第 29 卷 第 2 期
2003 年 3 月

自 动 化 学 报
ACTA AUTOMATICA SINICA

Vol. 29, No. 2
Mar., 2003

A General Judging and Constructing Method of SP Functions in Binary Neural Networks[1)]

LU Yang HAN Jiang-Hong WEI Zhen
(*Computer and Information College, Hefei University of Technology, Hefei* 230009)
(E-mail: luyang.gocom@163.com)

Abstract SP functions form a linearly separable series with a clear logical meaning and the set of PSP functions is only a special subset of SP functions. In this paper, some expression problems of SP function and PSP function in a binary neuron are discussed. Furthermore, some properties of the separating hyperplane for PSP functions are analyzed. Finally, a general judging and constructing method of SP functions and PSP functions is proposed.

Key words Binary neural network, linearly separable, PSP function, rule extraction

1 Introduction

Neurons in a binary neural network (BNN) employ a hard-limiter activation function. Each of them expresses a linearly separable (LS) structure in Boolean space[1]. Those LS structures that have similar properties can be grouped as a kind of LS series, for example, m-dimensional hypercubes in n-dimensional Boolean space. Hypercubes with different m have different structures. Their weights and threshold in a binary neuron are different too. But they have some similar space properties. So they can be analyzed as a special kind of LS series. On the other hand, the learning algorithm applied to a BNN is various[2,3]. When a BNN has been trained, some binary neurons perhaps belong to a kind of LS series, and some others perhaps belong to another kind of LS series. So for every kind of LS series, a general judging method and a clear logic meaning are very useful for the extraction of rules from a BNN. Jung H. Kim and Sung-Kwon Park defined a LS series in [2], called PSP (positive successive product) functions, and proposed an ETL learning algorithm of BNNs. They proved the output binary neuron in ETL is a PSP function, and gave a constructing method of PSP functions. But they did not solve the problem on how to judge that an arbitrary binary neuron is (not) a PSP function. In addition, PSP functions are only for positive logic inputs. In fact, PSP functions are a special case of SP (successive product) functions. So it is more effective to directly study SP functions.

SP functions are not only an LS series, but also an LS series that has a clear logical meaning. A through Study on SP function is helpful to the research about the extraction of rules from a BNN based on LS series. In this paper, a general judging and constructing method of SP functions is proposed.

2 SP functions

According to the concept of LS functions, suppose $F(x_1, x_2, \cdots, x_n)$ is an LS Boolean function, $F \in \{0,1\}$. $\sum_{i=1}^{n} w_i x_i - T = 0$ is a hyperplane of F. $\sum_{i=1}^{n} w_i x_i - T \geqslant 0$ when $F(x_1, x_2, \cdots, x_n) = 1$, and $\sum_{i=1}^{n} w_i x_i - T < 0$ when $F(x_1, x_2, \cdots, x_n) = 0$. If a binary neuron has a structure as follows.

1) Supported by the Science Research Program of Anhui Province of P. R. China(01041175)
Received September 4, 2001; in revised form May 20, 2002
收稿日期 2001-09-04; 收修改稿日期 2002-05-20

$$S = U\left(\sum_{i=1}^{n} w_i x_i - T\right),$$

$$U(x) = \begin{cases} 1, & x \geqslant 0 \\ 0, & x < 0 \end{cases}$$

then S is called Boolean function $F(x_1, x_2, \cdots, x_n)$, and can be expressed as $S(\boldsymbol{W}, \boldsymbol{X}, T)$, where $\boldsymbol{W} = (w_1, w_2, \cdots, w_n)$ and $\boldsymbol{X} = (x_1, x_2, \cdots, x_n)$.

In order to avoid confusion, in this paper, "1" represents logic value 1, "0" represents logic value 0.

Definition 1. An SP function is a Boolean function that can be expressed as follows.

$$B(x_1, x_2, \cdots, x_n) = x_1^* O(x_2^* O(\cdots O(x_{n-1}^* O x_n^*))\cdots) \tag{1}$$

where O is the logic AND operator $\wedge$, or the logic OR operator $\vee$, function variables x_i^* represent x_i or $\overline{x_i}$.

Also, SP functions can be expressed in a recursive form as follows:

$$\begin{cases} B(x_1, x_2, \cdots, x_n) = x_1^* O B'(x_2, x_3, \cdots, x_n) \\ B(x_{n-1}, x_n) = x_{n-1}^* O x_n^* \end{cases} \tag{2}$$

When all the input logic variables are positive (not containing NOT logic), SP functions are just PSP functions. For example, Eq. (3) is a PSP function, and Eq. (4) is an SP function.

$$B_1(x_1, x_2, x_3, x_4, x_5, x_6) = x_1 \wedge (x_2 \vee (x_3 \wedge x_4 \wedge (x_5 \vee x_6))) \tag{3}$$

$$B_2(x_1, x_2, x_3, x_4, x_5, x_6) = x_1 \wedge (\overline{x_2} \vee (x_3 \wedge \overline{x_4} \wedge (x_5 \vee x_6))) \tag{4}$$

Theorem 1. An SP function is an LS function.

Theorem 2. The domain of inputs and outputs of binary neurons is $\{0, 1\}$, $\boldsymbol{X} = (x_1^*, x_2^*, \cdots, x_j^*, \cdots, x_n^*)$, $\boldsymbol{X}' = (x_1^*, x_2^*, \cdots, \overline{x_j^*}, \cdots, x_n^*)$. If $S(\boldsymbol{X}, \boldsymbol{W}_1, T_1)$ denotes the Boolean function $F(\boldsymbol{X})$, then $P(\boldsymbol{X}, \boldsymbol{W}_2, T_2)$ denotes the Boolean function $F(\boldsymbol{X}')$, where

$$\begin{cases} w_{2i} = w_{1i} (i \neq j) \\ w_{2j} = -w_{1j} \\ T_2 = T_1 - w_{1j} \end{cases}$$

and $\boldsymbol{W}_1 = (w_{11}, w_{12}, \cdots, w_{1n})$, $\boldsymbol{W}_2 = (w_{21}, w_{22}, \cdots, w_{2n})$.

Theorem 2 indicates: For a Boolean function $F(\boldsymbol{X}')$ that contains some NOT logic input variables $\overline{x}_j$, its corresponding binary neuron P can be changed to the binary neuron that only contains positive logic input variables and represents function $F(\boldsymbol{X})$, and vice versa. So if a binary neuron S denotes a PSP function $F(\boldsymbol{X})$, then by modifying the weights and threshold of S, another binary neuron P that denotes the corresponding SP function $F(\boldsymbol{X}')$ can be constructed.

For a known PSP function, [2] gives a method of constructing a hyperplane as follows.

Method 1. A constructing method of a hyperplane for a PSP function:

1) The method starts from the innermost logic of the PSP function. Suppose

$$\mathrm{net}_n = x_n - 1.$$

2) Let $k = n$.

3) For logic $x_{k-1} \vee x_k$, construct $\mathrm{net}_{k-1} = (-\min[\mathrm{net}_k])x_{k-1} + \mathrm{net}_k$

and for logic $x_{k-1} \wedge x_k$, construct

$$\mathrm{net}_{k-1} = (\max[\mathrm{net}_k] + 1)x_{k-1} + \mathrm{net}_k - (\max[\mathrm{net}_k] + 1)$$

where $\min[\mathrm{net}_k]$ is the minimum value of net_k, $\max[\mathrm{net}_k]$ is the maximum value of net_k.

4) $k = k - 1$, and continue the above process until $k = 1$. Then $\mathrm{net}_1 = 0$ is a hyperplane of the PSP function.

In this paper, the main aim of research is to find out a method that can be used to judge if a binary neuron denotes an SP function. But firstly, it is necessary to discuss the general constructing method of hyperplanes for an SP function. Every SP function has

countless hyperplanes which form a plane series. If the constructing method of the plane series can be established, then the problem is solved.

3 General constructing method of hyperplanes for SP functions

An SP function has many logic levels, and Method 1 is just a process based on those logic levels. In Method 1, when the hyperplane of each logic level is deduced, only the logic relation (AND or OR) between the whole net and the next logic variable is necessary. According to that point, after analyzing and modifying Method 1, we propose the general constructing method of hyperplanes for SP functions as follows.

Method 2. The general constructing method of hyperplanes for an SP function:

1) Modify the SP function to the corresponding PSP function.

2) The method starts from the innermost logic of the PSP function. Suppose

$$\mathrm{ne}\, t_n = px_n - q \quad \left(p > 0,\ 0 < \frac{q}{p} \leqslant 1\right).$$

3) Let $k=n$.

4) For logic $x_{k-1} \vee x_k$, construct

$$\mathrm{ne}\, t_{k-1} = (-\min[\mathrm{ne}\, t_k] + \alpha_{k-1})x_{k-1} + \mathrm{ne}\, t_k \quad (\alpha_{k-1} \geqslant 0).$$

For logic $x_{k-1} \wedge x_k$, construct

$$\mathrm{ne}\, t_{k-1} = (\max[\mathrm{ne}\, t_k] + \beta_{k-1})x_{k-1} + \mathrm{ne}\, t_k - (\max[\mathrm{ne}\, t_k] + \beta_{k-1}) \quad (\beta_{k-1} > 0).$$

5) $k=k-1$, and continuing the above process until $k=1$. Then $\mathrm{ne}\, t_1 = 0$ is a hyperplane of the PSP function.

6) According to Theorem 2, modify $\mathrm{ne}\, t_1$ to $\mathrm{ne}\, t_1'$, then $\mathrm{ne}\, t_1' = 0$ is a hyperplane of the SP function.

As an example, consider the SP function of Eq. (4), and construct a hyperplane of it.

Firstly, modify Eq. (4) to the corresponding PSP function as Eq. (3), then construct a hyperplane of Eq. (3) by Method 2. The process is as follows:

$$\begin{aligned}
&\mathrm{ne}\, t_6 = x_6 - 0.1, && (p = 1, q = 0.1)\\
&\mathrm{ne}\, t_5 = 0.3x_5 + x_6 - 0.1, && (\alpha_5 = 0.2)\\
&\mathrm{ne}\, t_4 = 1.5x_4 + 0.3x_5 + x_6 - 1.6, && (\beta_4 = 0.3)\\
&\mathrm{ne}\, t_3 = 1.6x_3 + 1.5x_4 + 0.3x_5 + x_6 - 3.2, && (\beta_3 = 0.4)\\
&\mathrm{ne}\, t_2 = 3.2x_2 + 1.6x_3 + 1.5x_4 + 0.3x_5 + x_6 - 3.2, && (\alpha_2 = 0)\\
&\mathrm{ne}\, t_1 = 4.9x_1 + 3.2x_2 + 1.6x_3 + 1.5x_4 + 0.3x_5 + x_6 - 8.1. && (\beta_1 = 0.5)
\end{aligned}$$

After the hyperplane $net_1 = 0$ for Eq. (3) is constructed, it is modified according to Theorem 2. Then a hyperplane of the SP function of Eq. (4) can be constructed as follows: $\mathrm{ne}\, t_1' = 4.9x_1 - 3.2x_2 + 1.6x_3 - 1.5x_4 + 0.3x_5 + x_6 - 3.4$.
$\mathrm{ne}\, t_1' = 0$ is a hyperplane of the SP function.

4 General judging method of SP functions

When we consider the problem that how a binary neuron P can be judged to denote an SP function, our thought is: change the binary neuron P to the corresponding binary neuron S that only contains positive weights according to Theorem 2. If S can be judged to denote a PSP function, then P must denote an SP function. So the general judging method of PSP functions becomes the key point.

Next, two new concepts, PSP-AND functions and PSP-OR functions, are defined.

Definition 2. A PSP-AND function is a Boolean function that can be expressed as Eq. (5), A PSP-OR function is a Boolean function that can be expressed as Eq. (6).

$$B_{AND}(x_k, x_{k+1}, \cdots, x_n) = x_k \wedge B(x_{k+1}, x_{k+2}, \cdots, x_n) \tag{5}$$

$$B_{OR}(x_k, x_{k+1}, \cdots, x_n) = x_k \vee B(x_{k+1}, x_{k+2}, \cdots, x_n) \tag{6}$$

where $B(x_{k+1}, x_{k+2}, \cdots, x_n)$ is a PSP function, x_k is called the main element of the k-th log-

ic level.

Theorem 3. Let $net_1=\sum_{i=1}^{n}w_ix_i-T$ and let $net_1=0$ be a hyperplane of a PSP function, which is constructed according to Method 2. Then $\forall i\in\{1,2,\cdots,n\}$, satisfies $w_i>0$ and $T>0$.

Proof. The following proving process is mathematical induction.

1) $\because$ $net_n=px_n-q$ $\left(p>0,0<\frac{q}{p}\leqslant 1\right)$, $\therefore$ $w_n=p>0, T=q>0$

2) In net_k, suppose w_i, $i\in\{k,k+1,\cdots,n\}$, satisfy $w_i>0$, and threshold $T_k>0$. Then for a PSP-OR function, $net_{k-1}=(-min[net_k]+\alpha_{k-1})x_{k-1}+net_k=(T_k+\alpha_{k-1})x_{k-1}+net_k$.

$\because$ $\alpha_{k-1}\geqslant 0$, $\therefore$ $w_{k-1}=T_k+\alpha_{k-1}>0$, and $T_{k-1}=T_k>0$.

For a PSP-AND function, $net_{k-1}=(\max[net_k]+\beta_{k-1})x_{k-1}+net_k-$

$$(\max[net_k]+\beta_{k-1})=\left(\sum_{i=k}^{n}w_i-T_k+\beta_{k-1}\right)x_{k-1}+net_k-\left(\sum_{i=k}^{n}w_i-T_k+\beta_{k-1}\right)$$

$\because$ When $x_k,x_{k+1},\cdots,x_n$ are all "1", a PSP function must be true, $\therefore$ $\sum_{i=k}^{n}w_i-T_k\geqslant 0$.

Moreover, $\because$ $\beta_{k-1}>0$, $\therefore$ $w_{k-1}=\sum_{i=k}^{n}w_i-T_k+\beta_{k-1}>0$.

For the threshold $T_{k-1}=\sum_{i=k}^{n}w_i+\beta_{k-1}$, according to the supposition, i.e., $w_i>0$, $i\in\{k,k+1,\cdots,n\}$, so $T_{k-1}>0$.

When net_{k-1} of the PSP-AND or PSP-OR function is constructed, for $i\in\{k,k+1,\cdots,n\}$, the values of w_i are not changed, and according to the supposition, all of them are greater than 0. So the conclusion of the proposition is correct. □

Theorem 3 indicates that a binary neuron may denote a PSP function only when all the weights and threshold of it are greater than 0. Next, two important properties about weights of hyperplanes for a PSP-AND function and a PSP-OR function are discussed..

Theorem 4. Suppose $B_{OR}(x_k,x_{k+1},\cdots,x_n)$ is a PSP-OR function, and that $\sum_{i=k}^{n}w_ix_i-T_k=0$, which is constructed according to Method 2, is a hyperplane of B_{OR}. If x_k is the main element of the k-th logic level, then $w_k\geqslant T_k$.

Proof. Let $B_{OR}(x_k,x_{k+1},\cdots,x_n)=x_k\vee B(x_{k+1},\cdots,x_n)$, and $net_k=0$ be a hyperplane of it. Then $net_k=(-\min[net_{k+1}]+\alpha_k)x_k+net_{k+1}=(T_{k+1}+\alpha_k)x_k+net_{k+1}$

$\because$ $\alpha_k\geqslant 0$, $\therefore$ $T_k=T_{k+1}\leqslant T_{k+1}+\alpha_k=w_k$. □

Theorem 5. Suppose $B_{AND}(x_k,x_{k+1},\cdots,x_n)$ is a PSP-AND function, and that $\sum_{i=k}^{n}w_ix_i-T_k=0$, which is constructed according to Method 2, is a hyperplane of B_{AND}. Then $\forall i\in\{k,k+1,\cdots,n\}$, $w_i<T_k$.

Proof. Let $B_{AND}(x_k,x_{k+1},\cdots,x_n)=x_k\wedge B(x_{k+1},\cdots,x_n)$, where x_k is the main element of the k-th logic level, and $net_k=0$ be a hyperplane of B_{AND}. Then

$$net_k=(\max[net_{k+1}]+\beta_k)x_k+net_{k+1}-(\max[net_{k+1}]+\beta_k)=$$

$$\left(\sum_{i=k+1}^{n}w_i-T_{k+1}+\beta_k\right)x_k+net_{k+1}-\left(\sum_{i=k+1}^{n}w_i-T_{k+1}+\beta_k\right),$$

$\therefore$ $T_k=\sum_{i=k+1}^{n}w_i+\beta_k$.

According to Theorem 3, $\forall i\in\{k+1,k+2,\cdots,n\}$ satisfies $w_i>0$. Since $\beta_k>0$,

$\therefore$ $\forall i\in\{k+1,k+2,\cdots,n\}$, $T_k>w_i$.

Moreover, $w_k=\sum_{i=k+1}^{n}w_i-T_{k+1}+\beta_k=T_k-T_{k+1}<T_k$,

$\therefore$ $\forall i\in\{k,k+1,\cdots,n\}, T_k>w_i$. □

Theorem 4 and Theorem 5 indicate that a binary neuron, all the weights and threshold

of which are greater than 0, may denote a PSP-OR logic only when one of the weights is greater than the threshold, denote a PSP-AND logic only when all the weights are less than the threshold.

Definition 3. When a PSP function $B(x_k, x_{k+1}, \cdots, x_n)$ can be expressed as one of the two following equations, it is said that a logic conversion happens at the k-th logic level in the PSP function.

a) $B(x_k, x_{k+1}, \cdots, x_n) = x_k \vee B'(x_{k+1}, \cdots, x_n)$, where $B'(x_{k+1}, \cdots, x_n)$ is a PSP-AND function.

b) $B(x_k, x_{k+1}, \cdots, x_n) = x_k \wedge B'(x_{k+1}, \cdots, x_n)$, where $B'(x_{k+1}, \cdots, x_n)$ is a PSP-OR function.

And the case of a) is called a logic conversion of type Ⅰ, the case of b) is called a logic conversion of type Ⅱ.

Theorem 6. Suppose $\sum_{i=k}^{n} w_i x_i - T_k = 0$, which is constructed according to Method 2, is a hyperplane of PSP function $B(x_k, x_{k+1}, \cdots, x_n)$. If a logic conversion happens at the k-th logic level in $B(x_k, x_{k+1}, \cdots, x_n)$, and if x_k is the main element of the k-th logic level, then $\forall i \in \{k+1, \cdots, n\}$, $w_k > w_i$.

Proof. For a logic conversion of type Ⅰ, $B(x_k, x_{k+1}, \cdots, x_n) = x_k \vee B'(x_{k+1}, \cdots, x_n)$

$$\text{ne}\, t_k = (-\min[\text{ne}\, t_{k+1}] + \alpha_k) x_k + \text{ne}\, t_{k+1} = (T_{k+1} + \alpha_k) x_k + \text{ne}\, t_{k+1},$$

$\therefore$ $w_k = T_{k+1} + \alpha_k$.

Further, $\because$ $B'(x_{k+1}, \cdots, x_n)$ is a PSP-AND function,

$\therefore$ According to Theorem 5, in $\text{ne}\, t_{k+1}$, $\forall i \in \{k+1, \cdots, n\}$ satisfies $T_{k+1} > w_i$,

$\therefore$ $\forall i \in \{k+1, \cdots, n\}$ satisfies $w_k = T_{k+1} + \alpha_k \geqslant T_{k+1} > w_i$.

For a logic conversion of type Ⅱ, $B(x_k, x_{k+1}, \cdots, x_n) = x_k \wedge B'(x_{k+1}, \cdots, x_n)$

$\because$ $B'(x_{k+1}, \cdots, x_n)$ is a PSP-OR function,

$\therefore$ Let x_{k+1} be the main element of the $(k+1)$-th logic level. Then $B'(x_{k+1}, \cdots, x_n) = x_{k+1} \vee B''(x_{k+2}, \cdots, x_n)$. And no matter what kind of PSP function $B''(x_{k+2}, \cdots, x_n)$ is, it can be expressed as $B'(x_{k+1}, \cdots, x_n) = x_{k+1} \vee \cdots \vee x_{k+j} \vee B'''(x_{k+j+1}, \cdots, x_n)$,
where $j \geqslant 1$, $B'''(x_{k+j+1}, \cdots, x_n)$ is a PSP-AND function. Then for the $(k+1)$-th logic level, $\text{ne}\, t_{k+1} = w_{k+1} x_{k+1} + w_{k+2} x_{k+2} + \cdots + w_{k+j} x_{k+j} + \text{ne}\, t_{k+j+1}$.
Let $w_{k+t} = \max(w_{k+1}, w_{k+2}, \cdots, w_{k+j})$. Then according to the concept of OR logic, when x_{k+t} is "0" and all of $x_{k+1}, x_{k+2}, \cdots, x_{k+t-1}, x_{k+t+1}, \cdots, x_n$ are "1", $B'(x_{k+1}, \cdots, x_n)$ is true,

$$\therefore \qquad \text{ne}\, t_{k+1} = \sum_{\substack{i=k+1 \\ i \neq k+t}}^{n} w_i \cdot 1 + w_{k+t} \cdot 0 - T_{k+1} = \sum_{\substack{i=k+1 \\ i \neq k+t}}^{n} w_i - T_{k+1} \geqslant 0 \tag{7}$$

Consider the $\text{ne}\, t_k$ again, and we have

$$\text{ne}\, t_k = (\max[\text{ne}\, t_{k+1}] + \beta_k) x_k + \text{ne}\, t_{k+1} - (\max[\text{ne}\, t_{k+1}] + \beta_k),$$

where $w_k = \max[\text{ne}\, t_{k+1}] + \beta_k = \sum_{\substack{i=k+1 \\ i \neq k+t}}^{n} w_i - T_{k+1} + \beta_k = w_{k+t} + \sum_{\substack{i=k+1 \\ i \neq k+t}}^{n} w_i - T_{k+1} + \beta_k$.

According to Eq. (7), $w_k \geqslant w_{k+t} + \beta_k > w_{k+t}$. Moreover, the proved result about the logic conversion of type Ⅰ gives this fact: for $\forall r \in \{k+j+1, \cdots, n\})$, $w_{k+t} \geqslant w_{k+j} > w_r$. So $\forall i \in \{k+1, \cdots, n\}$, $w_k > w_i$. □

Theorem 6 indicates: When we analyze to see if a binary neuron denotes a PSP function $B(x_k, x_{k+1}, \cdots, x_n) = x_k O B'(x_{k+1}, \cdots, x_n)$, the input that has the maximum value of the weights must be considered as the main element x_k of the k-th logic level firstly. But Theorem 4, Theorem 5 and Theorem 6 are not yet enough to build a general method that can be used to judge if a binary neuron denotes a PSP function, and more criteria are needed.

Theorem 7. In Boolean space, $F(x_1, x_2, \cdots, x_n)$ is a n-dimensional LS function, its hyperplane is $\sum_{i=1}^{n} w_i x_i - T = 0$, and $w_i > 0$. $F'(x_2, x_3, \cdots, x_n)$ is a $(n-1)$-dimensional LS func-

tion, its hyperplane is $\sum_{i=2}^{n} w_i x_i - T = 0$. Then F denotes the logic relation $F = x_1 \vee F'$ if and only if $w_1 \geqslant T$.

Proof. Sufficiency: When $x_1 =$ "1",

$$\sum_{i=1}^{n} w_i x_i - T = w_1 + \sum_{i=2}^{n} w_i x_i - T \tag{8}$$

$\because$ $w_i > 0$

$\therefore$ No matter what kind of logic values $x_2, x_3, \cdots, x_n$ are, according to Eq. (8),

$$\sum_{i=1}^{n} w_i x_i - T = w_1 + \sum_{i=2}^{n} w_i x_i - T \geqslant w_1 - T \geqslant 0 ,$$

$\therefore$ $F(\text{"1"}, x_2, \cdots, x_n) = \text{"1"}$.

When $x_1 =$ "0", $\sum_{i=1}^{n} w_i x_i - T = \sum_{i=2}^{n} w_i x_i - T$,

$\therefore$ $F(\text{"0"}, x_2, \cdots, x_n) = F'(x_2, x_3, \cdots x_n)$. The sufficiency has been proved.

Necessity: Suppose $F = x_1 \vee F'$. Then $F(\text{"1"}, x_2, \cdots, x_n) = \text{"1"}$ when $x_1 =$ "1". So

$$\sum_{i=1}^{n} w_i x_i - T = w_1 + \sum_{i=2}^{n} w_i x_i - T$$

Let $x_2, x_3, \cdots, x_n$ be all "0". Then

$\sum_{i=1}^{n} w_i x_i - T = w_1 - T \geqslant 0$, $\therefore$ $w_1 \geqslant T$. The necessity has been proved. □

Theorem 8. In Boolean space, $F(x_1, x_2, \cdots, x_n)$ is a n-dimensional LS function, its hyperplane is $\sum_{i=1}^{n} w_i x_i - T = 0$, and $w_i > 0$. $F'(x_2, x_3, \cdots, x_n)$ is a $(n-1)$-dimensional LS function, its hyperplane is $\sum_{i=2}^{n} w_i x_i - (T - w_1) = 0$. Then F denotes the logic relation $F = x_1 \wedge F'$ if and only if $\sum_{i=2}^{n} w_i < T$.

Proof. The process of proof is similar to that of Theorem 7, so it is ignored. □

Now, we can describe the whole general method that is used to judge if a binary neuron P denotes an SP function.

Method 3. The general judging method of SP functions.

1) According to Theorem 2, for the binary neuron P , multiply each weight that is less than 0 by -1, and modify the threshold accordingly to form a new binary neuron S.

2) Let $k = 1$, and create a hyperplane $\text{ne}_k = 0$ with weights $(w_1, w_2, \cdots, w_n)$ and threshold T of S, where $\text{ne}_k = \sum_{i=1}^{n} w_i x_i - T$.

3) In ne_k, choose $w_t = \max(w_1, w_2, \cdots, w_n)$. When $w_t \geqslant T$, the logic relation of the k-th logic level can be expressed as follows:

$$F_k(x_1, x_2, \cdots, x_n) = x_t \vee F_{k+1}(x_1, \cdots, x_{t-1}, x_{t+1}, \cdots, x_n)$$

If $k = n-1$, then the judging process ends and the whole PSP function is analyzed successfully; else let $k = k+1$ and $\text{ne}_k = \sum_{\substack{i=1 \\ i \neq t}}^{n} w_i x_i - T$, then return to 3) to continue the judging process.

4) When $w_t < T$, calculate $A = \sum_{\substack{i=1 \\ i \neq t}}^{n} w_i$. If $A < T$, then the logic relation of the k-th logic level can be expressed as follows: $F_k(x_1, x_2, \cdots, x_n) = x_t \wedge F_{k+1}(x_1, \cdots, x_{t-1}, x_{t+1}, \cdots, x_n)$.

If $k = n-1$, then the judging process ends and the whole PSP function is analyzed successfully; else let $k = k+1$ and $\text{ne}_k = \sum_{\substack{i=1 \\ i \neq t}}^{n} w_i x_i - (T - w_t)$, and return to 3) to continue the judging process.

5) In step 4), if $A \geqslant T$, then F_k is not a PSP function, the judging process ends.

6) If the binary neuron S is analyzed and judged to be a PSP function, then the binary neuron P must be an SP function. According to Theorem 2, the SP function denoted by P can be written out directly by modifying the corresponding inputs to $\overline{x_i}$.

5 Example

In this section, an example which explains Method 3 is discussed. In this example, the sequence numbers $\sum_{i=1}^{n} x_i 2^{i-1}$ represent the input binary samples.

In 6-dimensional Boolean space, $X=(x_1, x_2, x_3, x_4, x_5, x_6)$, the set of input binary samples are $R_1=\{5,9,12,13,15,17,20,21,23,24,25,27,28,29,30,31,45,53,57,60,61,63\}$ and $R_2=\{1,5,9,13,17,21,23,25,29,33,37,39,41,45,49,53,55,57,61\}$. In their respective 6-dimensional Boolean spaces, the sample values of all elements in R_1 and R_2 are "1", and "0" for others. Fig. 1 and Fig. 2 are their Karnaugh maps.

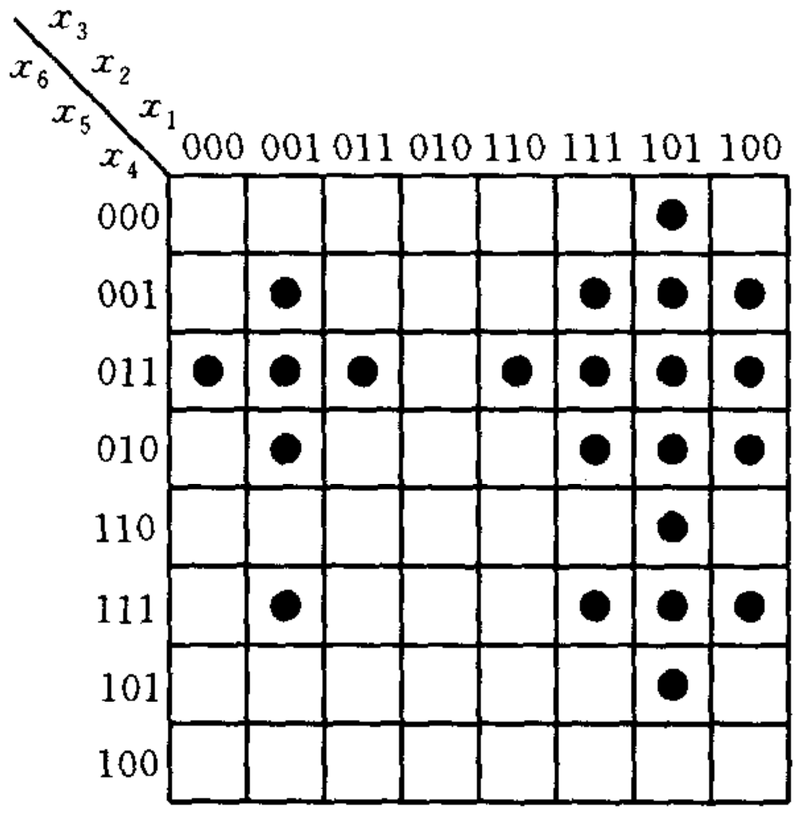

Fig. 1 Karnaugh map of R_1

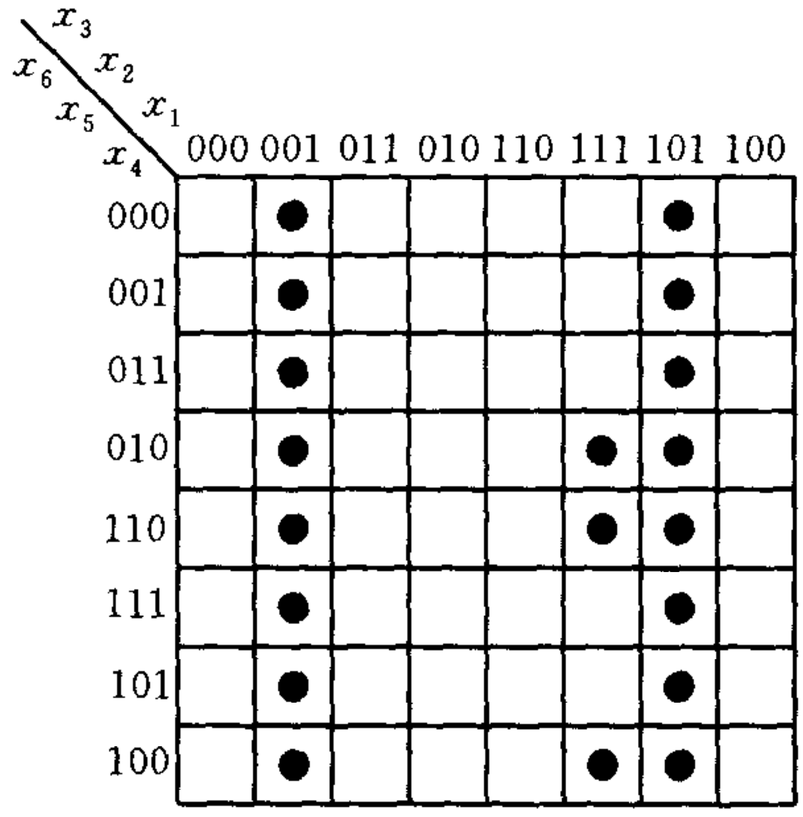

Fig. 2 Karnaugh map of R_2

With initial weights $W^0=\{0.1,-0.03,0.4,0.01,-0.1,-0.25\}$, initial threshold $T^0=0.35$ and step $h=0.05$, after R_1 and R_2 are trained in a binary neuron by the perceptron training algorithm, the convergent results about the weights and threshold are as follows:

R_1: $W_{R_1}=(0.25,-0.18,0.25,0.26,0.15,-0.25)$, $T_{R_1}=0.35$,

R_2: $W_{R_2}=(0.85,-0.58,0.2,-0.24,0.15,0.05)$, $T_{R_2}=0.5$.

5.1 Judging the training result of R_1

According to Method 3, firstly, W_{R_1} and T_{R_1} are modified as follows:

$$W'_{R_1}=(0.25,0.18,0.25,0.26,0.15,0.25),\quad T'_{R_1}=0.78.$$

All the weights in W'_{R_1} are less than T'_{R_1}, so select the maximum value $w_t=0.26$ in W'_{R_1}, then calculate $\sum_{\substack{i=1\\ i\neq t}}^{n} w_i=1.08>T'_{R_1}=0.78$. So R_1 is not a SP function. The judging process ends.

5.2 Judging the training result of R_2

Similarly, W_{R_2} and T_{R_2} are modified as follows:

$$W'_{R_2}=(0.85,0.58,0.2,0.24,0.15,0.05),\quad T'_{R_2}=1.32$$

All the weights in W'_{R_2} are less than T'_{R_2}. Select the maximum value $w_t=0.85$ in W'_{R_2}, then calculate $\sum_{\substack{i=1\\ i\neq t}}^{n} w_i=1.22<T'_{R_2}=1.32$. So the first level is AND logic, and is expressed

as $R_2' = x_1 \wedge F_1$.

F_1 is the logic function denoted by the binary neuron $W_{F_1} = (0.58, 0.2, 0.24, 0.15, 0.05)$ and $T_{F_1} = 1.32 - 0.85 = 0.47$. Continue to select the maximum value in W_{F_1}, that is, $0.58 > T_{F_1} = 0.47$. So the second level is OR logic, and is expressed as $R_2' = x_1 \wedge (x_2 \vee F_2)$.

F_2 is the logic function denoted by the binary neuron $W_{F_2} = (0.2, 0.24, 0.15, 0.05)$ and $T_{F_2} = 0.47$. Continue the process until the innermost logic level. The final result is $R_2 = x_1 \wedge (x_2 \vee (x_3 \wedge x_4 \wedge (x_5 \vee x_6)))$. So it is concluded that R_2 is a PSP function. Its logic relation is $R_2 = x_1 \wedge (\overline{x_2} \vee (x_3 \wedge \overline{x_4} \wedge (x_5 \vee x_6)))$.

After the above discussion, R_2 is judged to be the logic relation as Eq. (4). In fact, R_1 is a Hamming sphere[4], with center at (1,0,1,1,1,0) and radius of 2.

6 Conclusion

Besides SP functions, there exist other LS series that have clear logical meanings. It is very important to establish general judging methods of all these LS series. For a binary neural network, if every binary neuron in it can be judged to belong to a certain LS series that has a clear logical meaning, then the extraction of rules from a BNN becomes possible.

References

1 Ma Xiao-Min, Yang Yi-Xian, Zhang Zhao-Zhi. Nonlinear shift register synthesis based on binary neural network. *Acta Electronica Sinica*, 2000, **28**(1): 70～73(in Chinese)

2 Kim J H, Park S. The geometrical learning of binary neural networks. *IEEE Transactions on Neural Networks*, 1995, **6**(1): 237～247

3 Zhu Da-Ming, Ma Shao-Han, Wei Dao-Zheng. A geometrical learning algorithm for binary mapping neural networks and its application. *Acta Automatica Sinica*, 2000, **26**(3): 339～346(in Chinese)

4 Ma Xiao-Min, Yang Yi-Xian, Zhang Zhao-Zhi. An efficient algorithm for Boolean neural network. *Journal of China Institute of Communication*, 1999, **20**(12): 13～18(in Chinese)

LU Yang An associate professor and a Ph. D. candidate at the Computer and Information College, Hefei University of Technology. His research interests are intelligent control, pattern recognition and signal processing.

HAN Jiang-Hong A professor at the Computer and Information College, Hefei University of Technology. His research interests are intelligent control, embedded systems and computer information systems.

WEI Zhen An associate professor and a Ph. D. candidate at the Computer and Information College, Hefei University of Technology. His research interests are theory of reliability and intelligent control.

二进神经网络中 SP 函数的一般判别和构造方法

陆　阳　　韩江洪　　魏　臻

（合肥工业大学计算机与信息学院　合肥　230009）

（E-mail: luyang.gocom@163.com）

摘　要　SP 函数是一类具有明确逻辑意义的线性可分结构系，PSP 函数是 SP 函数的特殊子集. 文中讨论了二进神经元对 SP 函数和 PSP 函数的表达问题，通过研究 PSP 函数分类超平面的某些性质，建立了 SP 函数和 PSP 函数的一般判别和构造方法.

关键词　二进神经网络，线性可分，PSP 函数，规则提取

中图分类号　TP18