专题: 面向类脑计算的物理电子学

基于忆阻器阵列的下一代储池计算*

任宽[1)2)#] 张握瑜[1)3)#] 王菲[1)3)] 郭泽钰[1)3)] 尚大山[1)3)†]

1) (中国科学院微电子研究所, 微电子器件与集成技术重点实验室, 北京 100029)

2) (西南交通大学超导与新能源研究开发中心, 磁浮技术与磁浮列车教育部重点实验室, 成都 610031)

3) (中国科学院大学, 北京 100049)

(2022 年 1 月 12 日收到; 2022 年 1 月 26 日收到修改稿)

储池计算是类脑计算范式的一种, 具有结构简单、训练参数少等特点, 在时序信号处理、混沌动力学系统预测等方面有着巨大的应用潜力. 本文提出了一种基于存内计算范式的储池计算硬件实现方法, 利用忆阻器阵列完成非线性向量自回归过程中的矩阵向量乘法操作, 有望进一步提升储池计算的能效. 通过忆阻器阵列仿真实验, 在 Lorenz63 时间序列预测任务中验证了该方法的可行性, 以及该方法在噪声条件下预测结果的鲁棒性, 并探究忆阻器阵列阻值精度对预测结果的影响. 这一结果为储池计算的硬件实现提供了一种新的途径.

关键词: 储池计算, 忆阻器, 存内计算, 非线性向量自回归

PACS: 07.05.Mh, 84.35.+i, 85.40.–e, 87.15.A– **DOI**: 10.7498/aps.71.20220082

1 引 言

理解生物大脑中信息的加工、处理模式, 并在此基础上构建类脑计算硬件系统是现代信息科学的前沿研究之一[1]. 研究表明, 生物大脑等效于一个复杂神经网络动力学系统[2], 其处理外界信息的机能依赖于神经网络的动力学过程[3]. 如何理解大脑的神经动力学过程、构建类脑动力学系统, 是类脑计算硬件系统实现的核心问题[4]. 自然界中的信息大部分是用时序数据来定义的. 大脑的动力学系统受外部时序信号刺激, 并将刺激产生的数据进行编码和存储[5,6], 进而形成各类认知过程. 循环神经网络 (recurrent neural network, RNN)[7] 是一种具有短时记忆能力的神经网络, 其中的神经元通过具有环路的网络结构, 不仅可以接受其他神经元的信息, 也可以接受自身的信息, 从而使网络具有处理时序数据的能力, 因此, 更加适合模拟大脑的动力学系统. 当前, RNN 已经被广泛应用于语音识别、自然语言处理等任务中. 然而, 由于梯度消失和爆炸问题[8] , RNN 需要的超参数多, 而且训练过程复杂. 因此, RNN 在硬件系统实现上依然面临结构复杂、训练时间长和能耗高等问题[9].

储池计算 (reservoir computing, RC) 是 RNN 的一种简化形式. RC 概念最初的提出是为了模拟生物大脑中具有大量循环连接的皮质纹状体系统处理视觉空间序列信息的过程[10]. 随后人们基于 RNN 的框架, 构建了统一的 RC 计算框架[11–13] (如图 1(a)). RC 的核心是一个被称为“储池”的循环神经网络隐藏层. 该网络能够将时序输入信号转换到高维空间中. 经过高维转换后, 输入信号的特征就可以更容易地通过简单线性回归方法有效读出. 目前, RC 在时序信号处理[14]、混沌动力学系统预测[15]等动力学系统学习方面有良好的功能. 值得注意的

* 国家重点基础研究发展计划 (批准号: 2018YFA0701500)、国家自然科学基金 (批准号: 61874138) 和中国科学院战略性先导科技专项 (批准号: XDB44000000) 资助的课题.

同等贡献作者.

† 通信作者. E-mail: shangdashan@ime.ac.cn

是, 与标准 RNN 相比[16], RC 中只需要训练输出层权重, 并且不需要反向传播算法, 有效避免了梯度消失问题, 因此, 可以有效降低训练复杂度和训练时间.

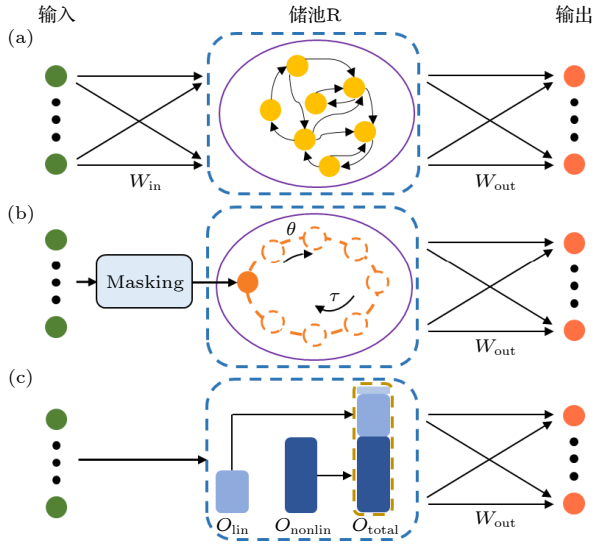

图 1 三种 RC 结构 (a) 传统 RC 结构; (b) 单节点延时 RC 结构; (c) 非线性向量自回归 RC 结构

Fig. 1. Three types of RC frameworks: (a) Conventional RC; (b) RC using a single nonlinear node reservoir with time-delayed feedback; (c) NGRC, which is equivalent to nonlinear vector autoregression.

传统 RC 主要包括由 Jaeger[11] 提出的 ESN 模型 (echo state network) 和 Maass 等[13] 提出的 LSM (liquid state machine) 模型, 他们的结构都如图 1(a) 所示, 包含输入、储池和输出三部分. ESN 模型的储池和 LSM 模型的储池都为基于 RNN 框架, 由多个神经元随机连接而成的结构. 所不同的是, ESN 模型中的神经元是离散时间人工神经元, 而 LSM 模型中的神经元是具有兴奋性和抑制性的脉冲神经元. 储池计算模型结构 (以 ESN 为例) 可以用以下公式描述:

$$\boldsymbol{x}(n)=f(\boldsymbol{W}_{\rm in}\boldsymbol{u}(n)+\boldsymbol{W}\boldsymbol{x}(n-1)), \tag{1}$$

$$\boldsymbol{y}(n)=\boldsymbol{W}_{\rm out}\boldsymbol{x}(n), \tag{2}$$

其中, $\boldsymbol{u}(n)$为输入向量, n 为离散的时间, f 表示储池层单元的非线性激活函数, $\boldsymbol{W}_{\rm in}$为输入连接储池的权重矩阵, $\boldsymbol{x}(n)$是所有离散时间人工神经元的回声状态向量, $\boldsymbol{W}$为神经元间的连接矩阵, $\boldsymbol{W}_{\rm out}$为储池连接输出的权重矩阵, $\boldsymbol{y}(n)$为输出向量. 储池计算只需要训练输出权重矩阵 $\boldsymbol{W}_{\rm out}$, 其性能取决于储池神经元间的连接矩阵 $\boldsymbol{W}$. 当 $\boldsymbol{W}$ 的谱半径小于 1 时 (即特征值的绝对值的最大项), 对任意输入都可以得到对应的回声状态属性[17]. 节点的回声状态属性等效于节点具备“衰退记忆”[18]. 人们从理论上证明了, 由离散时间“衰退记忆”节点构成的 RC 网络在输入有界的情况下具备动力学通用逼近能力[19].

这类 RC 的硬件实现方法可大致分为两种: 一种方法是通过使用神经网络硬件或神经形态计算技术实现, 如用模拟电路[20]、FPGA (field-programmable gate array)[21,22]、大规模集成电路[23]、忆阻器[24–26] 等直接构造储池中多个随机连接的神经元. 这种方法可灵活调整神经元间连接的拓扑结构以改善性能, 但是构造神经元需要的器件众多, 并且计算中每一个时间步都需要进行大量计算以及存储大量神经元的状态. 另一种方法是采用具备“衰退记忆”的物理节点代替随机连接的神经元, 构成储池的动力学系统, 如纳米线网络[27,28]、光学器件网络[29]、易失性忆阻器网络[30] 等. 这种方法利用物理节点的“衰退记忆”特性进行计算. 储池的存与算在节点网络中同时进行. 然而由随机连接的物理节点构成的动力学系统无法调整节点连接的拓扑结构, 故这种方法在面对不同任务时, RC 的性能具有一定的不稳定性.

为了提高 RC 的性能, 研究人员对 RC 结构进行了多种改进 (如多储池计算[31]、进化储池计算[32] 等), 以及将 RC 与其他特征提取方法 (如卷积神经网络[33]、强化学习[34]、注意力学习[35] 等) 相结合. 目前, 传统 RC 及其改进方法已经成功地被应用于众多领域, 如生物医学、声音识别、无线电等[36]. 然而, 由于储池结构中神经元很多, 神经元状态存储以及更新需要大量的硬件资源, 并且由于神经元连接拓扑结构难以调整, 导致储池计算的参数优化困难.

RC 中神经元节点间相互作用产生的高维信号可以通过延时动力学系统来实现. 其状态方程描述为[37]

$$\frac{{\rm d}x}{{\rm d}t}=F(t,x(t),x(t-\tau)), \tag{3}$$

其中, t 为连续时间信号, $x(t)$为系统的状态, F 为系统函数, τ 为延时时间. 2011 年, Appeltant 等[38] 提出基于延时动力学系统的单延时节点 RC 结构 (见图 1(b)). 输入信号通过掩码函数 (masking) 进行时间复用, 然后输入到单个物理节点在时间维度上展开的虚拟节点中. 虚拟节点通过平等分割 τ 的

N个时间点上来设置. 两个虚拟节点之间的时间间隔为$\theta=\tau/N$. 所有虚拟节点$x\left[t-(N-i)\theta\right], i=1,\cdots,N$共同作为$t$时刻节点状态, 并通过输出层得到计算结果.

单节点延时 RC 的提出, 使得 RC 的硬件实现变得更加便捷, 在一定程度上解决了传统储池硬件实现中, 由于神经元数量多而导致的神经元状态存储和更新硬件资源问题. 这种单节点延时储池已经在光子器件[39]、FPGAs[40] 中得到硬件实现, 用于语音识别、图像分类和混沌预测等任务中. 我们在前期工作中, 利用铁电隧道结 (FTJ) 中超薄铁电层的退极化效应产生的电流延时特性, 实现了单节点延时储池计算功能[41]. 为了拓展单节点延时 RC 功能, 我们采用了多个单延时节点储池并联的方式, 提高了计算的维度, 实现了对动态数字序列的识别功能[41]. 然而, 由于虚拟节点是通过时间切分获得, 所以其连接拓扑结构是按时间顺序固定的. 这意味着这种延时储池同样存在着参数优化困难的问题.

最近, Gauthier 等[42] 提出了一种新型 RC, 称为下一代储池计算 (NGRC). NGRC 是一种特殊的非线性向量自回归过程, 其等效于具有线性激活节点的储池与一个非线性读出层的结合, 如图 1(c) 所示. NGRC 模型描述为

$$\boldsymbol{O}_{\mathrm{lin},i}=\left[\boldsymbol{X}\left(i\right),\boldsymbol{X}\left(i-s\right),\cdots,\boldsymbol{X}\left(i-(k-1)\,s\right)\right]^{\mathrm{T}}, \tag{4}$$

$$\boldsymbol{O}_{\mathrm{nonlin},i}=\boldsymbol{O}_{\mathrm{lin},i}\lceil\otimes\rceil\boldsymbol{O}_{\mathrm{lin},i}, \tag{5}$$

$$\boldsymbol{O}_{\mathrm{total},i}=c\bigotimes\boldsymbol{O}_{\mathrm{lin},i}\bigotimes\boldsymbol{O}_{\mathrm{nonlin},i}, \tag{6}$$

$$\boldsymbol{Y}\left(i\right)=\boldsymbol{W}_{\mathrm{out}}\boldsymbol{O}_{\mathrm{total},i}, \tag{7}$$

其中, i为离散时间, $\boldsymbol{O}_{\mathrm{lin},i}$为线性特征向量, $\boldsymbol{X}\left(i\right)$为第 i时刻的输入向量, s为时间间隔, k为构成线性特征向量的组数, $\boldsymbol{O}_{\mathrm{nonlin},i}$为第 i时刻非线性特征向量, $\lceil\otimes\rceil$功能为将符号两边项进行外积、并收集外积结果的唯一单项式的运算符号, $\boldsymbol{O}_{\mathrm{total},i}$为第$i$时刻总特征向量, c为常数修正项, $\boldsymbol{Y}\left(i\right)$为输出值, $\boldsymbol{W}_{\mathrm{out}}$为储池连接输出的权重矩阵. NGRC 目前被证实在完成短期动态预测、长期混沌预测、推断动力学系统不可见数据等三个方面有很好的性能. 相对于传统 RC 和延时 RC, NGRC 使用更小的数据集进行训练, 并避免了 RC 的参数优化困难问题. 然而, 非线性向量自回归过程本身仍需要大量硬件计算资源用于乘法计算操作.

忆阻器是近年广受关注的一种具有记忆功能的器件[43]. 由忆阻器器件构成的交叉阵列[44], 可以通过欧姆定律和基尔霍夫定律, 以存内计算的方式原位、并行、物理地完成矩阵向量乘运算, 有效减少了计算过程中数据的搬运, 从而具有功耗低、速度快的优点[45–47]. 本文将 NGRC 过程通过矩阵向量乘法操作简化, 提出了一种 NGRC 的存内计算硬件实现方法, 并利用忆阻器阵列完成矩阵向量乘法操作. 通过进行忆阻器阵列仿真完成了 Lorenz63 时间序列预测任务, 验证了该方法的可行性, 并研究了忆阻器件电阻精度和波动性对 NGRC 预测精度的影响. 这一结果为高能效 RC 提供了一种新的途径.

2 NGRC 的存内实现方法

传统 RC 过程中, 每一个时间步都需要更新大量具有“衰退记忆”特性的神经元的状态, 然而具有“衰退记忆”特性的线性神经元组成的储池与二次非线性读出层组合, 在数学上等效于一种特殊的非线性向量自回归过程. NGRC 是对这种特殊的非线性向量自回归过程的优化[42]. NGRC 过程与传统储池计算过程的相同点在于都只需要训练输出权重, 但是在输入数据的选择和将输入数据进行高维空间非线性转换的方式上有所不同.

输入数据方面, 传统储池输入数据一般为当前时刻的数据, 而 NGRC 的输入数据中, 除了当前时刻的数据, 还包括之前时刻所对应的数据. 高维空间非线性转换方式方面, 传统储池的高维空间非线性转换通过储池中具备“衰退记忆”神经元的非线性激活函数达成. NGRC 结构储池的高维空间非线性转换可分为 3 个过程 (见图 1(c)): 1) 选择不同时刻输入数据构成线性特征向量$\boldsymbol{O}_{\mathrm{lin}}$; 2) 由线性特征向量构造非线性特征向量$\boldsymbol{O}_{\mathrm{nonlin}}$; 3) 由线性特征向量与非线性特征向量构造总特征向量$\boldsymbol{O}_{\mathrm{total}}$. 3 个过程中, 线性特征$\boldsymbol{O}_{\mathrm{lin}}$向量是由选择的输入数据直接拼接而成; 总特征向量$\boldsymbol{O}_{\mathrm{total}}$是由固定常数$c$、线性特征向量$\boldsymbol{O}_{\mathrm{lin}}$与非线性特征向量$\boldsymbol{O}_{\mathrm{nonlin}}$直接拼接而成; 而由线性特征向量$\boldsymbol{O}_{\mathrm{lin}}$构造非线性特征向量$\boldsymbol{O}_{\mathrm{nonlin}}$则需要经过一个非线性转换过程. NGRC 中的非线性转换过程将线性特征向量$\boldsymbol{O}_{\mathrm{lin}}$通过外积操作映射到一个高维空间中, 并在高维空间中去

除对应映射向量的重复部分, 得到非线性特征向量 $\boldsymbol{O}_{\text{nonlin}}$.

NGRC 的非线性转换过程虽然避免了传统储池中随机连接的性能不确定性与需要同时更新多个神经元状态的复杂性, 但其向量间的外积操作与除去高维向量重复部分的操作仍然需要大量硬件开销与时间开销. 我们注意到相同向量间的外积可以用矩阵向量乘法 (matrix vector multiplication, MVM) 来表示, 去除重复映射向量的操作可以通过保留外积后固定位置元素的值 (保留元素操作) 来实现. 硬件上, 使用忆阻器阵列进行 MVM 操作, 使用忆阻器阵列的选择线电路进行保留元素操作.

2.1 线性特征向量的构建

图 2(a) 为三维空间时序数据预测任务的 NGRC 储池的存内实现结构. 其中, $t_i = i \times \Delta t$, Δt 为采样间隔, i 为离散时间数, s 为时间间隔数, $x(t)$, $y(t)$, $z(t)$ 分别表示预测点在 t 时刻的 x 轴、y 轴、z 轴三维空间坐标, k 为每个线性特征向量所取数据的组数. 当已知 t_i 时刻及之前时刻点的轨迹坐标, 要预测 t_{i+1} 时刻点的坐标时, 取 $k = 2$, 即令 t_i 时刻和 t_{i-s} 时刻空间点的三维坐标构建第 i 个线性特征向量 $\boldsymbol{O}_{\text{lin},i}$, 有

$$\boldsymbol{O}_{\text{lin},i} = [x_i, y_i, z_i, x_{i-s}, y_{i-s}, z_{i-s}]^{\text{T}}, \tag{8}$$

其中, $[x_i, y_i, z_i]$ 与 $[x_{i-s}, y_{i-s}, z_{i-s}]$ 为 t_i 时刻与 t_{i-s} 时刻点的三维坐标. 将构建的线性特征向量 $\boldsymbol{O}_{\text{lin},i}$ 用时序电压脉冲幅值编码和电导编码, 编码的电压序列矩阵 $\boldsymbol{V}_{\text{lin},i}$ 为

$$\boldsymbol{V}_{\text{lin},i} = [\boldsymbol{V}_{\text{lin},i,1}, \boldsymbol{V}_{\text{lin},i,2}, \boldsymbol{V}_{\text{lin},i,3}, \boldsymbol{V}_{\text{lin},i,4}, \boldsymbol{V}_{\text{lin},i,5}, \boldsymbol{V}_{\text{lin},i,6}]^{\text{T}}, \tag{9}$$

其中列向量 $\boldsymbol{V}_{\text{lin},i,a}$ 的第 a 行的值, 为线性特征向量 $\boldsymbol{O}_{\text{lin},i}$ 中第 a 个元素对应的量化电压值, 列向量中的其他值为零. 线性特征向量 $\boldsymbol{O}_{\text{lin},i}$ 编码的电导矩阵为

$$\boldsymbol{G}_{\text{lin},i} = \begin{bmatrix} \boldsymbol{G}_{\text{lin},i,1} & \boldsymbol{G}_{\text{lin},i,2} & \boldsymbol{G}_{\text{lin},i,3} & \boldsymbol{G}_{\text{lin},i,4} & \boldsymbol{G}_{\text{lin},i,5} & \boldsymbol{G}_{\text{lin},i,6} \\ 0 & \boldsymbol{G}_{\text{lin},i,2} & \boldsymbol{G}_{\text{lin},i,3} & \boldsymbol{G}_{\text{lin},i,4} & \boldsymbol{G}_{\text{lin},i,5} & \boldsymbol{G}_{\text{lin},i,6} \\ 0 & 0 & \boldsymbol{G}_{\text{lin},i,3} & \boldsymbol{G}_{\text{lin},i,4} & \boldsymbol{G}_{\text{lin},i,5} & \boldsymbol{G}_{\text{lin},i,6} \\ 0 & \cdots & 0 & \boldsymbol{G}_{\text{lin},i,4} & \boldsymbol{G}_{\text{lin},i,5} & \boldsymbol{G}_{\text{lin},i,6} \\ \cdots & \ddots & \cdots & 0 & \boldsymbol{G}_{\text{lin},i,5} & \boldsymbol{G}_{\text{lin},i,6} \\ 0 & \cdots & 0 & 0 & 0 & \boldsymbol{G}_{\text{lin},i,6} \end{bmatrix}, \tag{10}$$

其中, $\boldsymbol{G}_{\text{lin},i,a}$ 为 $\boldsymbol{O}_{\text{lin},i}$ 中的第 a 个元素对应的量化电导值. 将电导序列使用差分编码存储到忆阻器阵列中; 再将电压序列 $\boldsymbol{V}_{\text{lin},i}$ 通过 Bitline 输入到忆阻阵列中, 具体过程如图 2(b) 所示. 需要指出的是, 电压脉冲幅值编码需要经过一个数模转换器 (DAC). DAC 的精度与忆阻器本身的精度影响着整体精度.

2.2 非线性特征向量的构建

构建非线性特征向量需要对线性特征向量进行外积操作与保留元素操作. 忆阻器阵列中, 每通过一个电压序列向量 $\boldsymbol{V}_{\text{lin},i,a}$, 能从阵列输出端 (SL) 得到一个电流向量 $\boldsymbol{I}_{\text{lin},i,a}$, 总电流矩阵由欧姆定律和基尔霍夫定律可表达为

$$\begin{aligned} \boldsymbol{I}_{\text{lin},i} &= \boldsymbol{V}_{\text{lin},i} \cdot \boldsymbol{G}_{\text{lin},i} \\ &= \left[\boldsymbol{I}_{\text{lin},i,1}^{\text{T}}, \boldsymbol{I}_{\text{lin},i,2}^{\text{T}}, \boldsymbol{I}_{\text{lin},i,3}^{\text{T}}, \boldsymbol{I}_{\text{lin},i,4}^{\text{T}}, \boldsymbol{I}_{\text{lin},i,5}^{\text{T}}, \boldsymbol{I}_{\text{lin},i,6}^{\text{T}}\right]^{\text{T}}. \end{aligned} \tag{11}$$

保留 $\boldsymbol{I}_{\text{lin},i}$ 矩阵中的非零元素, $\boldsymbol{I}_{\text{lin},i,a}$ 保留元素操作后的向量为 $\boldsymbol{I}_{\text{lin},i,ap}$, 保留元素操作可通过只读取忆阻器阵列电流输出中对应位置的输出实现, 过程如图 2(a) 中绿框部分所示, 将选择读取的输出电流组成非线性特征向量 $\boldsymbol{O}_{\text{nonlin},i}$, 可表达为

$$\begin{aligned} \boldsymbol{O}_{\text{nonlin},i} = [&\boldsymbol{I}_{\text{lin},i,1p}, \boldsymbol{I}_{\text{lin},i,2p}, \boldsymbol{I}_{\text{lin},i,3p}, \\ &\boldsymbol{I}_{\text{lin},i,4p}, \boldsymbol{I}_{\text{lin},i,5p}, \boldsymbol{I}_{\text{lin},i,6p}] \end{aligned} \tag{12}$$

2.3 总特征向量的构建及输出

t_i 时刻的总特征向量 $\boldsymbol{O}_{\text{total},i}$ 是由固定常数 c、线性特征向量 $\boldsymbol{O}_{\text{lin},i}$ 与非线性特征向量 $\boldsymbol{O}_{\text{nonlin},i}$ 直接拼接而成, 表示为

$$\boldsymbol{O}_{\text{total},i} = [c, \boldsymbol{O}_{\text{lin},i}, \boldsymbol{O}_{\text{nonlin},i}]. \tag{13}$$

t_{i+1} 时刻点的预测位置可直接由总特征向量乘以输出权重得出:

$$[x_{i+1}, y_{i+1}, z_{i+1}] = \boldsymbol{O}_{\text{total},i} \cdot \boldsymbol{W}_{\text{out}}, \tag{14}$$

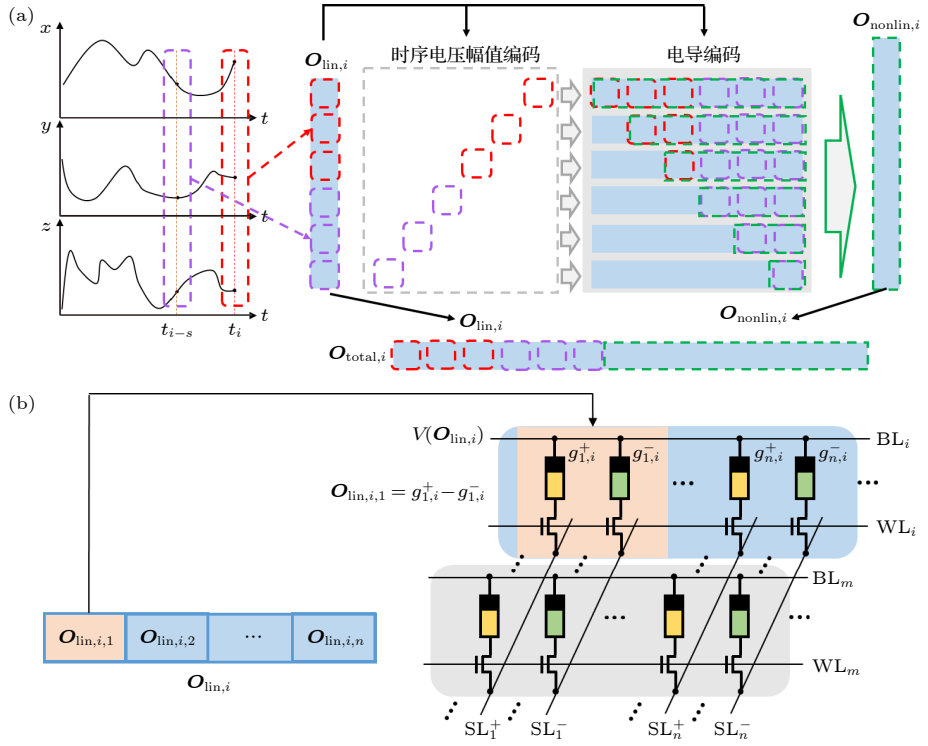

图 2 基于忆阻阵列的 NGRC 储池结构 (a) 用于预测三维时序信号的 NGRC 储池结构. 输入为三维时序信号; 提取 t_i 时刻 (红色框) 和 $t_{i\text{-}s}$(紫色框) 时刻信号的值组成线性特征向量 $\boldsymbol{O}_{\mathrm{lin}}$, 将第 i 个线性特征向量编码为时序电压和电导, 时序电压作为忆阻器阵列的输入, 电导映射到忆阻器阵列上作为权重; 非线性特征向量 $\boldsymbol{O}_{\mathrm{nonlin}}$ 由忆阻器阵列特定单元 (绿色方框) 的输出构成; 总特征向量由 $\boldsymbol{O}_{\mathrm{lin}}$ 与 $\boldsymbol{O}_{\mathrm{nonlin}}$ 直接拼接而成. (b) 图 (a) 中的线性特征向量 $\boldsymbol{O}_{\mathrm{lin},i}$ 映射到忆阻器阵列的方式. $\boldsymbol{O}_{\mathrm{lin},i}$ 中的每一个值都由两个忆阻器电导的差分 g^+, g^- 表示

Fig. 2. Structure of the NGRC based on memristor-based crossbar. (a) Structure of the NGRC reservoir for three dimensional (3D) timing signals predicting. The input is a 3D timing signal. The linear feature vector $\boldsymbol{O}_{\mathrm{lin}}$ is formed by extracting the signal values of t_i time (red box) and $t_{i\text{-}s}$ time (purple box). The ith linear feature vector is encoded as timing voltage and conductance, and the timing voltage is the input of the memristor array, and the conductance is mapped to the memristor array as weight. The nonlinear feature vector $\boldsymbol{O}_{\mathrm{nonlin}}$ consists of the outputs of specific elements of the memristor array (green boxes). The total feature vector is directly spliced by $\boldsymbol{O}_{\mathrm{lin}}$ and $\boldsymbol{O}_{\mathrm{nonlin}}$. (b) The way the linear feature vector $\boldsymbol{O}_{\mathrm{lin},i}$ in panel (a) mapping to the memristor array. The g^+ and g^- represent the device conductance values for the positive and negative weights in the differential pair, respectively.

其中, $[x_{i+1}, y_{i+1}, z_{i+1}]$为所预测的$t_{i+1}$时刻点的三维坐标, $\boldsymbol{W}_{\mathrm{out}}$为预先用岭回归方法训练好的输出权重矩阵. 值得注意的是, 忆阻器电流值的读出需要一个模数转换器 (ADC), ADC 的精度也会对最终预测精度造成一定影响.

2.4 训练过程

储池训练过程只训练输出层 $\boldsymbol{W}_{\mathrm{out}}$, 训练采用岭回归方法, 先用训练数据集得到由特征向量组成的特征矩阵 $\boldsymbol{O}_{\mathrm{total}}$, 以及所有特征向量对应的输出组成的结果矩阵 $\boldsymbol{Y}_d$, 岭回归方法表达为

$$\boldsymbol{W}_{\mathrm{out}} = \boldsymbol{Y}_d \boldsymbol{O}_{\mathrm{total}}^{\mathrm{T}} \left(\boldsymbol{O}_{\mathrm{total}} \boldsymbol{O}_{\mathrm{total}}^{\mathrm{T}} + a\boldsymbol{I}\right)^{-1}, \tag{15}$$

其中, $\boldsymbol{O}_{\mathrm{total}}^{\mathrm{T}}$是训练数据集的总特征向量的转置矩阵, a为岭回归参数, $\boldsymbol{I}$为单位矩阵.

2.5 仿真平台

仿真平台基于 python 3.8, pytorch1.9.1(主机 GPU 型号 NVIDIA GeForce RTX 3080, CPU 型号 i9-11980HK) 构建, 其结构示意图如图 3 所示, 可分为输入部分、权重部分和输出部分.

输入部分模拟将外界信号转换为电压信号的过程, 最高转换精度为定点 32 bit. 权重部分模拟将外界信号映射到忆阻器阵列中 (将带符号的权重映射到一对忆阻器差分电导上) 并进行运算的过程, 量化电导映射公式为

$$\Delta G = (G_{\max} - G_{\min})/2^{n-1}, \tag{16}$$

$$G = [W/\Delta G] \times \Delta G, \tag{17}$$

其中 n 为权重精度 (单位 bit), $G_{\max}$ 和 $G_{\min}$ 为忆阻器可变化的最大电导与最小电导, $[x]$ 为取整操作,

W 为需要映射的权重, G 为映射到忆阻器阵列对应位置的电导值. 输出部分模拟忆阻器阵列输出经过 ADC 转变为电脑可处理数据的过程, 输出精度为所使用 ADC 的精度.

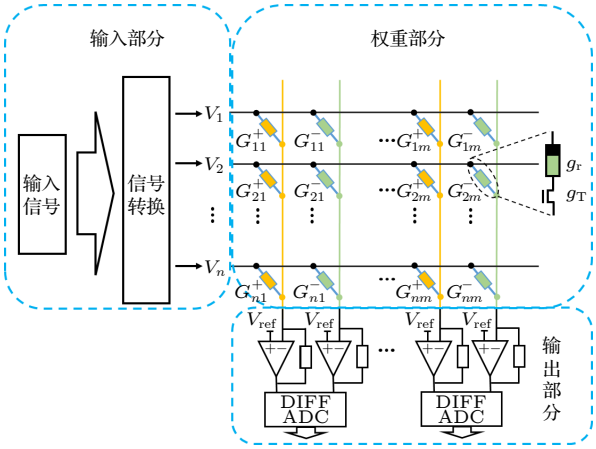

图 3 基于忆阻器阵列 (包括正、负列) 的矩阵乘法运算仿真平台结构示意图, g_r 为忆阻器的电导, g_T 为晶体管电导

Fig. 3. Simulation platform of memristor array (including positive and negative arrays) as analog dot-product engine. The memristor conductance corresponds to g_r and the transistor conductance corresponds to g_T.

3 实验结果与讨论

动力学系统的短期预测能力与动力学系统的长期预测能力通常被用来作为衡量 RC 性能的基准, 我们将用经典混沌动力学系统模型-Lorenz63 模型的短期预测任务与长期预测任务, 验证基于忆阻器阵列实现的 NGRC 结构的可行性及其对噪声的鲁棒性. Lorenz63 是 1963 年洛伦兹[48] 提出来的天气预测模型, 由 3 个方程组成:

$$\frac{\mathrm{d}x}{\mathrm{d}t}=a(y-x),\ \frac{\mathrm{d}y}{\mathrm{d}t}=ax(b-z)-y,\ \frac{\mathrm{d}z}{\mathrm{d}t}=xy-cz,\quad (18)$$

其中状态 $\boldsymbol{X}(t)\equiv[x,\,y,\,z]^{\mathrm{T}}$ 是一个分量为 Rayleigh-Bénard 的对流可观测量的矢量, $a=10$, $b=28$, $c=8/3$. Lorenz63 模型确定性的混沌行为体现在其对初始条件的敏感依赖 (蝴蝶效应), 以及在相空间轨迹形成奇异吸引子 (图 3).

动态系统的预测任务中, 用原序列 (由动态系统方程得到的序列) 与预测序列 (储池不断将此时刻输出值作为下一时刻的输入值进行预测得到的序列) 之间的结构相似度来衡量预测效果. 就 Lorenz63 时间序列预测任务而言, 归一化均方根误差 (NRMSE) 可在一定程度上衡量短预测期内的结构相似度, 但难以反映长期预测的结构相似度. Lorenz63 时间序列预测的 z 回归图能直观地反映 z 变量的长期行为, 比较原序列与预测序列的 z 回归图可以定性地比较两个序列长时段的结构相似度. 在之后的 Lorenz63 时间序列预测任务中, NRMSE 衡量短期预测 (1 个李雅普诺夫周期) 的结构相似度, 通过比较量原序列与预测序列的 z 回归图衡量长期预测的结构相似度.

在基于忆阻器阵列实现的 NGRC 结构的可行性验证实验中, 维持系统的输入精度不变, 通过改变系统的权重精度 (忆阻阵列中忆阻器的量化映射比特数) 和输出精度 (忆阻器阵列输出 ADC 比特数), 研究不同权重精度和输出精度对预测结果的结构相似度的影响. 在基于忆阻器阵列实现的 NGRC 结构对噪声的鲁棒性验证实验中, 维持输入精度和输出精度不变, 研究不同权重精度以及不同噪声大小对预测结果的结构相似度的影响.

3.1 可行性验证实验

保持输入精度为定点 32 bit, 输出精度为定点 64 bit, 在权重精度为 4, 6, 8, 16, 32 和 64 bit 情况下进行预测实验. 800 个时间步的预测结果及其 xz 截面图如图 4 所示. 可以看出, 权重精度在 4 和 6 bit 时无法产生混沌现象 (无洛伦兹吸引子); 当权重精度达到 8 bit 及以上时, 开始产生明显的洛伦兹混沌吸引子. 这一结果意味着权重精度对混沌的产生有重要影响. 当忆阻器阵列对应的权重精度达到一定值时, 基于忆阻器阵列实现的 NGRC 结构构成的系统能由稳定状态过渡到混沌状态.

保持输入精度为定点 32 bit, 通过改变权重精度以及输出精度, 在达到混沌状态的前提下探究短期预测结构相似度与权重精度的关系. 图 5 为短期预测 (1 个李雅普诺夫周期) 的 NRMSE 随不同权重精度 (8, 16, 32, 64 bit) 和不同输出精度 (8, 16, 32, 64 bit) 的变化. 结果显示, 当基于忆阻器阵列实现的 NGRC 结构构成的系统达到产生混沌所需的权重精度和输出精度时, 短期预测的性能随着权重精度、输出精度的增加而增加. 在权重精度不变的情况下, 当输出精度达到 16 bit, 输出精度的增加对短期预测结构相似度几乎无影响; 在输出精度不变的情况下, 短期预测结构相似度随着权重精度的增加而变高 (NRMSE 变小), 8 bit 权重精度下的 NRMSE 低于 0.05, 16 bit 权重精度下的 NRMSE 接近于 0, 当权重精度超过 16 bit 时, 权重精度的增加对短期预测结构相似度几乎无影响.

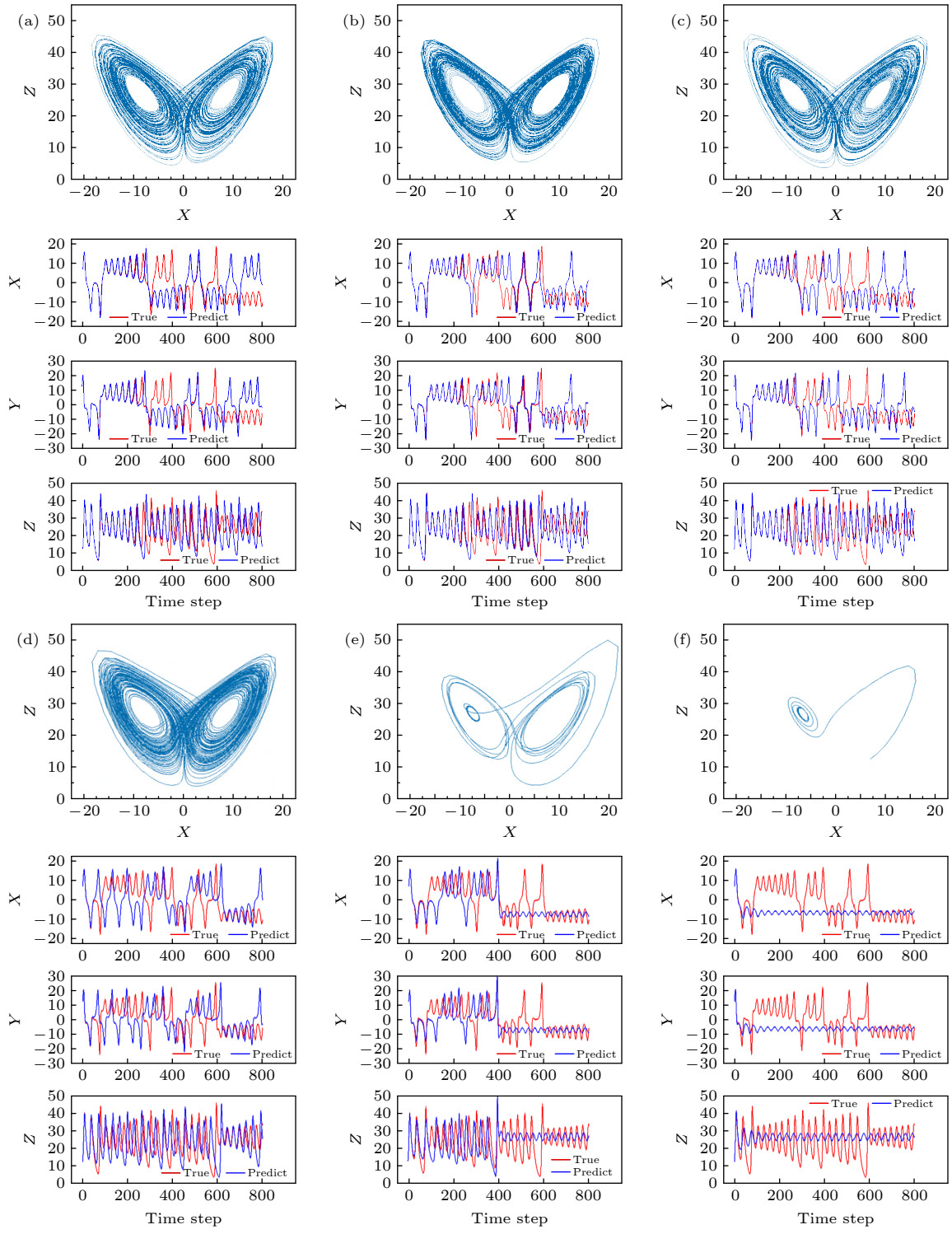

图 4 输入精度为定点 32 bit, 输出精度为定点 64 bit, 不同权重精度下 800 个时间步的预测 XZ 截面图 (a) 64 bit; (b) 32 bit; (c) 16 bit; (d) 8 bit; (e) 6 bit; (f) 4 bit

Fig. 4. The XZ cross sections of 800 time steps with different weight precision, when input precision of integer is 32 bit and output precision of integer is 64 bit: (a) 64 bit; (b) 32 bit; (c) 16 bit; (d) 8 bit; (e) 6 bit; (f) 4 bit.

Lorenz63 系统的 z 分量在连续的局部极大值之间具有函数关系, 通过找到 z 分量的连续局部极大值 M_i 并根据 M_{i+1} 画出 M_i 形成 z 回归图, 可以简洁地展现 z 变量的长期行为. Lorenz63 系统的 z 回归图如图 6 所示, 紫色点为真实序列 z 回归图, 其他颜色为输入精度为定点 32 bit, 输出精度为定点 64 bit, 权重精度分别为 8, 16, 32, 64 bit 的 z 回归图. 结果显示, 权重精度为 8 bit 时的 z 回归图相

比真实序列的回归图有明显偏移; 当权重精度在 16 bit 及以上时, 预测的回归图几乎完全覆盖了真实序列的回归图; 随着权重精度的增加, 预测的回归图往真实序列的回归图收敛.

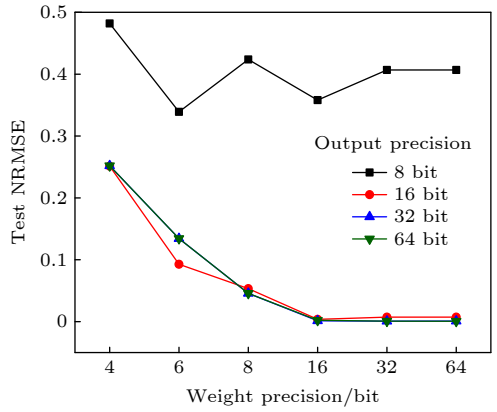

图 5 短期预测 (1 个李雅普诺夫周期) 的 NRMSE 随不同权重精度 (8, 16, 32, 64 bit) 和不同输出精度 (8, 16, 32, 64 bit) 的变化

Fig. 5. The variation diagram of NRMSE for short-term prediction (1 Lyapunov cycle) with different weight precision (8, 16, 32, 64 bit) and different output precision (8, 16, 32, 64 bit).

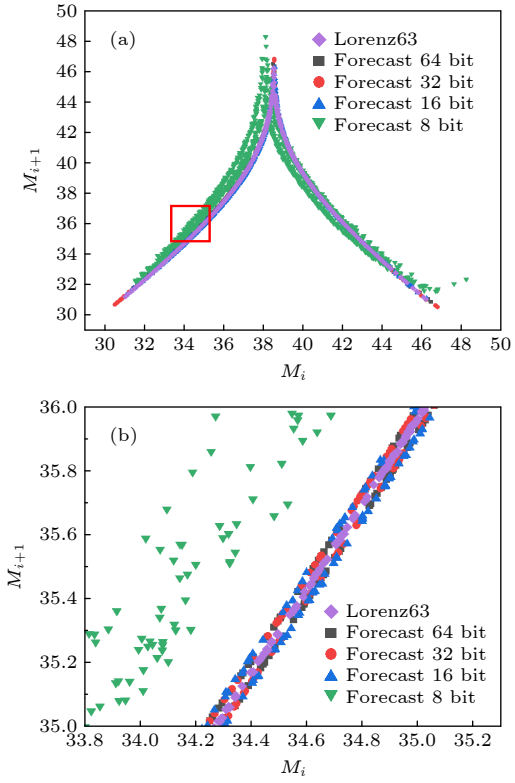

图 6 (a) Lorenz63 的 z 回归图 (紫色) 与不同权重精度下预测的 z 回归图; (b) 图 (a) 红框区域标记中的放大图

Fig. 6. (a) The z return map of Lorenz63 (purple) overlaid with the z return map under different weight accuracy; (b) detail of the region marked in Fig. (a).

3.2 噪声鲁棒性验证实验

在保持输入精度为定点 32 bit, 输出精度为定点 64 bit, 给权重 (即忆阻器电导 G) 添加高斯噪声 Noise $N(0,\sigma)$, 其中 $\sigma = G \times 10^{-4} \times \text{percent}$ 为方差, percent 表示噪声强度百分比. 短期预测结构相似度的 NRMSE 随权重噪声强度变化的结果如图 7 所示. 当权重精度在 8 bit 时, 随着 σ 的增大, 短期预测结构相似度的 NRMSE 会先降低后升高; 当权重精度在 16 bit 时, 短期预测结构相似度的 NRMSE 也会先降低后升高, 但降低点对应的噪声强度比权重精度在 8 bit 时小; 当权重精度在 16 bit 以上时, 短期预测的 NRMSE 会随着 percent 的增大而增大. 由于量化本身具备一定的抗噪声能力, 故权重精度越低, 噪声对短期预测结构相似度的 NRMSE 的影响越小; 值得注意的是, 一定程度的噪声有利于提升短期预测性能, 并且量化的比特数越高, 能带来增益的噪声强度越小.

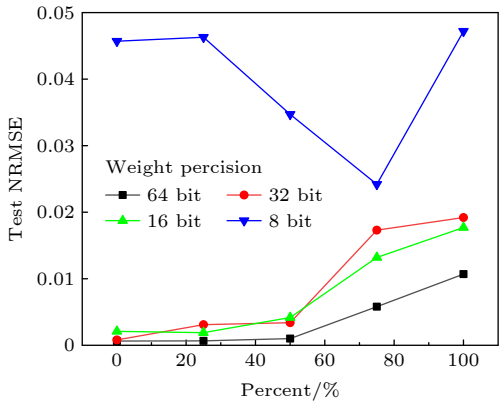

图 7 短期预测结构相似度的 NRMSE 在不同权重精度条件下随权重噪声强度的变化

Fig. 7. The variation of NRMSE under different weight precision conditions for short-term prediction with increasing weight noise intensity.

3.3 讨 论

基于忆阻器阵列的 NGRC 存内实现具备两方面的优势: 第一, 就 NGRC 算法本身而言, 相对于传统储池计算和基于延时的储池计算, NGRC 的储池具备更短的激活时间、更少的参数训练量以及更快的训练和推理速度[42]; 第二, 就存内计算方面而言, NGRC 中提取非线性特征向量的过程需要大量的乘法操作, 而忆阻器阵列相比传统 CMOS 电路, 在矩阵向量乘法方面具备更快的计算速度和更低的功耗[49]. 然而, 使用忆阻器阵列进行 NGRC

的过程中, 每一次推理过程都需要在忆阻器阵列中写入采样数据; 同时, 仿真结果表明, 忆阻器阵列中每个忆阻器精度达到 8 bit, Lorenz63 才能有较好的预测结果. 考虑到当前忆阻器还存在各种非理想性因素, 因此, 如何进一步提高写入效率, 同时降低所需忆阻器的阻值精度还需进一步探索.

4 结 论

储池计算自提出至今可以分为传统储池计算、延时储池计算和下一代储池计算三个阶段. 储池计算性能上的优势不仅来自于算法自身, 而且与硬件的实现方式密切相关. 本文在总结储池计算发展历程的基础上, 提出一种基于存内计算范式的硬件实现方法, 将 NGRC 过程通过矩阵向量乘法操作简化, 并利用忆阻器阵列完成矩阵向量乘法操作. 忆阻器阵列仿真实验验证了这一方法在 Lorenz63 三维时间序列预测任务中的可行性. 仿真实验结果表明, 预测效果与输出精度和权重精度密切相关. 当输出精度达到 16 bit, 进一步提高输出精度对预测效果的影响可忽略不计, 并且具有良好的抗噪声能力; 当权重精度达到 8 bit, 对 Lorenz63 三维时间序列预测的短期预测 (1 个李雅普诺夫时间) 就可以有良好的预测效果 (NRMSE 小于 0.05), 并可以在一定程度进行长期预测. 这些结果为 NGRC 的硬件实现提供了一种新的途径, 同时也展现了忆阻器阵列在开发基于储池计算的实时、低能耗边缘计算系统方面的潜力.

参考文献

[1] Guillem C, Jordi F 2015 *Front. Psychol.* **6** 818
[2] Dayan P, Abbott L F 2001 *J. Cogn. Neurosci.* **15** 154
[3] Vogels T P, Rajan K, Abbott L F 2005 *Annu. Rev. Neurosci.* **28** 357
[4] Tian Y, Li G, Sun P 2021 *Phys. Rev. Res.* **3** 043085
[5] Borst A, Theunissen F E 1999 *Nat. Neurosci.* **2** 947
[6] Amit D J, Gutfreund H, Sompolinsky H 1987 *Phys. Rev. A* **35** 2293
[7] Danilo P, Mandic J A C 2001 *Recurrent Neural Networks Architecture* (Hoboken: John Wiley & Sons Ltd) pp69–89
[8] Choi E, Schuetz A, Stewart W F, Sun J 2016 *J. Am. Med. Inform. Assoc.* **24** 361
[9] Pascanu R, Mikolov T, Bengio Y 2013 *Proceedings of the 30 th International Conference on Machine Learning* Atlanta, Georgia, USA, June 16–21, 2013 p1310
[10] Dominey P, Arbib M, Joseph J P 1995 *J. Cogn. Neurosci.* **7** 311
[11] Jaeger H 2001 *German National Research Institute for Computer ScienceGerman National Research Centre for Information Technology, GMD Technical Reports* Bonn, Germany, January 01, 2001 p13
[12] Jaeger H, Haas H 2004 *Science* **304** 78
[13] Maass W, Natschlager T, Markram H 2002 *Neural Comput.* **14** 2531
[14] Kan S, Nakajima K, Takeshima Y, Asai T, Kuwahara Y, Akai-Kasaya M 2021 *Phys. Rev. Appl.* **15** 024030
[15] Pathak J, Hunt B, Girvan M, Lu Z, Ott E 2018 *Phys. Rev. Lett.* **120** 024102
[16] Chattopadhyay A, Hassanzadeh P, Subramanian D 2020 *Nonlinear Processes Geophys.* **27** 373
[17] Lukoševičius M, Jaeger H, Schrauwen B 2012 *KI - Künstliche Intelligenz* **26** 365
[18] Boyd S, Chua L 1985 *IEEE Trans. Circuits Syst.* **32** 1150
[19] Grigoryeva L, Ortega J P 2018 *Neural Networks* **108** 495
[20] Zhao C, Li J, Liu L, Koutha L S, Liu J, Yi Y 2016 *Proceedings of the 3 rd ACM International Conference on Nanoscale Computing and Communication* New York, New York, USA, September 28–30, 2016 p1
[21] Canaday D, Griffith A, Gauthier D 2018 *Chaos:An Interdisciplinary Journal of Nonlinear Science* **28** 123119
[22] Yi Y, Liao Y, Fu X 2016 *Microprocess. Microsyst.* **46** 175
[23] Bertschinger N, Natschlager T 2004 *Neural Comput.* **16** 1413
[24] Yang X, Chen W, Wang F 2016 *Analog Integr. Circuits Signal Process.* **87** 263
[25] Merkel C, Saleh Q, Donahue C, Kudithipudi D 2014 *5 th Annual International Conference on Biologically Inspired Cognitive Architectures (BICA)* MIT Campus, Cambridge, Massachusetts, USA, November 7–9, 2014 p249
[26] Donahue C, Merkel C, Saleh Q, Dolgovs L, Ooi Y, Kudithipudi D, Wysocki B 2015 *IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)* Verona, New York, May 26–28, 2015 p24
[27] Demis E, Aguilera R, Scharnhorst K, Aono M, Stieg A, Gimzewski J 2016 *Jpn. J. Appl. Phys.* **55** 1102B2
[28] Lilak S, Woods W, Scharnhorst K, Dunham C, Teuscher C, Stieg A, Gimzewski J 2021 *Front. in Nanotechnol.* **3** 1
[29] Vandoorne K, Mechet P, van Vaerenbergh T, et al. 2014 *Nat. Commun.* **5** 3541
[30] Milano G, Pedretti G, Montano K, Ricci S, Hashemkhani S, Boarino L, Ielmini D, Ricciardi C 2021 *Nat. Mater.* doi: 10.1038/s41563-021-01099-9
[31] Gallicchio C, Micheli A, Pedrelli L 2017 *Neurocomputing* **268** 87
[32] Qiao J, Li F, Han H G, Li W 2016 *IEEE Trans. Neural Networks Learn. Syst.* **28** 391
[33] Tong Z Q, Tanaka G 2018 *24th International Conference on Pattern Recognition (ICPR)* Beijing, China, August 20–24, 2018 p1289
[34] Murakamli M, Kroger B, Birkholz P, Triesch J 2015 *5th IEEE Joint International Conference on Development and Learning and on Epigenetic Robotics (IEEE ICDL-EpiRob)* Providence, Rhode Island, August 13–16, 2015 p208
[35] Sussillo D, Abbott L F 2009 *Neuron* **63** 544
[36] Tanaka G, Yamane T, Heroux J B, et al. 2019 *Neural Networks* **115** 100
[37] Lepri S, Giacomelli G, Politi A, Arecchi F T 1994 *Physica D* **70** 235
[38] Appeltant L, Soriano M C, van der Sande G, et al. 2011 *Nat. Commun.* **2** 468 468
[39] Brunner D, Penkovsky B, Marquez B A, Jacquot M, Fischer I, Larger L 2018 *J. Appl. Phys.* **124** 152004
[40] Penkovsky B, Larger L, Brunner D 2018 *J. Appl. Phys.* **124** 162101
[41] Yu J, Li Y, Sun W, et al. 2021 *Symposium on VLSI*

Technology Kyoto, Japan, June 13–19, 2021 p1
[42] Gauthier D J, Bollt E, Griffith A, Barbosa W A S 2021 *Nat. Commun.* **12** 5564
[43] Strukov D B, Snider G S, Stewart D R, Williams R S 2008 *Nature* **453** 80
[44] Li H, Wang S, Zhang X, Wang W, Yang R, Sun Z, Feng W, Lin P, Wang Z, Sun L, Yao Y 2021 *Adv. Intell. Syst.* **3** 2100017
[45] Li Y, Loh L, Li S, Chen L, Li B, Bosman M, Ang K W 2021 *Nat. Electron.* **4** 348
[46] Kim H, Mahmoodi M R, Nili H, Strukov D B 2021 *Nat. Commun.* **12** 5198
[47] Xiao T P, Bennett C H, Feinberg B, Agarwal S, Marinella M J 2020 *Appl. Phys. Rev.* **7** 031301
[48] Lorenz E N 2004 *The Theory of Chaotic Attractors* (New York: Springer New York) pp25–36
[49] Zhang W, Gao B, Tang J, Yao P, Yu S, Chang M F, Yoo H J, Qian H, Wu H 2020 *Nat. Electron.* **3** 371

SPECIAL TOPIC—Physical electronics for brain-inspired computing

Next-generation reservoir computing based on memristor array*

Ren Kuan [1)2)#] Zhang Wo-Yu [1)3)#] Wang Fei [1)3)]
Guo Ze-Yu [1)3)] Shang Da-Shan [1)3)†]

1) (*Key Laboratory of Microelectronics Devices and Integrated Technology, Institute of Microelectronics, Chinese Academy of Sciences, Beijing* 100029, *China*)
2) (*Key Laboratory of Magnetic Levitation Technologies and Maglev Trains, Ministry of Education, Superconductivity and New Energy R&D Center, Southwest Jiaotong University, Chengdu* 610031, *China*)
3) (*University of Chinese Academy of Sciences, Beijing* 100049, *China*)

(Received 12 January 2022; revised manuscript received 26 January 2022)

Abstract

As a kind of brain-inspired computing, reservoir computing (RC) has great potential applications in time sequence signal processing and chaotic dynamics system prediction due to its simple structure and few training parameters. Since in the RC randomly initialized network weights are used, it requires abundant data and calculation time for warm-up and parameter optimization. Recent research results show that an RC with linear activation nodes, combined with a feature vector, is mathematically equivalent to a nonlinear vector autoregression (NVAR) machine, which is named next-generation reservoir computing (NGRC). Although the NGRC can effectively alleviate the problems which traditional RC has, it still needs vast computing resources for multiplication operations. In the present work, a hardware implementation method of using computing-in memory paradigm for NGRC is proposed for the first time. We use memristor array to perform the matrix vector multiplication involved in the nonlinear vector autoregressive process for the improvement of the energy efficiency. The Lorenz63 time series prediction task is performed by simulation experiments with the memristor array, demonstrating the feasibility and robustness of this method, and the influence of the weight precision of the memristor devices on the prediction results is discussed. These results provide a promising way of implementing the hardware NGRC.

Keywords: reservoir computing, memristor, in-memory computing, nonlinear vector autoregression

PACS: 07.05.Mh, 84.35.+i, 85.40.–e, 87.15.A– **DOI:** 10.7498/aps.71.20220082

* Project supported by the National Basic Research Program of China (Grant No. 2018YFA0701500), the National Natural Science Foundation of China (Grant No. 61874138), and the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDB44000000).

# These authors contributed equally.
† Corresponding author. E-mail: shangdashan@ime.ac.cn