


# Memristive neural network circuit with fault tolerance for character recognition

Mei Guo(郭梅), Jikang Liu(刘继康), and Jingzhi Xu(徐景芝)

**Citation:**Chin. Phys. B, 2026, 35 (**6**): 060702. DOI: 10.1088/1674-1056/ae3f94
http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

**What follows is a list of articles you may be interested in**

---

## Disturbance observer-based fuzzy fault-tolerant control for high-speed trains with multiple disturbances

Qian-Ling Wang(王千龄), Cai-Qing Ma(马彩青), and Xue Lin(林雪)
Chin. Phys. B, 2023, 32 (**10**): 100701. DOI: 10.1088/1674-1056/acdfbe

## A spintronic memristive circuit on the optimized RBF-MLP neural network

Yuan Ge(葛源), Jie Li(李杰), Wenwu Jiang(蒋文武), Lidan Wang(王丽丹), and Shukai Duan(段书凯)
Chin. Phys. B, 2022, 31 (**11**): 110702. DOI: 10.1088/1674-1056/ac6b1d

## Memristor-based multi-synaptic spiking neuron circuit for spiking neural network

Wenwu Jiang(蒋文武), Jie Li(李杰), Hongbo Liu(刘洪波), Xicong Qian(钱曦聪), Yuan Ge(葛源), Lidan Wang(王丽丹), and Shukai Duan(段书凯)
Chin. Phys. B, 2022, 31 (**4**): 040702. DOI: 10.1088/1674-1056/ac380b

## Investigation on the dynamic behaviors of the coupled memcapacitor-based circuits

Zhi Zhou(周知), Dong-Sheng Yu(于东升), Xiao-Yuan Wang(王晓媛)
Chin. Phys. B, 2017, 26 (**12**): 120701. DOI: 10.1088/1674-1056/26/12/120701

## Composite behaviors of dual meminductor circuits

Zheng Ci-Yan (郑辞晏), Yu Dong-Sheng (于东升), Liang Yan (梁燕), Chen Meng-Ke (陈孟科)
Chin. Phys. B, 2015, 24 (**11**): 110701. DOI: 10.1088/1674-1056/24/11/110701

---

**SPECIAL TOPIC — Biophysical circuits: Modeling & applications in neuroscience**

# Memristive neural network circuit with fault tolerance for character recognition

Mei Guo(郭梅)[1], Jikang Liu(刘继康)[1], and Jingzhi Xu(徐景芝)[2,†]

[1] *College of Electrical Engineering and Automation, Shandong University of Science and Technology, Qingdao* 266590, *China*
[2] *Department of Science and Technology Management, Shandong University of Science and Technology, Qingdao* 266590, *China*

(Received 17 December 2025; revised manuscript received 30 January 2026; accepted manuscript online 30 January 2026)

Memristor-based neural networks are one of the most promising approaches for the hardware implementation of artificial neural networks. In this paper, a memristor-based neural network circuit based on a one-memristor–one-resistor (1M1R) synaptic array structure is designed for character recognition. Compared with other memristive synaptic arrays, the 1M1R structure can reduce the number of memristors used. However, memristors may malfunction due to fabrication defects and the influence of external factors, resulting in a decrease in the accuracy of the circuit's character recognition, and a suitable solution needs to be found to improve the stability and durability of the circuit. Therefore, in this paper, a fault-tolerant module with feedback adjustment capability is designed in the memristive neural network circuit that can re-adjust the weights of the memristors through *in-situ* training to solve multiple faults in the memristive neural network. The effect of fault tolerance is verified by character recognition. The experimental results show that the designed memristive neural network circuit can accurately realize character recognition, and the designed fault-tolerant circuit can well tolerate multiple faults, ensuring stable operation of the circuit under fault conditions.

**Keywords:** memristor, neural network circuit, character recognition, fault-tolerant, feedback

**PACS:** 07.05.Mh, 45.30.+s

**DOI:** 10.1088/1674-1056/ae3f94 **CSTR:** 32038.14.CPB.ae3f94

## 1. Introduction

Due to the limitations of semiconductor materials and current manufacturing technologies, reducing transistor size to fabricate high-density complementary metal–oxide–semiconductor circuits has become more difficult. Memristor-based circuit design is one of the possible solutions to this problem.[1,2] In 1971, Chua predicted the existence of such devices, and Williams *et al*. realized the first physical model.[3,4] An important feature of a memristor is that its $I$–$V$ characteristic curve is a pinched hysteresis loop under periodic excitation. Memristors have the advantages of a nanostructure and high reliability,[5–7] making them suitable for integration.[8–10] Additionally, their variable-resistance characteristic gives them storage ability. Compared with traditional computing devices, memristive devices offer lower power consumption and higher computational efficiency.[11,12] Based on these advantages, memristor-based neural networks have developed rapidly.[13–16] Memristive neural networks have strong learning and memory capabilities by simulating the connections and signal transmission between neurons in the human brain.

Memristive neural networks have become a key vehicle for realizing energy-efficient neuromorphic computing hardware due to their brain-like parallel architecture and integrated memory-computing characteristics.[17–27] In recent years, memristive neural networks have been successfully applied in many fields, such as speech recognition[28,29] and medical treatment.[30,31] However, as the application of memristive neural networks expands, efficient and accurate computational characteristics become particularly important. This also means that the traditional memristive neural network faces the challenge of fault tolerance.

Due to the inherent characteristics of memristor technology, including noise interference, nonlinear response, and manufacturing defects,[32–35] memristive neural networks are prone to faults and errors in practical applications. These faults and errors can lead to performance degradation, limited learning capabilities, or even complete failure of the network.[36] To ensure the reliability and stability of memristive neural networks, researchers have employed hardware fault tolerance or software algorithm strategies to detect, correct, or mitigate faults and errors so as to improve the robustness and performance of the networks. A new March RC detection algorithm was proposed,[37] which can detect multiple fault types and significantly reduce the testing time. Hardware redundancy was reduced by iteratively pruning critical faults using pruning techniques.[38] A methodology to perform integrated crossbar circuit design with defect tolerance was proposed,[39] which utilizes fault-tolerant algorithms to achieve fault tolerance. A unified robust network training

†Corresponding author. E-mail: skd994298@sdust.edu.cn

© 2026 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

framework was proposed for memristor crossbar-based neural computing systems.[40] Traditional algorithmic fault tolerance is not applicable to effectively dealing with memristor faults in the memristive neural network circuits proposed in that paper, and circuit-level solutions need to be designed for neural network circuits. In this paper, the circuit implementation of a memristive neural network is carried out, and its feasibility is verified through character recognition. Faults that may occur in memristors within the circuit are analyzed, and a fault-tolerant method based on *in situ* learning is proposed.

This work is the first to integrate a closed-loop feedback mechanism with one memristor–one resistor (1M1R) memristor synapses, achieving adaptive fault tolerance entirely within the analog domain without requiring a digital co-processor. In contrast to existing methods based on hardware redundancy (e.g., two cross-coupled memristor (2M) structures), algorithmic retraining, or offline remapping for repair, our scheme enables real-time, *in situ* adjustment of conductance states upon fault occurrence to restore network functionality. Furthermore, this mechanism operates without relying on additional memory units or complex control logic, significantly reducing hardware overheads.

The rest of the paper is organized as follows. Section 2 introduces the Knowm physical memristor model. Section 3 constructs the memristive neural network and validates it with a character recognition application. Section 4 analyzes the types of faults and proposes a fault tolerance scheme based on the *in situ* learning principle and compares it with other works. Section 5 concludes the paper.

## 2. Introduction of the memristor model

All kinds of memristive materials, such as metal oxides, chalcogenide semiconductors and solid electrolytes, have been investigated; these all provide divergent memristive mechanisms.[41–44] Appropriate memristor models are essential for memristor-based circuit and system design. In our previous work, an adjustable voltage threshold memristor model with deterministic parameters was established based on the Knowm memristor.[45] The relationship between current and voltage in this model is[45]

$$v(t) = [R_{\mathrm{H}} - h(t)\cdot(R_{\mathrm{H}} - R_{\mathrm{L}})]\, i(t), \tag{1}$$

where $h(t)$ is the state variable that represents the normalized thickness of the conductive region of the memristor, $R_{\mathrm{H}}$ and $R_{\mathrm{L}}$ are the high- and low-resistance states of the memristor, $R_{\mathrm{H}} = 45\ \Omega$ and $R_{\mathrm{L}} = 20\ \Omega$.[45] The derivative of state variable $h(t)$ is[45]

$$\frac{\mathrm{d}h(t)}{\mathrm{d}t} = \begin{cases} a_1\big(N[v(t)] + b_1\big) i(t) f\big[h(t)\big], & v(t) \geq 0, \\ a_2\big(N[v(t)] + b_2\big) i(t) f\big[h(t)\big], & v(t) < 0. \end{cases} \tag{2}$$

The parameters $a_1$, $a_2$, $b_1$, and $b_2$ fitted by Simulink in Eq. (2) are $a_1 = 2500$, $a_2 = 3000$, $b_1 = 0$, and $b_2 = 0.005$. $N[V(t)]$ represents the threshold voltage function, and $f[h(t)]$ represents the window function of the memristor.[45] The voltage threshold function $N[V(t)]$ is formulated as follows:[45]

$$N\big[v(t)\big] = \begin{cases} \left[\dfrac{v(t)}{v_{\mathrm{ON}}}\right]^{k_1}, & v(t) \geq 0, \\ \left[\dfrac{v(t)}{v_{\mathrm{OFF}}}\right]^{k_2}, & v(t) < 0, \end{cases} \tag{3}$$

where $V_{\mathrm{ON}}$ and $V_{\mathrm{OFF}}$ represent the positive and negative threshold voltages of the memristor: $V_{\mathrm{ON}} = 0.7$ V and $V_{\mathrm{OFF}} = -0.9$ V. The parameters $k_1 = 25$, $k_2 = 100$, $p_1 = 3.8$, and $p_2 = 15$ are obtained.[45] The window function $f[n(t)]$ is described as follows:[45]

$$f\big[h(t)\big] = \begin{cases} \big[1 - h(t)\big]^{p_1}, & i(t) \geq 0, \\ \big[h(t)\big]^{p_2}, & i(t) < 0. \end{cases} \tag{4}$$

## 3. Memristive neural network circuits for character recognition

### 3.1. Memristor synaptic neuron circuit design

In this section, a neuronal circuit is designed with a memristor and a resistor as a synaptic structure, as shown in Fig. 1. Compared with the traditional dual-memristor structure, this structure uses one less memristor, which simplifies the circuit structure, reduces circuit overhead, and improves its stability. The blue box in the figure contains the synaptic unit module consisting of the basic synaptic structure. $M_i$ represents the memristor, $R_{\mathrm{g}}$ is the fixed resistor in the synaptic structure, $V_i$ is the input signal of the neuron circuit, $I_i^-$ represents the current flowing through the memristor, and $I_i^+$ represents the current flowing through the fixed resistor $R_{\mathrm{g}}$. The yellow box represents the activation circuit, with $R_1 = R_2 = R_3 = R_4 = 45\ \mathrm{k}\Omega$. $V_{\mathrm{b}}$ is the reference voltage, with $V_{\mathrm{b}} = 0$ V.

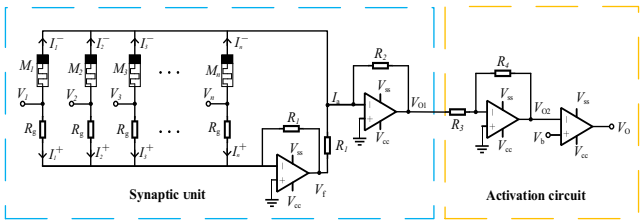



**Fig. 1.** Memristive synaptic neuronal circuits with 1M1R structure.

The current $I_i^+$ flowing through the resistor $R_{\mathrm{g}}$ is the product of the voltage $V_i$ and the conductance $G_{\mathrm{g}}$, and the current $I_i^-$ through the memristor $M_i$ is the product of the voltage $V_i$ and the conductance $G_{\mathrm{m}}$, as shown below:

$$\begin{cases} I_i^+ = V_i \times G_{\mathrm{g}}, \\ I_i^- = V_i \times G_{\mathrm{m}}. \end{cases} \tag{5}$$

According to Kirchhoff's law, we can obtain

$$V_{\mathrm{f}} = -R_1 \sum_{i=1}^{n} V_i G_{\mathrm{g}}, \tag{6}$$

$$I_{\mathrm{a}} = I^{-} + (-I^{+}) = \sum_{i=1}^{n} V_i G_{\mathrm{m}i} + V_{\mathrm{f}} G_1 = \sum_{i=1}^{n} V_i \left(G_{\mathrm{m}i} - G_{\mathrm{g}}\right). \tag{7}$$

The current $I_{\mathrm{a}}$ flows through the amplifier to obtain the voltage $V_{\mathrm{O1}}$,

$$V_{\mathrm{O1}} = I_{\mathrm{a}} \times R_2 = \sum_{i=1}^{n} V_i \left(G_{\mathrm{m}i} - G_{\mathrm{g}}\right) R_2. \tag{8}$$

After being input into the activation circuit and passing through the inverting amplifier, the voltage $V_{\mathrm{O2}}$ is obtained by

$$\begin{aligned} V_{\mathrm{O2}} &= \frac{-V_{\mathrm{O1}}}{R_3} \times R_4 = \sum_{i=1}^{n} V_i \left(G_{\mathrm{m}i} - G_{\mathrm{g}}\right) R_2 \\ &= \frac{\sum_{i=1}^{n} V_i \left(G_{\mathrm{g}} - G_{\mathrm{m}i}\right) R_2 R_4}{R_3}. \end{aligned} \tag{9}$$

In the above equation, $i = 1, 2, \ldots, n$, $G_{\mathrm{g}} = 1/R_{\mathrm{g}}$, $G_1 = 1/R_1$, and $G_{\mathrm{m}i}$ represents the conductance of the memristor in the $i$th column. The value of $G_{\mathrm{g}}$ is selected according to requirements and ranges between the maximum and minimum conductance values of the memristor. By adjusting the resistance value of the memristor, positive, negative, and zero weights can be realized. The mathematical expression for weight adjustment is

$$\begin{cases} G_{\mathrm{Max}} = \mathrm{Max}\, G_{\mathrm{m}} - G_{\mathrm{g}}, \\ G_{\mathrm{Min}} = \mathrm{Min}\, G_{\mathrm{m}} - G_{\mathrm{g}}. \end{cases} \tag{10}$$

This weight adjustment method allows the neural network to dynamically adjust synaptic weights during operation to adapt to different input patterns and learning tasks, thus improving the flexibility and performance of the network.

This weight adjustment method adjusts the resistance of the memristor by varying the duration of the applied pulse to achieve the desired weight value. As shown in Fig. 2, after applying a reverse voltage of $-0.4$ V and a forward voltage of 0.43 V to the memristor, the voltage across the memristor reaches 0.83 V, which exceeds the positive threshold voltage of the memristor. At this point, the resistance value of the memristor begins to change, gradually decreasing from the initial 550 Ω, resulting in a gradual increase in the current through the memristor. After 0.25 s the reverse voltage is removed, and the voltage across the memristor is now 0.43 V, which is below the positive threshold voltage of the memristor, resulting in no further change in the resistance value of the memristor. This process demonstrates how the timing characteristics of the pulse signal can be used to accurately adjust the resistance value of the memristor, thereby achieving accurate weight modulation.

As shown in Table 1, we conducted a comparative analysis of the synaptic structure designed in this paper with the commonly used 2M design or the design based on crossbars. We quantitatively analyzed the three mainstream structures from the perspectives of area, power consumption, number of devices, and reliability.

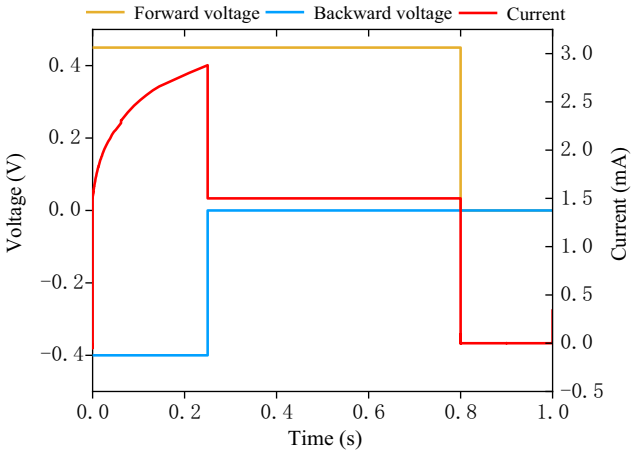



**Fig. 2.** Weight adjustment process.

**Table 1.** Performance comparison of different memristor-based synaptic structures (1M1R versus 2M versus crossbar).

| Metric | 1M1R (this work) | 2M (differential pair) | 1T1R (crossbar) |
|---|---|---|---|
| Memristors/synapse | 1 | 2 | 1 |
| Selector device | fixed resistor | fixed resistor × 2 | 1 transistor |
| Relative area | 1.0× | ∼1.8× | ∼2.5× |
| Write power | medium | high | low |
| Supports negative weights | yes | yes | no |
| Fault-tolerance flexibility | high | medium | low |

### 3.2. Activation function

An activation function is a nonlinear transformation function of a neuron that is used to introduce nonlinear properties to the output values of the neuron. In this paper, a binary activation function is used: the binary activation function converts the input values to binary output and mainly solves binary classification problems. Since the final output in this experiment is in the form of binary conversion, the binary activation function can be used to better solve the problem. The mathematical expression for the binary activation function is

$$f(x) = \begin{cases} 1, & x \geq 0, \\ 0, & x < 0. \end{cases} \tag{11}$$

When the input value $x_i$ is greater than or equal to 0, the output is 1. When the input value $x_i$ is less than 0, the output is 0. The schematic diagram of this activation function is shown in Fig. 3.

According to the principle of the binary activation function, the activation function circuit is designed for the synaptic

neuron network, as shown below:

$$V_{\mathrm{O}} = \begin{cases} 1, & V_{\mathrm{O2}} \geq V_{\mathrm{b}}, \\ 0, & V_{\mathrm{O2}} < V_{\mathrm{b}}, \end{cases} \tag{12}$$

where $V_{\mathrm{b}}$ is the comparison voltage. When the output voltage $V_{\mathrm{O2}}$ is greater than or equal to the comparison voltage $V_{\mathrm{b}}$, $V_{\mathrm{O}}$ outputs 1. When the output voltage $V_{\mathrm{O2}}$ is less than the comparison voltage $V_{\mathrm{b}}$, $V_{\mathrm{O}}$ outputs 0.

The use of a binary activation function (sign function) is primarily to align with the bipolar output characteristics of the 1M1R structure and to simplify the design of the peripheral comparator circuitry.

The core fault-tolerance mechanism is decoupled from the specific activation function: fault detection is based on the error between the target output and the actual output, and the feedback adjustment acts solely on the memristor conductance, independent of the form of the activation function. If a continuous activation function is to be employed, it would only require replacing the hard-threshold comparator in the judgment module with an analog subtractor plus an analog-to-digital converter to generate a continuous error signal for adjustment.

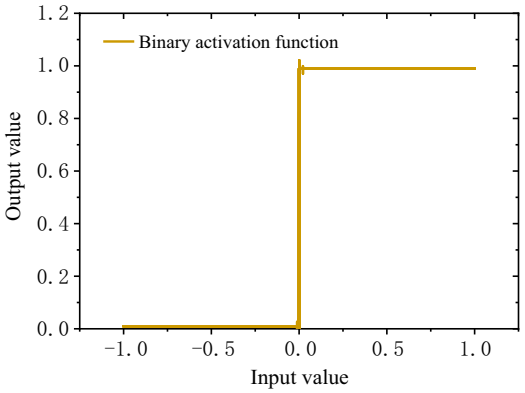



**Fig. 3.** Binary activation function.

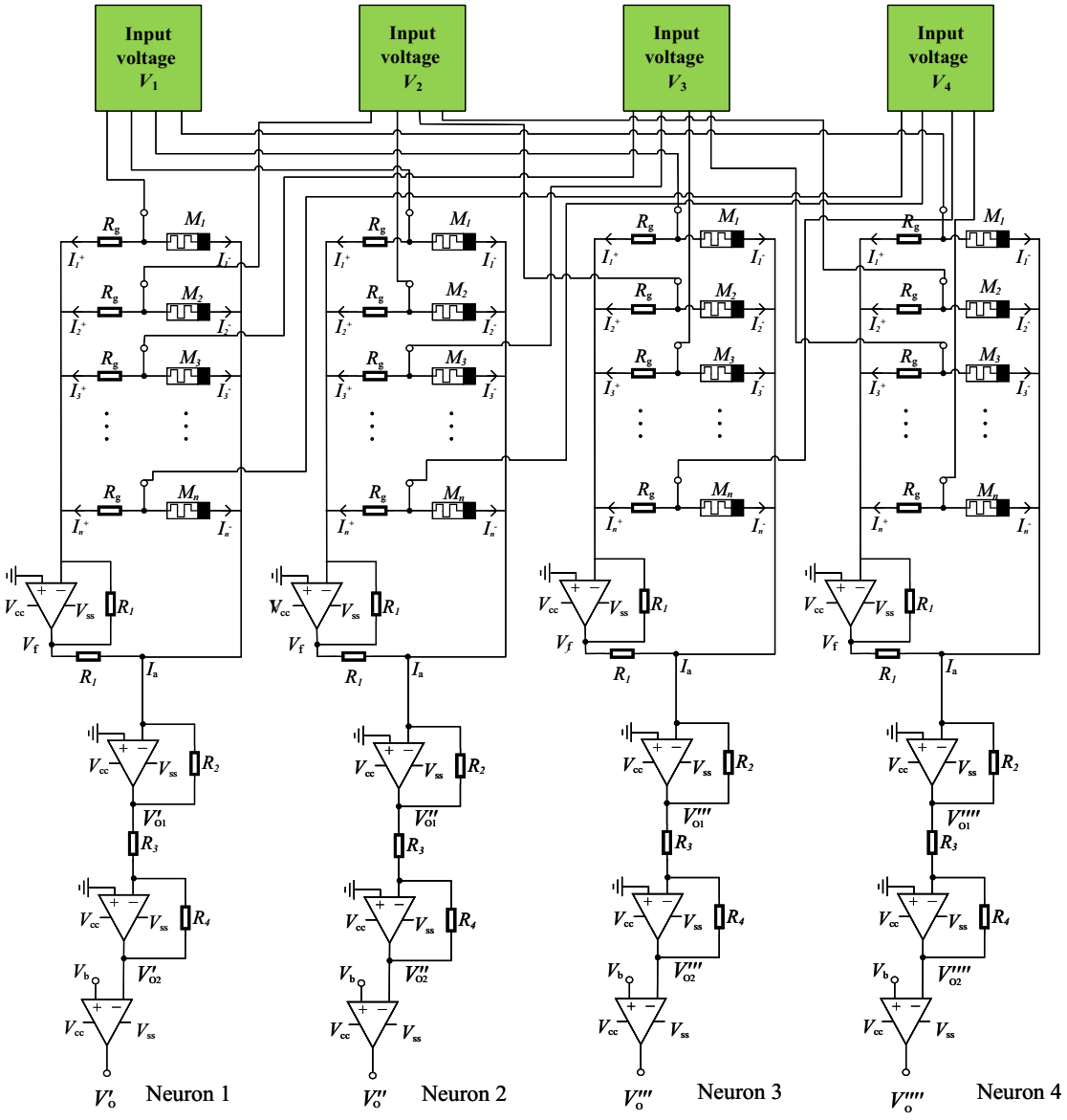



**Fig. 4.** Circuit implementation of the memristive neural network.

### 3.3. Circuit implementation of memristive neural network

As shown in Fig. 4, $V_1$, $V_2$, $V_3$, and $V_4$ represent the voltage input modules, each of which contains four input voltages for a total of 16 input voltages, corresponding to each pixel point ($X_1$–$X_{16}$) in the $4 \times 4$ image in Fig. 5.

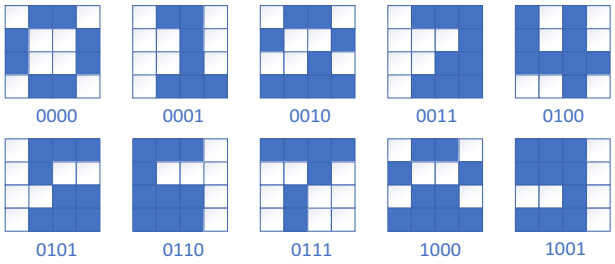



**Fig. 5.** Digital pixel points corresponding to numbers 0–9.

The output layer is binary coded with either a 0 or 1 output for each neuron. The final output value in the neural network then results from all four neurons. For example, for the digit 1 the recognized output is 0001, for the digit 2 the recognized output is 0010, and so on. This design enables the neural network to effectively identify and classify the input image data and expresses the recognition of different digits through the output results of the neurons.

To make the character recognition task more efficient and accurate, weight training of the neural network is a crucial step. The training of weights is a continuous iterative process that continues until the desired goal is reached or the loss value converges. Since the datasets are linearly separable, there exists a set of $16 \times 4$ weight matrices $W_{ij}$ that make successful classification possible. These weight values cannot be obtained by simple parsing calculations but need to be obtained by optimizing the training of the algorithm through Matlab. Figure 6 shows the results of training using the algorithm through Matlab. It can be clearly seen that the fitted values match the actual values exactly, which indicates that the neural network has completed training and can successfully implement the function of digit classification.

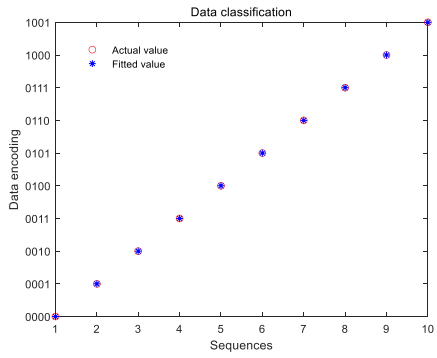



**Fig. 6.** Results of training a neural network in Matlab.

After obtaining the optimal weights in Matlab, they were imported into the circuit created in LTspice through weight mapping. The weight adjustment process took place between 0 and 0.8 s, with the specific adjustment times calculated based on the weight values. After importing the weights, inputs were applied according to the pixel values corresponding to each digit. There are a total of 16 input voltages, where 0.1 V and −0.1 V represent blue and white pixel points, respectively. Taking the input voltage for the digit 0 as an example, $X_1$–$X_4$ are −0.1 V, 0.1 V, 0.1 V and −0.1 V; $X_5$–$X_8$ are 0.1 V, −0.1 V, −0.1 V and 0.1 V; $X_9$–$X_{12}$ are 0.1 V, −0.1 V, −0.1 V and 0.1 V; and $X_{13}$–$X_{16}$ are −0.1 V, 0.1 V, 0.1 V and −0.1 V. Figure 7 shows the output of neuron 1 after applying the pixel value voltages. The blue line represents the activation voltage ($V_{O2}$) before passing through the activation function. The green line represents the output voltage ($V_O$) after the activation function. It can be observed that the activation voltages for the first eight digits are below 0 V, resulting in an output voltage of 0 V after the activation function, whereas the activation voltages for digits 8 and 9 are above 0 V, leading to an output voltage of 1 V after the activation function.

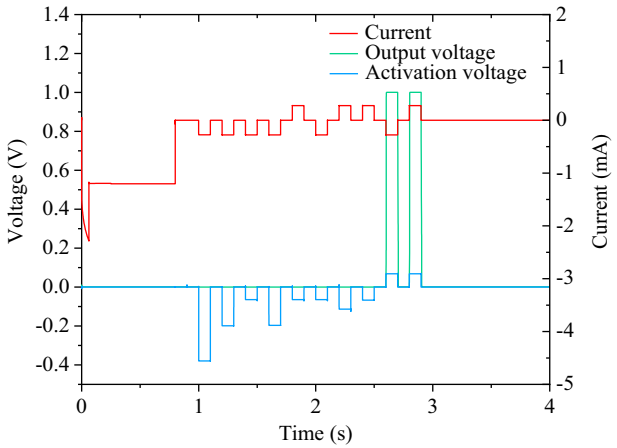



**Fig. 7.** Output image of the first column of the memristive neural network for digit recognition.

The output of the neural network is converted into corresponding voltage values, which are processed by the activation function and finally presented in binary representation. The results of pattern classification obtained by the memristive neural network are shown in Fig. 8. In each synapse unit, the first column of voltage represents the digit 0, the second column represents the digit 1, and so on, with the tenth column representing the digit 9. For example, since the binary representation of the digit 1 is 0001, the second column outputs of the four synaptic units were 0 V, 0 V, 0 V, and 1 V, respectively. As shown in Fig. 8, the circuit effectively classifies digits 0 to 9 and exhibits a basic digit recognition function.

Next, the recognition process is explained in detail using the digit 7 as an example. Figure 9 shows the recognition process. The input voltages for digit 7 are as follows: $X_1$–$X_4$ are 0.1 V, 0.1 V, 0.1 V, and 0.1 V; $X_5$–$X_8$ are −0.1 V, −0.1 V, 0.1 V, and −0.1 V; $X_9$–$X_{12}$ are −0.1 V, 0.1 V, −0.1 V, and −0.1 V; and $X_{13}$–$X_{16}$ are −0.1 V, 0.1 V, −0.1 V, and −0.1 V.

In the output results shown in Fig. 9, the eighth column of each synapse unit corresponds to the output for digit 7. The activation voltage for synapse unit 1 is −0.0678 V, and the output voltage is 0 V after processing through the activation function. The activation voltage for synapse unit 2 is 0.113 V, and the output voltage is 1 V after processing through the activation function. The activation voltage for synapse unit 3 is 0.129 V, and the output voltage is 1 V after activation function processing. The activation voltage for synapse unit 4 is 0.227 V, and the output voltage is 1 V after activation function processing. Therefore, the output for digit 7 is 0111 after passing through the activation function.

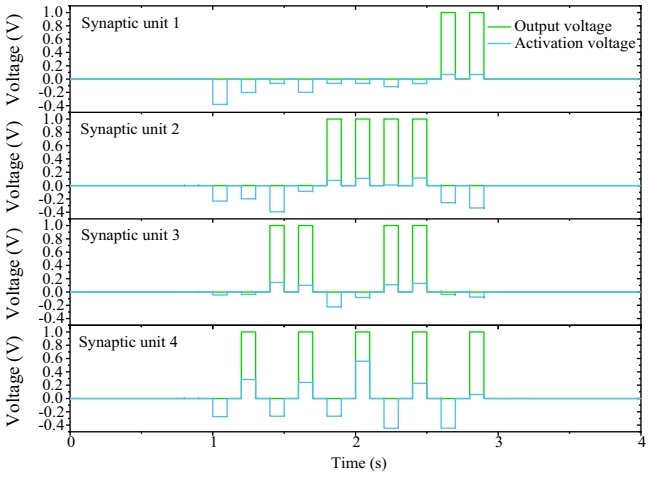



**Fig. 8.** Output of digit recognition by four synaptic units.

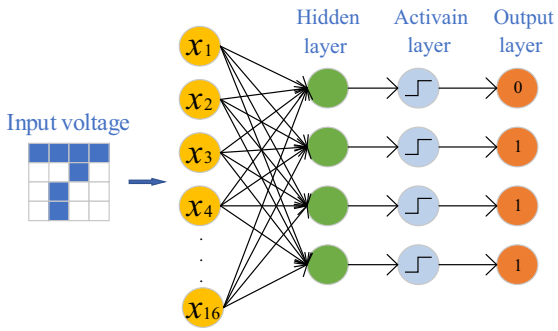



**Fig. 9.** Recognition process for digit 7.

This article validates the proposed neural network through a $4 \times 4$ small-scale digital recognition task. In the future, when expanding the input scale or designing a deeper neural network, several challenges remain regarding the complexity of the circuit, training time, and fault accumulation.

**Circuit complexity** Each neuron needs to independently judge the module (including operational amplifiers, comparators, etc.). If a network with $N$ layers and $L \times L$ neurons per layer is constructed, the number of judgment modules is $O(L_2N)$, and the area cost is significant. We propose a modular tile architecture as a solution—dividing the large network into multiple subarrays and sharing some control logic.

**Training time** The current approach employs offline Matlab training followed by LTspice mapping. When extended to deep networks, weight mapping errors could accumulate layer by layer. A hybrid training strategy is proposed for future work to address this.

**Fault accumulation** Faults in multiple layers may cascade and amplify. We refer to Ref. [48] and point out that in a network with local fault tolerance, the overall accuracy has a nonlinear characteristic with respect to the attenuation of faults.

## 4. Fault analysis and tolerance schemes for a memristive neural network

### 4.1. Simulation analysis of common faults in memristive neural networks

Memristor-based neural network circuits have demonstrated their potential for efficient neuromorphic computing. However, traditional neural networks have not adequately considered the impact of potential faults in memristors. During the fabrication process of a memristor, any deviation in materials can affect the resistance values. For example, the relationship between the length and cross-sectional area of a memristor is $\Delta R = \rho(\Delta L/A)$ and changes in cross-sectional area can influence its effective resistance $R_{\text{effective}} = R + \rho\left[L\Delta A/\left(A\left(A-\Delta A\right)\right)\right]$. Therefore, these factors can be taken into account by modeling the variations in the high-resistance state $R_{\text{H}}$ and low-resistance state $R_{\text{L}}$ of the memristor. Additionally, variations in doping concentration may alter the mobility of the charge carriers, leading to variations in threshold voltages $V_{\text{ON}}$ and $V_{\text{OFF}}$.

Moreover, two other fault types could have a significant impact on neural network circuits, namely SA0 and SA1 faults. An SA0 fault is characterized by the memristor being stuck in the high-resistance state, where conductivity reaches its minimum. An SA0 fault is usually caused by an open-circuit defect. Conversely, an SA1 fault involves the memristor being stuck in the low-resistance state, where conductivity reaches its maximum. An SA1 fault is often due to a short-circuit defect. SA0 and SA1 are hard faults, which cause the network output to deviate from the expected value, causing serious damage to the circuit performance. Repairing such faults may involve adjusting the neural network structure, retraining the network, or implementing hardware-level fixes.

Fault analysis of the circuit is performed using input digit 7. When the memristor in the neuron circuit is not faulted, the output value after inputting the voltage representing the digit 7 through the 16-bit pixel dot is 0111. Based on the previous fault analysis, the faults can be modeled as $R_{\text{H}}/R_{\text{L}}$ faults, $V_{\text{ON}}/V_{\text{OFF}}$ faults and SA0/SA1 faults.

First, $R_{\text{H}}/R_{\text{L}}$ faults were considered, where the high- and low-resistance values without faults were 550 Ω and 175 Ω. When $R_{\text{H}}$ faults occurred, the high resistance was simulated to change to 500 Ω and 600 Ω. These two faults were randomly injected into the neural network with a 50% probability. The experimental results are shown in Figs. 10 and 11.

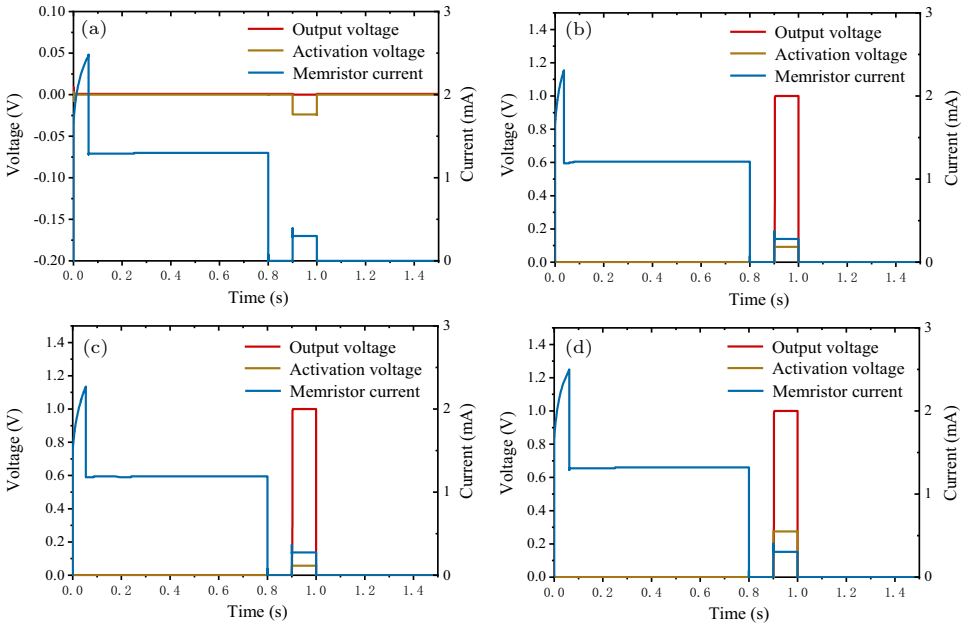



**Fig. 10.** Fault analysis when $R_{\mathrm{H}}$ is 500 Ω: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3; (d) output of synaptic unit 4.

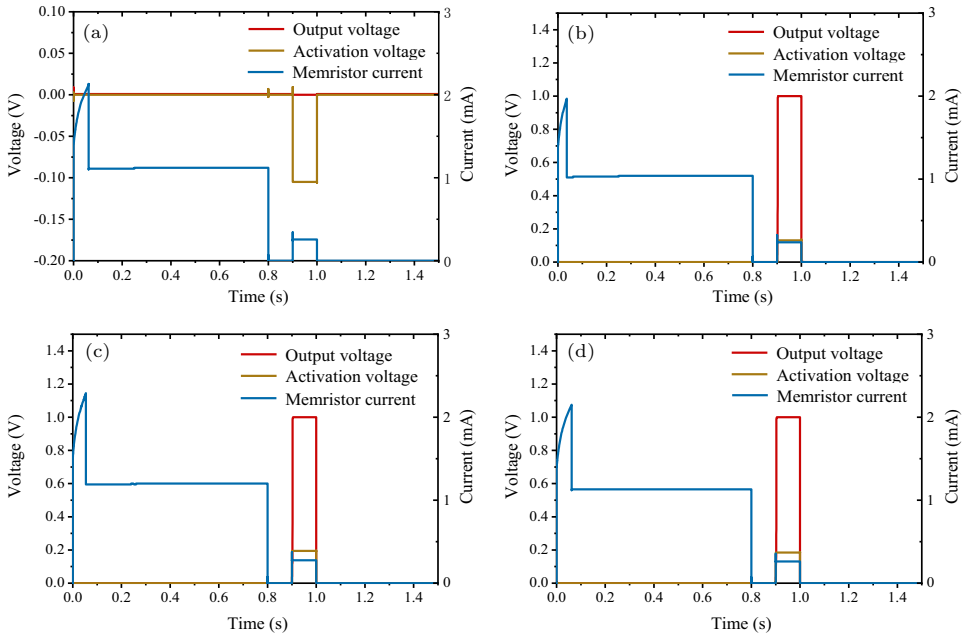



**Fig. 11.** Fault analysis when $R_{\mathrm{H}}$ is 600 Ω: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3; (d) output of synaptic unit 4.

To verify the robustness of the circuit under extreme conditions, we chose a 50% high fault injection rate. This was mainly based on the following three considerations.

Firstly, in actual deployments, there may be extreme scenarios where the local failure rate is much higher than the average. A 50% failure rate is used to simulate a harsh environment where more than half of the synapses fail in order to test the circuit's ability to maintain basic functionality.

Secondly, under low failure rates, the performance differences of various fault-tolerant schemes are not obvious. However, when the failure rate reaches 50%, the performance of the benchmark network without fault-tolerant design usually deteriorates sharply (for example, the accuracy drops to the level of random guessing). In this scenario, if our solution still maintains high accuracy, it can form a clear contrast and effectively prove its effectiveness.

Finally, in the field of memristor fault-tolerance research, it is common practice to test with a 30%–60% high failure rate. Choosing 50% as the unified test point is convenient for horizontal comparison with similar works and conforms to the general evaluation conventions of this subfield.

After injecting 50% $R_{\mathrm{H}}$ faults, it can be seen that although the activation voltage changed compared with before the fault injection, the output voltage after passing through the activa-

tion function remained 0111. The final output of the neural network was not affected by the fault, thus verifying that the neural network we designed is inherently resistant to interference, which ensures that the correct output is maintained when there is a small range of noise or anomalies in the input data.

When a pulse much larger than the threshold voltage is applied to the memristor, it can cause a permanent change in the high-resistance value of the memristor, resulting in an altered resistance range. When 50% of the high-resistance faults ($R_{\mathrm{H}} = 300\ \Omega$) are injected into the neural network, the significant change in the high resistance of the memristor leads to errors in the weight adjustment results. As shown in Fig. 12, by comparing the output voltages with the output voltages when there is no fault, it can be observed that, after the faults occur, the output result of the four synapse units changes to 1001, which does not match the original output of 0111. In other words, under fault conditions, when the input is the digit 7, the output result is 9.

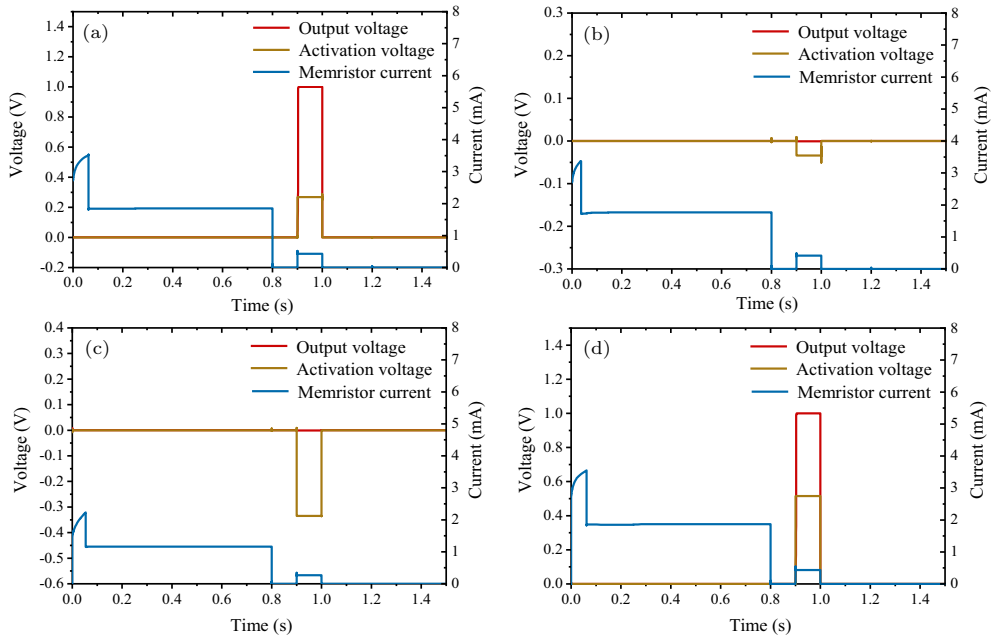



**Fig. 12.** Fault analysis when $R_{\mathrm{H}}$ is 300 Ω: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3; (d) output of synaptic unit 4.

When a memristor experiences faults, its low-resistance value may also change. This can cause distortion in the weights associated with the faulty memristor, leading to the loss or corruption of information stored in the device. Consequently, previously learned patterns and weight memories may be affected, ultimately reducing the overall performance of the network. In the fault simulation, 50% of low-resistance faults ($R_{\mathrm{L}} = 50\ \Omega$) were injected into the neural network. As shown in Fig. 13, it can be observed that the output voltage of synapse unit 1 is 0 V, the output voltage of synapse unit 2 is 1 V, the output voltage of synapse unit 3 is 0 V, and the output voltage of synapse unit 4 is 1 V. Thus, the output after the fault is 0101.

During the fabrication process, slight changes in the thickness or microstructure of the metal film in a thin-film memristor can affect its positive and negative threshold voltages. Additionally, in operation, environmental factors such as temperature, stress, and high-intensity use can impact the electrochemical performance of the memristor, which may lead to changes in the positive and negative thresholds of the memristor. When these changes occur, the impact on a trained neural network can be significant. The following is a simulation analysis of fault types caused by changes in positive and negative threshold voltages. First, 50% of negative threshold voltage faults are injected into the neural network. The original negative threshold voltage of the memristor is $-0.9$ V, but when a fault occurs, it is simulated to change to $-0.5$ V. As shown in Fig. 14, the output voltage of synapse unit 1 becomes 1 V, the output voltage of synapse unit 2 is 0 V, the output voltage of synapse unit 3 is 0 V, and the output voltage of synapse unit 4 is 1 V. Therefore, when the negative threshold voltage fault occurs, the final output of the neural network is 1001.

The positive threshold voltage of the memristor in the normal state is 0.7 V. A fault where the positive threshold voltage changes to 1 V is simulated. Fifty percent of the positive threshold voltage faults are injected into the neural network. When the normal input pulse voltage representing the digit 7 is applied at the input, Fig. 15 shows that after the activation function, the output voltage of synapse unit 1 is 0 V, the output voltage of synapse unit 2 is 1 V, the output voltage of synapse unit 3 is 1 V, and the output voltage of synapse unit 4 is 0 V. Therefore, when the positive threshold voltage changes to 1 V due to the fault, the final output of the neural network changes from the original 0111 to 0110.

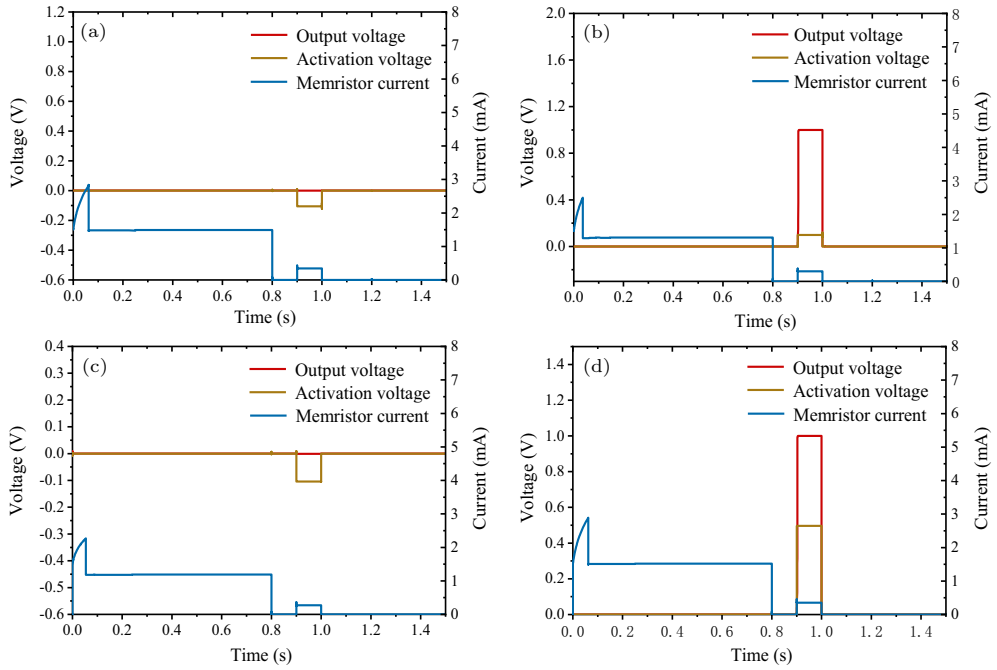



**Fig. 13.** Fault analysis when $R_L$ is 50 Ω: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3; (d) output of synaptic unit 4.

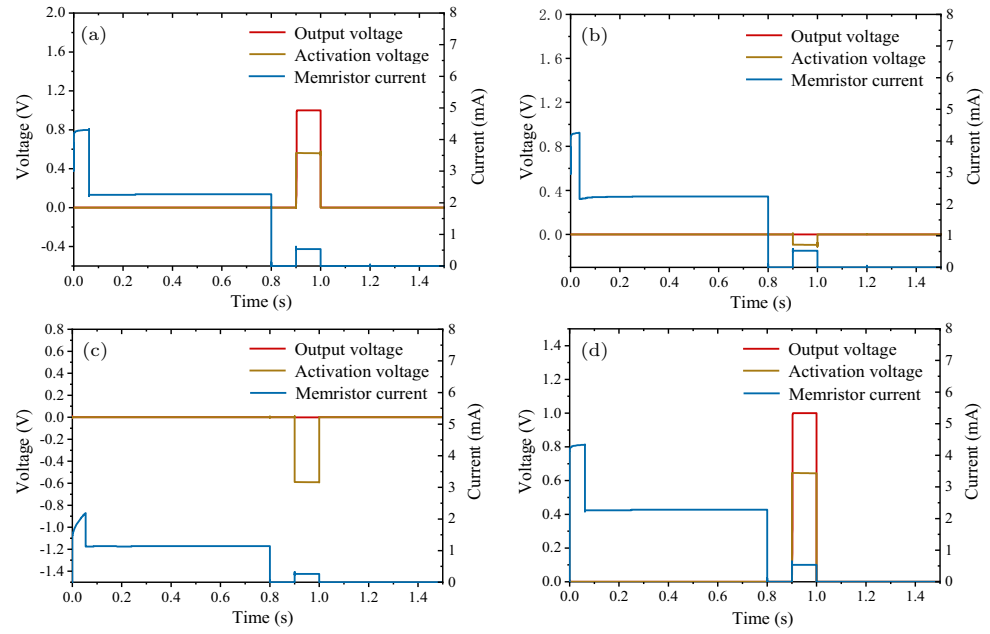



**Fig. 14.** Fault analysis when $V_{OFF}$ is −0.5 V: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3; (d) output of synaptic unit 4.

In memristors, SA0 and SA1 represent being stuck in a high-resistance state and a low-resistance state, respectively. SA0 and SA1 faults can be caused by various factors, including manufacturing defects, wear and aging, temperature and environmental effects, and excessively high or low voltages.

To analyze SA0 and SA1 faults, we model the faults by injecting 50% SA0 faults into the neural network. Due to the SA0 faults, the weights stored in the memristors are altered. As shown in Fig. 16, after randomly injecting 50% SA0 faults into the neural network, the output voltage of synapse unit 1 becomes 0 V, synapse unit 2 outputs 1 V, synapse unit 3 outputs 1 V, and synapse unit 4 outputs 0 V. Therefore, the final output of the neural network changes from the original 0111 to 0110.

In addition to the above-mentioned changes, the recognition accuracy of this neural network has also increased accordingly. After applying the proposed fault-tolerant scheme under 50% SA0 fault injection, the recognition accuracy was restored from 41.2% to 98.5%, with an average single-fault recovery time of approximately 1.2 ms. This figure clearly illustrates the entire process from fault occurrence, through error detection and feedback adjustment, to state recovery, demonstrating that the fault-tolerance mechanism can effectively restore system functionality even under extreme fault densities.

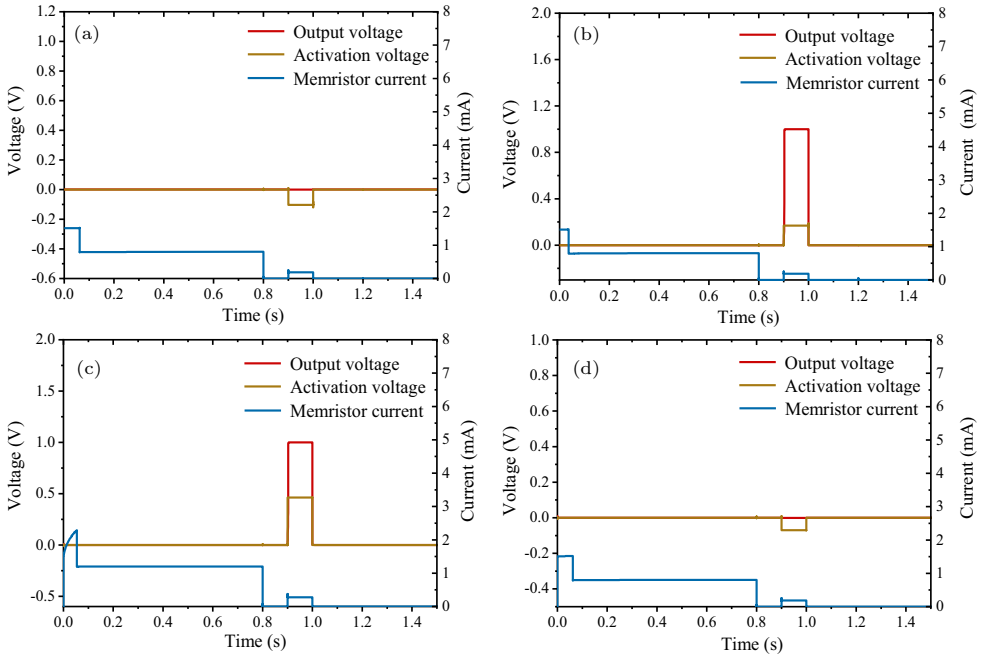



**Fig. 15.** Fault analysis when $V_{ON}$ is 1 V: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3; (d) output of synaptic unit 4.

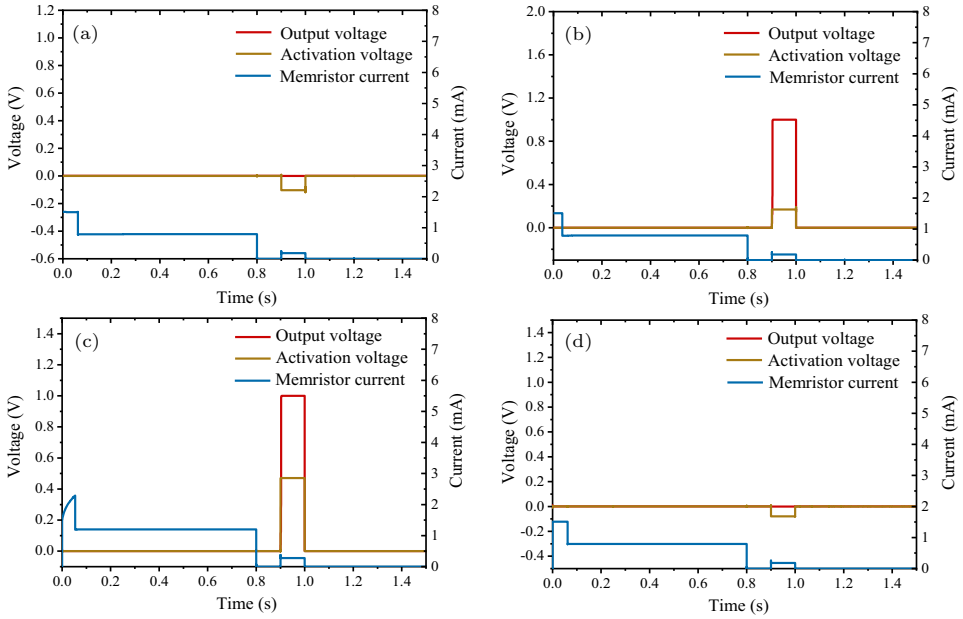



**Fig. 16.** Fault analysis when 50% SA0 faults occur: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3; (d) output of synaptic unit 4.

The proposed scheme can achieve stable weight mapping and dynamic adjustment, ensuring robustness during long-term operation.

### 4.2. Neuronal circuits with adaptive fault tolerance

Due to faults in the memristors, the originally trained neural network produced computational errors. To ensure the accuracy of neural network calculations, we propose a self-learning feedback fault-tolerant control scheme based on the *in situ* learning theory. This scheme is implemented by combining forward circuits and feedback circuits. During the process of adjusting weights using multiple fixed pulses, it effectively overcomes the problem of reduced neural network accuracy. The self-learning feedback fault-tolerant scheme based on *in situ* learning theory not only corrects computational errors caused by memristor faults but also provides an effective solution for enhancing the stability and reliability of the neural network.

The circuit implementation of this method is shown in Fig. 17 and consists of five major modules: the computation module, the signal conversion module, the judgment module, the feedback module, and the output module. After the input signal is processed through the neural network, the amplified signal $V_{O1}$ is obtained, which is then fed into an inverting cir-

cuit to generate the output $V_{O2}$. This output is then passed to the judgment module, which determines whether fault tolerance is required.

The working principle of the judgment module is shown in Fig. 18. This judgment module is composed of a subtraction circuit and a judgment circuit, including components such as voltage-controlled switches, comparators, NOT gates, and XNOR gates. The subtraction circuit is designed using an operational amplifier based on the principles of "virtual short" and "virtual open". The base voltage $V_t$ has two scenarios.

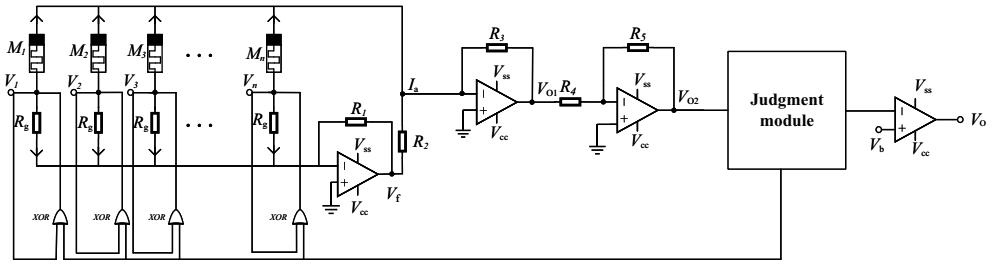



**Fig. 17.** Neuronal circuits with adaptive fault tolerance.

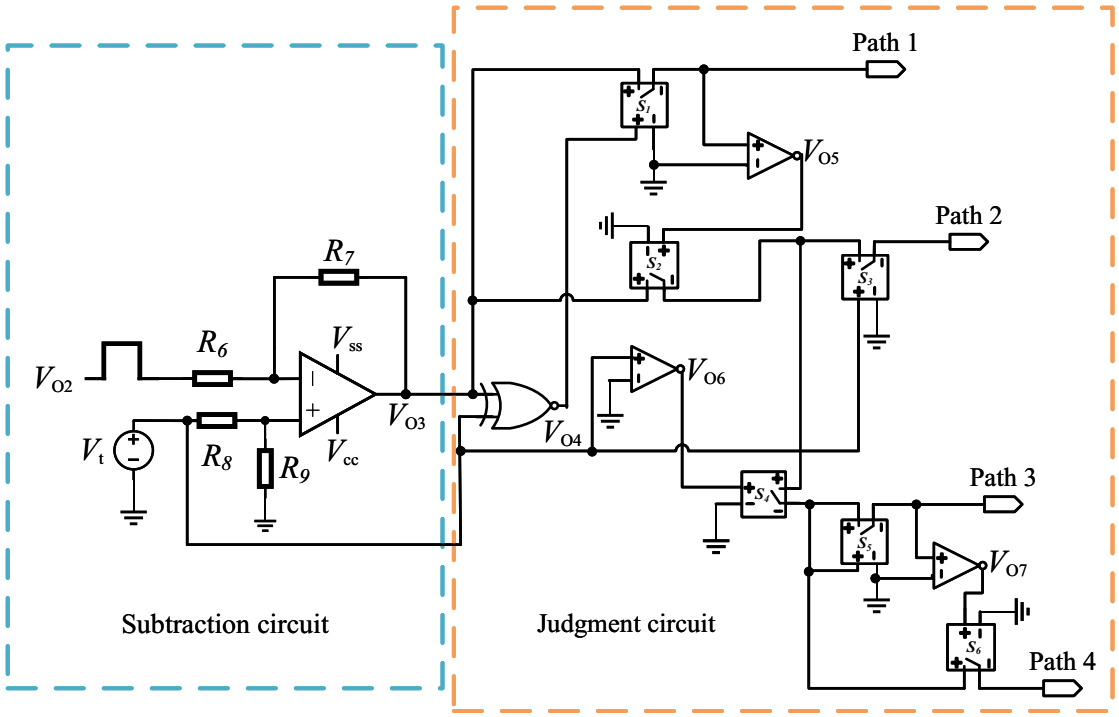



**Fig. 18.** Judgment module in fault tolerance.

For fault-free conditions, when the final output of the neural synapse unit is 1, the base voltage $V_t$ selected for the fault-tolerance process is 0.1 V. And when the final output of the neural synapse unit is 0, the base voltage $V_t$ selected for the fault-tolerance process is 0.1 V. When the signal $V_{O2}$ from the neural network activation function is input into the judgment module, it is compared with the base voltage $V_t$. The working principle of the subtraction circuit is analyzed below. When $V_{O2}$ acts alone, setting $V_t = 0$ makes it equivalent to an inverse proportional arithmetic circuit, yielding the output voltage $V_{O31}$ generated by $V_{O2}$,

$$V_{O31} = -\frac{R_7}{R_6} \times V_{O2}. \tag{13}$$

When $V_{O2}$ acts alone, setting $V_t = 0$ makes it equivalent to an in-phase proportional arithmetic circuit, yielding the output voltage $V_{O32}$ generated by $V_t$

$$V_{O32} = \left(1 + \frac{R_7}{R_6}\right) \times U_+. \tag{14}$$

The voltage value of $U_+$ is equal to the voltage division of $V_t$ by $R_8$ and $R_9$, which can be expressed as

$$U_+ = \frac{R_9}{R_8 + R_9} \times V_t. \tag{15}$$

Substituting $U_+$ into Eq. (14) gives

$$V_{O32} = \left(1 + \frac{R_7}{R_6}\right) \times \frac{R_9}{R_8 + R_9} V_t. \tag{16}$$

The output voltage $V_{O3}$ is the result of the simultaneous action of the input voltages $V_{O2}$ and $V_t$. Let $R_6 = R_7 = R_8 = R_9 = 1$ kΩ; it can be obtained that $V_{O3}$ is

$$V_{O3} = V_t - V_{O2}. \tag{17}$$

When the base voltage is 0.1 V, the correct output of the circuit should be 1 when there is no fault, and the output of the erroneous result after the fault occurs is 0. The judgment is made according to the difference between the input voltage $V_{O2}$ and the base voltage $V_t$, which is $V_{O3}$. If the circuit erroneously outputs 0, fault-tolerant adjustment is required. At this time, $V_{O3}$ and $V_t$ are greater than the threshold voltage of the XNOR gate, which is 0.02 V, and the output of the XNOR gate is high. At this time, the voltage-controlled switch $S_1$ is closed, and the signal is output from path 1. If the circuit correctly outputs 1, there is no need for fault tolerance. At this time, $V_{O3}$ is less than 0.02 V, the output of the XNOR gate $V_{O4}$ is low, the voltage-controlled switch $S_1$ is open, and the output voltage $V_{O5}$ through NOT gate 1 is high. At this time, the voltage-controlled switches $S_2$ and $S_3$ are closed, and the signal is output from path 2.

When the base voltage is $-0.1$ V, the correct output of the circuit should be 0 when there is no fault, and the output of the error result after the fault occurs is 1. If the circuit correctly outputs 0, there is no need to carry out fault tolerance. At this time, $V_{O3}$ is greater than the threshold voltage of the XNOR gate, and the switches $S_1$, $S_3$, and $S_6$ are open, while $S_2$, $S_4$, and $S_5$ are closed. The signal is directly output through path 3. If the circuit erroneously outputs 1, it is necessary to carry out fault tolerance. At this time, $V_{O3}$ is less than the threshold voltage of the XNOR gate. At this time, $S_5$ is open and $S_6$ is closed, allowing the signal to be output through path 4 for feedback fault tolerance. The overall process is as follows:

(1) Input the fault mode T1 (digit 0) as the first input sample to the neural network and calculate the output voltage of the four neurons.

(2) Fault detection is performed for each synaptic unit by the judgment module, and judgment signals are obtained through pathways 1–4.

(3) Convert the judgment signals obtained in the previous step into weight adjustment voltages to modify the memristor synaptic weights.

(4) Then, input the fault mode T2 (digit 1) as the second input mode into the network for fault tolerance and repeat steps 1–3.

(5) Finally, input the fault mode T10 (digit 9) into the network for fault tolerance and repeat steps 1–3. At this point, one fault tolerance cycle is completed.

(6) Fault detection is performed again for each synaptic unit through the judgment module. If the error output is still present, the circuit returns to step 1 and performs a new fault-tolerance cycle.

For the overall fault-tolerant circuit design, there is a performance overhead required in the hardware. As shown in Table 2, we have summarized the key hardware cost indicators of fault-tolerant circuits.

**Table 2.** Performance overhead comparison of fault-tolerant schemes.

| Performance metric | Fault-free baseline | This paper fault-tolerant scheme | Absolute increase | Relative increase |
|---|---|---|---|---|
| Area /neuron | 700 μm$^2$ | 1,850 μm$^2$ | +1150 μm$^2$ | +164% |
| Inference power (static + dynamic) | 8.2 μW | 12.1 μW | +3.9 μW | +48% |
| Peak power during fault tolerance phase | – | 380 μW | – | – |
| Average power (fault tolerance triggered once per 1000 inferences) | 8.2 μW | ≈ 8.24 μW | +0.04 μW | < 1% |
| Inference latency | 42 ns | 47 ns | +5 ns | +12% |
| Single fault tolerance cycle latency | – | ≈ 1.2 ms | – | – |

The primary latency during the fault-recovery process originates from the physical write time of the memristor, which is a limitation inherent to the device itself, not the circuit design. If faster memristors are employed, the fault-recovery cycle can be shortened to the order of hundreds of microseconds. Despite the increases in area and instantaneous power, the core advantages of this scheme lie in: achieving purely analog-domain autonomous recovery without the need for a digital co-processor or external memory; preventing complete chip failure due to a small number of faults; and significantly enhancing the long-term reliability of edge AI devices. For application scenarios that demand high reliability and low maintenance costs — such as Internet of things endpoints, implantable medical devices, and aerospace electronics — this level of overhead is acceptable.

### 4.3. Simulation analysis of common faults in memristive neural networks

For fault tolerance where the correct output should be 1 but the actual output is 0 due to a fault, a reference voltage of $V_t = 0.1$ V is used. When the difference $V_{O3}$ is less than 0, it indicates that the output value is greater than the reference voltage, which meets the condition for the output of 1, meaning no fault-tolerance adjustment is needed. When the difference is greater than 0, it indicates that the output value is less than the reference voltage, and there are two cases to consider: the output value is either less than 0 or it is greater than 0 but less than 0.1 V. For the second case, although it also meets the judgment condition for the output of 1 ($V_{OUT} > 0$), the system is not stable in this state, as any slight variation could affect the final output. Therefore, we also classify the case where the

output value is greater than 0 but less than 0.1 V as requiring fault tolerance.

For fault tolerance where the correct output should be 0 but the actual output is 1 due to a fault, a reference voltage of $V_t = 0.1$ V is used. When the difference $V_{O3}$ is greater than 0, it indicates that the output value is less than the reference voltage, which meets the condition for the output of 0, meaning no fault-tolerance adjustment is needed. When the difference is less than 0, there are also two cases to consider: the output value is either greater than 0 or it is less than 0 but greater than $-0.1$ V. Although the second case satisfies the condition for the output of 0, fault-tolerance processing is still required to ensure system stability.

The specific fault-tolerance process is as follows. For the fault tolerance process when the correct output should be 1, if the output voltage is too low, the weight is adjusted to increase the output voltage. For memristors with a negative input pulse voltage, a negative voltage adjustment is used, which increases the memristor's resistance, causing the absolute value of the current flowing through the memristor to decrease. For positive inputs, a positive voltage adjustment is applied, decreasing the resistance of the memristor, which increases the current flowing through the memristor and consequently increases the overall output.

Conversely, for the fault-tolerance process when the correct output should be 0, if the output voltage is too high, the weight is adjusted to reduce the output voltage. For negative inputs, positive voltage adjustment is applied, decreasing the memristor's resistance, which increases the absolute value of the current flowing through the memristor. For positive inputs, negative pulse adjustment is applied, increasing the memristor's resistance, which reduces the current flowing through the memristor, ultimately decreasing the current output.

### 4.4. Analysis of fault-tolerant results

According to the relationship between the adjustment voltage, input voltage, and $V_{O3}$ in Table 3, fault-tolerant simulations are performed for each type of fault. In the previous fault analysis, it was found that when the high-resistance value $R_H$ of the memristor changed from the original 550 Ω to 300 Ω due to a fault, the final output of the neural network changed from 0111 to 1001. This means that synaptic units 1, 2, and 3 produced erroneous results due to the fault. The proposed feedback fault-tolerant method based on *in situ* learning is used to correct the faulty synaptic units.

**Table 3.** Correspondence of regulation voltage in the fault-tolerance process.

| Reference voltage $V_t$ | Input pulse voltage | $V_{O3}$ (difference) | Adjustment of pulse voltage | Value of memristor resistance | Output result |
|---|---|---|---|---|---|
| $V_t = 0.1$ V | +0.1 V | $V_{O3} > 0$ | positive pulse (1.5 V) | decrease | increase |
| | −0.1 V | $V_{O3} > 0$ | negative pulse (−1.5 V) | increase | decrease (absolute value) |
| $V_t = -0.1$ V | +0.1 V | $V_{O3} < 0$ | negative pulse (−1.5 V) | increase | increase |
| | −0.1 V | $V_{O3} < 0$ | positive pulse (1.5 V) | decrease | decrease (absolute value) |

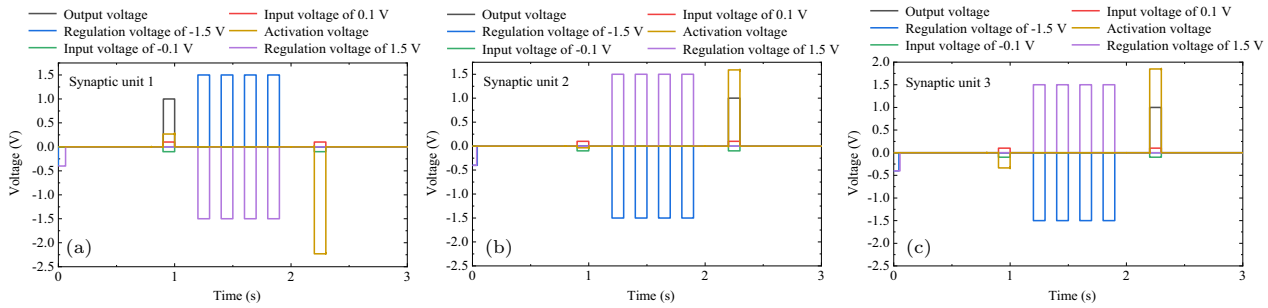



**Fig. 19.** Fault-tolerant process when $R_H$ is 300 Ω: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3.

Figure 19(a) shows the fault-tolerant process of synaptic unit 1. Between 0.9 s and 1 s, the activation voltage of synaptic unit 1 was greater than 0 V, and the output voltage after the activation function was 1 V, while the normal output voltage of synaptic unit 1 should be 0 V. According to Table 3, between 1.2 s and 1.9 s, a regulation voltage of −1.5 V was applied to the memristor with an input voltage of +0.1 V, and a regulation voltage of +1.5 V was applied to the memristor with an input pulse of −0.1 V.

In synaptic units 2 and 3, the fault caused the activation voltage to decrease between 0.9 s and 1 s, requiring regulation voltages to increase the activation voltage of these synaptic units. Between 1.2 s and 1.9 s, a regulation voltage of +1.5 V was applied to the memristor with an input voltage of +0.1 V, and a regulation voltage of −1.5 V was applied to the memristor with an input pulse of −0.1 V.

Between 2.2 s and 2.3 s, the input voltage representing the digit 7 was rewritten to the synaptic units, and it can be seen that the final output voltage of these synaptic units after the fault-tolerant process was corrected to 1 V.

In the previous fault analysis, when the low-resistance value $R_L$ changed from 175 Ω to 50 Ω, only synaptic unit 3

produced an erroneous output. Therefore, only synaptic unit 3 needs fault-tolerant processing, and the fault-tolerant results are shown in Fig. 20. During the time period of 0.9–1 s, the output voltage should be 1 V without the fault, but the output voltage after the fault occurred was 0 V. In this case, the fault-tolerant method with a reference voltage of $V_t = 0.1$ V was selected.

According to Table 3, between 1.2 s and 1.9 s, a regulation voltage of +1.5 V was applied to the memristor with an input voltage of +0.1 V, and a regulation voltage of −1.5 V was applied to the memristor with an input voltage of −0.1 V. After the fault-tolerant process was completed, during the time period from 2.2 s to 2.3 s, the input voltage representing the digit 7 was re-entered into synaptic unit 3. The results showed that the output voltage after the fault-tolerant process was corrected to 1 V.

Under normal conditions, the output voltages of synaptic unit 1, synaptic unit 2, and synaptic unit 3 should be 0 V, 1 V, and 1 V, respectively. However, when a fault occurred, causing $V_{OFF}$ to change from −0.9 V to −0.5 V, the output voltages of synaptic units 1, 2, and 3 changed to 1 V, 0 V, and 0 V, respectively. Therefore, the proposed fault-tolerant method needs to be applied to these synaptic units for fault-tolerant processing.

As shown in Fig. 21, during the period from 1.2 s to 1.9 s, a regulation voltage of −1.5 V was applied to the memristor in synaptic unit 1 with an input voltage of +0.1 V, and a regulation voltage of +1.5 V was applied to the memristor with an input voltage of −0.1 V. Similarly, a regulation voltage of +1.5 V was applied to the memristors in synaptic units 2 and 3 with input voltages of +0.1 V, and a regulation voltage of −1.5 V was applied to the memristors with input voltages of −0.1 V.

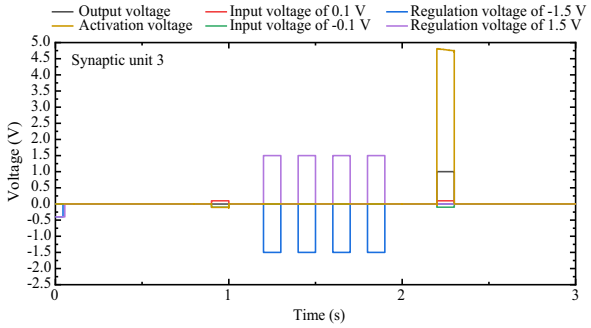



**Fig. 20.** The output of the fault-tolerant process for synapse unit 3 when $R_L$ is 50 Ω.

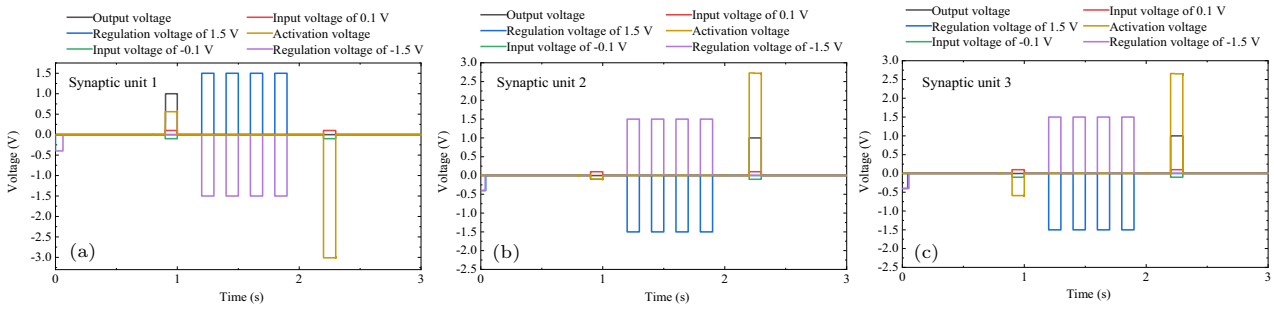



**Fig. 21.** Fault-tolerant process when $V_{OFF}$ is −0.5 V: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3.

After applying the regulation voltages for fault tolerance, during the time period from 2.2 s to 2.3 s, the input voltages representing the digit 7 were re-entered. The results showed that the output voltage of synaptic unit 1 returned to 0 V while the output voltages of synaptic units 2 and 3 returned to 1 V, successfully completing the fault-tolerant process.

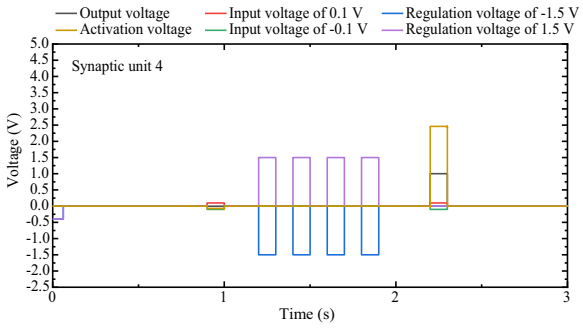



**Fig. 22.** The output of the fault-tolerant process for synapse unit 4 when $V_{ON}$ is 1 V.

When $V_{ON}$ changes from 0.7 V to 1 V due to the fault, only synaptic unit 4 experiences an error, and the output voltage changes from 1 V to 0 V because the synaptic unit itself has a certain degree of fault tolerance. As shown in Fig. 22, during the period from 1.2 s to 1.9 s, a regulation voltage of 1.5 V is applied to the memristor with an input voltage of 0.1 V, and −1.5 V is applied to the memristor with an input voltage of −0.1 V. Between 2.2 s and 2.3 s, after applying the input voltage representing the digit 7, the final output voltage is 1 V, indicating the completion of the fault-tolerant process.

When 50% SA0 faults occur in the neural network, as previously analyzed, only synapse unit 4 exhibits an erroneous output. Under normal conditions, the output of synapse unit 4 should be 1 V, but it changes to 0 V after the fault. During the fault-tolerant process, a regulation voltage of 1.5 V is applied to the memristor with an input voltage of 0.1 V, and −1.5 V is applied to the memristor with an input voltage of −0.1 V, as shown in Fig. 23. Between 2.2 s and 2.3 s, the result shows that after the fault-tolerant process the output voltage of synapse

unit 4 is restored to 1 V.

When fault SA1 occurs in the neural network, it indicates that the memristor is stuck in a low-resistance state. Based on previous analysis, synaptic units 1, 2 and 3 all exhibit incorrect outputs. For the memristor in synaptic unit 1 with an input voltage of 0.1 V, a regulation voltage of −1.5 V is applied. For the memristor with an input voltage of −0.1 V, a regulation voltage of 1.5 V is applied. In synaptic units 2 and 3, a regulation voltage of 1.5 V is applied to the memristor with an input voltage of 0.1 V, and a regulation voltage of −1.5 V is applied to the memristor with an input voltage of −0.1 V. In Fig. 24, it is observed that during the time period from 2.2 s to 2.3 s the output voltage of synapse unit 1 becomes 0 V, the output voltage of synapse unit 2 becomes 1 V, and the output voltage of synapse unit 3 becomes 1 V, indicating that after the fault-tolerant process each synaptic unit has returned to its normal output.

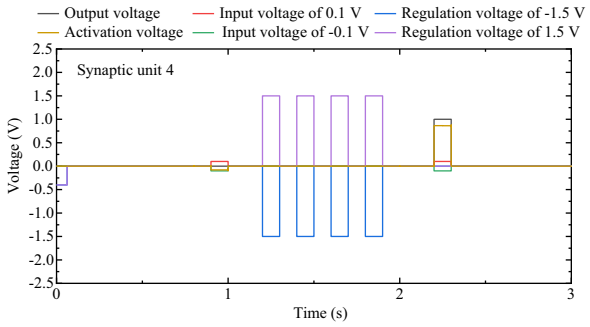



**Fig. 23.** The output of the fault-tolerant process for synapse unit 4 when 50% SA0 faults occur.

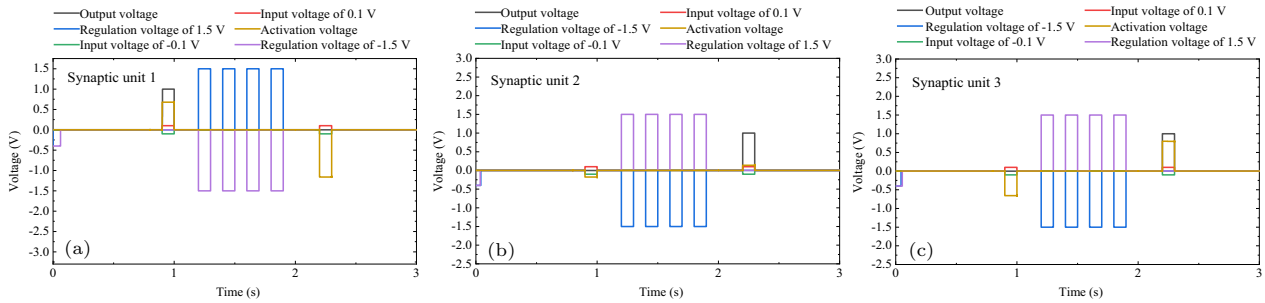



**Fig. 24.** Fault-tolerant process when 50% SA1 faults occur: (a) output of synaptic unit 1; (b) output of synaptic unit 2; (c) output of synaptic unit 3.

The proposed method uses input regulation pulses to modify weights, ensuring that the output voltage in each case can be adjusted to the ideal range. The experimental results above fully validate the feasibility and effectiveness of the method. By analyzing the fault-tolerance process for each fault, it is clear that the fault-tolerance scheme proposed in this section can effectively address the problem of accuracy degradation caused by memristor faults.

### 4.5. Comparison with previous works

Table 4 compares this work with other fault-tolerance methods in neural networks. Reference [46] proposed a memristive neural network circuit, which solved the problem of recognizing pictures with three types of characters but did not analyze the impact of potential memristor faults on the circuit. Reference [47] proposed a very lightweight approach that exploits a statistical property present in the activation data of a neural network layer to address the impact of non-idealities. However, this method only addresses the impact of SA0 and SA1 faults and does not consider other types of faults. Reference [48] proposed a bit-aware fault-tolerant hybrid retraining and remapping scheme, but the retraining and remapping can be more costly. Reference [49] proposed an efficient mapping method that solves the problem of SAF as well as the variation of the memristor, but it ignored the potential advantages of the circuit implementation.

**Table 4.** Comparison between different fault-tolerance methods.

| | Circuit implementation | Address the SA1 and SA0 faults | Address the resistance state faults | Address the threshold voltage faults |
|---|---|---|---|---|
| [46] | √ | × | × | × |
| [47] | × | √ | × | × |
| [48] | × | √ | × | × |
| [49] | × | √ | √ | × |
| Work in this paper | √ | √ | √ | √ |

Table 5 provides a clear comparison between the work presented in this paper and other related studies in terms of fault types, fault-tolerance mechanisms, circuit complexity, and performance differences. As shown in Table 5, this work achieves a unique balance in the following aspects: it simultaneously addresses hard faults, parameter drift, and device variability, rather than targeting only a single type of fault; it is the first to achieve real-time fault tolerance entirely within the analog domain without requiring the involvement of any digital system, making it suitable for fully analog edge intelligence chips; it achieves an accuracy of 98% even under an extreme fault rate of 50%, outperforming existing solutions; and it avoids the area doubling caused by redundancy while circumventing the computational overhead of retraining and the storage requirements of mapping tables.

**Table 5.** Comparative analysis of state-of-the-art fault-tolerant memristive systems.

| Work | Fault model | Fault-tolerance mechanism | Requires digital processor | Circuit complexity | Maximum validation scale | Accuracy recovery capability |
|---|---|---|---|---|---|---|
| [12] | SA0/SA1 | redundant backup | no | high | $4\times4$ | 40% → 95% |
| [28] | parameter drift | online retraining | yes | low | MNIST | 70% → 92% |
| [28] | variability | weight remapping | yes | medium | CIFAR-10 | 65% → 88% |
| Work in this paper | SA0/SA1 + drift + variability | analog-domain adaptive feedback | no | medium | $4\times4/8\times8$ | 40% → 98% |

**Table 6.** Comparison of fault-tolerance strategies in memristive neural networks.

| Feature | Redundancy-based approach | Algorithm-level fault tolerance | Proposed method |
|---|---|---|---|
| Fault-tolerance mechanism | hardware redundancy (e.g., 2M differential cells) | software retraining or weight remapping | *in situ* analog feedback repair |
| Increases memristor count | yes | no | no |
| Requires digital processor | no | yes | no |
| Can dynamically repair faults | no | yes | yes |
| Target application | high-reliability systems with relaxed area/power constraints | hybrid systems with digital backend support | pure analog edge device |

This paper designs a memristor-based digital recognition neural network circuit and analyzes the potential faults that may occur in memristors within this network. Simultaneously, a self-learning feedback fault-tolerant control scheme is proposed, which not only addresses the problems of internal parameters of the memristor, such as high- and low-resistance states, threshold voltage changes, and stuck-at faults, but also provides a new research approach for neural networks to achieve hardware fault-tolerant control.

This is different from hardware-level fault tolerance based on redundancy and algorithm-level fault tolerance based on re-training/mapping. Table 6 shows the differences between the fault-tolerance scheme of this paper and the other two schemes.

## 5. Conclusion

A memristive neural network circuit with fault tolerance is an innovative design. Such a circuit not only enables digital character recognition but also realizes adaptive fault tolerance in the presence of memristor faults. In this paper, a memristor synapse array with a 1M1R structure is designed, where the mapping of weights is achieved by adjusting the relationship between memristors and resistors, thereby reducing the number of memristors required. This structure is connected to an activation function to implement character recognition in the perception neural network. The trained weights were imported into the LTspice circuit for simulation, and the results showed that the constructed perception neural network circuit successfully achieved character recognition functionality. A thorough analysis of potential memristor faults was conducted, and an adaptive feedback fault-tolerant circuit based on *in situ* learning was designed to adjust the weights of the memristors, ensuring that the neural network could operate normally even in the presence of faults. This scheme provides a new idea for the development of neural network circuits and holds significant reference value for similar fault studies. It is hoped that this design can be further improved in the future to play a greater role in practical applications.

## Acknowledgments

Project supported by the Natural Science Foundation of Shandong Province (Grant No. ZR2022MF225) and the National Natural Science Foundation of China (Grant Nos. 62176143 and 62371275).

## References

[1] Wang X Y, Zhou P F, Eshraghian J K, Lin C Y, Iu H H C, Chang T C and Kang S M 2021 *IEEE Trans. Circuits Syst. I-Regul. Pap.* **68** 264
[2] Rao M Y, Tang H, Wu J B, Song W H, Zhang M, Yin W B, Zhuo Y, Kiani F, Chen B J M, Jiang X Q, Liu H F, Chen H Y, Midya Y, Ye F, Jiang H, Wang Z R, Wu M C, Hu M, Wang H, Xia Q F, Ge N, Li J and Yang J J 2023 *Nature* **615** 823
[3] Chua L 1971 *IEEE Trans. Circuit Theory* **18** 507
[4] Strukov D B, Snider G S, Stewart D R and Williams R S 2008 *Nature* **453** 80
[5] Karimi A and Rezai A 2019 *Int. J. Circuit Theory Appl.* **47** 1933
[6] Rziga F O, Mbarek K, Ghedira S and Besbes K 2019 *J. Comput. Electron.* **18** 1055
[7] Mozaffari S N, Tragoudas S and Haniotakis T 2017 *IEEE Trans.* **36** 1018
[8] Singh A 2020 *IETE J. Res.* **66** 182
[9] Papandroulidakis G, Serb A, Khiat A and Merrett G V 2019 *IEEE Trans. Circuits Syst. I-Regul. Pap.* **66** 3041
[10] Gul F 2019 *IEEE Electron Device Lett.* **40** 643
[11] Guo M, Zhang X W, Dou G and Iu H H C 2025 *IEEE Trans. Circuits Syst. I-Regul. Pap.* **72** 7120
[12] Lee J Y, Han J Y, Kang B X, Hong J Y, Lee C Z and Jeon I 2025 *Adv. Mater.* **37** 2413916
[13] Guo M, Zhang X W, Guo W H, Dou G, Chen D, Wang L H, Iu H H C 2025 *IEEE Trans. Circuits Syst. I-Regul.*
[14] Sun J W, Zhai Y, Liu P and Wang Y F 2025 *IEEE Trans. Neural Netw. Learn. Syst.* **36** 3618
[15] Guo M, Sun J H, Dou G and Iu H H C 2025 *IEEE Trans. Circuits Syst. I-Regul. Pap.* **72** 5998

[16] Dou G, Li D G, Guo M and Iu H H C 2025 *IEEE Trans. Circuits Syst. I-Regul. Pap.* **72** 2566
[17] Wang Y F, Tao K F, Wang Z C and Sun J W 2025 *IEEE Trans. Ind. Inform.*
[18] Sun J W, Gao P L, Liu P and Wang Y F 2025 *IEEE Trans. Ind. Inform.*
[19] Guo M, Kong L T, Dou G and Iu H H C 2024 *IEEE Trans. Circuits Syst. I-Regul. Pap.* **71** 4676
[20] Guo M, Zhang D Y, Guo W H, Dou G and Sun J W 2024 *IEEE Trans. Cogn. Dev. Syst.* **17** 155
[21] Guo M, Zheng C G, Dou G and Iu H H C A 2026 *IEEE Trans. Circuits Syst. I-Regul. Pap.* **73** 850
[22] Lei Z, Guo Y T and Ma J 2025 *Nonlinear Dyn.* **113** 29999
[23] Wang B C, Guo Y T, Jia J N and Ma J 2025 *Euro. Phys. J. Plus* **140** 706
[24] Ma J 2025 *Nonlinear Dyn.* **113** 25365
[25] Lu J L, Gao J, Cai J Y, Chen L X, Zhang G Y and Xu Q 2025 *Chin. Phys. B* **34** 018703
[26] Yan X, Wu F Q, Yao W, Zhou P and Liu Y Z 2024 *Chin. Phys. B* **33** 028705
[27] Lu J Y, Wang G Y, Xu Y, Ren G D and Ma J 2024 *Chin. Phys. B* **33** 048701
[28] Li J, Xin Y, Sun B, Gu D S, Liao C R, Hu X F, Wang L D, Duan S K and Zhou G D 2025 *J. Semicond.* **46** 012604
[29] Xiao H, Zhou Y, Gao T T, Duan S K, Chen G R and Hu X F 2023 *IEEE J. Emerg. Sel. Top. Circuits Syst.* **13** 344
[30] Zhao H, Liu Z W, Tang J S, Gao B, Qin Q, Li J M, Zhou Y, Yao P, Xi Y, Lin Y D and Wu H Q 2023 *Nat. Commun.* **14** 2276
[31] Kim N, Oh J, Kim S, Cha J H, Choi J, Im S G, Choi S Y and Jang B C 2024 *Adv. Funct. Mater.* **34** 2305136
[32] Wu X L, Dang B J, Wang H, Wu X L and Yang Y C 2021 *Adv. Intell. Syst.* **4** 2100151
[33] Yang L, Zeng Z G and Shi X M 2019 *Neurocomputing* **363** 114
[34] Chen C Y, Shih H C, Wu C W, Lin C H, Chiu P F, Sheu S S and Chen F T 2015 *IEEE Trans. Comput.* **64** 180
[35] Sun X Y and Yu S M 2019 *IEEE J. Emerg. Sel. Top. Circuits Syst.* **9** 570
[36] Zhang B G, Uysal N, Fan D L and Ewetz R 2020 *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **39** 2448
[37] Liu P, You Z Q, Wu J G, Liu B S, Han Y H and Chakrabarty K 2021 *IEEE Trans. Circuits Syst.* **68** 4444
[38] Chen C Y and Chakrabarty K 2022 *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **41** 2301
[39] Morgul M C, Frontini L, Tunali O, Anghel L, Ciriani V, Vatajelu E I, Moritz C A, Stan M R, Alexandrescu D and Altun M 2021 *IEEE Trans. Nanotechnol.* **20** 39
[40] Bi Y T, Xu Q, Geng H, Chen S and Kang Y 2023 *IEEE Trans. Circuits Syst. II-Express Briefs* **70** 2221
[41] Wang Z W, Yin M H, Zhang T, Cai Y M, Wang Y Y, Yang Y C and Huang R 2016 *Nanoscale* **8** 14015
[42] Chen A 2016 *Solid-State Electron.* **125** 25
[43] Berdan R, Prodromakis T and Toumazou C 2012 *Electron. Lett.* **48** 1105
[44] Ghofrani A, Lastras-Montano M A, Gaba S, Payvand M, Lu W, Theogarajan L and Cheng K T 2015 *J. Emerg. Technol. Comput. Syst.* **12** 1
[45] Guo M, Zhu Y L, Liu R Y, Zhao K X and Dou G 2022 *Neurocomputing* **472** 12
[46] Zhang X R, Wang X Y, Ge Z Y, Li Z L, Wu M Y and Borah S 2022 *Micromachines* **13** 2074
[47] Quan C H, Fouda M E, Lee S, Jung G J, Lee J, Eltawil A E and Kurdahi F 2023 *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **42** 2174
[48] Reddy M A, Pandeeswari R and Ko S B 2023 *IEEE Trans. Circuits Syst. II-Express Briefs* **70** 959
[49] Shi F, Wang K H, Cao Y H and Mou J A 2026 *J. Emerg. Technol. Comput. Syst.* **195** 108921