# Neuromorphic Extreme Learning Machines with Bimodal Memristive Synapses

Zhekang Dong [a,b], Chun Sing Lai [c,d], Zhaowei Zhang [a], Donglian Qi [b*], Mingyu Gao [a], Shukai Duan [e]

[a] *College of Electronic Information, Hangzhou Dianzi University, Hangzhou 310018, China*
[b] *College of Electrical Engineering, Zhejiang University, Hangzhou 310027, China*
[c] *Department of Electronic and Computer Engineering, Brunel University, London, UB8 3PH, United Kingdom*
[d] *Department of Electrical Engineering, School of Automation, Guangdong University of Technology, Guangzhou 510006, China*
[e] *School of Artificial Intelligence, Southwest University, Chongqing 310027, China*

**Abstract**
The biology-inspired intelligent computing system for the neuromorphic hardware implementation is useful in high-speed parallel information processing. However, the traditional Von Neumann computer architecture and the unsatisfactory signal transmission approach have jointly limited the overall performance of the specific hardware implementation. In this paper, a compact extreme learning machine (ELM) architecture synthesized with the spintronic memristor-based synaptic circuit, the biasing circuit, and the activation function circuit is presented with a hardware-friendly learning method. Notably, due to the threshold characteristic of the memristive device, the synaptic circuit has a bimodal behavior. Namely, it is capable to provide the constant and adjustable network weights between the adjacent layers in the ELM. Furthermore, two major limitations (process variations and sneak path issue) are taken into account for the detailed robustness analysis of the whole network. Finally, the entire scheme is verified with case studies in single image super-resolution (SR) reconstruction.

*Keywords:* Bimodal behavior, extreme learning machine, image super-resolution, neuromorphic hardware implementation

## 1. Introduction

Over the past decades, the traditional Von Neumann computer architecture has encountered a bottleneck in efficiency limitation. The demand for designing the neotype intelligent computing system/unit is increasingly urgent and challenging [1]. As a remedy for this issue, the hardware implementation of neuromorphic systems has been an active research field. Recent work [2-4] has proven that the neuromorphic hardware systems can provide the capabilities of high-speed parallel computation, real-time information processing, and biological perception, especially for some industrial applications such as the pattern recognition [4].

Actually, an artificial neural network (ANN) can be deemed as a simplified abstract of the biological system that is capable of solving a series of problems in prediction, optimization, and recognition [5]. From an application point of view in computational intelligence [6], the success of a neural network hardware implementation depends on a tradeoff among the accuracy, processing speed, and occupied area. Based on this, an efficient approach of hardware-level assistance is necessary.

The discovery of the fourth circuit element, memristor, has opened up a new path in the neural network hardware field [7]. This nanoscale element has several superior properties including non-volatile multi-level memory capability, ultra-low power consumption, nanoscale geometry, and good compatibility with the complementary metal-oxide-semiconductor (CMOS) technology, which makes it a potential candidate for realizing the large-scale intelligent neuromorphic systems [8, 9]. Notably, due to the fact that the memristor behaves similarly to the biological synapse, a lot of efforts have been made to develop memristor-based hardware synapses [10-14]. In [10], a memristor bridge synapse consisting of four memristors and a CMOS differential pair is proposed, and the literature [11] and [12] respectively illustrate the corresponding multilayer neural network and the learning method employing the synaptic circuit in [10]. The work in [13] presents a neuromorphic character recognition system using Al/$Pr_{0.7}Ca_{0.3}MnO_3$ (PCMO) memristor-based synapses. Similarly, a neuromorphic hardware system for visual pattern recognition is designed by PCMO memristor array and CMOS neurons [14]. In particular, on account of the specific input/output (I/O) state (usually, voltage is served as the input and current is served as the output) of the above memristive synaptic circuits, the corresponding neurons have to convert their outputs from a current into a voltage, which leads to low efficiency and additional cost. Moreover, for these existing synaptic circuits, it is difficult to achieve the constant weights under the cases of voltage inputs and current outputs. In fact, the constant weights are extremely important in many neural networks [15, 16], for example the extreme learning machines (ELMs) [16] which is also the main research area in this paper.

Corresponding email: qidl@zju.edu.cn (D. Qi).

The ELM originally presented by Huang *et al*. is a kind of feedforward neural network with fixed random weights between the input and hidden layer [16-18]. Since only the output weights (linking the hidden layer to the output layer) need to be trained through the simple generalized inverse operation, the learning speed of ELM is much faster than that of traditional learning methods, while tending to have the better generalization performance [17, 18]. Meanwhile, the availability of compact fast circuitry for the support of ELM architecture is a long-standing and critical requirement for many important applications, especially where frequent and fast training, or even real-time training is required (e.g., classification and regression).

For these reasons, a variety of hardware implementations of ELM are investigated in recent years [19-28]. Specifically, Sergio Decherchi *et al*. introduced a novel learning procedure for the ELM model, which can properly trade off classifier complexity and generalization ability. Based on this, two alternative field programmable gate array (FPGA) based ELM hardware implementation strategies were proposed [19]. Similarly, other different FPGA implementations of ELM system were also proposed, which offers a guideline on required resources and level of performance that an FPGA based ELM training can provide [20, 21]. Considering the mature CMOS technology, [22] proposed a hardware implementation of ELM using spiking neural circuits which is composed of silicon neurons, synapse circuits, and address event representation (AER) circuits. Then, [23] presented a low-power and compact hardware implementation of random feature extractor (RFE) core, and further applied this RFE core as the first stage of ELM for handwritten digit recognition. Also, a simplified artificial intelligence accelerator is proposed for realizing the ELM based inference engine. Then, [24] described a compact low-power and high-performance hardware implementation of ELM for machine learning applications, where both regression and classification are demonstrated and a design space tradeoff between speed, power, and accuracy is explored. Recently, researchers have invented memristor device with high density and low power to realize the hardware implementation of ELM. In [25], a current-mode ELM architecture comprised of CMOS current-mode neuron circuits, memristor-based synapse circuits, and a hardware-friendly training method is proposed. Similarly, [26] proposed a novel ELM with the memristor based activation function, instead of the sigmoid function. Compared with the regular ELMs, this kind of ELM is able to shorten time and improve accuracy in the classification tasks.

Although all these above-mentioned approaches have constituted new design paradigms for the hardware implementation of ELM, they still suffer from several limitations. For FPGA based ELM, a primary drawback may be the limitation of FPGA resources (including the block/distributed RAM, DSP block, etc.), especially for VLSI implementation of ELM. The majority of CMOS based ELM implementations suffer from low density, long runtime, and high power consumption. For existing memristor based ELMs, there is no straight-forward way to implement weights with voltage inputs and current outputs. In addition, the customized ELM chip has been fabricated [27], and it has been proved effective in texture recognition for analyzing spiking activity [28]. However, the existing ELM chips are commonly expensive and custom-made for particular applications.

In this paper, a novel memristor-based ELM (M-ELM) hardware implementation with a proper training method is presented. Notably, the memristor model applied in the entire scheme is the spintronic memristor [29, 30], and all subsequent experiments are carried out with MATLAB and PSpice software platforms. In summary, the main contributions can be concluded as below:

1) A novel bimodal memristive synapse circuit is designed to achieve two kinds of weights, i.e., the constant weight and the adjustable weight.

2) The specific M-ELM hardware circuit construction is presented.

3) Two major limitations (process variations and sneak path issue) are taken into account for the robustness analysis of the presented M-ELM, which offers huge benefits in terms of network stability and fault tolerance.

4) The proposed M-ELM is further applied to the image super-resolution (SR) reconstruction, which is in favor of facilitating the hardware implementation of image reconstruction.

The remainder of the paper is organized as follows. In Section 2, the spintronic memristor model with its basic properties is briefly reviewed. Section 3 provides the specific implementation of the M-ELM architecture and describes a hardware-friendly training method. For the verification purpose, the presented M-ELM is applied to the image SR reconstruction in Section 4. Finally, Section 5 concludes the entire work.

## 2. Spintronic memristor

Among all the spintronic memristor physical structures proposed in [29] and [30], the model based on magnetic-domain-wall motion is most widely used for its simplicity and reliability. The specific three-dimensional (3D) structure and its equivalent circuit model are exhibited in Fig. 1, respectively.

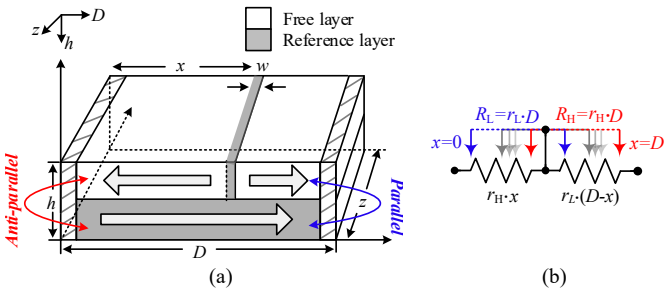



Fig. 1: Spintronic memristor based on magnetic-domain-wall motion. (a) The 3D structure. (b) The equivalent circuit.

From Fig. 1(a), the spintronic memristor can be deemed as a long spin-value strip with the size of $D$ (length) × $z$ (width) × $h$ (thickness). It includes two up-down ferromagnetic layers, i.e., the reference layer and the free layer. The former one (reference layer) is an integral whole with fixed magnetization direction; and the latter one (free layer) is further divided by a mobile domain wall (width: $w$) into two segments with totally opposite magnetization directions. The resistance per unit length of each segment depends completely on the relative magnetic directions of the free layer and the reference layer [29]. The mathematical expression of the resistance of a spintronic memristor can be written as [29]:

$$M(x) = r_{\mathrm{H}} \cdot x + r_{\mathrm{L}} \cdot (D - x) \tag{1}$$

where $r_{\mathrm{H}}$ and $r_{\mathrm{L}}$ are the highest and lowest resistance per unit length respectively. Correspondingly, the limiting resistances of the spintronic memristor can be calculated as $R_{\mathrm{H}}=r_{\mathrm{H}}\cdot D$ and $R_{\mathrm{L}}=r_{\mathrm{L}}\cdot D$. $x$ denotes the position of the domain wall, and its dynamic function is given by [29]:

$$\frac{dx}{dt} = \begin{cases} \Gamma v \cdot J, & J \geq J_{\mathrm{cr}} \\ 0, & J < J_{\mathrm{cr}} \end{cases} \tag{2}$$

where $\Gamma v$ is the domain wall velocity coefficient, $J$ and $J_{\mathrm{cr}}$ are the real-time and critical current density respectively.

After the differential operation, Eq. (1) leads to:

$$\frac{dM}{dt} = \Delta r \cdot \frac{dx}{dt} = \begin{cases} \Delta r \cdot \Gamma v \cdot J, & J \geq J_{\mathrm{cr}} \\ 0, & J < J_{\mathrm{cr}} \end{cases} \tag{3}$$

where $\Delta r$ denotes the difference of $r_{\mathrm{H}}$ and $r_{\mathrm{L}}$.

Actually, Eq. (3) clearly shows the threshold property of the spintronic memristor. Specifically, when $J<J_{\mathrm{cr}}$, the memristance variation is equal to zero, the spintronic memristor behaves like an ordinary resistor. Contrarily, when $J\geq J_{\mathrm{cr}}$, the memristance changes with time $t$, and Eq. (3) can be rewritten as:

$$\frac{dM}{dt} = \Delta r \cdot \Gamma v \cdot J = \frac{\Delta r \cdot \Gamma v}{h \cdot z} \cdot \frac{V}{M} \tag{4}$$

where $V$ is the voltage applied to the spintronic memristor.

Integrating both sides of Eq. (4), the flux-controlled spintronic memristor can be obtained by:

$$M(\varphi) = \begin{cases} R_{\mathrm{H}}, & \varphi > \varphi_{\mathrm{th2}} \\ \sqrt{M_0^{\,2} + 2A\varphi}, & \varphi_{\mathrm{th1}} \leq \varphi \leq \varphi_{\mathrm{th2}} \\ R_{\mathrm{L}}, & \varphi < \varphi_{\mathrm{th1}} \end{cases}, \; where \begin{cases} \varphi_{\mathrm{th1}} = \dfrac{R_{\mathrm{L}}^{\,2} - M_0^{\,2}}{2A} \\ \varphi_{\mathrm{th2}} = \dfrac{R_{\mathrm{H}}^{\,2} - M_0^{\,2}}{2A} \end{cases} \tag{5}$$

where $\varphi$ is the magnetic flux flowing through the spintronic memristor. $\varphi_{\mathrm{th1}}$ and $\varphi_{\mathrm{th2}}$ are the relevant thresholds determined by the limiting memristances, initial memristance $M_0$, as well as the auxiliary variable $A$ ($A=\Delta r\cdot\Gamma v/h/z$).

Next, the corresponding spintronic memristor SPICE model with the specific sub-circuit description is provided in **Table 1**, which is beneficial to facilitate the circuit simulation in the subsequent sections.

**Table 1**: Sub-circuit description of spintronic memristor SPICE model

```
* Spintronic memristor model
.SUBCKT Spintronic memristor Plus Minus Flux Charge PARAMS:
+D=1000E-9 h=70E-10 z=10E-9 rl=3E9 rh=6E9 Jcr=5E11
Taov=1.3517E-11
******* Differential equation modeling*******
Gx 0 x value={if(abs(I(Emem)/(h*z))<Jcr,0,Taov*I(Emem)/(h*z))}
Cx x 0 1 IC={0}
Raux x 0 1T
******* Resistive port of the memristor*******
Emem plus aux value={if(V(x)<=D,I(Emem)*V(x)*(rh-rl),
I(Emem)*D*(rh-rl))}
Roff aux minus {rl*D}
*******Flux computation *******
Eflux flux 0 value={SDT(V(plus, minus))}
*******Charge computation *******
Echarge charge 0 value={SDT(I(Emem))}
.ENDS spintronic memristor
```

Furthermore, the spintronic memristance variation rule under different voltage pulses is investigated through a series of computer simulations, which is important for the implementation of different synapses in ELM. Concretely, five different voltage pulses ($V_j$, $j$=[1,2,3,4,5]) within the time interval of [0, 200ns] are applied to the spintronic memristor respectively, the specific memristance variation rule can be clearly illustrated by the simulation results exhibited in Fig. 2. Notably, the necessary parameter setting is provided in **Table 2**.

**Table 2**: Collection of specific technical parameters

| Parameters | Physical meaning | Values |
|---|---|---|
| $r_L$ | Low resistance per unit length (Ω/m) | 4e9 |
| $r_H$ | High resistance per unit length (Ω/m) | 6e9 |
| $D$ | Length (nm) | 1000 |
| $h$ | Thickness (Å) | 70 |
| $z$ | Width (nm) | 10 |
| $J_{cr}$ | Critical current density (A/cm$^2$) | 5e7 |
| $\Gamma v$ | Domain wall velocity coefficient | 1.3517e-11 |
| $[M_{01}, M_{02}]$ | Initial memristance (kΩ) | [6, 4] |
| $V_i, i$=[1, 2] | Applied voltage amplitude (V) | [-0.1,-0.3] |
| $V_j, j$=[3,4,5] | Applied voltage amplitude (V) | [0.05,0.15,0.25] |
| $\varphi_0$ | Initial magnetic flux (Wb) | 0 |

**Note:** we define $I_{cr}=J_{cr}\cdot h\cdot z$ as the critical current.

In Fig. 2(a), when a negative voltage is applied to the spintronic memristor (initial memristance: $M_{01}$), the memristance variation can be divided into two cases:

*Case a*: If $V>-I_{cr}\cdot M_{01}$ ($V_1$, for example), the real-time current density $J$ is smaller than the critical current density $J_{cr}$, and the memristance remains in the initial state, i.e., $M=M_{01}$.

*Case b*: If $V\leq -I_{cr}\cdot M_{01}$ (like $V_2$), the real-time current density always satisfies $J\geq J_{cr}$, the memristance directly decreases to its lowest value ($r_L\cdot D$) within a very short time.

Similarly, when the input voltage is a positive one as shown in Fig. 2(b), the initial memristance is set to $M_{02}$=4kΩ. The memristance variation can be summarized in three cases:

*Case a:* If $V<I_{cr}\cdot M_{02}$ (such as $V_3$), the real-time current density $J<J_{cr}$, the final memristance is equal to the initial memristance, namely $M=M_{02}$.

*Case b:* If $I_{cr}\cdot M_{02}\leq V< I_{cr}\cdot R_H$ ($V_4$, for instance), the initial current density is larger than the critical current density. The memristance increases sharply while the real-time current density changes in the opposite direction. Until the applied current density reduces to the critical value $J_{cr}$, the memristance tends to be steady.

*Case c:* If $V\geq I_{cr}\cdot R_H$ (like $V_5$), the real-time current density always satisfies $J\geq J_{cr}$, and the memristance goes up to its highest value ($r_H\cdot D$) rapidly.

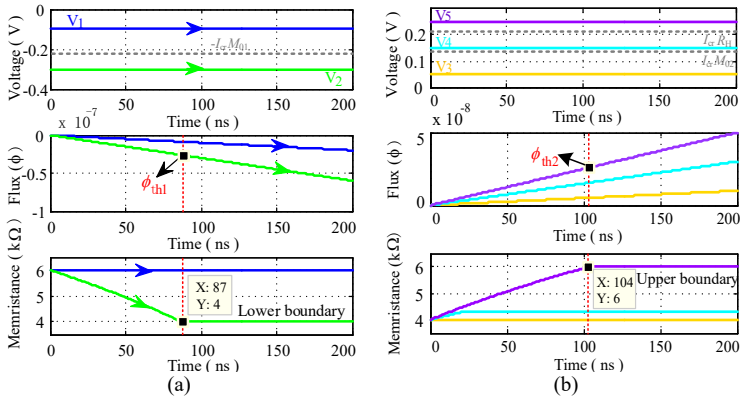



Fig. 2. Memristance variation rule of the spintronic memristor. (a) The memristance variation under negative voltages. (b) The memristance variation under positive voltages. $\varphi_{th1}$ and $\varphi_{th2}$ are the threshold fluxes, which can be calculated by Eq. (5).

## 3. Hardware circuit design and robustness analysis

According to the literature [29, 30], the spintronic memristor has demonstrated its superiorities in terms of nanoscale geometry, nonvolatile memory capability, as well as the fast resistance transformation (nano-second level). These unique properties make it a promising candidate for the hardware implementation of the ELM. In this section, a novel ELM circuit design scheme with the relevant training method is presented. Then, some limitations probably affecting the overall performance of the entire ELM system will be discussed in detail.

### *3.1 Hardware circuit design of the ELM*

Theoretically, the classical ELM architecture is a kind of single-hidden layer feedforward neural network (SLFNN) with two types of synaptic weights, i.e., the constant and adjustable synaptic weights [16, 17]. As shown in Fig. 3(a), assuming the number of nodes in the three layers (namely, the input layer, the hidden layer and the output layer) is $N$, $L$, and $N'$ respectively. The corresponding output of the ELM architecture with the activation function $g(\cdot)$ can be mathematically expressed by:

$$f_{outj}=\sum_{i=1}^{L}\beta_i g\left(a_i\cdot x_j+b_i\right),\ j=1,\ldots,N \tag{6}$$

where $\beta_i=[\beta_{i1}, \beta_{i2},\ldots, \beta_{iL}]^T$ denotes the weight vector between the $i^{th}$ hidden node and the output nodes, $a_i=[a_{i1},a_{i2},\ldots a_{iN}]^T$ is the weight vector connecting the $i^{th}$ hidden node and the input nodes, and $b_i$ is the additional bias (threshold) of the $i^{th}$ hidden node. Commonly, the output of the $i^{th}$ hidden node is indicated as $h_i(x)$ which can be calculated by $h_i(x)= g_i(a_i \cdot x_j+b_i)$. Note that, the weight vector $a_i$ and the hidden layer bias $b_i$ are not necessarily tuned once these two parameters are assigned randomly at the beginning of the learning procedure, which results in the hidden layer output $h_i(x)$ being invariable.

For clarity and demonstration purposes, the single-hidden layer of the ELM architecture in Fig. 3(a) is further divided into three compositions which are used to realize the sum operation (the purple circle), biasing (the green circle) and the activation function (the light blue circle) respectively. The remaining two regions labeled by red dotted lines denote the bimodal synapses (i.e., the constant synapse and the adjustable synapse) connecting the input (output) layer with the hidden layer in ELM. Correspondingly, the specific hardware implementation of the ELM can also be separated into three components, i.e., the bimodal synaptic circuit, the biasing circuit, and the activation function circuit, as shown in Fig. 3(b). The relevant circuit analysis with mathematical description is provided below:

*1) Bimodal synaptic circuit*

The hybrid memristor/resistor crossbar array and the additional amplifier module (Amp) jointly constitute the bimodal synaptic circuit, which is utilized to mimic the synapses connecting the input (output) layer and the hidden layer. Here, the former one (crossbar array configuration) is mainly responsible for the matrix-vector multiplication in ELM, and the later one (Amp) in the bottom central inset is used to realize the sum operation.

Assuming $R_{a1}=R_{a2}=1\Omega$, the output of the synaptic circuit $V_{O\lambda}$ can be obtained with the Kirchhoff's voltage law as follows:

$$V_{O\lambda i}=\sum_{j=1}^{row}\left(\frac{R_{a3}}{M_{\lambda ij}}-\frac{R_{a3}}{R_{\lambda ij}}\right)\cdot V_{I\lambda j},\ i=[1,2,\ldots,col],\ \lambda=[1,2] \tag{7}$$

where $V_{I\lambda}$ denotes the input voltage of the synaptic circuit. $\lambda$=1 or 2 represents the specific type of the synapse. Specifically, when $\lambda$=1, the synaptic circuit is actually used to model the constant synapse; while $\lambda$=2, the synaptic circuit is able to realize the

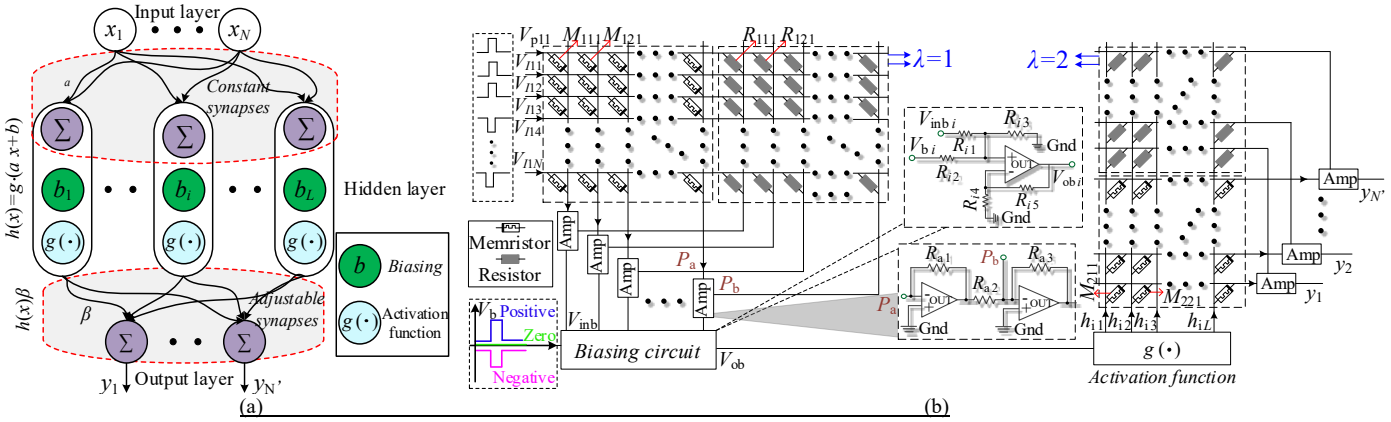



Fig. 3: Complete schematic of the ELM. (a) Classical ELM architecture. (b) Conceptual diagram of the ELM circuit

adjustable synapse. Parameters *row* and *col* denote the numbers of the inputs and outputs of the synaptic circuit respectively. $M_{\lambda ij}$ ($R_{\lambda ij}$) is the value of the memristors (resistors) installed at the crossbar array intersection. Especially, when $R_{a3}$ and $R_{\lambda ij}$ are uniformly set to $2R_L$ ($R_L$ is the lowest memristance of the spintronic memristor), Eq. (7) can be rewritten as:

$$V_{O\lambda i}=\sum_{j=1}^{r}\left(\frac{2R_L}{M_{\lambda ij}}-1\right)\cdot V_{I\lambda j} \tag{8}$$

Then, the synaptic weight can be expressed as:

$$\psi_{\lambda ij}=\frac{2R_L}{M_{\lambda ij}}-1 \tag{9}$$

which is theoretically within the scope of [-1, 1].

Notably, on account of the natural properties of the ELM architecture and in spite of having the same mathematical expression, the weights $\psi_{1ij}$ (input layer→hidden layer) are completely different from the weights $\psi_{2ij}$ (hidden layer→output layer). Specifically, the weights between the input layer and the hidden layer $\psi_{1ij}$ are referred to as the constant weights, whose values totally depend on the initial memristances. This type of weights is not required to be updated and can be easily implemented by controlling the input voltage within an appropriate range. On the contrary, the other type of weights $\psi_{2ij}$ connecting the hidden and output layer has two separate states, as follows:

• During the weighting (multiplication) process, the weights $\psi_{2ij}$ are invariable just like the constant weights $\psi_{1ij}$;

• During the learning (programming) process, the weights $\psi_{2ij}$ are adjustable and the specific learning methodology will be discussed in the next part.

Based on these processes, the presented synaptic circuit has the unique bimodal behavior, which means that it possesses the ability to simulate two kinds of synapses which are capable of achieving constant weights and adjustable weights respectively. The specific comparison information of these two kinds of synapses (referring to the circuit structure, location, input/output mode, weight size, and weight variation) is summarized in Table 3.

**Table 3**: Comparison of the two kinds of synapses in ELM

| Types | Constant synapse | Adjustable synapse | |
|---|---|---|---|
| Circuit structure | Same | Same | |
| Location | Input layer→ Hidden layer | Hidden layer → Output layer | |
| Input /Output | Voltage-mode | Voltage-mode | |
| Weight size | $L\times N$ | $N'\times L$ | |
| Weight variation | Random and constant | Weighting | Constant |
| | | Programming | Adjustable |

Additionally, compared to the work in [9], the proposed hybrid memristor/resistor crossbar array effectively reduces the amount of the memristors by 50%, and the presented Amp also requires fewer amplifiers and resistors. The great reduction in number of the circuit components is in favor of energy conservation and reliability increment.

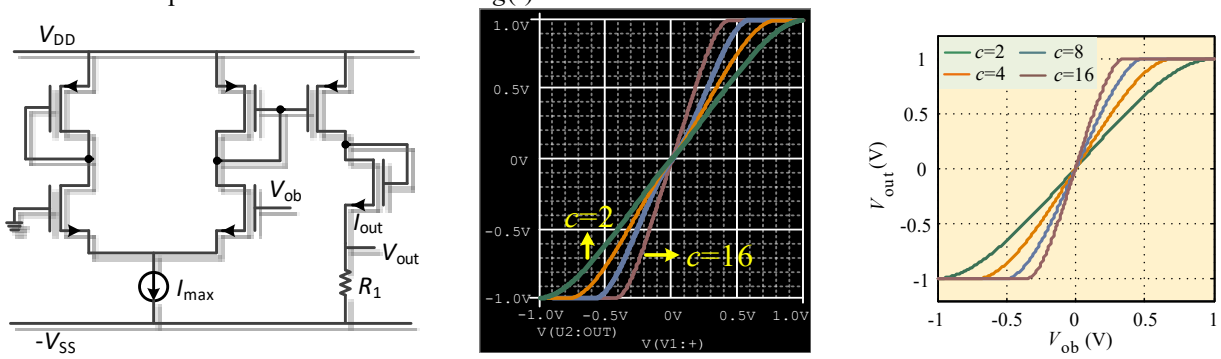
*2) Biasing circuit*

The top central inset in Fig. 3(b) is the biasing circuit, where $V_{bi}$, $i$=[1, 2, …, $L$] represents the bias of the $i^{th}$ hidden node. Notably, the bias $V_{bi}$ can be randomly set to a positive, zero, or negative value and remains unchanged thereafter. The other input $V_{inbi}$, $i$=[1,2,…,$L$] generated by the front synaptic circuit is the inner product of $x_j$,$j$=[1, 2, …, $N$] and the relevant weight vector $a_i$, $i$=[1, 2, …, $L$].

Assuming the resistors $R_{i1}$= $R_{i2}$=$R_{i3}$=$R_{i4}$=0.5$R_{i5}$=1Ω, the final output of the biasing circuit can be calculated by:

$$V_{obi} = V_{inbi} + V_{bi}, i = [1,2,...,L] \tag{10}$$

which are indeed the inputs to the activation function $g(\cdot)$.




(a) (b) (c)

Fig. 4: Analog implementation of the sigmoid activation function. (a) Schematic circuit diagram of activation function. (b) PSpice simulation of the sigmoid activation function. (c) MATLAB simulation of the sigmoid activation function

*3) Activation function circuit*

Furthermore, an appropriate design scheme for the activation function in ELM is presented in Fig. 4(a), where the input voltage $V_{ob}$ is connected to one side of an NMOS source-coupled pair, biased with a current sink $I_{max}$. According to [25], the output current $I_{out}$ (taken as the right branch current) can be described by:

$$I_{out} = I_n \cdot I_{max} \tag{11}$$

where $I_n$ is the normalized current, it can be given by [25]:

$$I_n = \begin{cases} 0, & V_{ob} < -\sqrt{\dfrac{2}{c}} \\ \dfrac{1}{2} + \dfrac{V_{ob}}{4}\sqrt{4c - V_{ob}^2 c^2}, & |V_{ob}| \le \sqrt{\dfrac{2}{c}} \\ 1, & V_{ob} > \sqrt{\dfrac{2}{c}} \end{cases} \tag{12}$$

where $c$=$\varepsilon$/$I_{max}$ and $\varepsilon$ is a constant gain parameter.

Based on the Ohm's law, the output voltage $V_{out}$ can be obtained by:

$$V_{out} = I_n \cdot I_{max} \cdot R_1 - V_{ss} \tag{13}$$

where $R_1$ is a constant resistor.

Then, a representative case study is further provided to verify the validity of the activation function circuit. The specific parameter setting is given below: the input voltage is $V_{ob}(t)=t$-1 within the time interval of [0, 2], $R_1$=4kΩ, and $V_{ss}$=1.0V. The relevant experiment results obtained by PSpice and MATLAB software platforms are exhibited in Fig. 4(b) and Fig. 4(c), which are like the classical sigmoid activation functions in ELM.

From Fig. 4(b) and Fig. 4(c), the input-output curves (i.e., the sigmoid activation function) under different values of $c$ can be achieved by modifying the current sink $I_{max}$. It is clear that a larger value of $c$ will lead to a higher slope of the activation function. Meanwhile, the range of the activation function can also be changed after resetting the value of the constant resistor $R_1$ and the voltage source $V_{ss}$. In other words, the presented activation function circuit in Fig. 4(a) can be regarded as a general one, which is able to realize other types of activation functions utilizing in most neural networks including back-propagation neural network (BPNN), cellular neural networks (CNNs), deep neural networks (DNNs), recurrent neural networks (RNNs), and so forth.

### *3.2 Training method*

The hardware implementation of the training algorithms in ELM architecture is complex and difficult, which may be even aggravated due to the inevitable variations occurring in the interconnected analog circuit components. In this sub-section, a hardware-friendly training algorithm with four phases is described as follows:

**Step 1**: Initialization for the memristor/resistor crossbar array: At the beginning of the training procedure, the resistances of all the devices installed at the intersections of the crossbar array should be initialized to a proper value. Specifically, the resistor $R_{\lambda ij}$ is set to $2R_L$ and the memristor $M_{\lambda ij}$ is suggested to be an intermediate value between $R_L$ and $R_H$.

**Step 2**: Forward network computations: According to the given training vector, the forward network computation is performed in the entire ELM circuit. The output of the hidden layer (i.e., $h_i(x)$) is read out and saved in a workstation through a specialized ELM/workstation interface.

**Step 3**: Weight updating calculation: The weight updating (i.e., the calculation of Moore-Penrose generalized inverse matrix) is performed with the workstation. Notably, the regularization strategy [17, 31, 32] has been added into the updating phase for better performance. The weight updating can be expressed by:

$$\beta = \left( \frac{I}{\gamma} + \mathbf{H}^T \mathbf{H} \right)^{\dagger} \mathbf{H}^T \mathbf{T} \tag{14}$$

where $I$ is the unit matrix, $\gamma$ means the weight factor. It is clear that the original ELM is just a special case of the regularized ELM when $\gamma \to \infty$. Then, the expected memristances and the relevant programming voltages can be computed by Eq. (5) and Eq. (9), respectively.

**Step 4**: Weight programming: The acquired programming voltages are applied to the memristors interconnected in the adjustable synaptic circuit. The corresponding memristances will change to the desired values rapidly and the weight programming is completed.

For clarity, the entire procedure is further demonstrated in Fig. 5. It is noted that the communication overhead of the ELM/workstation interface in this training scheme is very low, and the additional auxiliary circuits for readout of the hidden layer outputs are no longer necessary, which makes the whole circuit design simpler and more efficient.

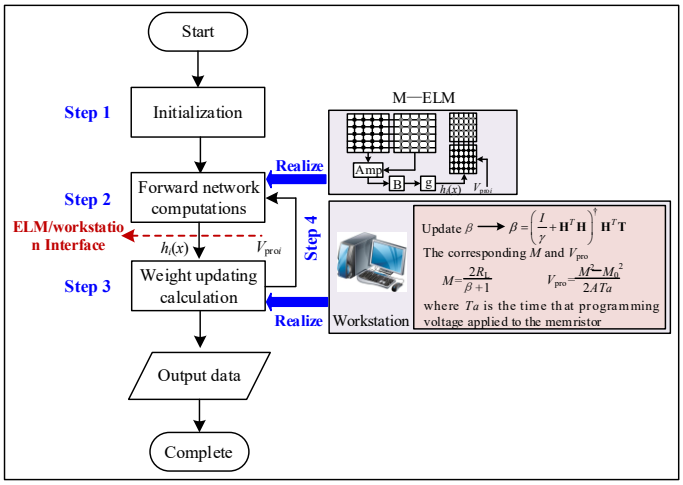



Fig. 5: The training scheme for ELM

### *3.3 Design robustness considerations*

Theoretically, when the proposed ELM hardware system is performed under the ideal condition, the obtained results are the same as the mathematical algorithm. However, the noise induced by process variations and sneak path issue may significantly

affect the circuit performance. In this section, these two major limitations (i.e., the process variations and the sneak path issue) are considered and discussed in detail as follows:

*1) Process variations*

Based on literature [33, 34], the device parameter fluctuations induced by the process variations will inevitably affect the electrical characteristics of the devices, and even the overall performance of the related composite circuits. As for the aforementioned spintronic memristor, the process variations mainly refer to the cross-sectional area ($S=h\times z$) variations and length variations, which may arise from lithographic patterning methods and deposition processes, respectively [34]. Hence, it is necessary to investigate the different impacts on the memristive device and even the entire spintronic memristor-based ELM circuit brought by these two types of process variations. As the matter of convenience, the impact of process variations on any given device parameter can be expressed as a factor $\theta$, and it can be calculated by the ratio of the actual value to the ideal value.

*Cross-sectional area:* According to [34], the actual highest and lowest memristance per unit length under the cross-sectional area variation can be calculated by:

$$r'_{\mathrm{L(H)}}=r_{\mathrm{L(H)}}\cdot\frac{S}{S'} \tag{15}$$

where $S$ and $S'$ are the ideal and actual cross-sectional area respectively.

From Eq. (1), the actual overall memristance under cross-sectional area variations can be written as:

$$M'(x)=r'_{\mathrm{H}}\cdot x+r'_{\mathrm{L}}\cdot(D-x)=\frac{S}{S'}M(x)=\frac{1}{\theta_S}M(x) \tag{16}$$

where $\theta_S$ denotes the variations caused by the cross-sectional area fluctuations.

*Length:* The length variations have no impact on the largest and lowest memristance per unit length, i.e., $r'_{\mathrm{L(H)}}= r_{\mathrm{L(H)}}$. And the actual overall memristance under the length variations can be gotten by

$$M'(x')=r'_{\mathrm{H}}\cdot x'+r'_{\mathrm{L}}\cdot(D'-x') \tag{17}$$

where $D'$ is the actual length of the spintronic memristor.

Assuming $x'/x=D'/D$, Eq. (17) can be rewritten as

$$M'(x')=\frac{D'}{D}\left[r_{\mathrm{H}}\cdot x+r_{\mathrm{L}}\cdot(D-x)\right]=\theta_D M(x) \tag{18}$$

where $\theta_D$ is the variations resulted from the length fluctuations.

Furthermore, the impact of the variances in these two device parameters (i.e., cross-sectional area and length) mainly reflects on the synaptic weights in ELM circuit. Hence, from Eq. (9), the actual synaptic weights under these two process variations can be obtained by:

$$\begin{cases}\psi'_S=\dfrac{2\theta_S R_{\mathrm{L}}}{M}-1=(\psi+1)\cdot\theta_S-1\\ \psi'_D=\dfrac{2R_{\mathrm{L}}}{\theta_D M}-1=\dfrac{(\psi+1)}{\theta_D}-1\end{cases} \tag{19}$$

where $\psi$ is the ideal synaptic weights.

*2) Sneak path issue*

When the crossbars are utilized as memories, only one word line (WL) is raised up, which means one or a few bit lines (BLs) are accessible each time. Such a single-input-single-output (SISO) access inevitably gives rise to currents passing through the unintended paths, which is the so-called sneak path issue [35]. It is noted that the sneak path issue existing in the passive resistive network might greatly limit the size of crossbar arrays, as well as their utilization in memory design.

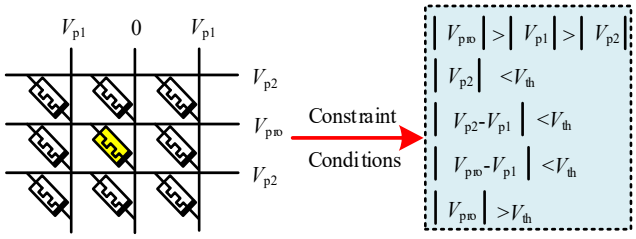



Fig. 6: The specific method for dealing with the sneak path issue during the training process

During the forward neural computation process, our proposed strategy accesses the crossbar array in a multi-input-multi-output (MIMO) mode. Therefore, the sneak path issue can be ignored. While during the training process, a set of protect voltages $V_{pi}$ with relevant constraint conditions is provided in Fig. 6, which is a proper approach to relieve the sneak path issue.

In Fig. 6, $V_{\mathrm{pro}}$ is the desired programming voltage calculated by the workstation, $V_{\mathrm{p1}}$ and $V_{\mathrm{p2}}$ are the two additional protect voltages following the constraint conditions in the black dashed box. $V_{\mathrm{th}}$ is the threshold voltage with the amplitude of $I_{\mathrm{cr}}\cdot R_{\mathrm{L}}$. If the programming voltage $V_{\mathrm{pro}}$ is a positive one, the protect voltages ($V_{\mathrm{p1}}$ and $V_{\mathrm{p2}}$) are larger than zero, the corresponding

memristance (highlighted in yellow) increases and the synaptic weight decreases. On the contrary, if the acquired $V_{pro}$ is a negative one, the protect voltages are set to $V_{p1}$ ($V_{p2}$)<0, the target memristance decreases while the relevant synaptic weight increases. Notably, the voltages applied to the other memristors are all smaller than the threshold voltage $V_{th}$, which means that the unexpected memristance changes will not occur and the sneak path issue can be effectively resolved.

## 4. Applications in image super-resolution

Image super-resolution (SR) is actually an ill-posed inverse problem which aims to obtain the high-resolution (HR) image from a set of (or single) low-resolution (LR) image(s) via proper signal processing techniques [36, 37]. For the sake of verification, the presented M-ELM is utilized to perform the single-image SR reconstruction in this section.

### *4.1 Algorithm description of image SR reconstruction*

In general, the problem of single image SR can be mathematically expressed by:

$$Y = F_D KP + \eta \tag{20}$$

where $Y$ is the LR image, $P$ is the corresponding HR image, $F_D$ and $K$ are the sampling operator and blurring operator respectively. Parameter $\eta$ denotes the additive noise. The goal of the single image SR is to achieve an estimation of HR image $P$ from a LR image $Y$.

Specifically, the entire process of the M-ELM based single-image SR can be completed in two phases, i.e., the training and testing. The relevant algorithm description with the detailed illustration (as seen in Fig. 7) is provided as follows:

**Training phase:**

**Step 1**: Based on a given HR training image $P$', the corresponding LR training image $Y$' can be obtained through the downscale operation (i.e., Eq. (20)).

**Step 2**: An amplified image $W$' with the same size of the HR image $P$' is acquired by the basic bilinear interpolation (magnification factor: $F_D$).

**Step 3**: The high-frequency (HF) information $W_{HF}$ can be gotten by the subtraction operation between the HR image $P$' and the amplified image $W$'. Notably, the HF information $W_{HF}$ is also the target output of the M-ELM during the training process.

**Step 4**: The image feature vectors $Fv$, as the inputs fed to the M-ELM, are extracted from the LR image $Y$'. The feature vectors can be given by:

$$Fv(i,j) = [Pa(i,j), d(i,j), d^2(i,j)]^T \tag{21}$$

where $Pa(i, j)$ is a row vector reshaped from a local image patch (size: $l_p \times l_p$) centered at the location $(i, j)$ of the LR image $Y$'. $d(i, j)$ and $d^2(i, j)$ are the corresponding first and second order derivatives in horizontal and vertical directions [38]. Once the obtained training dataset $[Fv(i, j), W_{HF}(i, j)]$ is stacked to the M-ELM, and the training process begins.

**Step 5**: When the training is completed, a well-trained M-ELM model is generated. It directly reflects the mapping relationship between the initially interpolated image (feature vectors) and the HF information.

**Testing phase:**

**Step 1**: Given a LR image $Y$, its amplified version $W$ can be obtained by the same bilinear interpolation.

**Step 2**: The feature vectors of the image $W$ are injected to the well-trained M-ELM for testing. The predicted values $W_{HF}$ can be obtained from the output terminal of the M-ELM.

**Step 3**: The final HR image $P$ can be easily obtained by:

$$P = W_{HF} + W \tag{22}$$

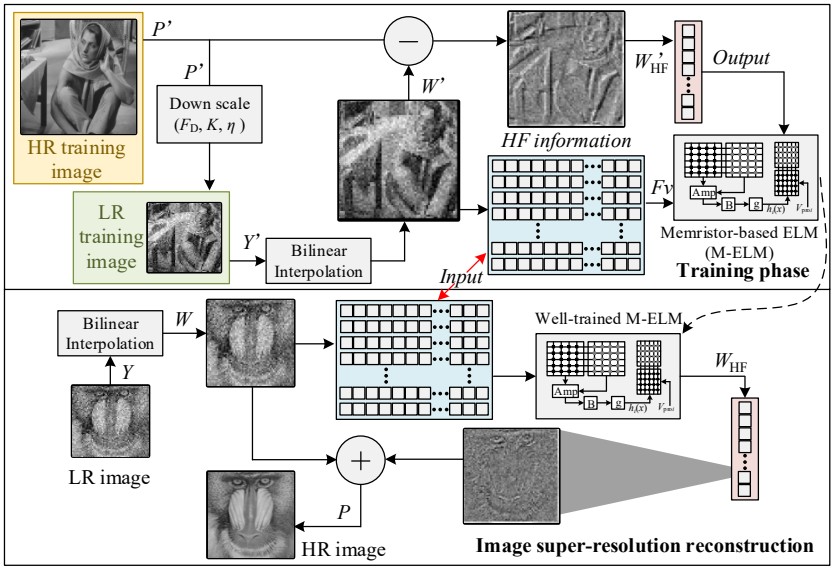


Fig. 7: The block diagram of the image SR algorithm

*4.2 Experimental results and analysis*

In this subsection, a series of experiments with the relevant analysis are carried out. Notably, the entire experiment process is performed on a desktop workstation with the Intel Core i7 6700K processor, 3.4 GHz CPU and Windows 10 OS.

As discussed in Section 3, the process variations on the device parameters may have an impact on the overall performance of the presented M-ELM, especially for the synaptic weights. Hence, the relevant computer simulations are necessarily conducted before the image SR reconstruction, and the experimental results are demonstrated in Fig. 8.

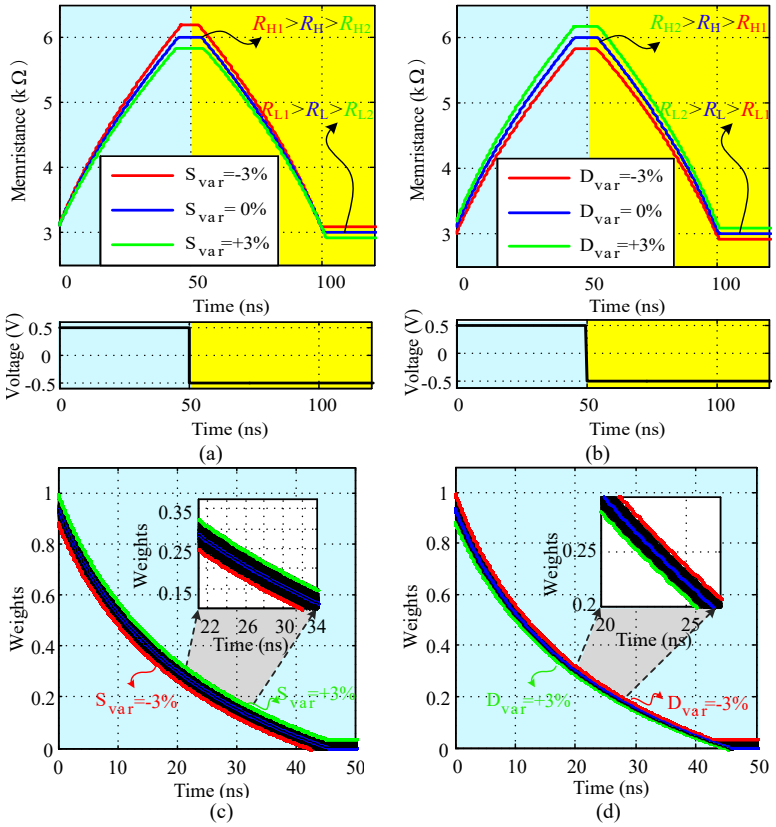



Fig. 8: Impacts of the cross-sectional area variation and length variation. (a) Memristance under the cross-sectional area variation. (b) Memristance under the length variation. (c) Synaptic weight under the cross-sectional area variation. (d) Synaptic weight under the length variation

**Note:** $S_{var}$ ($D_{var}$) is defined as the ratio between $\Delta S$ and $S$ ($\Delta D$ and $D$), where $\Delta$ means the difference of the actual and the ideal value.

From Fig. 8(a) and Fig. 8(b), process variations on the cross-sectional area and on the length affect the memristance in an opposite way. Namely, a negative $S_{var}$ (-3%) or a positive $D_{var}$ (+3%) leads to a larger memristance; while a positive $S_{var}$ (+3%) or a negative $D_{var}$ (-3%) results in a smaller memristance. Meanwhile, 200 times repeated simulations with the $S_{var}$ ($D_{var}$)=[-3%, +3%] are conducted to visually demonstrate the overall impact of the process variations on the synaptic weights, as shown in Fig. 8(c) and Fig. 8(d). The smaller cross-sectional area ($S_{var}$<0) and the larger length ($D_{var}$>0) result in the reduction of the synaptic weight; the larger cross-sectional area ($S_{var}$>0) and the smaller length ($D_{var}$<0) result in the increase of the synaptic weight. Additionally, the technical parameter setting is provided in **Table 1**, and the applied voltage is a step signal with the amplitude of ±0.5V.

Then, the HR images with the size of 512×512 can be obtained following the presented image SR method. The primary parameters used in the image SR are listed in **Table 4**. Other four existing methods, i.e., the context-based image dependent (CBID) method [37], original ELM based method [16], Online Sequential (OS) ELM based method [39], and memristive ELM based method [26] are introduced for comparison purpose. Correspondingly, the MATLAB codes can be downloaded from the authors' websites.

Table 4: Collection of technical parameters

| Parameters | Values (Unit) | Parameters | Values (Unit) |
|---|---|---|---|
| $r_L$ | 3e8 Ω/m | $L$ | 30 |
| $r_H$ | 6e9 Ω/m | $V_{th}$ | 35 μv |
| $M_0$ | 3 kΩ | $V_{p1}$ | ±30 μv |
| $D$ | 1000 nm | $V_{p2}$ | ±20 μv |
| $h$ | 7 nm | $F_D$ | 2 |
| $z$ | 10 nm | $l_p$ | 3 |

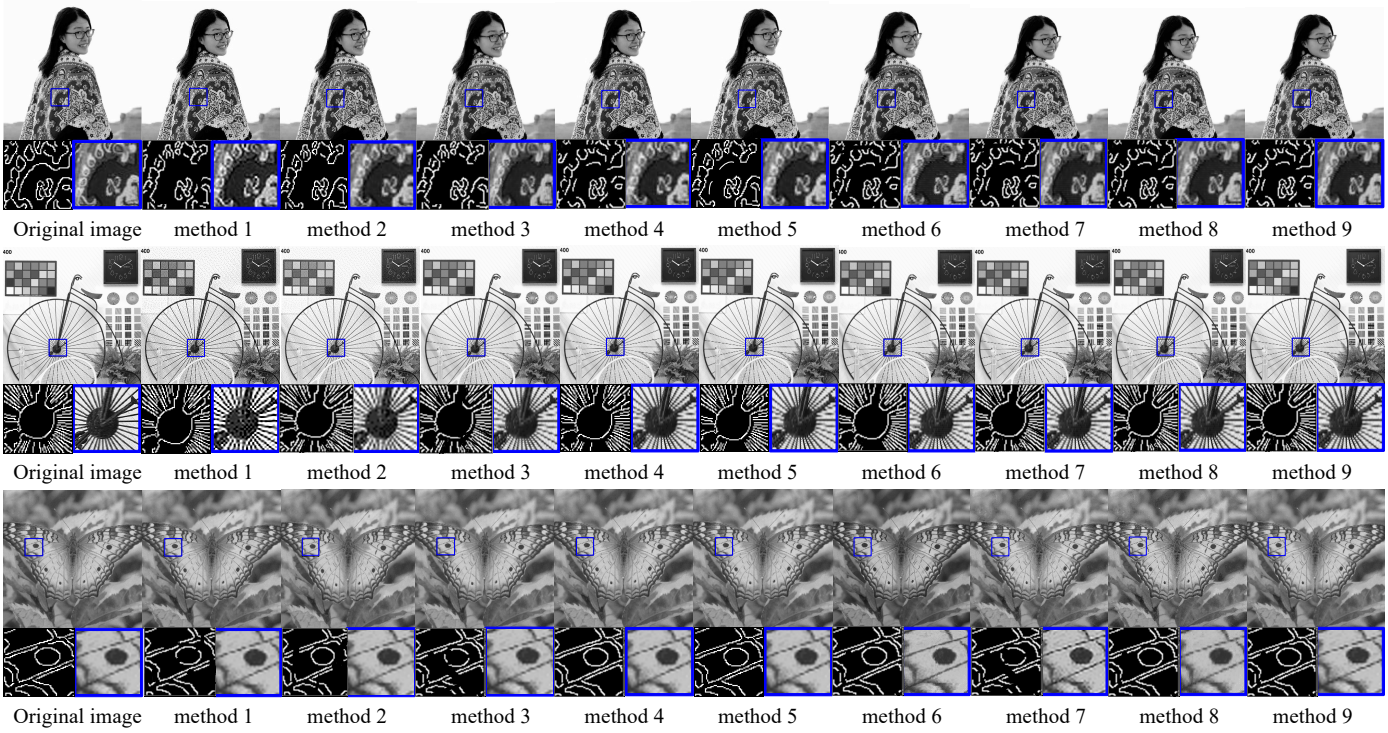



Fig. 9 Results comparison of the presented method with the other competitors. Specifically, method 1→CBID method, method 2→Original ELM based method, method 3→OS ELM based method, method 4→memristive ELM based method, method 5→the proposed method, method 6→the proposed method with $S_{var}$=+3%, method 7→the proposed method with $D_{var}$=+3%, method 8→the proposed method with $S_{var}$=−3%, and method 9→the proposed method with $D_{var}$=−3%.

From Fig. 9, the leftmost column images are the original HR images, the middle four columns are the obtained HR images generated by the CBID method, the original ELM based method, the OS ELM based method, and the memristive ELM based method respectively. The remaining fifteen images are achieved by the proposed image SR method with the process variations ($S_{var}$, $D_{var}$)=[(0, 0), (+3%, 0), (0, +3%), (−3%, 0), (0, −3%)]. Notably, the locally enlarged sub-images (labeled by blue box) and the corresponding binary edge images (using canny operators) are provided for subjective analysis. According to the human visual system (HVS), the CBID method is not able to recover the high-frequency information effectively, and the obtained images suffer from the blurry issue. Meanwhile, the presented method and the other ELM based methods are able to produce more details (including the edge and texture information) on the final images. For method 6 ~ method 9, although the device process variations have an impact on the synaptic weights, the generated images seem not affected by the small fluctuations of the cross-sectional area and length of the memristor.

Notably, the above subjective visual analysis for image SR mainly depends on the HVS which refers to several factors, such as image category, observers' preference, mission requirement, and so forth. Especially, for the cases when the visual difference is small (for example, the HR images obtained by the ELM based methods), it is difficult to provide the precise analysis and sensible judgement for the final performance. Hence, an objective analysis is carried out through three common objective image assessment indexes, i.e., Peak Signal to Noise Ratio (PSNR), Structural Similarity (SSIM) and Information Entropy (IE) [40]. The overall information of PSNR, SSIM, and IE is collected in **Table 5**:

**Table 5**: The image quality assessment indexes for image super-resolution reconstruction

| Assessment index | Concrete meaning | Mathematical expression |
|---|---|---|
| Peak Signal to Noise Ratio | Representing a measure of the peak error. | $MSE = \frac{1}{m \cdot n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left[G(i,j)-F(i,j)\right]^2$<br>$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_F^2}{MSE}\right)$ |
| Structural similarity | Representing the similar degree between the Original HR image and the obtained HR image. | $SSIM(G,F) = \left(\frac{2\mu_g\mu_f + c_1}{\mu_g^2 + \mu_f^2 + c_1}\right) \cdot \left(\frac{2\sigma_g\sigma_f + c_2}{\sigma_g^2 + \sigma_f^2 + c_2}\right) \cdot \left(\frac{\mathrm{cov}_{gf} + c_3}{\sigma_g\sigma_f + c_3}\right)$ |
| Information entropy | Denoting the specific information content of the obtained HR image. | $IE = -\sum_{i=0}^{L_e-1} p_i \log_2(p_i)$ |

**Note**: $G$ and $F$ denote the original high-resolution (HR) image and the obtained HR image (size: $m \times n$), respectively. $MAX_F$ commonly equals to 255 representing the possible maximum pixel of HR image $F$. $\mu_g$ and $\mu_f$ are the average value of $G$ and $F$. $\sigma_g$ ($\sigma_f$) and $\mathrm{cov}_{gf}$ are the relevant standard deviation and covariance of $G$ and $F$. Parameters $c_1$, $c_2$ and $c_3$ are all constants. $L_e$ is the overall gray level and $p_i$ is the occurrence probability of pixel $i$.

Correspondingly, the calculated results are collected in **Table 6**:

**Table 6**: The objective performance evaluation of the images yielded by different methods

| | PSNR (dB) | | | SSIM | | | IE | | | Time |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Girl** | **Bicycle** | **Butterfly** | **Girl** | **Bicycle** | **Butterfly** | **Girl** | **Bicycle** | **Butterfly** | **(Sec)** |
| CBID method | 29.77 | 29.92 | 30.05 | 0.8702 | 0.8717 | 0.8813 | 6.1272 | 6.0120 | 5.8283 | 0.041 |
| Original ELM based method | 31.15 | 32.55 | 32.47 | 0.8819 | 0.9022 | 0.9010 | 6.7301 | 6.4186 | 6.3122 | 0.0033 |
| OS ELM based method | 32.09 | 32.77 | 32.50 | 0.8830 | 0.9072 | 0.9021 | 6.7331 | 6.4330 | 6.3227 | 0.0057 |
| memristive ELM based method | 31.02 | 32.39 | 32.32 | 0.8797 | 0.8991 | 0.8998 | 6.7123 | 6.3982 | 6.3111 | 0.0062 |
| Proposed method ($S_{var}$=0% $D_{var}$=0%) | 31.87 | 32.76 | 32.48 | 0.8823 | 0.9047 | 0.9012 | 6.7238 | 6.4200 | 6.3130 | 0.0012 |
| Proposed method ($S_{var}$=+3% $D_{var}$=0%) | 31.85 | 32.63 | 32.52 | 0.8821 | 0.9050 | 0.8977 | 6.7330 | 6.4312 | 6.2998 | 0.0012 |
| Proposed method ($S_{var}$=0% $D_{var}$=+3%) | 31.81 | 32.67 | 32.65 | 0.8816 | 0.9033 | 0.8971 | 6.7252 | 6.4329 | 6.3225 | 0.0013 |
| Proposed method ($S_{var}$=-3% $D_{var}$=0%) | 31.87 | 32.62 | 32.50 | 0.8811 | 0.9041 | 0.9013 | 6.7115 | 6.4173 | 6.2726 | 0.0013 |
| Proposed method ($S_{var}$=0% $D_{var}$=-3%) | 31.82 | 32.77 | 32.42 | 0.8819 | 0.9037 | 0.8983 | 6.7138 | 6.4228 | 6.2514 | 0.0013 |

From **Table 6**, it can be seen that the HR images (girl, bicycle, and butterfly) obtained by the ELM based methods achieve better PSNR, SSIM and IE results than the image obtained by the CBID method, which corresponds with the subjective visual analysis, i.e., the images generated by the ELM based methods are closer to the original image. Among the four ELM based methods, the proposed method and the OS ELM based method have larger PSNR, SSIM, and IE results, compared with the other two ELM based methods. Large PSNR and SSIM values illustrate that these two ELM methods are superior in terms of maintaining the image luminance, image contrast ratio, and image structure. Meanwhile, large IE indicates that less information is lost during the entire reconstruction process. Furthermore, for the proposed method and the OS ELM based method, it is clear that the differences of the three objective image assessment indexes (i.e., PSNR, SSIM, and IE) are very small. However, the algorithm running time of the proposed method is approximately quarter of the other method, which means that the proposed method is more efficient and less time-consuming. In addition, the small fluctuations ($\pm \leq 3\%$) of the device parameters have limited effects on the values of the PSNR, SSIM and IE results, which indicates the presented algorithm is an effective one with good fault tolerance and robustness.

## 5. Conclusions

In this paper, the hardware implementation of the extreme learning machine (ELM) is investigated. Specifically, the spintronic memristor with its resistance variation rule is examined via detailed formula derivation and numerical simulations. Then, multiple spintronic memristors with crossbar array configuration are utilized to realize a bimodal synaptic circuit, which is able to simulate two kinds of synapses between the adjacent layers in the ELM. Meanwhile, the compact M-ELM architecture is proposed, along with the concrete design of the biasing circuit and the activation function circuit. Moreover, during the robustness analysis, the impacts of the process variations (mainly referring to the cross-sectional area and length), as well as the sneak path issue are discussed. For the purpose of verification, the presented M-ELM is applied to the single image super-resolution reconstruction.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China grant number 61671194, Fundamental Research Funds for the Provincial Universities of Zhejiang grant number GK199900299012-010, and Brunel Research Initiative and Enterprise Fund.

## References

[1] R. Tessier, K. Pocek, and A. DeHon, Reconfigurable computing architectures, *Proc. IEEE*, 103(3) (2015) 332-354.
[2] P. Gao, B. V. Benjamin, and K. Boahen, Dynamical system guided mapping of quantitative neuronal models onto neuromorphic hardware, *IEEE Trans. Circuits Syst. I: Reg. Papers*, 59(10) (2012) 2383-2394.
[3] S. Friedmann, J. Schemmel, A. Grübl, A. Hartel, M. Hock, and K. Meier, Demonstrating hybrid learning in a flexible neuromorphic hardware system, *IEEE Trans. Biomed. Circuits Syst.*, 11(1) (2017) 128-142.
[4] R. Wang, C. S. Thakur, G. Cohen, T. J. Hamilton, J. Tapson, and A. van Schaik, Neuromorphic hardware architecture using the neural engineering framework for pattern recognition, *IEEE Trans. Biomed. Circuits Syst.*, 11(3) (2017) 574-584.
[5] X. Yao, Evolving artificial neural networks, *Proc. IEEE*, 87(9) (1999) 1423-1447.
[6] Kalantari, Ali, et al., Computational intelligence approaches for classification of medical data: State-of-the-art, future challenges and research directions, *Neurocomput.*, 276 (2018) 2-22.
[7] L. O. Chua, Memristor-the missing circuit element, *IEEE Trans. Circuit Theory*, 18(5) (1971) 507–519.
[8] K. Kim, S. Gaba, D. Wheeler, J. Cruz-Albrecht, T. Hussain, N. Srinivasa, and W. Lu, A functional hybrid memristor crossbar-array/CMOS system for data storage and neuromorphic applications, *Nano Lett.*, 12(1) (2011) 389–395.
[9] V. Milo, G. Malavena, et al., Memristive and CMOS devices for neuromorphic computing. *Mater.*, 13(1) (2020) 166-172.
[10] H. Kim, M. Sah, C. Yang, T. Roska, and L. Chua, Memristor bridge synapses, *Proc. IEEE*, 100(6) (2012) 2061–2070.
[11] S. Adhikari, C. Yang, H. Kim, and L. Chua, Memristor bridge synapse-based neural network and its learning, *IEEE Trans. Neural Netw. Learn. Syst.*, 23(9) (2012) 1426–1435.
[12] B. Li, Y. Zhao, and G. Shi, A novel design of memristor-based bidirectional associative memory circuits using Verilog-AMS, *Neurocomput.*, 330 (2019) 437-448.

[13] Y. Zhang, Y. Li, X. Wang, and E. G. Friedman, Synaptic characteristics of Ag/AgInSbTe/Ta-based memristor for pattern recognition applications, *IEEE Trans. Electron Device.*, 64(4) (2017) 1806-1811.
[14] P. J. Zhang, C. Li, T. Huang et al., Forgetting memristor based neuromorphic system for pattern training and recognition, *Neurocomput.*, 222 (2017) 47-53.
[15] K. Hamedani, L. Liu, R. Atat, J. Wu, and Y. Yi, Reservoir computing meets smart grids: attack detection using delayed feedback networks, *IEEE Trans. Ind. Informat.*, 14(2) (2018) 734-743.
[16] G. Huang, Q. Zhu, and C. Siew, Extreme learning machine: theory and applications, *Neurocomput.*, 70(1-3) (2006) 489-501.
[17] G. Huang, H. Zhou, X. Ding, and R. Zhang, Extreme learning machine for regression and multiclass classification, *IEEE Trans Syst. Man, Cybern., B, Cybern.*, 42(2) (2012) 513-529.
[18] G. Huang, G. Huang , S. Song, et al., Trends in extreme learning machines: A review. *Neural Netw.*, 61(1) (2015) 32-48.
[19] S. Decherchi, P. Gastaldo, A. Leoncini, et al. Efficient digital implementation of extreme learning machines for classification. *IEEE Trans Circuits Syst. II, Exp. Briefs*, 59(8) (2012) 496-500.
[20] V. Frances, V. Jose, et al., Hardware implementation of real-time Extreme Learning Machine in FPGA: analysis of precision, resource occupation and performance. *Comput. Electr. Eng.*, 51(1) (2016) 139-156.
[21] T. Yeam, N. Ismail, K. Mashiko, et al., FPGA implementation of extreme learning machine system for classification. *TENCON IEEE Region 10 Conference. IEEE*, (2017) 1868-1873.
[22] A. Basu, S. Shuo, H. Zhou, et al., Silicon spiking neurons for hardware implementation of extreme learning machines. *Neurocomput.*, 102 (2013) 125-134.
[23] M. Sharma, U. Lohani, V. Parmar, et al. Design of an optimized CMOS ELM accelerator, *IEEE 32nd International Conference on VLSI Design (VLSID)*, (2019) 443-447.
[24] E. Yao, A. Basu. VLSI extreme learning machine: A design space exploration. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 25(1) (2016) 60-74.
[25] C. Merkel and D. Kudithipudi, Neuromemristive extreme learning machines for pattern classification, *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, (2014) 77-82.
[26] H. Li, L. Wang, S. Duan, A Kind of Extreme Learning Machine Based on Memristor Activation Function, *International Conference on Extreme Learning Machine*. Springer, Cham, (2017) 210-218.
[27] Y. Chen, E. Yao, and A. Basu, A 128-channel extreme learning machine based neural decoder for brain machine interfaces, *IEEE Trans. Biomed. Circuits Syst.*, 10(3) (2016) 679–692.
[28] M. Rasouli, Y. Chen, A. Basu, et al., An extreme learning machine-based neuromorphic tactile sensing system for texture recognition. *IEEE Trans. Biomed. Circuits Syst.*, 2018, 12(2): 313-325.
[29] Y. Chen and X. Wang, Compact modeling and corner analysis of spintronic memristor, *In Proceedings of IEEE/ACM International Symposium on Nanoscale Architectures*, (2009) 7-12.
[30] X. Wang, Y. Chen, H. Xi, H. Li, and D. Dimitrov, Spintronic memristor through spin-torque-induced magnetization motion, *IEEE Electron Device Lett.*, 30(3) 294–297.
[31] W. Deng, Q. Zheng, L. Chen, Regularized extreme learning machine, *IEEE symposium on computational intelligence and data mining*, (2009) 389-395.
[32] H. Dai, J. Cao, T. Wang, et al., Multilayer one-class extreme learning machine. *Neural Netw.*, (115) (2019) 11-22.
[33] Z. Dong, C. S. Lai, Z. Xu, et al., A general memristor-based pulse coupled neural network with variable linking coefficient for multi-focus image fusion, *Neurocomput*., 25 (2018) 172-183.
[34] D. Niu, Y. Chen, C. Xu, and Y. Xie, Impact of process variations on emerging memristor, *In Proceedings of the 47th Design Automation Conference(DAC)*, (2010) 877-882.
[35] C. Jung, J. Choi, and K. Min, Two-step write scheme for reducing sneak-path leakage in complementary memristor array, *IEEE Trans. Nanotechnol.*, 11(3) (2010) 611-618.
[36] Z. Dong, C. S. Lai, Z. Xu, and D. Qi, Single Image Super-Resolution via the Implementation of the Hardware-Friendly Sparse Coding, *In 2018 37th Chinese Control Conference* (2018) 8132-8137.
[37] S. Prasad Jaiswal, V. Jakhetiya, A. Kumar, and A. Tiwari, A low complex context adaptive image interpolation algorithm for real-time applications, *In IEEE Instrumentation and Measurement Technology Conference*, (2012) 969–972.
[38] A. Laghrib, et al. Multiframe super-resolution based on a high-order spatially weighted regularization, *IET Image Process*., 12(6) (2018) 928-940.
[39] A. Safaei, Q. Wu, T. Akilan, et al., System-on-a-Chip (SoC)-Based Hardware Acceleration for an Online Sequential Extreme Learning Machine (OS-ELM). *IEEE Trans. Comput. Aided Design Integ. Circuits Syst.*, 38(11) (2019) 2127-2138.
[40] S. Bosse, D. Maniry, K. Müller, T. Wiegand, and W. Samek, Deep neural networks for no-reference and full-reference image quality assessment, *IEEE Trans. Image Process.*, 27(1) (2017) 206-219.