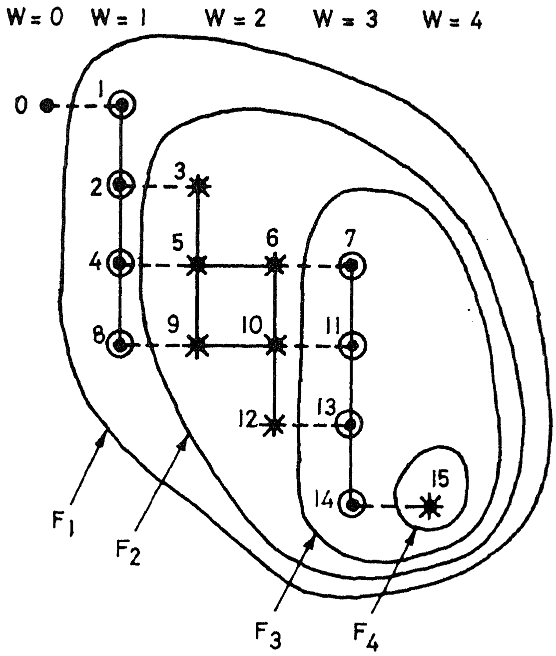

Figure 1. 4-Variable implied minterm structure, with on and star minterms of 4-variable odd parity function plotted on it. Also shown are nested compact zones of F_1, F_2, F_3, and F_4.

Solid lines (in Figure 1) are used to depict the implications due to unit positive permutations between two minterms of the same weight; and the dotted lines are used to show the implications due to elementary positive complementations, where implying minterm m_i has a weight w and the implied minterm m_j has a weight $w+1$.

To test the linear separability of a Boolean function, first make the function positive and ordered, and then plot the ON set minterms in the IMS structure. If the IMS now reveals at least one minterm that does not belong to the ON set of the function but is implied by one of the on-minterms, then the IMS of that function is said to have a *hole* in it and is therefore not compact. In this case the function is not LS. Hence, for a function to be linearly separable, its on-minterms must form a compact zone in the IMS, which may be defined as follows.

Definition 8

If a set of minterms in the IMS, whose every member has all (may be one or more) minterms implied by it also within the set, then the set of minterms is said to produce a *compact zone.*

The algorithm

The algorithm consists essentially of four procedures namely, TRANSFORM, ON-STAR, REALIZE and RESTORE.

We now solve two problems to explain the above procedures. Using the 4-variable odd-parity problem, we shall describe the procedures ON-STAR and REALIZE. Using the 5-variable mirror symmetry problem, we shall describe the procedures TRANSFORM and RESTORE.

4-Variable odd-parity problem

In this problem, the network learns to detect odd parity in bit strings of length 4. It can be seen that such strings of 4-variable minterms are 0001, 0010, 0100, 0111, 1000, 1011, 1101 and 1110. These have the decimal designations, 1, 2, 4, 7, 8, 11, 13 and 14. Hence the 4-variable odd parity Boolean function F_{OP} is given by the following sum-of-minterm form.

$$F_{OP} = \Sigma\,(1, 2, 4, 7, 8, 11, 13, 14).$$

It can be verified that the tabular form of this function is already positive and ordered. Now, plot the *on-minterms* of the function on the 4-variable IMS by circling the minterms (Figure 1). It is seen that the plotted on-minterms alone fail to produce a compact zone in the IMS. Hence, the function is not LS[5]. Therefore, to obtain an LS function we identify a compact zone which includes not only all on-minterms, but also some off-minterms. This produces the first compact zone, and represents an LS function F_1. The off-minterms in the first compact zone will be called *star-minterms* (shown by stars in Figure 1). Next we identify the second compact zone which is nested in the first compact zone. It must compulsorily include all star-minterms, and if need be, some on-minterms to make the zone compact. The second compact zone corresponds to the second LS function, F_2. The third compact zone (if any) will be nested in the second compact zone, and will compulsorily include all on-minterms within the second compact zone, and if need be, some star-minterms to avoid any hole. This produces another LS function, F_3. This process continues until the *n*th compact zone, which is a homogeneous zone (containing either only on or only star but not both types of minterms) is detected. In this function, the procedure terminates with the identification of the fourth LS function, F_4. It can now be easily verified that $F_{OP} = F_1 - F_2 + F_3 - F_4$, where $-$ and $+$ are the logical INHIBIT and logical OR operators. The identification of the on and star minterms, and the compact zones is done by the ON-STAR procedure of the algorithm. The procedure REALIZE now finds the set of weights and thresholds of the identified LS functions. The procedure achieves this by several iterations, based on the principle of Dertouzos' vector[6].

For the 4-variable odd parity function, the weights of the four functions (see Figure 1), F_1 to F_4, turn out to be identical (namely 1, 1, 1, 1) and their thresholds are 0.5, 1.5, 2.5 and 3.5 respectively. Since, the functions F_1 and F_3 are excitory, they are connected to the output unit by weight 1. F_2 and F_4 are inhibitory and, therefore, are connected to the output unit by weight −1. The threshold of the output unit is always 0.5. The representation learnt by the network is shown in Figure 2. This representation happens to be identical to that of Rumelhart *et al.*[1]

We can now state and prove a new and significant theorem which is at the heart of this algorithm. We shall call this the linearization theorem, since it converts a non-linearly separable function into a series of linearly separable functions.

THEOREM 1

(*The Linearization Theorem*) *A non-linearly separable function F can always be expressed as a convergent series of LS functions* $F_1, F_2, \ldots, F_n$ *connected by the logical OR* (+) *and the logical INHIBIT* (−) *operators, such that,*

$$F_1 > F_2 > \ldots F_n, \text{ and}$$
$$F = F_1 - F_2 + \ldots + F_n \text{ when } n \text{ is odd, and}$$
$$F = F_1 - F_2 + \ldots - F_n \text{ when } n \text{ is even.}$$

Proof: Without any loss of generality, we shall prove the theorem from the working of the algorithm while solving the 4-variable odd parity problem. It can be seen that the minterms of the first compact zone constitute the first LS function F_1. It consists of the on-

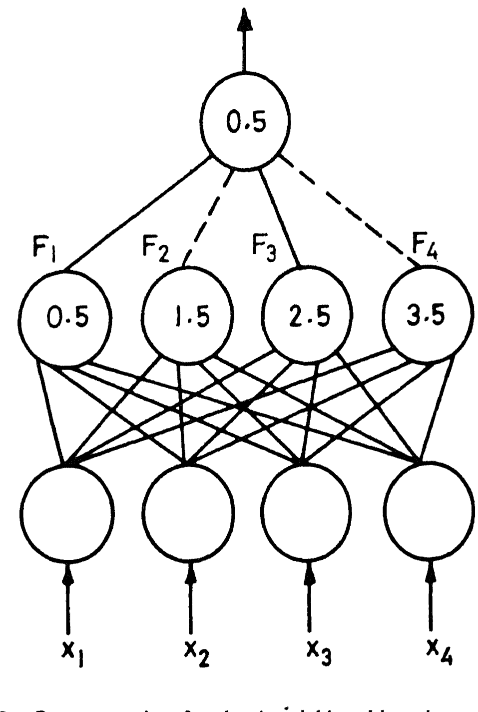

Figure 2. Representation for the 4-variable odd parity problem as obtained by the IMS algorithm. Solid lines show connections with positive weight, 1, and dotted lines connections with negative weight, −1. Gates are normally off, with output 0, and gets turned on producing an output 1, when the weighted sum exceeds the threshold shown in the circle.

minterms and the star-minterms. It is now definite that whatever may be the distribution of the on and star minterms in the first compact zone, the on and star minterms constituting the second compact zone must be at least one on-minterm less than those of the first compact zone. Thus the set of minterms constituting the second compact zone will always be a proper subset of the set of minterms constituting the first compact zone. Hence $F_1 > F_2$. This will also be true for F_2 and F_3, and so on. Hence, generalizing $F_i > F_{i+1}$. It also follows from the algorithm, that the minterms of F_2 must be subtracted and those of F_3 must be added. In general, for all $i > 1$, F_i must be subtracted if i is even and F_i must be added if i is odd. The relation $F_i > F_{i+1}$ also proves that the series is convergent. QED

While solving the 4-variable parity problem, we explained the procedures ON-STAR and REALIZE. Let us now solve the second problem.

5-Variable mirror symmetry problem

In this problem the network learns to detect symmetry in a bit string of length 5. It can be seen that minterms, 00000(0), 00100(4), 01010(10), 01110(14), 10001(17), 10101(21), 11011(27) and 11111(31) have symmetrical bit patterns. In the sum-of-minterm form the 5-variable mirror-symmetry function can, therefore be written as

$$F_{MS}(x_1, x_2, x_3, x_4, x_5) = \Sigma\,(0, 4, 10, 14, 17, 21, 27, 31).$$

If this function is now plotted on the 5-variable IMS, and the compact zones identified, then we get 11 LS functions of theorem 1 as shown by the nested expression given below.

(0(1–3(4(5–9, 16(10, 17(11–13, 18–20, 24(14, 21
(15, 22, 23, 25, 26, 28(27(29, 30(31))))))))))

where the notation 1–3 means 1, 2, 3.

This will result in a representation having 11 hidden units. Although such a representation is correct, it is not economical. A significant reduction in hidden units can be achieved with the help of the procedure TRANSFORM. This procedure finds a permutation and/or complementation of the columns of the tabular form of the function, such that the function not only becomes positive and ordered, but also in its tabular form, the least minterm of the tabular form has the highest decimal designation. For example, the tabular form of the 5-variable F_{MS} function is shown in Table 1*a*. Table 1*b* shows the transform $(x_4, x_2', x_1, x_5', x_3)$, wherein the columns x_2 and x_5 have been complemented and the columns x_4, x_2', x_1, x_5' and x_3 become the 1st, 2nd. 3rd, 4th and 5th columns of Table 1*b*. The 'orm also changes the decimal designations of the Jote that the least minterm of Table 1*a* is 0, of Table 1*b* is 10. This is the highest of the least minterm that may be obtained.

Table 1

(a)						(b)					
m_i	x_1	x_2	x_3	x_4	x_5	m_i	x_4	x_2'	x_1	x_5'	x_3
0	0	0	0	0	0	10	0	1	0	1	0
4	0	0	1	0	0	11	0	1	0	1	1
10	0	1	0	1	0	18	1	0	0	1	0
14	0	1	1	1	0	19	1	0	0	1	1
17	1	0	0	0	1	12	0	1	1	0	0
21	1	0	1	0	1	13	0	1	1	0	1
27	1	1	0	1	1	20	1	0	1	0	0
31	1	1	1	1	1	21	1	0	1	0	1
N(1)	4	4	4	4	4	N(1)	4	4	4	4	4
N(0)	4	4	4	4	4	N(0)	4	4	4	4	4

The minterms of Table 1*b* are now plotted on the 5-variable IMS, and the on and star minterms identified (Figure 3). It is interesting to note that now only two compact zones are found, corresponding to only two functions I_1 and I_2 of theorem 1. The REALIZE procedure now determines the weights of these two functions which turn out to be (2, 2, 1, 1, 0) for both I_1 and I_2. The RESTORE procedure now assigns these weights with proper signs to the input variables. Since in the transformed columns, the first column is headed by x_4, the first weight 2 is assigned to x_4. The second column is headed by x_2'. Therefore, the second weight with a negative sign is assigned to x_2. This way the weight vector (2, 2, 1, 1, 0) of the transformed variable $(x_4, x_2', x_1, x_5', x_3)$ becomes the weight vector (1, −2, 0, 2, −1) for the original variables $(x_1, x_2, x_3, x_4, x_5)$. Both the functions I_1 and I_2 have these weights. Thresholds are now computed with these weights.

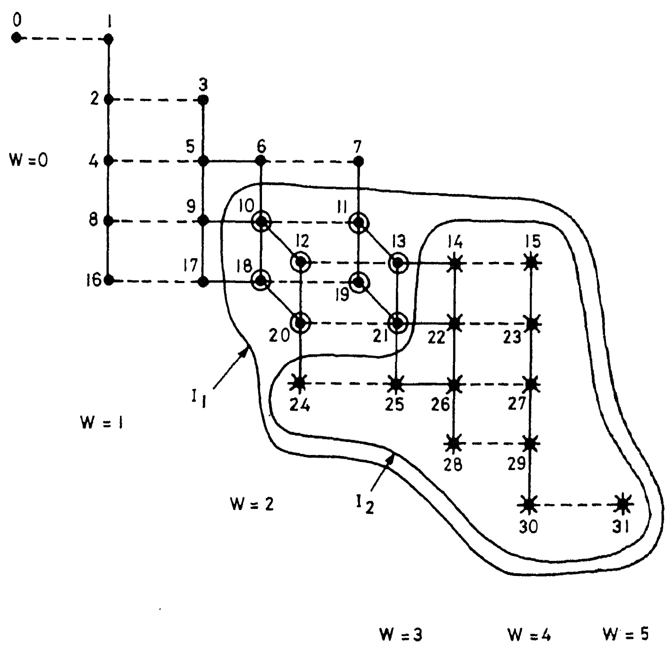

Figure 3. 5-Variable implied minterm structure, with on and star minterms of the 5-variable mirror-symmetry function plotted on it. Also shown are the nested compact zones of I_1 and I_2.