具有二维状态转移结构的随机逻辑及其在神经网络中的应用

季 渊*①② 陈文栋① 冉 峰①② 张金艺① David LILJA③
①(上海大学微电子研究与开发中心 上海 200072)
②(上海大学机电工程与自动化学院 上海 200072)
③(明尼苏达大学电子与计算机工程系 明尼阿波利斯 55455)

摘 要：随机计算是一种特殊的基于概率数据码流的数学计算方法，其优点在于可以采用非常简单的数字逻辑完成复杂数学运算，从而大幅降低硬件实现成本。该文首先讨论了随机计算的基本原理和主要运算逻辑，论述了传统线性状态机的不足，并分析了一种 2 维状态转移拓扑结构，推导了通过 2 维有限状态机实现高斯函数的方法。在此基础上，提出一种随机径向基函数神经网络模型，其硬件实现成本非常低，而性能与传统神经网络相当。两类模式识别实验结果显示，所提出的随机径向基函数神经网络的输出值均方误差与相应结构传统神经网络的差别小于 1.3%。FPGA 实验结果显示，数据宽度为 12 位时，随机中间神经元的电路面积仅为传统插值查表结构的 1.2%、坐标旋转数字计算方法(CORDIC)的 2%。通过改变输入码流长度，该神经网络可以在处理速度、功耗和准确性之间作出平衡，具有应用灵活性，适用于对成本、功耗要求较高的应用如嵌入式、便携式、穿戴式设备。

关键词：随机计算；人工神经网络；径向基函数；模式识别
中图分类号： TP302.7 **文献标识码**： A **文章编号**：1009-5896(2016)08-2099-08
DOI: 10.11999/JEIT151233

Stochastic Logics with Two-dimensional State Transfer Structure and Its Application in the Artificial Neural Network

JI Yuan①② CHEN Wendong① RAN Feng①② ZHANG Jinyi① David LILJA③
①(*Microelectronic Research and Development Center*, *Shanghai University*, *Shanghai* 200072, *China*)
②(*School of Mechatronic Engineering and Automation*, *Shanghai University*, *Shanghai* 200072, *China*)
③(*Department of Electrical and Computer Engineering*, *University of Minnesota*, *Minneapolis* 55455, *USA*)

Abstract: Stochastic computing is a special algorithm that performs mathematical operations with probabilistic values of bit streams rather than traditional deterministic values. The main advantage of stochastic computing is its great simplicity of hardware arithmetic units for mathematical operations to reduce the circuit cost. This paper discusses the principle of the stochastic computing and its main arithmetic logic. It analyzes a two-dimension state transition topology structure, and discusses the Gaussian function implementation method based on the two-dimension Finite State Machin (FSM). Then, a low cost stochastic radial basis function neural network model is proposed. Results from two pattern recognition tests show that the difference of the mean squared error between the stochastic network output value and the corresponding deterministic network output value can be less than 1.3%. FPGA implementation results show that the hardware resource requirement of the proposed stochastic hidden neuron is only 1.2% of the corresponding deterministic hidden neuron with the interpolated look-up table, and is 2.0% of the CORDIC algorithm. The accuracy, speed and power of the stochastic network can be tradeoff dynamically. This network is suitable for the low cost and low power applications like embedded, portable and wearable devices.

Key words: Stochastic computing; Artificial neural network; Radial Basis Function (RBF); Pattern recognition

1 引言

随机计算(stochastic computing)[1–4]是一种特殊的基于非确定性数据的计算方法，最初由文献[1]提出，它以比特码流中出现 1(或 0)的概率作为运算数值，在概率域中进行数学运算。与传统的基于确定性数的计算方法(deterministic computing)相比，随机计算的最大优势在于所需要的运算逻辑资源非常小，例如，两个数之间的乘法运算可以简单地用一个二输入与门来实现[5]。其次，由于随机计算采用概率数作为运算数据，其运算过程中的输入噪声或数据错误不会导致最终结果的完全偏离，因此，随

收稿日期：2015-11-03；改回日期：2016-04-08；网络出版：2016-05-24
*通信作者：季渊 jiyuan@shu.edu.cn
基金项目：国家自然科学基金(61376028)
Foundation Item: The National Natural Science Foundation of China (61376028)

机计算的抗干扰能力强于传统的确定性计算。第三，随机计算的运算精度与比特码流的长度相关，更长的码流可以产生更高的数据精度，这给系统设计带来了较好的灵活性，在同一个硬件系统中，可以根据应用场景特点采用不同的码流长度，实现运算速度、电路功耗和运算精度的动态调整。

正是由于这些特点，随机计算非常适合于并行计算[6]、容错计算[7]、可靠性计算[8]以及一些对数据精度要求并不十分严格的运算密集型应用，例如通信随机编码[9]、图像处理[10]，尤其是人工神经网络[11]。由于神经网络包含大量乘法、加法和指数运算，消耗大量逻辑运算资源，因此用传统方法进行硬件实现的成本较高[12,13]，而基于随机计算的神经网络(也称为随机神经网络)由于运算逻辑简单，更容易组成大规模网络结构。文献[5]提出了比例缩减加法器、线性有限状态机、随机除法器等随机计算逻辑单元，改进了随机神经网络的性能，并将这些改进型随机逻辑单元用于一种基于软竞争学习算法的神经网络中进行字符识别应用[14]。然而，这种随机神经网络仍采用了单一的 sigmoid 激活函数，网络类型和网络性能都受到一定限制。

近年来，文献[15]提出了一种基于马尔可夫链(Markov chain)的 2 维有限状态机(FSM)拓扑结构，本文在此基础上，推导了利用 2 维有限状态机来实现高斯函数的具体过程，并根据码流的全相关性，利用一个简单的 XNOR 门实现了减法运算，然后讨论了将随机逻辑应用于径向基函数(Radial Basis Function, RBF)神经网络的方法[16]，得到随机径向基函数神经网络，将该网络应用于两类模式识别实验(鸢尾花识别和光学字符识别)，最后在 FPGA 上得到验证。

2 随机逻辑运算单元

2.1 数值定义

随机计算中，传统的确定性数据都转化为概率数据。设 $X(t)$为一个长度为 L 的比特码流(L 为正整数)，$t=0,1,2,\cdots,L$, $X(t)\in[0,1]$，将 $X(t)$简记为 X，令 x表示 X 中出现数字 1 的概率，即 $x=P(X=1)$，例如，码流 $X=0010011001010001$ 对应的 x为 0.375。随机计算可采用两种数据格式：无符号数(Unipolar)和有符号数(Bipolar)，分别用 y_u和 y_s表示，其取值大小和范围由式(1)和式(2)给出：

$$y_u = x\left(0 \le y_u \le 1\right) \tag{1}$$

$$y_s = 2y_u - 1\left(-1 \le y_u \le 1\right) \tag{2}$$

根据式(1)和式(2)可以得到，码流 X 作为无符号数和有符号数时分别表示 0.375 和−0.25(即 y_u=0.375, $y_s=-0.25$)。这两种数据表示方式本质上相同，可以同时出现在同一个系统中。确定性数可通过比较器和随机数发器转换为随机码流，随机码流可通过二进制计数器转换为确定性数[5]。

2.2 传统随机逻辑运算单元

2.2.1 乘法 假定随机码流 A 和 B 均为伯努利(Bernoulli)序列[17]，用 $P(X)$表示码流 X 中所含有数字 1 的概率，若采用无符号数据格式，则码流 A 和 B 之间的乘法可以用一个逻辑与门(AND)来实现，若采用有符号数据格式，则 A 和 B 之间的乘法可以用一个异或非门(XNOR)来实现[5]。

2.2.2 加法 早期加法用或门(OR)实现，但存在误差[1]。文献[5]提出一种比例缩放加法器，利用多路选择器(MUX)实现加法，但是比例缩放加法器每经过一次运算，输出值精度减半，因此仍然存在一定误差。本文在实现径向基函数时，将加法运算转化为乘法运算，从而避免了精度损失问题，第 3 节将详述。

2.2.3 线性状态机 神经网络的激活函数一般为 sigmoid 函数。早期的随机神经网络采用二进制计数器、查找表、组合逻辑或者预定义码流来实现激活函数，但是性能都不太理想。文献[5]提出一种线性状态转移结构如图 1 所示，通过改变参数 P 和 Q 可以实现多种 sigmoid 函数。然而，当利用该线性状态机实现指数函数时，状态数 N 往往大于 64，需要较多硬件资源，且在零点的近似效果不理想，无法保证指数函数在零点对称。

2.3 增强型随机逻辑运算单元

2.3.1 减法 传统随机计算的输入序列必须保持不相关[1,5]。但是，如果使输入序列完全相关(例如，由相同的随机数源发生器产生)，则经过二输入 XNOR 门后，逻辑结果不再是乘法：当输入序列相应位的值不同时，输出值为 0，当输入序列相应位的值相同时，输出值为 1，这样，输出码流中 0 的数量代表 A 和 B 中不相同的位数，1 的数量代表 A 和 B 中相同的位数，因此，两个输入码流中数据不同的概率被检测出来，从而实现了减法。XNOR 门输入为两个无符号数码流，输出为这两个码流减法的绝对值(无符号数)。用 XOR 实现减法时，其输出为 XNOR 的逻辑取反。

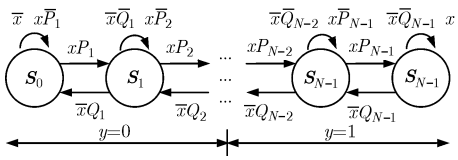


图1 线性状态转移结构

2.3.2 2 维状态机　考虑图 2 所示的状态转移拓扑结构，状态总数为 $M\times N$($\boldsymbol{S}_0$至$\boldsymbol{S}_{MN-1}$)，状态根据输入值在 2 维空间内向相邻状态转移，输入变量由输入码流 x和调制码流 k组成，共有 4 种组合：00, 01, 10, 11，该组合值决定了状态的跳转方向，例如 11 为向右跳转，00 为向左跳转，当状态处于最右或最左的状态时，即使输入码流为向右或向左跳转时，当前状态也保持不变，形成饱和；输入码流组合为 10 和 01 的过程类似。该状态变化过程可以由齐次时间的 Markov 链描述，其特征为一个不可约的、非周期性的、遍历性的状态机[15]。根据 Markov 定理，当状态转移次数足够多时，该结构可等效为一个与初值状态无关的概率分布。若用 P_{St}(t 为当前状态编号，t =0,1,⋯,MN–1)来表示状态 t 出现的概率，用 P_X 来表示输入随机码流 $\boldsymbol{x}$ 的概率值，用 P_K 来表示调制码流 $\boldsymbol{k}$ 的概率值，则 P_{St} 为

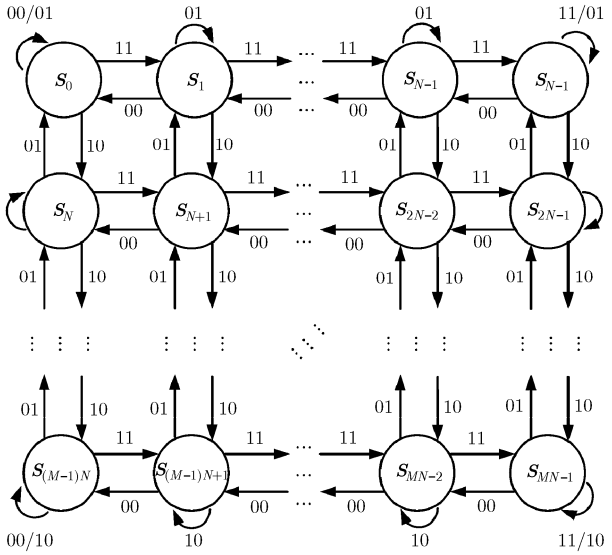


图 2　2 维状态转移结构

$$P_{St}=P_{i\times N+j}=\frac{t_x^i\cdot t_y^j}{\sum_{u=0}^{M-1}\sum_{u=0}^{N-1}t_x^u\cdot t_y^v} \tag{3}$$

其中，i 是水平方向上的状态编号，j 是垂直方向上的状态编号，t_x和 t_y为

$$t_x=\frac{P_X}{1-P_X}\cdot\frac{P_K}{1-P_K} \tag{4}$$

$$t_y=\frac{P_X}{1-P_X}\cdot\frac{1-P_K}{P_K} \tag{5}$$

由式(3)得到的概率可用于模拟复杂目标函数。图 3 给出了一种求近似函数的硬件架构，$\boldsymbol{x}$, $\boldsymbol{k}$, $\boldsymbol{q}_t$和 $\boldsymbol{y}$ 都是数据位宽为 1 bit 的随机码流，采用无符号数据格式，其中，$\boldsymbol{x}$ 和 $\boldsymbol{k}$ 为 2 维状态机(2D-FSM)的输

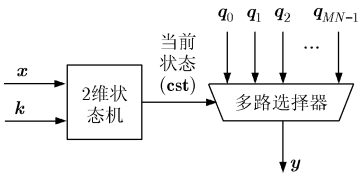


图 3 用随机计算实现复杂函数的逻辑电路结构

入码流，$\boldsymbol{q}_t$是一组预先定义好的随机参数码流，其值由 $\boldsymbol{P}_{qt}$表示。信号 **cst** 表示 2 维状态机的当前状态，连接到多路选择器(MUX)的选择端。若 **cst** 为 t(t=0, 1,⋯,MN–1)，则 $\boldsymbol{q}_t$就被选择作为输出端 $\boldsymbol{y}$ 的输出值。因此，$\boldsymbol{y}$ 的概率值(用 P_Y表示)可以由 **cst** 和 P_{qt}来调制[16]。

若设目标函数为 $T(P_X)$，则可定义目标误差为

$$\varepsilon=\int_0^1\left(T\left(P_X\right)-P_Y\right)^2\mathrm{d}\left(P_X\right) \tag{6}$$

由于 P_Y可以表示为 P_X, P_K和 P_{qt}的函数，因此通过求解二次函数 ε 的最小值，就可以得到调制码流 $\boldsymbol{k}$ 和 $\boldsymbol{q}_t$的概率值。表 1 列出了 3 种高斯函数作为目标函数的求解结果。由于高斯函数为对称函数，因此 P_K均为 0.5。P_{qt}通过 MATLAB 求得，其值关于中心点对称。P_{Y1}和 P_{Y2}的表达式相同，但是状态数不相同，它们都关于直线 $x=0.5$ 对称，逼近结果如图 4 所示。可以发现，在目标函数相同的情况下，8 状态和 16 状态的结构都较好逼近目标函数。与线性状态转移结构比较[5]，在实现相同函数时，2 维状态转移结构使用的状态数量大幅减少，输出曲线更光滑，且 2 维状态转移结构可以实现更多复杂函数。图 5 显示了 P_{Y3}的逼近效果，码流长度为 1 kbit 的输出误差较大，码流长度为 10 kbit 的输出误差大幅减少，说明输出函数的误差与随机码流的长度有关。图 4 和图 5 的结果表明，可以用一种简单的 2 维状态机来实现高斯函数。

3　随机神经网络模型

3.1 径向基函数神经网络分解

本文以一个 3 层径向基函数网络为例。网络输入层包含了输入对象的原始特征，这些特征用矢量 $\boldsymbol{x}_i$来表示(i =1,2,⋯,I, I 为输入神经元个数)。中间层包含了径向基函数对输入神经元提取的抽象特征，用矢量 $\boldsymbol{y}_j$来表示(j=1,2,⋯,J, J 为中间神经元个数)，采用高斯函数作为径向基函数，即

$$\boldsymbol{y}_j=\exp\left(\frac{\left\|\boldsymbol{x}_i-\boldsymbol{c}_{ij}\right\|^2}{-\sigma^2}\right) \tag{7}$$

其中，$\boldsymbol{c}_{ij}$ 为高斯函数的中心点，σ^2 控制了高斯函数的弯曲程度。矢量 $\boldsymbol{z}_k$为输出神经元(k =1,2,⋯,K, K

表 1 2 维状态转移结构近似目标函数中的参数

目标函数 $T(P_X)$	状态机状态数	P_K	P_{qt}
P_{Y1}=0.25exp[$(P_X$-0.5$)^2$/-0.08]	8 (M= 2, N= 4)	0.5	P_{q0}=0.011, P_{q1}=0.010, P_{q2}=0.973, P_{q3}=0, P_{q4}= 0, P_{q5}=0.973, P_{q6}=0.010, P_{q7}=0.011
P_{Y2}=0.25exp[$(P_X$-0.5$)^2$/-0.08]	16 (M= 4, N= 4)	0.5	P_{q0}=0.011, P_{q1}=0.070, P_{q2}=0, P_{q3}=0, P_{q4}=0.070, P_{q5}=0.60, P_{q6}=0.70, P_{q7}=0.60, P_{q8}=0.60, P_{q9}=0.70, P_{q10}=0.045, P_{q11}=0.07, P_{q12}=0, P_{q13}=0, P_{q14}=0.07, P_{q15}=0.011
P_{Y3}=exp(P_X^2/-2)	8 (M= 2, N= 4)	0.5	P_{q0}=1, P_{q1}=1, P_{q2}=1, P_{q3}=1, P_{q4}=0.990, P_{q5}=0.591, P_{q6}=0.867, P_{q7}=0.607

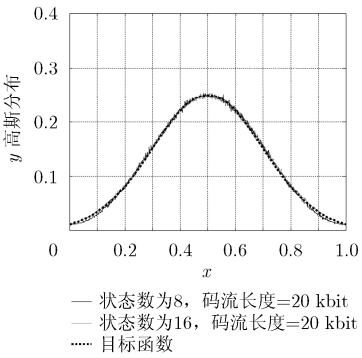


图 4 用 2 维状态机综合的高斯函数 P_{Y1} 和 P_{Y2}

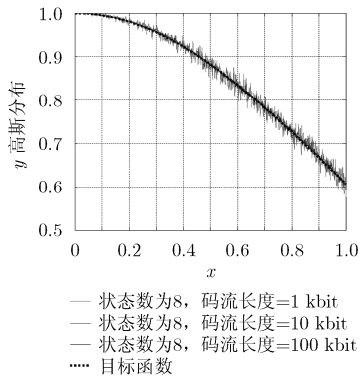


图 5 用 2 维状态机综合的高斯函数 P_{Y3}

为输出神经元个数)，其值是中间神经元的线性组合，即

$$\boldsymbol{z}_k = \boldsymbol{y}_j \boldsymbol{w}_{jk} + \boldsymbol{b}_k \tag{8}$$

其中，$\boldsymbol{w}_{jk}$是权值矩阵，$\boldsymbol{b}_k$是线性偏置。$\boldsymbol{c}_{ij}$, σ , $\boldsymbol{q}_t$, $\boldsymbol{w}_{jk}$和 $\boldsymbol{b}_k$的值在网络训练过程中确定。然而，式(7)表示的径向基函数并不容易直接用硬件实现，因此，把该范数展开得到：

$$\begin{aligned}\boldsymbol{y}_j &= \exp\left(\frac{\left(x_1 - c_{1j}\right)^2 + \left(x_2 - c_{2j}\right)^2 + \cdots + \left(x_I - c_{Ij}\right)^2}{-\sigma^2}\right) \\ &= \exp\left(\frac{\left(x_1 - c_{1j}\right)^2}{-\sigma^2}\right)\exp\left(\frac{\left(x_2 - c_{2j}\right)^2}{-\sigma^2}\right)\cdots \\ &\quad \cdot \exp\left(\frac{\left(x_I - c_{Ij}\right)^2}{-\sigma^2}\right)\end{aligned} \tag{9}$$

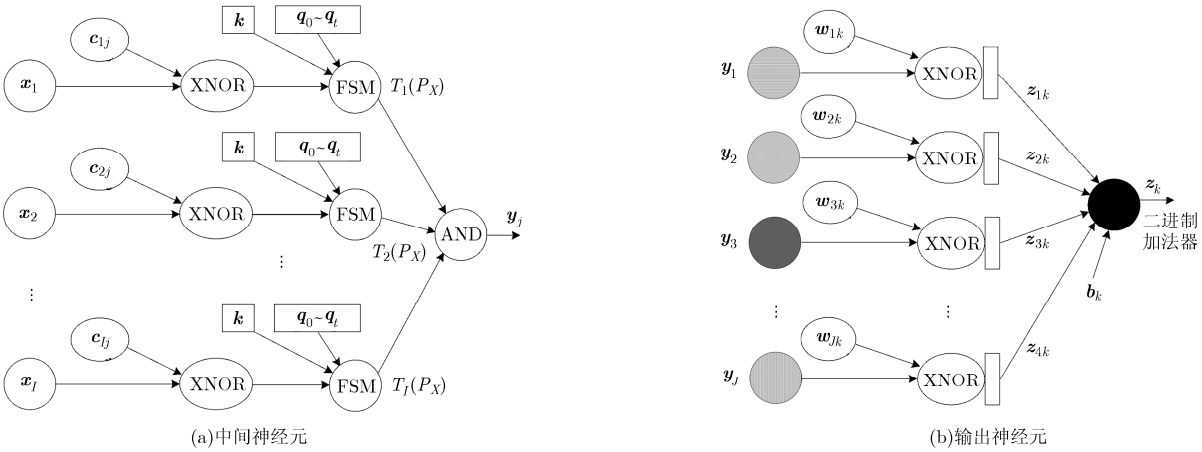
可见，$\boldsymbol{y}_j$的计算包含 3 个计算过程：(1)输入神经元 $\boldsymbol{x}_i$和高斯函数中心点 $\boldsymbol{c}_{ij}$的差值；(2)中心点为零的高斯函数；(3)各高斯函数间的乘积。

3.2 随机径向基函数神经网络

根据式(9)构造一个由随机逻辑实现的径向基函数神经网络(以下简称随机 RBF 网络)，其输入神经元将输入的确定性二进制数 $\boldsymbol{d}_i$转换为随机码流 $\boldsymbol{x}_i$，中间神经元和输出神经元的结构如图 6 所示，网络参数 $\boldsymbol{c}_{ij}$, $\boldsymbol{k}$, $\boldsymbol{q}_t$, $\boldsymbol{w}_{jk}$和 $\boldsymbol{b}_k$存储于外部存储器中，通过移位寄存器输入网络。

对于中间神经元，高斯函数的中心点 $\boldsymbol{c}_{ij}$被转换为随机码流，其所使用的随机源与输入神经元使用的随机源相同，以保证 $\boldsymbol{x}_i$和 $\boldsymbol{c}_{ij}$完全相关，从而使用 XNOR 门将随机码流 $\boldsymbol{x}_i$和高斯函数中心点 $\boldsymbol{c}_{ij}$相减(如 2.3.1 节所述)，所得 I个差值的绝对值经过 I个 2 维状态机(如图 3 所示)完成高斯函数计算，其中，调制码流 $\boldsymbol{k}$ 和 $\boldsymbol{q}_t$由独立的随机数发生器产生，以保证所得高斯函数的输出值 $T_i(P_X)$各不相关，从而可以使用一个 AND 门将该中间神经元中所有高斯函数的输出值相乘，得到中间神经元的输出 $\boldsymbol{y}_j$。

输出神经元可由两种方式实现，一种是采用随机逻辑，另一种是采用传统的确定性逻辑。对于传统的确定性逻辑，可以先将中间神经元输出的随机码流 $\boldsymbol{y}_j$转换为确定性二进制数，然后采用一组乘法器和加法器进行式(8)的线性迭代运算，得到输出神经元的值 $\boldsymbol{z}_k$，每个输出神经元的结构相同，但权值 $\boldsymbol{w}_{jk}$和偏置值 $\boldsymbol{b}_k$不同。对于随机逻辑，由于输出神经元的权值 $\boldsymbol{w}_{jk}$和偏置 $\boldsymbol{b}_k$可能大于 1，因此先要将 $\boldsymbol{w}_{jk}$和 $\boldsymbol{b}_k$进行等比例缩小，使其取值范围为[-1, 1]，然后使用 XNOR 门完成 $\boldsymbol{y}_j$和 $\boldsymbol{w}_{jk}$有符号乘法计算，最后将结果乘以原来的缩小系数，形成 J路随机码流 $\boldsymbol{z}_{ik}$。为了避免随机加法运算，该网络先将 J 路随机码流



(a)中间神经元　　(b)输出神经元

图6 随机径向基函数神经网络的神经元结构

z_{ik} 转换为确定性二进制数，利用确定数的二进制加法器将转换结果相加，得到输出神经元的输出结果 z_k，从而提高了精度。

4　实验与结果讨论

4.1 鸢尾花识别实验

鸢尾花数据组包含 3 种类型鸢尾花，每种花各有 50 组样本，每组样本均含有 4 种数据：萼片、花瓣的长度和宽度。为了识别花的类型，首先构建一个确定性数 RBF 网络，该网络包含 4 个输入神经元、8 个中间神经元和 3 个输出神经元，随机抽取 75 个样本作为训练集，剩余 75 个样本作为测试集，采用正交最小二乘法训练，得到测试样本识别率为 97.33%。然后构建两个同样结构(4×8×3)的随机 RBF 神经网络 M1 和 M2, M1 的中间神经元采用图 6(a)的随机逻辑、输出神经元采用确定逻辑，M2 的所有神经元均采用图 6 的随机逻辑。M1 和 M2 中间神经元 2 维状态机的状态数均为 8。对 M1 和 M2 分别进行鸢尾花种类识别测试，每个测试项重复 2000 次以减少随机数波动影响，测试结果如图 7 所示。MSE 用于衡量随机网络与确定网络的偏离程度。在 M1 中，当测试码流为 10 kbit 时，平均识别率为 93.4%；当测试码流为 500 kbit 时，平均识别率达到 96.7%。在 M2 中，当测试码流为 1 Mbit 时，MSE 为 0.051，比相同情况 M1 的 MSE 高 20%，这是因为 M2 的输出神经元采用了随机加法器。该误差可以通过增大随机码流长度得到改善，这一事实也证明了随机码流可以动态调整运算速度和精度。

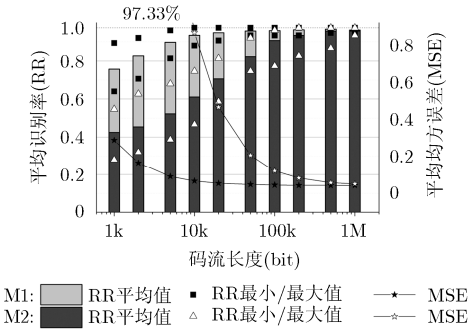


图 7 随机径向基函数网络 M1 和 M2 对鸢尾花的识别率(柱体)和 MSE(实线)

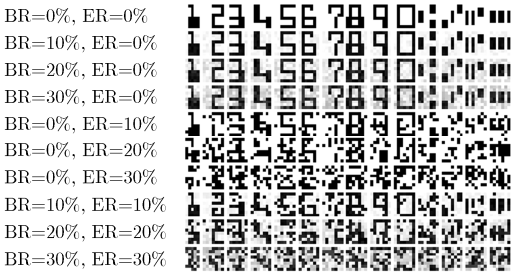
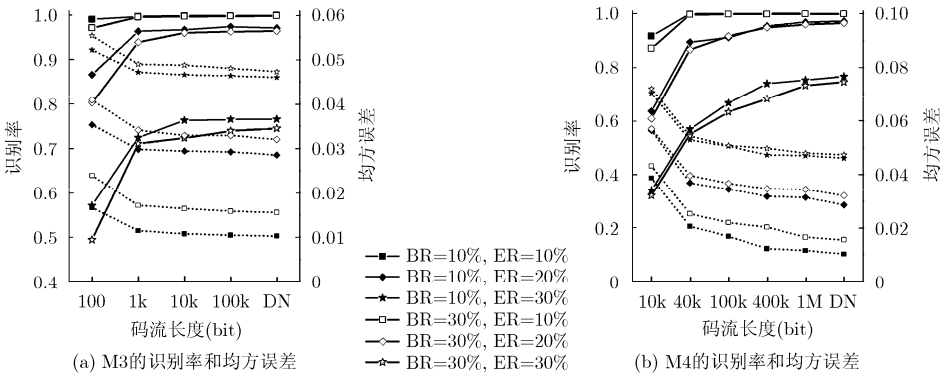


图 8　E-13B MICR 字体中注入数据噪声

4.2 光学字符识别实验

采用 E-13B MICR 字体作为识别对象。在标准 MICR 字体中加入两种随机数据噪声：像素模糊和像素错误，分别用 BR(Blur Rate)和 ER(Error Rate)表征，前者指像素灰度的改变程度，后者指像素灰度值完全错误的比例，如图 8 所示。每个字符均采用 7×8 像素阵列描述，像素灰度值经过标准化后输入至随机径向基函数神经网络，网络拥有 56 个输入神经元、15 个中间神经元和 14 个输出神经元，每一个输出神经元都代表了 MICR 字体识别结果。网络采用正交最小二乘法训练后得到参数 $\boldsymbol{c}_{ij}$, $\boldsymbol{k}$, $\boldsymbol{q}_t$, $\boldsymbol{w}_{jk}$ 和 $\boldsymbol{b}_k$。实验采用网络结构 M3 和 M4，其中，M3 的中间神经元采用随机逻辑、输出神经元采用确定性逻辑，M4 的中间神经元和输出神经元均采用随机逻辑。采用 MATLAB 对 M3 建模和仿真，识别注入随机数据噪声的 MICR 字体。图 9 示意了不同码流长度的识别结果。对于网络 M3，当码流长度从 1 kbit 增加到 100 kbit 时，字体识别率有 2%~3%的改善，



(a) M3的识别率和均方误差 (b) M4的识别率和均方误差

图 9 随机径向基函数网络和对应确定性网络的 MICR 字体识别率(实线)和 MSE(虚线)(DN：确定性神经网络)

MSE 有 5%~8%的改善，字体识别率和 MSE 最终向确定性网络(DN)的结果收敛。对于网络 M4，当 ER 为 10%，且码流长度大于 40 kbit 时，字体识别率几乎为 100%；当 ER 为 20%，码流长度从 40 kbit 增加到 1 Mbit 时，字符识别率从 91%增加到 96.7%，最终收敛于相应的确定性网络。

上述数据为 2000 次实验结果的平均值。图 10 记录了 100 次 MSE 实验结果。对于 M3，码流长度 1 kbit 时的 MSE 与相应确定性网络的 MSE 差别为 3.26%，码流长度 10 kbit 时的差别为 1.3%；对于 M4，码流长度为 100 kbit 时差别为 7.61%，码流长度为 1 Mbit 时的差别为 2.17%。这些 MSE 差别均小于文献[5]所提出的软竞争学习神经网络的字符识别实验[14]，在文献[5]实验中，确定性网络和随机网络的 MSE 差别在第 1 层软件竞争学习网络中为 31% (0.545 vs 0.715)，在第 2 层线性网络中为 10%(4.77 vs 5.26)。

4.3 硬件实现比较

采用 Altera Cyclone III FPGA(EP3C80F780C8)对网络结构 M1~M4 进行硬件实现，该 FPGA 共有 81264 个逻辑单元 LE 和 430 个管脚，综合结果如表 2 所示。网络参数($\boldsymbol{c}_{ij}$, $\boldsymbol{k}$, $\boldsymbol{q}_t$, $\boldsymbol{w}_{jk}$和 $\boldsymbol{b}_k$)存储于外部存储器中，通过移位寄存器移入网络。对于 M1 和 M3，

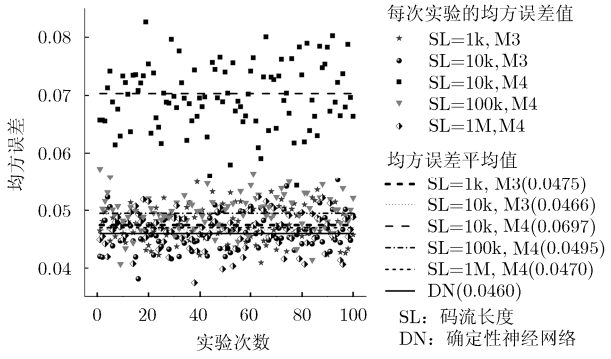


图 10 MSE 的随机波动(BR=10%, ER=30%)，SL：码流长度(Stream length)

中间神经元采用随机逻辑实现，输出神经元采用二进制加法器和乘法器实现，对于 M2 和 M4，中间神经元和输出神经元均采用随机逻辑实现。当数据位宽增加时，两个网络结构的总规模都呈现增长趋势，但是中间神经元的规模保持不变，说明基于随机逻辑的中间神经元规模与数据位宽无关。M2, M4 的输出神经元规模分别大于 M1, M3，随着数据位宽增加，其占系统总资源的比例不断扩大，说明与传统的确定性逻辑相比，随机逻辑的硬件资源优势随着网络规模的增大变得越来越明显。

4.4 综合性能比较

表 3 列出了给定 FPGA 的前提下，不同 RBF 网络实现方式的硬件资源和性能比较。A 为电路面积，以摊销至每一个网络输入所需要的中间神经元 LE 数量作为电路面积资源的衡量单位(电路面积除以输入神经元总数再除以中间神经元总数)；C 为时钟，以每完成一次识别操作所需要的时钟数作为网络性能的衡量单位；O 为操作次数，以每秒能够完成的最大操作次数作为网络性能的衡量单位。从表中可见，随机 RBF 网络的 LE 摊销值为 22 个，远小于其他 3 种方法。全查找表结构虽然只需要一个时钟周期来完成操作，但是所需要的查找表过于庞大。插值查找方式减少了查找表规模，但是增加了计算时间。CORDIC(坐标旋转数字计算)算法是一种流行的指数函数计算方法，它只需要二进制加法器，比查找表规模更小。这几种算法的性能通过 $R0, R1, R2$ 3 种指标衡量，其中，$R0$ 为面积 A 和时钟 C 的乘积的倒数($R0=1/(AC)$)，代表面积和性能的综合指标；$R1$ 为面积 A 的平方和时钟 C 的乘积的倒数($R1=1/(A^2C)$)，代表以面积为优先的衡量指标；$R2$ 为操作次数 O 除以面积 A($R2=(O/A)$)，代表单位面积可以达到的操作性能。其比较结果如图 11 所示，纵坐标为归一化后的计分值。总体看来，10 位随机网络结构拥有最好的 $R0, R1$ 和 $R2$ 指标，但是精度偏低。在同样 12 位的数据宽度情况下，若以 $R1$ 为考查指

表2 随机径向基函数神经网络的FPGA实现资源，单位：LE(逻辑单元)，网络规模：56×15×14

网络结构(输入×中间×输出)4×8×3					
网络结构	控制器	中间层	输出层	LFSR	总资源
M1(10 bit)	720(40%)	712(40%)	294(16%)	60(3%)	1786
M1(20 bit)	1440(44%)	712(22%)	986(30%)	120(4%)	3258
M2(10 bit)	720(44%)	712(44%)	132(8%)	70(4%)	1634
M2(20 bit)	1440(57%)	712(28%)	254(10%)	140(5%)	2546
网络结构(输入×中间×输出)56×15×14					
M3(10 bit)	11290(32%)	18765(53%)	4990(14%)	590(2%)	35635
M3(20 bit)	22580(39%)	18765(32%)	15609(27%)	1180(2%)	58134
M4(10 bit)	11290(33%)	18765(55%)	3668(11%)	590(2%)	34313
M4(20 bit)	22580(45%)	18765(37%)	8092(16%)	1180(2%)	50617

表3 不同实现方式的径向基函数神经网络的硬件资源和性能比较

硬件结构	A	C	O	$R0$	$R1$	$R2$
10 位随机网络	22	1024	244	1	1	1
12 位随机网络	22	4096	61	0.20	0.25	0.23
14 位随机网络	22	16384	15	<0.01	0.06	0.04
12 位全查找表	33898	1	10000	0.64	<0.01	<0.01
12 位查找表(插值)	1853	15	3333	0.80	0.01	0.11
12 位 CORDIC	1079	22	5455	0.95	0.02	0.28

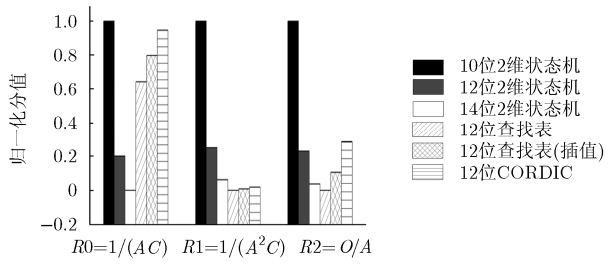


图 11 不同网络实现方式的性能比较

标(即强调面积优先)，则随机网络拥有最高分值，若以 $R2$ 为考查指标时(即强调单位面积性能)，则随机网络与 CORDIC 拥有较高的分值。若提高数据宽度至 14 位，则随机网络在面积上仍有优势。综上所述，随机网络的性能和传统电路相差并不明显，但是拥有非常大的电路面积优势。

随机计算给神经网络提供了一种并行解决方案。由于神经网络包含大量乘法、加法、指数运算，传统基于指令运算的微处理器需要非常多的时钟周期，对于本实验 M3 和 M4 的 56×15×14 网络结构，若每个节点都进行一次乘法和加法，则至少需要 56×15×14×2=23520 个计算时钟，再加上指数运算、控制指令、读写数据则需要更多时钟周期，而随机 RBF 网络仅需要 10k 个时钟就可以得到基本正确的结果，硬件开销远小于传统微处理器，因此随机计算的综合性能相对传统的微处理器的串行指令运行方式有较大优势。

5 结束语

计算机的处理能力正受到越来越严重的瓶颈限制。本文提出了一种思路，可以运用非常精简的随机运算逻辑来实现大规模神经网络。实验证明，本文所提出随机径向基函数神经网络的中间神经元的电路面积只有传统插值查表结构神经元的 1.2%, CORDIC 法的 2%，而其运算精度非常接近于传统网络，且其 MSE 波动范围非常小。从实验结果可以推论，这些随机逻辑可以应用于更为复杂的网络结构中。当硬件成本为主要考虑因素时，随机网络具有很大优势，尤其适应于对成本、功耗要求较高而对于精度要求并不高的应用如嵌入式、移动式、穿戴式设备。另一方面，随机网络的性能随着随机码流长度而改变，因此，在相同的网络结构下，可以通过改变码流长度来产成不同的计算精度，这给设计者带来了灵活性，可以在不改变硬件结构的情况下对网络性能、电路功耗和运算速度 3 个因素进行动态平衡。最后，由于随机计算是基于概率数的运算，网络可以容许输入数据的误差和电路错误，当电路中有个别数据发生改变时，其最终结果并不会发生根本变化，该特点将是今后的研究方向之一。

参 考 文 献

[1] GAINES B R. Stochastic Computing Systems (Chapters) in Advances in Information Systems Science[M]. New York: Plenum, 1969: 37–172.

[2] HAYES J P. Introduction to stochastic computing and its challenges[C]. 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, 2015: 1–3. doi: 10.1145/2744769.2747932.

[3] ALAGHI A and HAYES J P. Survey of stochastic computing[J]. *ACM Transactions on Embedded Computing Systems*, 2013, 12(2s): 1–19. doi: 10.1145/2465787.2465794.

[4] MOONS B and VERHELST M. Energy-efficiency and accuracy of stochastic computing circuits in emerging technologies[J]. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2014, 4(4): 475–486. doi: 10.1109/JETCAS.2014.2361070.

[5] BROWN B D and CARD H C. Stochastic neural computation. I. Computational elements[J]. *IEEE Transactions on Computers*, 2001, 50(9): 891–905. doi: 10.1109/12.954505.

[6] QIAN Weikang, LI Xin, RIEDEL M D, *et al*. An architecture for fault-tolerant computation with stochastic logic[J]. *IEEE Transactions on Computers*, 2011, 60(1): 93–105. doi: 10.1109/TC.2010.202.

[7] HAN Jie, CHEN Hao, LIANG Jinghang, *et al*. A stochastic computational approach for accurate and efficient reliability evaluation[J]. *IEEE Transactions on Computers*, 2014, 63(6): 1336–1350. doi: 10.1109/TC.2012.276.

[8] ALAWAD M and LIN Mingjie. FIR filter based on stochastic computing with reconfigurable digital fabric[C]. 2015 IEEE 23rd Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), Vancouver, BC, Canada, 2015: 92–95. doi: 10.1109/FCCM.2015.32.

[9] TEHRANI S S, NADERI A, KAMENDJE G A, *et al*. Majority-based tracking forecast Memories for Stochastic LDPC Decoding[J]. *IEEE Transactions on Signal Processing*, 2010, 58(9): 4883–4896. doi: 10.1109/TSP.2010.2051434.

[10] LI Peng, LILJA D J, QIAN Weikang, *et al*. Computation on stochastic bit streams digital image processing case studies[J]. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, 2014, 22(3): 449–462. doi: 10.1109/TVLSI.2013.2247429.

[11] ZHANG Da and LI Hui. A stochastic-based FPGA controller for an induction motor drive with integrated neural network algorithms[J]. *IEEE Transactions on Industrial Electronics*, 2008, 55(2): 551–561. doi: 10.1109/TIE.2007.911946.

[12] 王守觉, 李兆洲, 陈向东, 等. 通用神经网络硬件中神经元基本数学模型的讨论[J]. 电子学报, 2001, 29(5): 576–580.
WANG Shoujue, LI Zhaozhou, CHEN Xiangdong, *et al*. Discussion on the basic mathematical models of neurons in general purpose neurocomputer[J]. *Acta Electronica Sinica*, 2001, 29(5): 576–580.

[13] 吴大鹏, 赵莹, 熊余, 等. 基于小波神经网络的告警信息相关性挖掘策略[J]. 电子与信息学报, 2014, 36(10): 2379–2384. doi: 10.3724/SP.J.1146. 2013.01701.
WU Dapeng, ZHAO Ying, XIONG Yu, *et al*. Alarm information relevance mining mechanism based on wavelet neural network[J]. *Journal of Electronics & Information Technology*, 2014, 36(10): 2379–2384. doi: 10.3724/SP.J.1146. 2013.01701.

[14] BROWN B D and CARD H C. Stochastic neural computation. II. Soft competitive learning[J]. *IEEE Transactions on Computers*, 2001, 50(9): 906–920. doi: 10.1109/12.954506.

[15] LI Peng, LILJA D J, QIAN W K, *et al*. The synthesis of complex arithmetic computation on stochastic bit streams using sequential logic[C]. 2012 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), San Jose, CA, USA, 2012: 480–487. doi: 10.1145/2429384.2429483.

[16] JI Yuan, RAN Feng, MA Cong, *et al*. A hardware implementation of a radial basis function neural network using stochastic logic[C]. 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 2015: 880–883.

[17] 马承光, 仲顺安, LILJA D J, 等. 基于超几何分解的随机运算系统分析方法[J]. 电子与信息学报, 2013, 35(2): 355–360. doi: 10.3724/SP.J.1146.2012.00711.
MA Chengguang, ZHONG Shunan, LILJA D J, *et al*. Analysis method of stochastic computing system based on hypergeometric decomposition[J]. *Journal of Electronics & Information Technology*, 2013, 35(2): 355–360. doi: 10.3724/SP.J.1146.2012.00711.

季 渊： 男，1980 年生，副研究员，研究方向为大规模集成电路设计、神经网络与机器学习、硅基微显示器.

陈文栋： 男，1993 年生，硕士生，研究方向为神经网络与机器学习.

冉 峰： 男，1954 年生，教授，研究方向为大规模集成电路设计、微电子技术、半导体器件.

David LILJA： 男，教授，研究方向为计算机架构、并行计算、高性能计算.