Memristor-Based Echo State Network With Online Least Mean Square

Shiping Wen, *Member, IEEE*, Rui Hu, *Student Member, IEEE*, Yin Yang, *Member, IEEE*, Tingwen Huang, *Senior Member, IEEE*, Zhigang Zeng, *Senior Member, IEEE*, and Yong-Duan Song, *Senior Member, IEEE*

***Abstract*—In this paper, we propose a novel computational architecture of memristor-based echo state network (MESN) with the online least mean square (LMS) algorithm. Newman and Watts small-world network is adopted for the topological structure of MESN network with memristive neural synapses. In the MESN network, the state matrix of the reservoir layer, which is obtained by raising the dimension of input data, is utilized as an input of the LMS algorithm to train the output weight matrix on chip. After certain iterations, the resistance value of memristor is adjusted to a constant. Thus, the final weight output matrix is obtained. To verify the effectiveness of the proposed MESN network, car evaluation and short-term power load forecasting are employed with the effect evaluation of the node number and the connectivity degree of the reservoir layer. The research provides a novel way to design neuromorphic computing systems.**

***Index Terms*—Echo state network, least mean quare, memristor, neural network.**

I. Introduction

COMPARED with the traditional computer, human brain is not only characterized as a biological organization but also a large number of parallel nonlinear processing units with self-organized and self-learning properties. Biological neural networks are the most skillful and complex information processing systems with unparalleled information processing capabilities in real life [1], [2].

Artificial neural network (ANN) is utilized to simulate the structure and function of biological neural network [3]. Based on the rapid development of contemporary neurobiology, mathematics, physics, and computer science, ANN research becomes a hot topic [2], [4], [5] with a large number of proposed ANN models, such as the McCulloch–Pitts neuron model [6], linear threshold function neuron model [7], backpropagation neural network model [8], RNN model [9], and so on.

Compared with feed/forward neural networks, RNNs can describe the system dynamic characteristics better [10] and simulate the human brain more appropriately. Under normal circumstances, RNNs can infinitely approach any complex nonlinear dynamic system. These advantages make RNN widely used in prediction [11], nonlinear system identification [12], and adaptive filtering [13]. Nonetheless, some inevitable problems still exist in RNNs, for example, the network structure and training algorithm are complex, the computation is large, the convergence speed is slow, and the error gradient may disappear or produce distortion with the increase of training steps. Some improvements have been achieved in recent years [14]–[16]. However, it is difficult to solve the problem of the heavy computational burden in the training process.

In order to reduce the computational burden and overcome the problem of memory fading in the training process, in 2001 [17], Jaeger proposed ESN, which also solves the structural defects in traditional RNNs. In 2002, ESN is collectively referred to as "reservoir computing model" [18]. The most significant difference between ESN and traditional RNNs lies in the structure of reservoir layer. The weights of each layer in traditional RNNs are computed by the gradient descent algorithm. However, in ESN, only the weights between reservoir layer and output layer need to be trained, which greatly reduces the computational complexity of traditional RNNs. These advantages make ESN gradually become a research hot topic in the field of neural networks [19], [20].

At present, ANN is mainly implemented by computer software. The computer processor can only execute one instruction per core in a cycle, so the processing speed is slow and difficult to meet the real-time requirements. Meanwhile, in some harsh applications high stability are required, the software implementation of neural networks is not applicable. On the

Manuscript received February 7, 2018; accepted April 3, 2018. Date of publication May 4, 2018; date of current version August 16, 2019. This work was supported in part by the Natural Science Foundation of China under Grant 61673187, Grant 61673188, and Grant 61773081, in part by the Technology Transformation Program of Chongqing Higher Education University under Grant KJZH17102, and in part by NPRP from the Qatar National Research Fund (a member of Qatar Foundation) under Grant NPRP 9-466-1-103. This paper was recommended by Associate Editor J. Lu. *(Corresponding author: Zhigang Zeng.)*

S. Wen, R. Hu, and Z. Zeng are with the School of Automation, Huazhong University of Science and Technology, Wuhan 430074, China, and also with the Key Laboratory of Image Processing and Intelligent Control of Education Ministry of China, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: wenshiping226@hust.edu.cn; hurui315309@qq.com; zgzeng@hust.edu.cn).

Y. Yang is with the College of Science, Engineering, and Technology, Hamad Bin Khalifa University, Doha, Qatar (e-mail: yyang@hbku.edu.qa).

T. Huang is with Science Program, Texas A&M University at Qatar, Doha, Qatar (e-mail: tingwen.huang@qatar.tamu.edu).

Y.-D. Song is with the Key Laboratory of Dependable Service Computing in Cyber Physical Society of Ministry of Education, School of Automation, Chongqing University, Chongqing 400044, China (e-mail: ydsong@cqu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2018.2825021

other hand, hardware implementation, which includes optical technology, very-large-scale integration (VLSI) technology and molecular biotechnology, can exert the characteristics of large-scale parallel processing for ANN, especially in dealing with big data problems [21]. VLSI technology is widely applied for high precision, mature technology, and superior anti-noise capability. Therefore, the realization of neural networks usually use VLSI Technology. In 1989, Mead [22] proposed a concept of neuromorphic computing and a method to simulate the structure of the nervous system with a neural circuit. Subsequently, Smith and Hamliton [23] proposed a kind of VLSI to simulate the brain structure and function. Neural networks have been implemented on CPU, GPU [24], FPGA [25], and other computing platforms, but when the computation load of the system becomes larger, memory wall and other issues will arise [26]. In order to enhance scalability, a novel neural system based on pulse time signal has been proposed to achieve biological functions by the traditional CMOS devices directly. Each synaptic weight is stored in the SRAM unit, and takes the form of pulse signal to learn and recall [27].

The realization of CMOS-based computing hardware is of great significance to neuromorphic computing, but there still exist many issues left to be solved. For example, the leakage power is too high and the storage density of the SRAM unit is too low, which leads to a large amount of energy consumption in the process of storing, reading, and programming synaptic weights [28]. At present, the number of synapses in a single integrated circuit is only 10^2-10^5, but the number of synapses in brain is up to 10^{14}. In a word, the energy consumption of these traditional means is too large, and the density of synapses is not up to the level of biological brain.

In 1971, based on the relationships between charge, current, voltage, and magnetic flux, Chua [29] suspected the existence of fourth electronic components and proposed the concept of memristor. Until 2008, HP labs successfully fabricated the memristor with the semiconductor material [30]. The emergence of memristor promotes the development of neural networks with nonvolatile memory. Meanwhile, memristor can also perform logic operations with a very high degree of integration as a nanoscale device. The excellent characteristics of memristor provide great foundation to develop neuromorphic systems. Combined with memristor, several neural networks have been successfully designed [31]–[34].

The traditional reservoir layer is usually composed of a large number of neurons, this kind of complex physical topology puts forward a high requirement to the hardware technology. Therefore, the traditional model of reservoir layer is mainly based on software. In recent years, there are quite a lot of research results about hardware implementation of ESN [35]–[37]. Kudithipudi *et al.* [38], Saleh *et al.* [39], and Merkel *et al.* [40] optimized the architecture of reservoir computing. Based on the new structure and memristor, scalable computational circuits of echo state network are realized. The circuits are applied separately to biological signal processing, speech-emotion recognition, and epileptic seizure detection. Kulkarni and Teuscher [41] and Yang *et al.* [42] used memristor to realize random connection between neuron nodes of the reservoir layer. All these researches provide references for hardware implementation of ESN. Furthermore, hardware implementation of ESNs was first proposed to use memristor double crossbar arrays [43]. However, memristor is mainly used as a connection between synaptic nodes in the reservoir layer, the training of the weights between reservoir layer and output layer still adopts the method of calculating pseudo inverse matrix, which is difficult to realize with circuits.

In this paper, a new scheme is proposed to design memristor-based echo state network (MESN), in which reservoir layer is based on Newman and Watts (NW) small-world network, and the training of the output layer weights adopts the least mean square (LMS) algorithm. The state matrix of the reservoir layer, which is obtained by raising the dimension of input data, is utilized as an input of the LMS algorithm to train the output weight matrix on chip. After certain iterations, the resistance value of memristor is adjusted to a constant. Thus, the final weight output matrix is obtained. The proposed scheme is applied to car evaluation and short-term power forecasting. The rest of this paper is organized as follows. The training algorithm of ESN is discussed in Section II. Section III provides the improved architecture and the design of synapse circuit. Section IV discusses the application of car evaluation and short-term power load forecasting and Section V presents the conclusions.

II. ESN Training Algorithm

ESN, as a special class of RNN, was proposed by Jaeger in 2001 [17]. Different from the traditional RNNs, ESN only needs to adjust the output weights between the reservoir layer and output layer. Moreover, the reservoir layer of ESN is randomly generated with sparse connected neuron with excellent nonlinear processing capability and simple effective training process.

A. Basic Structure of Echo State Network

As a kind of neural networks, ESN also has three layers: 1) input layer; 2) hidden layer; and 3) output layer as shown in Fig. 1. The hidden layer is called reservoir layer, which is composed of a large scale and sparse connected recurrent neural network. The reservoir layer has the following characteristics: 1) the number of neurons is huge; 2) the connections between neurons are randomly generated; and 3) connections between neurons are sparse.

For the topology shown in Fig. 1, the state vectors of the reservoir layer can be iterated and updated as

$$\mathbf{x}(n+1) = f(\mathbf{W}\mathbf{x}(n) + \mathbf{W}_{\text{in}}\mathbf{u}(n) + \mathbf{W}_{\text{back}}\mathbf{y}(n)) \tag{1}$$

where $\mathbf{u}(n)$ is the ESN input signal connected to the reservoir layer through an input-connect matrix $\mathbf{W}_{\text{in}}$. In addition, the reservoir layer can also receive feedback $\mathbf{y}(n)$ from the output layer and the internal feedback of reservoir layer $\mathbf{x}(n)$. $\mathbf{W}_{\text{back}}$ is the feedback-connect matrix between the output and the reservoir layer. $\mathbf{W}$ is the weight matrix between neurons in the reservoir layer. f is the activation function of the reservoir layer, in general, it is sigmoid function or tanh function. Then,

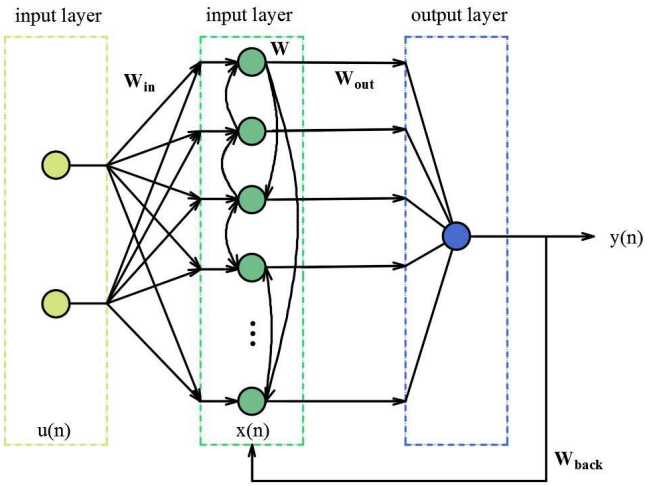


Fig. 1. Structure of ESN.

the output of ESN can be calculated as

$$\mathbf{y}(n+1) = f_{\text{out}}(\mathbf{W}_{\text{out}}\mathbf{x}(n+1)) \tag{2}$$

where $\mathbf{W}_{\text{out}}$ is the output weight matrix at the readout layer and f_{out} is the activation function of the readout layer. The output layer of the classical echo state network is a linear mapping layer, so f_{out} is generally a function of identity.

Meanwhile, the weights of the reservoir layer can be randomly set in advance rather than generated through training, the output layer weights can be trained separately to achieve the outstanding performance of ESN. Different from traditional neural networks, weight matrices W, W_{in} and W_{back} are randomly generated in the network initialization phase, and remain unchanged during the training process. Only the weight matrix W_{out} between reservoir layer and output layer needs to be trained. The ESN network separates the information expression and the weight training process of the network in an ingenious way. The training of the output weights is independent of the weight setting of reservoir layer, which greatly simplifies the training process of RNN. On the other hand, the dynamic reservoir layer maps low-dimensional input to high-dimensional feature space, which guarantees the network to be trained with linear training method. The training process only needs to solve a linear regression problem. Therefore, the training process of the network can be completed by using simple linear regression algorithms. This greatly reduces the computational complexity of traditional recurrent neural networks and speeds up the convergence.

B. Training Algorithm of Echo State Network

Classical ESN usually adopts supervised learning and batch learning methods. Suppose that ESN contains L input units, M output units, and N reservoir nodes. The state matrix of reservoir $\mathbf{x}(n)$ and the actual output $\hat{\mathbf{y}}(n)$ according to (1) and (2). To eliminate the influence of the initial system state of the dynamic characteristics, sample work is usually carried out after a period. It is assumed that the internal state of the reservoir layer is sampled from the time m. $\mathbf{x}_1(i), \mathbf{x}_2(i), \ldots, \mathbf{x}_n(i)(i = m, m+1, \ldots, M)$ is used to constitute the row vector of the matrix $\mathbf{X}(M-m+1, N)$, the label data $\mathbf{y}(n)$ were sampled at the same time, which constitutes the matrix $\mathbf{Y}(M-m+1, 1)$. The state matrix $\mathbf{x}(n)$ is linear with the system output $\hat{\mathbf{y}}(n)$, however, the target is to make the actual output of the network $\mathbf{y}(n)$ approach the expected output $\mathbf{y}(n)$ as shown in

$$\mathbf{y}(n) \approx \hat{\mathbf{y}}(n) = \sum_{i=1}^{L} \mathbf{w}_i^{\text{out}} \mathbf{x}_i(n). \tag{3}$$

To make the weights $\mathbf{w}_{\text{out}}$ satisfy the requirement that the mean square error of the system is minimized, the following optimization problems need to be solved as:

$$\min_{\mathbf{w}_i^{\text{out}}} \frac{1}{M-m+1} \sum_{n=m}^{M} \left(\left(\mathbf{y}(n) - \sum_{i=1}^{L} \mathbf{w}_i^{\text{out}} \mathbf{x}_i(n) \right)^2 \right). \tag{4}$$

If viewed from mathematics, it is a linear regression problem, so the solution process of the weights $\mathbf{W}_{\text{out}}$ can be converted to calculating the inverse matrix problem of matrix $\mathbf{X}$ as

$$\mathbf{W}^{\text{out}} = \mathbf{X}^{-1}\mathbf{Y}. \tag{5}$$

C. Online Training Algorithm Based on Least Mean Square

However, the training algorithm mentioned above is a basic offline training algorithm (or called batch algorithm). Since the matrix $\mathbf{X}$ may be singular, the inverse matrix of the matrix $\mathbf{X}$ can be calculated by using the pseudo inverse algorithm or the regularization technique, ridge regression algorithm [44] is commonly used. In order to solve the real-time requirements of the online problem, online training algorithm based on recursive least square algorithm is proposed with LMS to train the weights W_{out} of the readout layer [45].

Widrow and Hoff [46] proposed LMS adaptive filters in 1960. The weight adjustment with LMS algorithm is based on the error correction-learning rule, which is easy to be achieved and has been widely used. LMS algorithm can only be trained on a single layer network, thus it can only solve linearly separable problems. However, as mentioned above, reservoir layer guarantees good performance of the network with linear training method. Therefore, LMS algorithm is suitable for ESN and easier to be achieved with hardware rather than other learning methods.

The steps of the LMS algorithm are presented as follows.

Step 1: Define variables and parameters. In order to facilitate the processing, bias is combined with weights

$$\mathbf{w}(n) = [\mathbf{b}(n), \mathbf{w}_1(n), \mathbf{w}_2(n), \ldots, \mathbf{w}_N(n)]^T \tag{6}$$

where $b(n)$ is bias, n is iteration number.
The corresponding training sample is

$$\mathbf{x}(n) = [1, \mathbf{x}_1(n), \mathbf{x}_2(n), \ldots, \mathbf{x}_N(n)]^T. \tag{7}$$

Step 2: The initialization. Assign small random initial values to the weights $\mathbf{w}(n), n = 0$.

Step 3: Input the sample, calculate actual output $\mathbf{y}(n)$ and error $\mathbf{e}(n)$. According to the given expected output $\mathbf{d}(n)$, we can calculate

$$\mathbf{y}(n) = \mathbf{x}^T(n)\mathbf{w}(n) \tag{8}$$

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{y}(n). \tag{9}$$

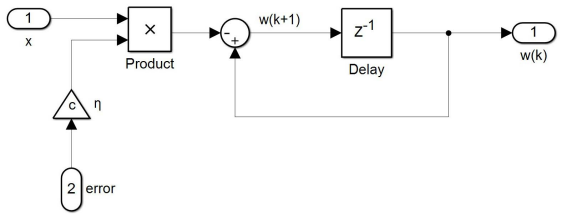


Fig. 2. Realization block diagram of (10).

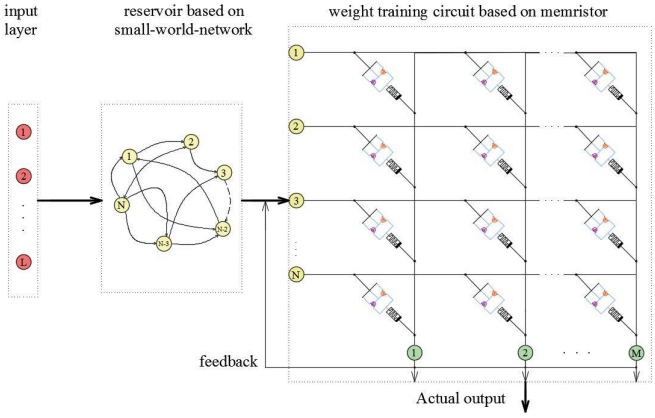


Fig. 3. Architecture overview of MESN.

Step 4: Adjust the weight vector. Set the learning rate η and calculate

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta \mathbf{x}^T(n)\mathbf{e}(n). \tag{10}$$

Realization block diagram of (10) is as shown in Fig. 2.

Step 5: Judge whether the algorithm is convergent. If the convergent condition is satisfied, the algorithm will be ended, otherwise $n = n+1$ and jump to the third step.

III. ESN in Hardware

A block diagram is shown in Fig. 3, which consists of three modules: 1) the input layer; 2) the reservoir based on small-world network; and 3) the weight training circuit based on memristor. Dealt with the reservoir based on small-world network, the L-dimensional input vector is mapped to N-dimensional feature space. The output of reservoir is treated as the input of weight training circuit based on memristor. The following two aspects will be described in detail.

1) Architectural topology of the reservoirs.
2) Training algorithm based on memristor.

A. Architectural Topology of the Reservoir Layer

The reservoir layer plays a key role in the whole network. Jaeger [17] stated that it is coordinated to research the reservoir layer and the support vector machine (SVM). Meanwhile, it is similar to optimize the reservoir layer and the kernel function of SVM. In the classical ESN, most of the network parameters and connections between neurons are randomly generated. However, randomly generated reservoir layer may not be optimal. Meanwhile, in the classical model as shown in Fig. 1, the reservoir layer is composed of a large number of interconnected nodes in order to complete the mapping from low-dimensional input to high-dimensional state space. It is difficult to implement the complex random topology with hardware. The reservoir layer is supposed to be designed as a regular network to reduce the complexity of hardware implementation. Due to the difference in internal structure, the networks with same number of neurons show a great difference in performance. Some methods have been used to subvert this structure [47]–[50], such as small-world networks and scale-free networks [51]. The results show that ESN with topology of low complexity can also obtain superior memory ability and high prediction accuracy.

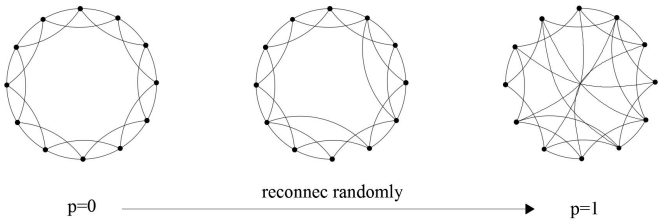


Fig. 4. WS small-world network.

Several schemes have been proposed to change the topology of ESN, such as regular network structure and small world network structure. Small world network, which is proposed by Watts and Strogatz (WS) in 1998 [52], is a new kind of network structure between regular network and random network, and also called WS small-world network. Combined with the advantages of regular network and random network, this network structure has not only a large clustering coefficient but also a smaller average path length. The network links are disrupted and reconnected with certain probability p. If $p = 0$, the reconnection will not happen, and finally only the original rule network will be obtained; if $p = 1$, all connections are reconnected, finally, a complete random network will be obtained as shown in Fig. 4. Due to the random reconnection, WS small-world network sometimes destroys the connectivity of the whole network. It limits the transmission of information in the network to some extent. To overcome this shortcoming, NW proposed another slightly modified network structure in 1999, commonly known as the NW small-world network [53]. The structure also starts with a regular network, adopts the method of adding edges by randomization with a certain probability p, so as to transforms regular network into small world network as shown in Fig. 5. This network structure can effectively avoid the generation of isolated nodes in the network.

In small-world networks, there are many shortcuts which can make the feature length of the network smaller than the regular network, but also have a large clustering coefficient, which can reflect the clustering characteristics. These features are similar to human brain. In addition, the actual social, ecological and other networks are small world, such a network structure makes the information transmission faster, and changing several connections can dramatically change the

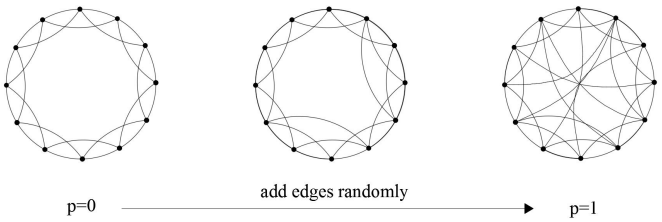


Fig. 5. NW small-world network.

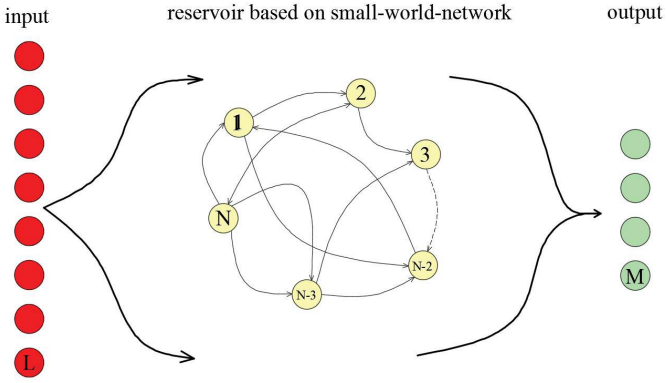


Fig. 6. ESN with NW small-world network.

performance of the network. The results show that the network performance is better than the regular network and random network because of the addition of some random shortcuts to the regular networks. Therefore, the small-world network topology is used to replace the internal topology of ESN. NW small-world network model is chosen since it can effectively avoid the generation of isolated nodes in the network. As shown in Fig. 6, ESN is presented with the topology of NW small-world network.

B. Training Algorithm Based on Memristor

1) Memristor: Memristor is a kind of passive two-terminal device. Besides resistance, capacitance, and inductance, it is the fourth basic component of electronic circuits. The concept of memristor was first developed by Prof. Chua [29] at UC Berkeley in 1971. Until 2008, the research team led by Williams of HP labs first produced memristor [30].

As is well known, four kinds of basic elements of the circuit include the voltage (v), current (i), charge (q), and magnetic flux (φ). Resistor (R) represents the differential relationship between voltage and current, capacitor (C) represents the differential relationship between charge and voltage, inductor (L) represents the differential relationship between magnetic flux and current. According to the theory of symmetry, Chua believes there should be a component that represents the differential relationship between the magnetic flux (φ) and charge (q) [30].

Prof. Chua put forward that memristor can be expressed by flux (φ) and charge (q) as

$$g(\varphi, q) = 0. \tag{11}$$

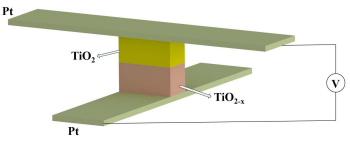


Fig. 7. Microstructure of the physical memristor produced by HP labs.

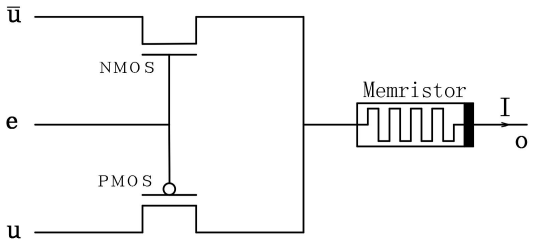


Fig. 8. Schematic of the single synapse.

In particular, when the formula above is represented by a single valued function of the charge, there is

$$\varphi = f(q). \tag{12}$$

Take the derivative of the formula (12) on both sides

$$\frac{d\varphi(t)}{dt} = \frac{df(q)}{dq} \cdot \frac{dq}{dt}. \tag{13}$$

According to the existing physical formula

$$v(t) = \frac{d\varphi(t)}{dt}, i(t) = \frac{dq}{dt}. \tag{14}$$

Substituting (14) and (12) into (13), we have

$$v(t) = \frac{d\varphi(q)}{dq} \cdot i(t) \tag{15}$$

where $i(t)$ represents the current flowing through memristor. Formula (15) is the expression of the charge controlled memristor, where $M(q) = (d\varphi(q)/dq)$ is called the value of memristor. On the other hand, when (11) is represented by a single valued function of magnetic flux, with the same argument, there will be

$$i(t) = \frac{dq(\varphi)}{d\varphi} \cdot v(t) \tag{16}$$

where $v(t)$ represents the voltage flowing through memristor. Formula (16) is the expression of the magnetically controlled memristor, where $W(\varphi) = (dq(\varphi)/d\varphi)$ is called memory conductance. Then the memristance can be expressed as $M(q) = (1/W(\varphi))$.

Since memristor has the same unit as the resistance (Ω), and has the characteristics of nonvolatile, which means that the resistance changes only in the case of current flow ($i = (dq/dt) \neq 0$). When the power is cut off, the resistance can keep unchanged. So, Prof. Chua named this component as memristor, which means a combination of memory and resistance. In 2008, HP labs first produced a physical memristor based on the gate structure of TiO2 materials, as shown in Fig. 7 [30]. Memristor becomes a new hot spot in fields of storage, ANNs, and logic computing.

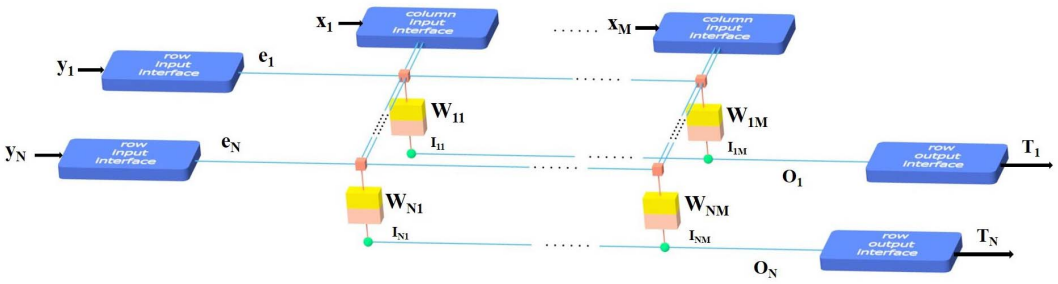


Fig. 9. Architecture of synaptic grid ($N \times M$) circuit.

2) Synapse Circuit Architecture and Operation: Based on the memristive synaptic circuit Soudry proposed [55], as shown in Fig. 8, neuromorphic computing circuits cost only 2% of the original network area and only 8% of the power consumption as the original network. In the synaptic circuit, input signal $\bar{u} = u$, $\bar{u} = -u$. The value of the control signal e is $V_{\rm DD}$, 0 or $-V_{\rm DD}$. Current I is the output signal. When $e = 0$, the two transistors are nonconduction. When $e = V_{\rm DD}$, nMOS is conducting, pMOS is nonconduction, the resistance of memristor becomes large. When $e = -V_{\rm DD}$, pMOS is conducting, nMOS is nonconduction, the value of memristor becomes small.

As shown in Fig. 9, a single layer neural network is composed with synaptic circuits in which the memristor is used to store time coding and weight values. By using Ohm's law, a multiplication operation can be completed, which greatly simplifies the hardware design and enhances the real-time reliability of the computing system. The network corresponds to a sample with M-dimensional input and N-dimensional output.

While in the computing phase, each computation phase is assumed to be $T_{\rm rd}$, then we can convert the sample input x to voltage input $u = ax$, where a is a constant less than 1 to avoid the situation that the input voltage is larger than the threshold of memristor. Meanwhile, the value of the control signal e is

$$e_n(t) = \begin{cases} V_{\rm DD}, & 0 \le t \le 0.5T_{\rm rd} \\ -V_{\rm DD}, & 0.5T_{\rm rd} \le t \le T_{\rm rd}. \end{cases} \tag{17}$$

The difference of memristor is presented as follows:

$$\Delta s_{nm} = \int_0^{0.5T_{\rm rd}} (ax_m)\, dt + \int_{0.5T_{\rm rd}}^{T_{\rm rd}} (-ax_m)\, dt = 0. \tag{18}$$

This design makes the value of memristor remain unchanged. As other storage devices, memristor achieves writing without loss. The circuit reads the output current I_{nm} at time 0+. According to Ohm's law and Kirchhoff's law, the obtained result r_n can be expressed as follows:

$$I_{nm} = x_m * w_{nm} \tag{19}$$

$$r_n = I_{n1} + I_{n2} + \cdots + I_{nm}. \tag{20}$$

Therefore, (8) mentioned is implemented as above. After sampling the result r_n, we can calculate the error y between the result r_n and the expected output d_n

$$r_n = d_n - y_n. \tag{21}$$

Formula (9) is achieved with the LMS algorithm.

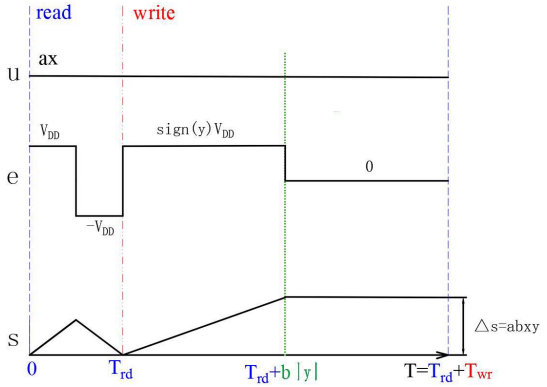


Fig. 10. Waveform illustration of each training iteration.

In the process to update weights, weights updating time is T_{wr}, u and $\bar{u}$ remain unchanged. Then control signal becomes

$$e_n(t) = \begin{cases} \text{sign}(y_n)V_{\rm DD}, & 0 \le t - T_{\rm rd} \le b|y_n| \\ 0, & b|y_n| \le t \le T_{wr}. \end{cases} \tag{22}$$

Therefore, the error is converted to the duration by the control signal e. b is a constant that transforms an error y_n into a unit of time, it makes the duration of the control signal less than the total updating time T_{wr}. Then, we can get the total change values of memristor as follows:

$$\Delta s_{nm} = \int_{T_{\rm rd}}^{T_{\rm rd}+b|y_n|} (a\ \text{sign}(y_n)x_m)\, dt = abx_m y_n \tag{23}$$

$$w(i+1) = w(i) + \Delta s_{nm} = w(i) + \eta y x^T. \tag{24}$$

Thus, (10) is achieved with the LMS algorithm. During the whole process, the time-varying values of signal and resistance are shown in Fig. 10.

IV. Simulation and Result

A. Application to Car Evaluation

With the development of automobile industry and living standards, cars are no longer luxuries for most ordinary families. Due to the market firerce competition, criteria for evaluating products has gradually changed from the performance of products to customer satisfaction. Thus, car evaluation is critical. In the paper, the purchase index is based on the specific attributes of the car. The data set "Car Evaluation Database" comes from standard data sets in the machine learning database UCI, which has no missing data, and the values of each attribute are discrete and uniformly distributed.

TABLE I
VALUES OF SIX ATTRIBUTES TO EVALUATE CARS

attributes	values			
buying	v-high	high	med	low
maint	v-high	high	med	low
doors	2	3	4	5more
persons	2	4	more	
trunk	small	med	big	
safety	low	med	high	

TABLE II
PART OF THE TRAINING SAMPLES FOR CAR EVALUATION

buying	maint	doors	persons	trunk	safety	class
vhigh	vhigh	2	2	small	low	unacc
vhigh	vhigh	2	4	small	med	unacc
high	high	2	4	small	med	unacc
high	high	2	4	small	high	acc
med	high	2	4	med	low	unacc
med	med	3	4	big	high	vgood
low	low	5more	more	big	low	unacc
low	low	5more	more	big	med	good

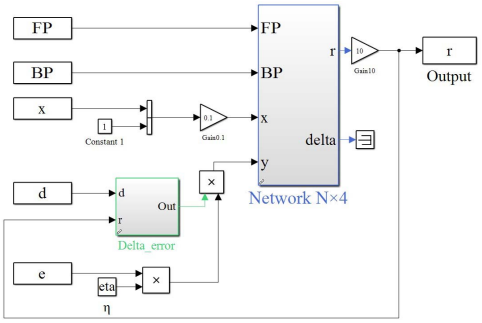


Fig. 11. Simulation model built in Simulink.

There are six attributes introduced to evaluate cars, such as purchase price (buying), maintenance costs (maint), number of gates (doors), number of seats (persons), luggage boot (trunk), and safety performance (safety). Their values are shown in Table I.

The results of vehicle evaluation are divided into four categories: 1) unacceptable (unacc); 2) acceptable (acc); 3) good; and 4) very good (vgood). Part of the training samples for car evaluation are shown in Table II.

In order to simplify the simulation work, first, the training data are processed by the reservoir computing in MATLAB. The input signals of the reservoir layer are a group of 6-dimensional (6-D) vectors. Let the number of nodes in the reservoir layer be n and sample size be s. After the reservoir computing, 6-D vectors are mapped to n-dimensional vectors, an $n * s$ matrix will be obtained. The matrix is then used as inputs of the circuit based on memristor to train the weights. The LMS circuit is established in Simulink as shown in Fig. 11.

As car evaluation is divided into four categories, the synaptic grid circuit model should be the size of $n * 4$ as shown in Fig. 12. The output value of the circuit is processed with the principle of winner-takes-all. From car evaluation dataset, 200 samples are randomly selected as training samples and 100

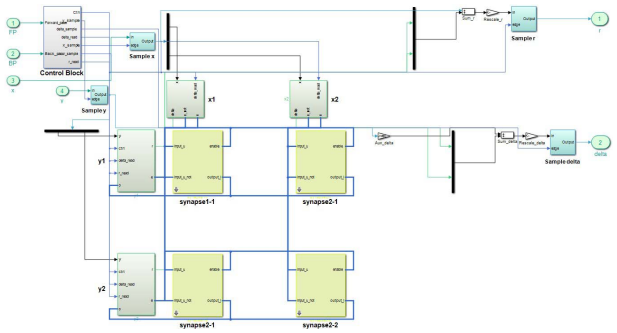


Fig. 12. 2×2 synaptic grid circuit built in Simulink.

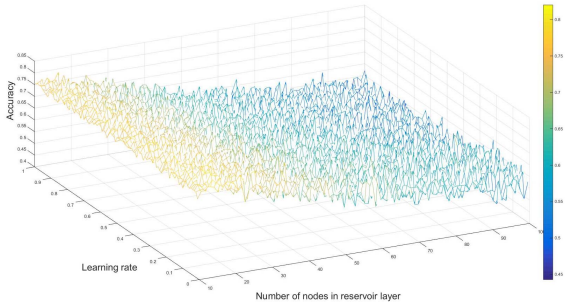


Fig. 13. Testing accuracy of each simulation.

samples are taken as test samples. The mean square value and the prediction accuracy are calculated in the end.

Most parameters in the classical reservoir layer network are randomly generated. However, it is widely acknowledged that the best solution is to design a network related to the specific application. Among these parameters, the number of nodes in reservoir layer and the connection degree are significant in the performance of ESN network.

In practice, the simulation using Simulink is not same as the real circuit with the characteristics of parallel computing for the existence of the algebraic loop of Simulink, thus it usually takes several hours to finish the circuit simulation each time. In order to analyze the proposed scheme, the simulation of ESN with online LMS was tested with MATLAB at first to find the best reservoir sizes and connectivity. In the course of testing, the number of nodes in the reservoir layer ranges from 10 to 100, the degrees of connectivity range from 0.01 to 1. The accuracy of each simulation is shown in Fig. 13. According to the simulation results, it shows that the optimal number of nodes in the reservoir layer is about 15 nodes. Therefore, the bigger of the number of nodes does not mean the better performance. Instead, an appropriate number of nodes is needed. The optimal connection degree is about 0.1, which also varies for different cases.

At the same time, in order to verify the convergence of the algorithm, the value of the weight is recorded with the number of iterations. If the scheme is convergent, the training error decreases with the number of iterations. According to (10), the change of the weight is close to 0 as well. Five weights were randomly selected, after about 1000 iterations. The weight converges in the vicinity of -0.5, -0.2, and 0.4

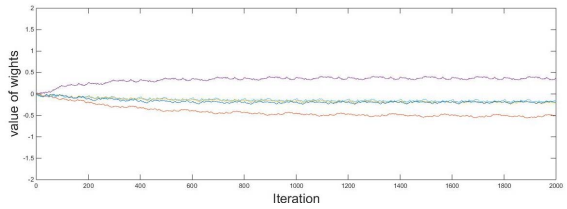


Fig. 14. Value of five weights.

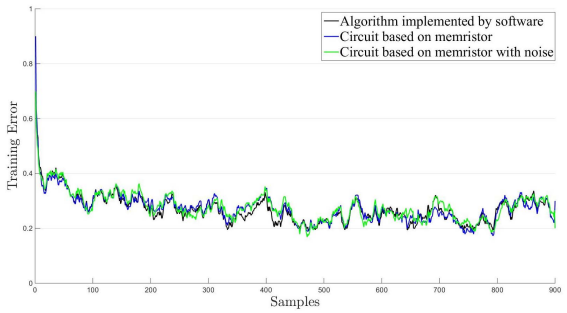


Fig. 15. Test errors of the proposed circuit, the circuit with noise and the algorithm.

as shown in Fig. 14. From simulation results, we can conclude that the proposed scheme is convergent.

The optimal number of nodes and connection degree in reserve pool are used in the process of circuit simulation. Compared with the software implementation, the fabricating technology of memristive circuits is not mature. Thus, the hardware implementation process is prone to interference of noise. Therefore, the proposed scheme is supposed to be robust with 10% noise in the circuit simulation. As shown in Fig. 15, the final training errors of different simulations are basically similar, which indicates that the circuit design can precisely implement ESN with online LMS. On the other hand, it is shown that the noise effect is relatively small for the system performance. Thus, the robustness of the proposed scheme is confirmed due to the robustness of gradient descent in the LMS algorithm.

B. Application to Short-Term Power Load Forecasting

The load forecasting of power system has become one of the hotspots in power systems [56]. Historical power load data is a typical time series. The ESN network is very suitable for fitting the time series with noise, strong randomness and non-stationary. Jaeger applied ESN to Mackey–Glass time series with great performances. Therefore, the proposed MESN is applied to the short-term power load forecasting. The historical power load data comes from a global online forecast competition held in August 1, 2001 by EUNITE Network. The organizers exposed every 30 min load data in 1997 and 1998 in eastern Slovakia, as well as daily temperature, holiday type, and other data. In this paper, the historical power load data of 1997 is used as the training set, and the historical power load data of 1998 is used as the test set. Some historical power load data are displayed as shown in Fig. 16. Meanwhile, for MESN, comparison is made between the BP algorithm and LMS algorithm. The test results are shown in Table III.

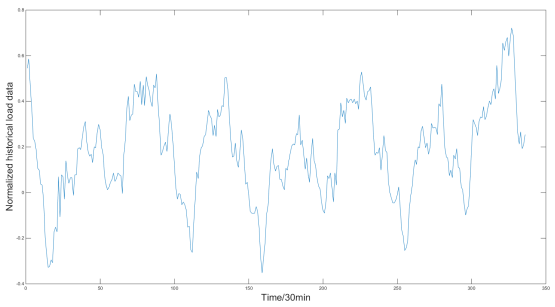


Fig. 16. Historical power load data of a week.

TABLE III
TEST RESULTS OF SHORT-TERM POWER LOAD FORECASTING

mdoel	reservoir nodes number	mse	time/s	number of weights
ESN-LMS	**10**	$\mathbf{5.77x10^{-3}}$	**4.75**	**10**
	100	$7.60x10^{-3}$	7.62	100
ESN-BP	10	$5.51x10^{-3}$	428.96	110
	100	$5.51x10^{-3}$	1111.48	2020

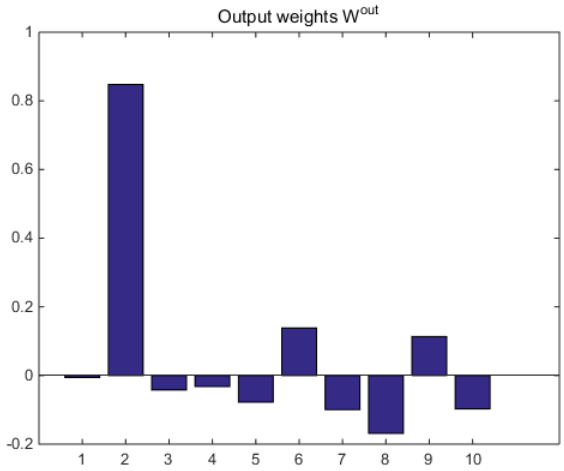


Fig. 17. Output weights W_{out}.

According to the test results, when using the LMS or BP algorithm to train the output weight matrix, we need not to use thousands of reservoir nodes as in the classical ESN network. A desired result can be obtained with only ten reservoir nodes. In addition, though the precision of ESN-BP may be higher, but there are more weights we need train. ESN-BP requires more memristor synaptic circuits with more areas and powers, as well as greatly increased training times. Moreover, the accuracy of ESN-LMS is not much lower than that of the ESN-BP network. Thus, the overall performance of ESN-LMS is better than that of ESN-BP in memristive hardware designs.

After more detailed debugging, the number of the pool layer nodes was selected as 8, the number of samples used to initialize the reservoir pool layer was 500. The weight distribution is shown for the output matrix after training in Fig. 17. The obtained MSE is 5.7×10^{-3} for the memristive circuit simulation. The partial prediction results are shown in Fig. 18, which demonstrates that the proposed scheme completes the prediction task well.

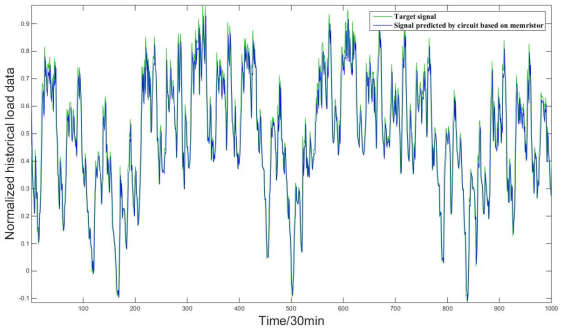


Fig. 18. Comparison between predicted value and target value.

V. Conclusion

This paper proposed a computational architecture of MESN with online LMS algorithm. The topological structure adopted NW small-world network. The training process was implemented by the memristive synaptic circuit. In the proposed MESN network, the state matrix of the reservoir layer, which is obtained by raising the dimension of input data, was utilized as an input of the LMS algorithm to train the output weight matrix on chip. After certain iterations, the resistance value of memristor was adjusted to a constant. Thus, the final weight output matrix was obtained. The proposed scheme successfully performed car evaluation and short-term power load forecasting. We also studied the effect of node number and connectivity degree in reservoir. The research provided a new idea for the hardware implementation of neural networks. Future work will focus on improving the accuracy of the model and the hardware implementation of the memristive reservoir layer to achieve a fully analog MESN network.

References

[1] Y. Xu, Z. Wang, D. Y. Yao, R. Q. Lu, and C.-Y. Su, "State estimation for periodic neural networks with uncertain weight matrices and Markovian jump channel states," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published.

[2] X. P. Xie, D. Yue, and C. Peng, "Multi-instant observer design of discrete-time fuzzy systems: A ranking-based switching approach," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 5, pp. 1281–1292, Oct. 2017.

[3] D. O. Hebb, *The Organization of Behavior: A Neuropsychological Theory*. New York, NY, USA: Wiley, 1949, pp. 100–136.

[4] Y. Xu, C. Liu, J.-Y. Li, C.-Y. Su, and T. W. Huang, "Finite-Horizon H∞ state estimation for time-varying neural networks with periodic inner coupling and measurements scheduling," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published.

[5] Y. Xu, C. Liu, R. Q. Lu, and C.-Y. Su, "Remote estimator design for time-delay neural networks using communication state information," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published.

[6] R. Brette and W. Gerstner, "A logical calculus of the ideas immanent in nervous activity," in *Neurocomputing: Foundations of Research*. Cambridge, MA, USA: MIT Press, 1988, pp. 115–133.

[7] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biol. Cybern.*, vol. 43, no. 1, pp. 59–69, 1982.

[8] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[9] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, no. 8, pp. 2554–2558, 1982.

[10] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Comput. Sci. Rev.*, vol. 3, no. 3, pp. 127–149, 2009.

[11] F. Zafari, G. M. Khan, and M. Rehman, "Evolving recurrent neural network using cartesian genetic programming to predict the trend in foreign currency exchange rates," *Appl. Artif. Intell.*, vol. 28, no. 6, pp. 597–628, 2014.

[12] T. Wang, H. Gao, and J. Qiu, "A combined adaptive neural network and nonlinear model predictive control for multirate networked industrial process control," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 2, pp. 416–425, Feb. 2016.

[13] H. Zhao, X. Zeng, and Z. He, "Low-complexity nonlinear adaptive filter based on a pipelined bilinear recurrent neural network," *IEEE Trans. Neural Netw.*, vol. 22, no. 9, pp. 1494–1507, Sep. 2011.

[14] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," *Comput. Sci.*, vol. 2014, pp. 338–342, 2014.

[15] P. Liu, Z. Zeng, and J. Wang, "Multistability of recurrent neural networks with nonmonotonic activation functions and mixed time delays," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 4, pp. 512–523, Apr. 2016.

[16] P. Li, Y. Li, and Q. Xiong, "International journal of electrical power with energy systems," in *Proc. IEEE Int. Symp. Ind. Electron.*, vol. 55, 2014, pp. 749–759.

[17] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks-with an erratum note," German Nat. Res. Center Inf. Technol., Bonn, Germany, GMD Rep. 148, p. 13, 2001.

[18] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural Comput.*, vol. 14, no. 11, pp. 2531–2560, 2002.

[19] L. Wang, Z. Wang, and S. Liu, "An effective multivariate time series classification approach using echo state network and adaptive differential evolution algorithm," *Expert Syst. Appl.*, vol. 43, no. 12, pp. 237–249, 2016.

[20] S. Scardapane, D. Wang, and M. Panella, "A decentralized training algorithm for echo state networks in distributed big data applications," *Neural Netw.*, vol. 78, pp. 65–74, Jun. 2016.

[21] S. Jung and S. S. Kim, "Hardware implementation of a real-time neural network controller with a DSP and an FPGA for nonlinear systems," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 265–271, Feb. 2007.

[22] C. Mead, *Analog VLSI and Neural Systems*. Boston, MA, USA: Springer, 2012.

[23] L. S. Smith and A. Hamilton, *Neuromorphic Systems: Engineering Silicon From Neurobiology*. Singapore: World Sci., 1998.

[24] A. Dundar, J. Jin, B. Martini, and E. Culurciello, "Embedded streaming deep neural networks accelerator with applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 7, pp. 1572–1583, Jul. 2017.

[25] A. R. Omondi and J. C. Rajapakse, *FPGA Implementations of Neural Networks*. New York, NY, USA: Springer, 2006.

[26] M. Laiho *et al.*, "FPAA/memristor hybrid computing infrastructure," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 3, pp. 906–915, Mar. 2015.

[27] F. Walter, F. Röhrbein, and A. Knoll, "Neuromorphic implementations of neurobiological learning algorithms for spiking neural networks," *Neural Netw.*, vol. 72, no. 12, pp. 152–167, 2015.

[28] J. Park, T. Yu, S. Joshi, C. Maier, and G. Cauwenberghs, "Hierarchical address event routing for reconfigurable large-scale neuromorphic systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2408–2422, Oct. 2017.

[29] L. O. Chua, "Memristor—The missing circuit element," *IEEE Trans. Circuit Theory*, vol. CT-18, no. 5, pp. 507–519, Sep. 1971.

[30] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, no. 7191, pp. 80–83, 2008.

[31] Z. Guo, J. Wang, and Z. Yan, "Global exponential synchronization of two memristor-based recurrent neural networks with time delays via static or dynamic coupling," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 2, pp. 235–249, Feb. 2015.

[32] M. Hu, Y. Chen, J. J. Yang, Y. Wang, and H. H. Li, "A compact memristor-based dynamic synapse for spiking neural networks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 8, pp. 1353–1366, Aug. 2017.

[33] P. Zhang, C. Li, T. Huang, L. Chen, and Y. Chen, "Forgetting memristor based neuromorphic system for pattern training and recognition," *Neurocomputing*, vol. 222, pp. 47–53, Jan. 2017.

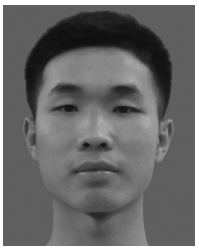
[34] X. Liu *et al.*, "Harmonica: A framework of heterogeneous computing systems with memristor-based neuromorphic computing accelerators," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 63, no. 5, pp. 617–628, May 2016.

[35] C. T. Yen, W.-D. Weng, and Y. T. Lin, "FPGA realization of a neural-network-based nonlinear channel equalizer," *IEEE Trans. Ind. Electron.*, vol. 52, no. 2, pp. 472–479, Apr. 2004.

[36] L. Larger *et al.*, "Photonic information processing beyond turing: An optoelectronic implementation of reservoir computing," *Opt. Express*, vol. 20, no. 3, pp. 3241–3249, 2012.
[37] F. Jiang, H. Berry, and M. Schoenauer, "Supervised and evolutionary learning of echo state networks," in *Parallel Problem Solving From Nature*. Heidelberg, Germany: Springer, 2008, pp. 215–224.
[38] D. Kudithipudi, Q. Saleh, and C. Merkel, "Design and analysis of a neuromemristive reservoir computing architecture for biosignal processing," *Front. Neurosci.*, vol. 9, p. 502, Feb. 2016.
[39] Q. Saleh, C. Merkel, D. Kudithipudi, and B. Wysocki, "Memristive computational architecture of an echo state network for real-time speech-emotion recognition," in *Proc. IEEE Symp. Comput. Intell. Security Defense Appl. (CISDA)*, Verona, NY, USA, 2015, pp. 1–5.
[40] C. Merekl, Q. Saleh, C. Donahue, and D. Kudithipudi, "Memristive reservoir computing architecture for epileptic seizure detection," in *Proc. Int. Conf. Biol. Inspired Cogn. Archit.*, Cambridge, MA, USA, 2014, pp. 249–254.
[41] M. S. Kulkarni and C. Teuscher, "Memristor-based reservoir computing," in *Proc. IEEE/ACM Int. Symp. Nanoscale Archit. (NANOARCH)*, Amsterdam, The Netherlands, 2012, pp. 226–232.
[42] X. Yang, W. Chen, and F. Z. Wang, "Investigations of the staircase memristor model and applications of memristor-based local connections," *Analog Integr. Circuits Signal Process.*, vol. 87, no. 2, pp. 263–273, 2016.
[43] A. M. Hassan, H. H. Li, and Y. Chen, "Hardware implementation of echo state networks using memristor double crossbar arrays," in *Proc. Int. Joint Conf. Neural Netw.*, Anchorage, AK, USA, 2017, pp. 2171–2177.
[44] Z. W. Shi and M. Han, "Ridge regression learning in ESN for chaotic time series prediction," *Control Decis.*, vol. 22, no. 3, pp. 258–261, 2007.
[45] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," *Networks*, vol. 8, no. 9, p. 17, 2003.
[46] B. Widrow and M. E. Hoff, "Adaptive switching circuits," *IRE WESCON Convention Record*, vol. 4, no. 1, pp. 96–104, 1960.
[47] A. Rodan and P. Tino, "Minimum complexity echo state network," *IEEE Trans. Neural Netw.*, vol. 22, no. 1, pp. 131–144, Jan. 2011.
[48] Q. Song and Z. Feng, "A new method to construct complex echo state networks," *J. Xi'an Jiaotong Univ.*, vol. 43, no. 4, pp. 1–4, 2009.
[49] Y. Xue, L. Yang, and S. Haykin, "Decoupled echo state networks with lateral inhibition," *Neural Netw.*, vol. 20, no. 3, pp. 365–376, 2007.
[50] Z. Deng and Y. Zhang, "Collective behavior of a small-world recurrent neural system with scale-free distribution," *IEEE Trans. Neural Netw.*, vol. 18, no. 5, pp. 1364–1375, Sep. 2007.
[51] L. Yang, X. Liang, T. Ma, and K. Liu, "Spectrum prediction based on echo state network and its improved form," in *Proc. 5th Int. Conf. Intell. Human Mach. Syst. Cybern. (IHMSC)*, vol. 1. Hangzhou, China, 2013, pp. 172–176.
[52] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
[53] M. E. J. Newman and D. J. Watts, "Renormalization group analysis of the small-world network model," *Phys. Lett. A*, vol. 263, no. 4, pp. 341–346, 1999.
[54] T. Song and H. Li, "Chaotic time series prediction based on wavelet echo state network," *Acta Physica Sinica*, vol. 61, no. 8, p. 963, 2012.
[55] D. Soudry, D. D. Castro, A. Gal, A. Kolodny, and S. Kvatinsky, "Memristor-based multilayer neural networks with online gradient descent training," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2408–2421, Oct. 2015.
[56] R. Hu, S. Wen, Z. Zeng, and T. Huang, "A short-term power load forecasting model based on the generalized regression neural network with decreasing step fruit fly optimization algorithm," *Neurocomputing*, vol. 221, pp. 24–31, Jan. 2017.

Shiping Wen (M'16) received the M.Eng. degree in control science and engineering from the School of Automation, Wuhan University of Technology, Wuhan, China, in 2010 and the Ph.D. degree in control science and engineering from the School of Automation, Huazhong University of Science and Technology, Wuhan, in 2013.

He is currently an Associate Professor with the School of Automation and also with the Key Laboratory of Image Processing and Intelligent Control of Education Ministry of China, Huazhong University of Science and Technology. His current research interests include memristor-based circuits & systems and machine learning.



Rui Hu (S'16) received the B.Eng. degree in Control Science and Engineering, from the School of Automation, Wuhan University of Technology, Wuhan, China, in 2016. He is currently pursuing the M.Eng. degree from the Huazhong University of Science and Technology, Wuhan.

His current research interests include memristor-based circuit, neural networks, data mining, and pattern recognition.

Yin Yang (M'16) received B. Eng. Degree in Computer Science, from Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China (SJTU) in 2004, and his Ph.D degree in Computer Science, Department of Computer Science and Engineering, Hong Kong University of Science and Technology (HKUST), in 2009.

Hi is currently an Assistant Professor with the College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar. He has published extensively in top venues on differentially private data publication and analysis, and on query authentication in outsourced databases. He is currently researching on cloud-based big-data analytics, with a focus on fast streaming data. His current research interests include cloud computing, database security and privacy, and query optimization.

Tingwen Huang (SM'14) received the B.S. degree in Mathematics from Southwest Normal University (currently Southwest University), Chongqing, China, in 1990, the M.S. degree from Sichuan University, Chengdu, China, in 1993, and the Ph.D. degree from Texas A&M University, College Station, TX, USA, in 2002.

He is a Professor of Mathematics with Texas A&M University at Qatar, Doha, Qatar. His current research interests include dynamical systems, neural networks, complex networks, optimization and control, and traveling wave phenomena.

Zhigang Zeng (SM'07) received the B.S. degree in Mathematics from Hubei Normal University, Huangshi, China, in 1993, the M.S. degree from Hubei University, Wuhan, China, in 1996, and the Ph.D. degree from the Huazhong University of Science and Technology, Wuhan, in 2003.

He is a Professor with the School of Automation and also with the Key Laboratory of Image Processing and Intelligent Control of Education Ministry of China, Huazhong University of Science and Technology. His current research interests include neural networks, switched systems, computational intelligence, stability analysis of dynamic systems, pattern recognition, and associative memories.

Yong-Duan Song (M'90–SM'12) received the Ph.D. degree in electrical and computer engineering from Tennessee Technological University, Cookeville, TN, USA, in 1992.

He held a tenured Full Professor with North Carolina A&T State University, Greensboro, NC, USA, from 1993 to 2008 and a Langley Distinguished Professor with the National Institute of Aerospace, Hampton, VA, USA, from 2005 to 2008. He is currently the Dean of the School of Automation, Chongqing University, Chongqing, China, where he is also the Founding Director of the Institute of Smart Engineering. His current research interests include intelligent systems and bioinspired adaptive and cooperative systems.

Dr. Song was a recipient of several competitive research awards from the National Science Foundation, the National Aeronautics and Space Administration, the U.S. Air Force Office, the U.S. Army Research Office, and the U.S. Naval Research Office. He has served as an Associate Editor/Guest Editor for several prestigious scientific journals.