基于复数权神经元的布尔函数稳健学习算法

吕伟锋[1,2],林 弥[1],孙玲玲[1]
(1.杭州电子科技大学教育部射频电路与系统重点实验室,浙江杭州 310018;
2.浙江大学超大规模集成电路设计研究所,浙江杭州 310027)

摘 要: 复数权神经元由于引入了多阈值逻辑而具有更强的性能.文中根据其数学模型,结合二进感知器神经元稳健设计概念,提出了该神经元的稳健性数学定义,并根据定义,得到了简单逻辑和异或逻辑的单个神经元稳健实现方案.然后根据该方案提出了任意复杂布尔函数稳健实现算法,同时证明了该算法的正确性.最后通过具体实例演示该算法实现布尔函数的过程和方法,结果表明了基于该神经元的稳健学习算法实现布尔函数优点和灵活性,说明了其强大的逻辑处理能力.

关键词: 复数权值;多值神经元;布尔函数;稳健设计
中图分类号: TP183, TN431 **文献标识码:** A **文章编号:** 0372-2112 (2011) 11-2708-05

Robust Learning Algorithm for Boolean Function Based on Neurons with Complex-Valued Weights

LÜ Wei-feng[1,2], LIN Mi[1], SUN Ling-ling[1]
(1. *Key Laboratory of Ministry of Education for RF Circuits and Systems, Hangzhou Dianzi University, Hangzhou, Zhejiang* 310018, *China*;
2. *Institute of VLSI Design, Zhejiang University, Hangzhou, Zhejiang* 310027, *China*)

Abstract: Neurons with complex-valued weights have a stronger performance due to the multi-threshold logic. In the paper, the robustness for this kind of neurons is presented in accordance with its mathematical model and the binary perceptron's definitions of robustness. The robust design for basic digital logics of multiple variables or XOR logic is proposed after that and the robust algorithm for any Boolean functions base on the neurons is given, also the correctness of the algorithm is proved. The final use of specific examples not only shows that the algorithm can realize an arbitrary Boolean function. It also shows that the neurons used to achieve the flexibility and advantage of realizing any digital logic function, illustrates its powerful logical processing capabilities.

Key words: complex-valued weights; multi-valued neurons (MVNs); Boolean function; robust design

1 引言

1992 年,美国 IEEE 出版的神经网络理论基础与分析文集将神经网络定义为:用大量简单的"神经元"并行叠联而成的任何计算结构[1].显然,神经元是神经网络的基本组成和处理单元,其功能的强弱直接影响到整个神经网络的规模、复杂度及稳健性的优劣.近年来,有关文献提出了复数权多阈值神经元的概念,这是一种引入了复数权值而具有多阈值逻辑的神经元,复数扩大了信息处理范围,增强信息处理能力;多阈值逻辑使得传输线上携带更多信息,提高信息密度.因此,这种神经元被认为具有更强的逻辑映射能力和收敛速度,已被广泛应用于联想式记忆、图像分割、识别,时间序列预测等领域[2~7].随着研究的深入,该神经元又被应用于组建 BP 或 Hopfield 神经网络并应用于交通信号控制等领域[8~11].

布尔函数(数字逻辑函数)实现是神经元网络应用的一个重点和核心领域,也是直观反映神经元功能强弱的有效手段,并已取得大量研究成果[12~17],其中的一个重要内容是对神经元稳健性的讨论和应用[14~17].但到目前为止,稳健性的概念在最简单的感知器神经元中建立最好,对其它类型的神经元研究较少[16],也使得任意复杂布尔函数的复数权神经元稳健设计缺乏基础.本文着重讨论和解决该问题,并提出相应的算法.

2 复数权神经元数学模型

复数权神经元的基本思想是对神经元权值取复数,并通过对实数单位 1 的多次方根和对激活函数的编码达到不但使输出可以是多阈值信号,而且使其学习算法具有更快收敛速度的目的,从而扩展了神经元的功能.

收稿日期:2009-09-25;修回日期:2010-10-26

设 $x_1, x_2, \cdots, x_n$,是 n 个 $k(k \geqslant 3)$值或二值输入变量,$w_0, w_1, \cdots, w_n$ 是神经元的 $n+1$ 个复数权(其中 w_0 通常也被称为阈值),则该神经元的输出可表示为:

$$f(x_1, \cdots, x_n) = P(w_0 + w_1 x_1 + \cdots + w_n x_n) \tag{1}$$

这里 P 是神经元激活函数,可由下面的表达式定义:

$$\text{当} \quad 2\pi(j+1)/k > \arg(z) > 2\pi j/k \text{ 时}, \quad P(z) = \exp(i2\pi j/k) \tag{2}$$

其中 $j \in \{0,1,\cdots,k-1\}$,是 k 值逻辑的其中一个值,i 是虚数单位,而 $\arg(z)$表示复数 z 的幅角,$z = w_0 + w_1 x_1 + \cdots + w_n x_n$ 是其加权和,函数和变量的值通过实数单位 1 的 k 次方根进行编码,即:$\varepsilon^j = \exp(i2\pi j/k)$,而 $j \rightarrow \varepsilon^j$.因此,若复数 z 属于由式(2)划分的复平面上的第 j 个部分,神经元的输出即为 ε^j,如图 1 所示即为其激活函数定义.若将它应用于普通二进逻辑中,则式(2)可表示为:

$$\text{当} \quad [2\pi(j+1)/m] > \arg(z) \geqslant (2\pi j/m) \text{时}, \quad P(z) = (-1)^j \tag{3}$$

式中,m 是正整数 $0 < m \leqslant 3n$,j 是非负整数 $0 \leqslant j < m$.在这种情况下,变量和实现的逻辑函数值是二进的,取自集合$\{-1,1\}$.

3　二进复数权神经元稳健设计

复数权神经元稳健性的定义可与感知器神经元类似[14,17]:假定红色和蓝色分别代表逻辑 1 和逻辑 0,设

$$\text{红色顶点:} \boldsymbol{X}^{(1)} = (x_1^{(1)}, x_2^{(1)}, \cdots, x_n^{(1)}),$$

$$\text{蓝色顶点:} \boldsymbol{X}^{(2)} = (x_1^{(2)}, x_2^{(2)}, \cdots, x_n^{(2)})$$

为 n 维超立体中相邻的两点,分别对应于加权和后的复矢量 z_1和 z_2,若分类复矢量:

$$\arg(B_1 x_1 + B_2 x_2 + \cdots + B_n x_n + \theta) = L\pi, L \in \{0,1\}$$

划分上述两个顶点满足:

$$\begin{cases} \arg(B_1 x_1^{(1)} + B_2 x_2^{(1)} \cdots B_n x_n^{(1)} + \theta) = \dfrac{\pi}{2} \\ \arg(B_1 x_1^{(2)} + B_2 x_2^{(2)} \cdots B_n x_n^{(2)} + \theta) = \dfrac{3\pi}{2} \end{cases} \tag{4}$$

若 $\pi(j+1) > \arg(z) \geqslant \pi j, j \in \{0,1\}$称由下式定义的神经元:$y = P(z) = (-1)^j$ 为稳健二进复数权神经元.此时,该神经元的模型可表示为图 2 所示.在该情况下,可建立简单逻辑神经元稳健设计的矩阵方程,然后得到的相应权矢量.

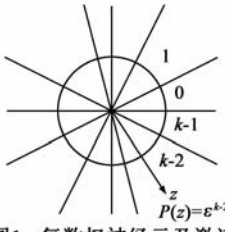


图1　复数权神经元及激活函数的定义

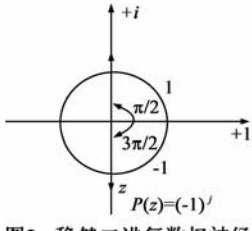


图2　稳健二进复数权神经元及激活函数

3.1　与运算

与运算,即当所有输入均为 -1(编码后)时,输出为 -1,否则为 1.其相对应的幅角分别为 $3\pi/2$ 和 $\pi/2$,则其复数值分别为 $-\alpha i$ 和 $+\alpha i$(α 为实数).由此可得如式(5)所示矩阵方程,式中 $c_0, c_1, \cdots, c_{2^n-2} > 0$ 且 $c_{2^n-1} < 0$ 均是待定实系数.由方程可猜想一组最简单权矢量解为 $\boldsymbol{W} = [(n-1)i, i, \cdots, i]$.

$$\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 1 & 1 & \cdots & -1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & -1 & -1 & \cdots & 1 \\ 1 & -1 & -1 & \cdots & -1 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} c_0 i \\ c_1 i \\ \vdots \\ c_{2^n-2} i \\ c_{2^n-1} i \end{bmatrix} \tag{5}$$

证明　n 个输入变量共存在 2^n 个不同的状态,这些状态可分为两种情况:(a)所有变量都是 -1;(b)至少存在一个 1.

对于(a),有:

$$z = (n-1)i + i(-1) + i(-1) + \cdots + i(-1) = (n-1)i + (-n)i = -i$$

则
$$P(z) = (-1)^1 = -1 \tag{6}$$

对于(b),有:

$$z/i = (n-1) + (-1) + \cdots + 1 + \cdots + (-1) \geqslant 1$$

因此
$$P(z) = (-1)^0 = 1 \tag{7}$$

证毕.

权矢量确定后,系数 $c_N (0 \leqslant N \leqslant 2^n - 1)$值可按如下步骤计算:将十进制数 N 转化为二进制数,将二进制位变换为$\{0,1\} \rightarrow \{1,-1\}$,再将该变换后的二进制位求和再加上 $n-1$.如在 5 个变量的情况下,求得系数 c_{21}是 3,而系数 c_{31}是 -1.

3.2　或运算

或运算即当输入至少有一个 -1 时(编码后),输出为 -1,否则为 1,可得到方程如式(8)所示,式中,$d_0 > 0$,且 $d_1, \cdots, d_{2^n-2}, d_{2^n-1} < 0$ 均为待定实系数.可猜想得到一组最简单的权矢量解为 $\boldsymbol{W} = [-(n-1)i, i, \cdots, i]$.

$$\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 1 & 1 & \cdots & -1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & -1 & -1 & \cdots & 1 \\ 1 & -1 & -1 & \cdots & -1 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} d_0 i \\ d_1 i \\ \vdots \\ d_{2^n-2} i \\ d_{2^n-1} i \end{bmatrix} \tag{8}$$

证明　n 个变量存在的 2^n 个不同状态,可分为两种不同情况:(a)所有变量都是 1;(b)至少存在一个 -1.

对于(a),有:

$$z = -(n-1)i + i + i + \cdots + i = -(n-1)i + ni = i$$

则
$$P(z) = (-1)^0 = 1 \tag{9}$$

对于(b),有:

$$z/i = -(n-1) + 1 + \cdots + -1 + \cdots + 1 \leqslant -1$$

因此 $$P(z)=(-1)^1=-1 \tag{10}$$

证毕.

系数 $d_N(0\leqslant N\leqslant 2^n-1)$的确定类似于上述计算系数 c_N,不同的是在最后一步中不是加上而是减去 $n-1$. 因此在 5 个变量情况下,得到系数 d_{21}是 -5,而系数 d_0 是 1.

3.3 非运算

非运算对应的矩阵方程如下:

$$\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}w_0\\ w_1\end{bmatrix}=\begin{bmatrix}e_0 i\\ e_1 i\end{bmatrix} \tag{11}$$

式(11)中,$e_0<0$,且 $e_1>0$ 是待定系数. 易知最简单的解为 $\boldsymbol{W}=[0,-\mathrm{i}]$,系数 $e_0=-1$,$e_1=1$. 根据上述规律,容易得到多变量简单逻辑的稳健设计,一般规律可由表 1 所示.

表 1 复数权神经元简单逻辑稳健设计规律

简单逻辑	输入 x 对应权		阈值 w_0
与	X	i	$(n-1)\mathrm{i}$
	$\bar{x}$	$-\mathrm{i}$	
或	X	i	$-(n-1)\mathrm{i}$
	$\bar{x}$	$-\mathrm{i}$	
非	X	$-\mathrm{i}$	0

不但如此,单个复数权神经元可以方便实现二变量异或运算,而其他如感知器神经元是无法做到的. 取 $m=2n=4$,此时,权矢量与分类复矢量的最大夹角为 $\pi/4$ 弧度,权矢量可取 $\boldsymbol{W}=[0,1,\mathrm{i}]$,虽然稳健性要次于简单逻辑的情况,但可达到为这种情况下的最大值(也可称次稳健设计)[16]. 正是由于复数权神经元的这个特性,使得其在实现布尔函数上具有很大的灵活性和方便性,也体现了与其他类型神经元的不同特点.

4 布尔函数稳健学习算法

基于复数权神经元的布尔函数学习算法的一个基本思想是:任意复杂的多变量布尔函数总可以用上述简单逻辑构成完备集,而每个简单逻辑又可以用一个神经元实现,因此只要按照布尔函数逻辑表达式(如以最小项之和 SOP 或化简后的 SOP 形式),就可以最多用只含有一个隐层的神经元网络实现. 算法如下:

Step 1 隐层神经元与布尔函数最小项(或化简后乘积项)一一相对应,个数相等设为 M,输出神经元个数为 1 个.

Step 2 隐层神经元权矢量的大小按照上述与运算规律选取,输出神经元的权矢量按照或运算的规律选取,则总共需要 $M+1$ 个神经元即可实现任意函数.

Step 3 设 $\boldsymbol{X}=(x_1,x_2,\cdots,x_n)$为其输入矢量,$w_{1hg}$ 是隐层第 g 个神经元与第 h 个输入变量的权,z_g 为隐层第 g 个神经元加权和,P_g 为第 g 个神经元输出,w_{2g}为输出层的连接权,z_o 为输出神经元加权和,P_o 为最后输出. 它们满足下面的关系式:

$$z_g=\sum_{h=1}^{n} w_{1hg}x_h+w_{10},\quad P_g=P(z_g) \tag{12}$$

$$z_o=\sum_{g=1}^{M} w_{2g}P_g+w_{20},\quad P_o=P(z_o) \tag{13}$$

对于隐层:
$$\begin{cases}w_{1hg}=\boldsymbol{x}_h^g\cdot\mathbf{i}\\ \boldsymbol{w}_{10}=(\boldsymbol{n}-1)\mathbf{i}\end{cases} \tag{14}$$

对于输出层:
$$\begin{cases}\boldsymbol{w}_{2g}=\mathbf{i}\\ \boldsymbol{w}_{20}=-(\boldsymbol{M}-1)\mathbf{i}\end{cases} \tag{15}$$

式(14)中,$\boldsymbol{x}_h^g$ 表示连接到第 $\boldsymbol{g}$ 个神经元的第 $\boldsymbol{h}$ 个输入变量,下面证明算法正确性.

证明 当网络的输入 $\boldsymbol{X}=\boldsymbol{X}^r=(x_1^r,x_2^r,\cdots,x_n^r)$是第 r 个 $f=1$ 的最小项. 则若 $g=r$,$z_g=z_r=\sum_{h=1}^{n} w_{1hg}x_h+w_{10}=n\mathrm{i}+(n-1)\mathrm{i}=-\mathrm{i}$,则 $P_g=P(z_g)=P(z_r)=-1$. 对任意的 $g\neq r$,$z_g/\mathrm{i}=(\sum_{h=1}^{n} w_{1hg}x_h+w_{10})/\mathrm{i}\geqslant-(n-2)+(n-1)\geqslant 1$,$P_g=P(z_g)=1$,该神经元处于抑制状态. 此时输出层神经元:$z_o/\mathrm{i}=(\sum_{g=1}^{M} w_{2g}P_g+w_{20})/\mathrm{i}\leqslant-1$,则输出可表示为 $P_o=P(z_o)=-1$,可见当输入为某个最小项时,输出确实为 -1. 而当输入 $\boldsymbol{X}=\boldsymbol{X}^q$ 不是某个最小项时:$z_g/\mathrm{i}=(\sum_{h=1}^{n} w_{1hg}x_h+w_{10})/\mathrm{i}\geqslant-(n-2)+(n-1)\geqslant 1$,所有隐层神经元都处于抑制状态. 输出为:$z_o/\mathrm{i}=(\sum_{g=1}^{M} w_{2g}P_g+w_{20})/\mathrm{i}=M-(M-1)=1$,$P_o=P(z_o)=1$. 神经元为抑制状态,输出 1.

证毕.

举例 设某个四变量函数用最小项表示为 $f_1(X)=((2,3,4,7,10,11,14,15)$,用该神经元实现. 由于该函数含 8 个最小项,则按上述基本算法隐层用 8 个神经元对应每个最小项,1 个输出神经元,总共可用 9 个神经元实现该函数. 但这种算法的效率不高,如果把其化简为 $f_1(X)=\bar{x}_1x_2\bar{x}_3\bar{x}_4+x_3x_4+\bar{x}_2x_3+x_1x_3$,则隐层只需要 4 个神经元,因此总共 5 个神经元就能实现该函数,效率得到了提高. 要进一步提高学习算法的效率,可采用卡诺图化简和最小项抑制的思想,如图 3 所示,先把它化为 $f_1(X)=\bar{x}_1x_2\bar{x}_3\bar{x}_4+x_3$,此时隐层只要 2 个神经元,权矢量的选取仍参照上述方法. 但最小项 $X=(1-1-11)$出现时,其输出为错误的 -1,因此必须抑制掉. 方法是隐层增加一个抑制神经元,该神经元的权矢量与普通最小项一致为$[3\mathrm{i},-\mathrm{i},\mathrm{i},\mathrm{i},-\mathrm{i}]$. 所不同的是,该神经元与输出神经元相连的权矢量不是正常情况下

式(15)中的 i,而应改为 – i,这时候若当该项出现时,输出的结果又会回到正确的 1,用这种方法后实现 $f_1(X)$ 函数只需要 4 个神经元,进一步提高了算法的效率.而采用复数权神经元实现布尔函数的特殊优点在于能够设法将某些复杂的布尔函数化成异或运算,如此可大大提高算法的实现效率,这具有其他神经元无法比拟的优势.该算法中关键的一步是将布尔函数表达成较为简单的异或(⊕)运算,再用神经元来实现之.

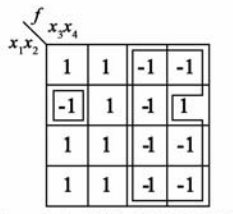


图3　布尔函数卡诺图化简与最小项抑制

一般布尔函数的 SOP 形式转换为异或运算的方法有多种.这里采用适合计算机使用的代数法[2,18].先引入 Kronecker 矩阵: $K_1 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$,其 n 次方为 $K_n = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes n}$,它的一个性质是逆阵即为其本身,即 $K_n^{-1} = K_n$,接下来的转换过程经过两步:

Step 1　将函数 $f(\boldsymbol{X})$用最小项表示成(0,1)形式的矢量 $\boldsymbol{F}$,并求得 $\boldsymbol{F}' = K_n^{-1}\boldsymbol{F}$;

Step 2　在实数域将 $\boldsymbol{F}'$变换成 $\boldsymbol{F}'' = K_n\boldsymbol{F}'$,矢量 $\boldsymbol{F}''$即包含了多阈值信息,而该多阈值信息可以反映输出逻辑值.

举例　三变量函数用最小项和代数形式表示为: $f_2(\boldsymbol{X}) = (1,2,4,7) = x_1 \oplus x_2 \oplus x_3$,将其表示成矢量形式为 $f_2(\boldsymbol{X}) = (01101001)$,因为 $n = 3$,所以 $\boldsymbol{F}_2' = K_3^{-1}f_2 = K_3 f_2 = (01101000)$, $\boldsymbol{F}_2'' = K_3\boldsymbol{F}_2' = (01121223)$,与 $f_2(\boldsymbol{X})$对照可知,该函数包含的多阈值信息可以很好的反映其最小项,即只有最小项(1,2,4,7)对应的阈值为(1113),编码后输出是 – 1,而其他项输出均为 1.此时只要令式(3)中的 $m = 2n = 6$,就能用 1 个神经元实现上述的三变量异或运算,计算的权矢量为 $\boldsymbol{W} = \lfloor 0, \sqrt{3}, \sqrt{3}, 2\mathrm{i} \rfloor$,且具有较好的稳健性能,而采用一般稳健算法需 5 个神经元.又如上述的函数 $f_1(\boldsymbol{X})$也可表示成 $f_1(\boldsymbol{X}) = x_3 \oplus \bar{x}_1 x_2 \bar{x}_4$,则实现该函数只需要用 2 个神经元,1 个实现与运算,1 个实现异或运算,这是目前稳健神经元实现布尔函数最简便有效的方法.

5　结论

本文实现了一种基于复数权神经元的布尔函数稳健学习算法,该算法的基本思想是简单逻辑可由单个稳健神经元稳健设计实现,而任意复杂布尔函数可由简单逻辑组成.通过具体实例研究表面,应用该算法不但能很好的实现基于神经元的布尔函数的稳健设计,而且实现的算法具有很大的灵活性和优越性,说明了该神经元较强的信息处理能力,也为各种类型的复杂神经元实现布尔函数稳健设计提供了借鉴思路.

参考文献

[1] C Lau edited. Neural Networks: Theoretical Foundation and Analysis[M]. New York: IEEE Press, 1992.

[2] N N Aizenberg, I N Aizenberg. CNN-like networks based on multi-valued and universal binary neurons: learning and application to image processing[A]. Proceedings of Third IEEE International Workshop on Cellular Neural Networks and Their Applications[C]. Rome, Italy, 1994.153 – 158.

[3] N N Aizenberg, I N Aizenberg, G Krivosheev. Multi-valued and universal binary neurons: mathematical model, learning, networks, application to image processing and pattern recognition[A]. Proceedings of the 13 International Conference on Pattern Recognition[C]. Vienna, Austria, 1996.185 – 189.

[4] S Jankowski, A Lozowski, M Zurada. Complex-valued multistate neural associative memory[J]. IEEE Trans on Neural Networks, 1996, (7): 1491 – 1496.

[5] I N Aizenberg, N N Aizenberg, C Butakov, Farberov E. Image recognition on the neural network based on multi-valued neurons[A]. Proceedings of the 15 International Conference on Pattern Recognition[C]. Barcelona, Spain, 2000.989 – 992.

[6] S L Goh, D P Mandic. Nonlinear adaptive prediction of complex-valued signals by complex-valued PRNN[J]. IEEE Trans on Signal Processing, 2005, 53(5): 1827 – 1836.

[7] M K Muezzinoglu, C Guzelis, J M Zurada. A new design method for the complex-valued multistate Hopfield associative memory[J]. IEEE Trans on Neural Networks, 2003, 14(4): 891 – 899.

[8] P K Kalra, D Mishra, K Tyagi. A novel complex-valued counterpropagation network[A]. IEEE Symp on Computational Intelligence and Data Mining[C]. Honolulu, HI, 2007.81 – 87.

[9] I Nishikawa, K Sakakibara, T Iritani, Y Kuroe. 2 types of complex-valued hopfield networks and the application to a traffic signal control[A]. Int Joint Conf on Neural Networks[C]. Montreal, Canada, 2005.782 – 787.

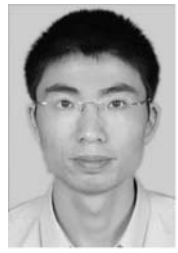
[10] I Nishikawa, T Iritani, K Sakakibara. Improvements of the traffic signal control by complex-valued hopfield networks [A]. Int Joint Conf on Neural Networks[C]. Vancouver, BC, 2006.459 – 464.

[11] I Nishikawa, K Hayashi, K Sakakibara. Complex-valued neuron to describe the dynamics after hopf bifurcation: An example of CPG model for a biped locomotion[A]. Proc Int Joint Conf on Neural Networks[C]. Orlando, FL, 2007.327 – 332.

[12] D L Gray, A Miched. A training algorithm for binary feedforward neural networks[J]. IEEE Transaction on Neural Net-

works,1992,3(2):176 – 194.
[13] A K Hundiwal. An improved algorithm for learning representations in Boolean neural networks[J]. IEEE Transaction on Circuits and Systems,1993,16 – 18:592 – 595.
[14] 张军英,保铮.二进网络的最稳健设计[J].电子学报,1997,25(10):37 – 43.
Zhang Jun-ying, Bao Zheng. The most-robust design of binary networks[J]. Acta Electronica Sinica, 1997, 25(10): 37 – 43. (in Chinese)
[15] 张军英,保铮.前向网络隐空间分类超平面的构造[J].电子学报,1999,27(1):136 – 139.
Zhang Jun-ying, Bao Zheng. Construction of classification hyperplanes in hidden space for feedforward neural networks[J]. Acta Electronica Sinica, 1999, 27(1): 136 – 139. (in Chinese)
[16] 吕伟锋,林弥,华柏兴.基于复数权值神经元的数字逻辑最稳健设计[J].电路与系统学报,2008,13(4):77 – 80.
Lv Wei-feng, Lin Mi. Robust design of digital logic based on neurons with complex-valued weights[J]. Journal of Circuits and Systems, 2008, 13(4): 77 – 80. (in Chinese)
[17] Wei-feng Lü, Mi LIN, Ling-ling SUN. The most robust design for digital logics of multiple variables based on neurons with complex-valued weights[J]. Journal of Zhejiang University (science A), 2009, 10(2): 184 – 188.
[18] 陈偕雄,沈继忠.近代数字理论[M].杭州:浙江大学出版社,2001.1 – 19.
Chen Xie-xiong, Shen Ji-zhong. Modern Digital Theory[M]. Hangzhou: Zhejiang University Press, 2001. 1 – 19. (in Chinese)

作者简介



吕伟锋 男,1977 年生于浙江省桐乡市,杭州电子科技大学教师,浙江大学在职博士生.主要研究方向为,神经网络、稳健设计、集成电路统计 CAD 算法等.
E-mail:lvwf@hdu.edu.cn