Full-Circuit Implementation of Transformer Network Based on Memristor

Chao Yang, Xiaoping Wang, *Senior Member, IEEE*, and Zhigang Zeng, *Fellow, IEEE*

***Abstract*— As an emerging in-memory element, memristor has been widely used in various neural network circuits to represent the weights and accelerate the calculation. However, the Transformer Network (TN), one of the most important models for machine vision and natural language processing in recent years, has not yet been full-circuit implemented using memristors due to the complex calculation process and data storage. In order to carry out the computation of the TN more efficiently, this work proposes a memristor-based full-circuit implementation of the TN capable of: 1) a memristor crossbar module to preserve the weights of the TN and perform the vector-matrix multiplications; 2) an analog signal memory module to store the analog signal directly in near-memory mode; 3) function circuit modules to achieve five transformations, namely Softmax, Layer Normalization, ReLU, Multiply-add and Residual; 4) a timing signal generation module to schedule operations of the circuit. The proposed TN circuit can complete all calculations directly based on the analog signal without using any analog-digital converter (ADC), digital-analog converter (DAC) and digital memory. In addition, character image recognition experiments are carried out in PSPICE to verify the functional correctness of the designed circuit. The corresponding signal retention rates of the analog memory, the performance of the whole circuit, and the non-idealities of the memristors are also analyzed. The results indicate that the circuit has advantages in terms of area overhead, energy efficiency and anti-noise.**

***Index Terms*— Memristor, transformer network, analog signal memory, circuit implementation.**

I. Introduction

IN RECENT years, with the rapid development of artificial intelligence, all walks of life have set off a storm of intelligence, more and more attention has been drawn to the neural network and its algorithms. In order to solve more complex problems, deeper networks have been proposed [1], [2], which also mean that more memory units and computing resources are needed to meet the growing parameters and computing requirements. Thus, the traditional von Neumann computing system suffers from increasingly severe memory wall problem, especially in Internet of Things (IoT) terminals. Many different methods have been proposed to mitigate the above problems, mainly including the following two types: one is a more compact and efficient network [3], [4], and the other is an acceleration method [5]–[7]. These methods alleviated the pressure to a certain degree, but did not get rid of the dilemma caused by the separation of storage and calculation fundamentally. Therefore, it is necessary to construct a new architecture that integrates storage and calculation to effectively solve memory wall problem [8], [9].

Since Chua [10] proposed the theoretical memristor from the symmetry and completeness perspective in 1971, memristor has drawn great attention. In 2008, the first device with memristive characteristics was finally manufactured by HP labs [11]. As a two-terminal device, the memristor has many excellent characteristics such as resistance plasticity, in-memory computing, low power consumption and conducive to integration. This makes the memristor become a promising candidate for solving the memory wall problem and realizing various neural network circuits. Over the past years, it has been widely used in various neural networks such as Spiking Neural Network [12], [13], Multilayer Neural Network [14]–[16], Hopfield Neural Network [17], [18], Convolutional Neural Network [19], [20], and Long Short-term Memory Network [21]. These neural network circuits have achieved obvious advantages in terms of power consumption, computational acceleration and so on.

However, as tasks become increasingly difficult, more complex and deeper networks are still being proposed. Among them, Transformer Network (TN) has become one of the most important models. The TN was first proposed by Vaswani *et al*. [22] in 2017. After that, Devlin *et al*. [2] further proposed the BERT (Bidirectional Encoder Representations from Transformers) based on the TN in 2018 and achieved the best results in 11 natural language processing tasks. Since the TN are believed to be able to capture long-distance dependencies in the sequence by using attention mechanism [23], [24], it has also been widely introduced into other fields such as image recognition [25], multimodal information fusion and recognition [26], [27], speech synthesis [28], image description [29], target detection [30] and so on. The TN has shown outstanding performance in various fields, but its huge parameters ($65 * 10^6$ parameters

Manuscript received June 21, 2021; revised November 11, 2021; accepted December 13, 2021. Date of publication January 4, 2022; date of current version March 29, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61876209, Grant 61936004, and Grant U1913602. This article was recommended by Associate Editor Y. Zhang. *(Corresponding author: Xiaoping Wang.)*

Chao Yang is with the Institute of Artificial Intelligence, School of Artificial Intelligence and Automation, and the Key Laboratory of Image Processing and Intelligent Control of Education Ministry of China, Huazhong University of Science and Technology, Wuhan 430074, China, and also with the School of Electrical, Electronic and Computer Science, Guangxi University of Science and Technology, Liuzhou 545006, China.

Xiaoping Wang and Zhigang Zeng are with the Key Laboratory of Image Processing and Intelligent Control of Education Ministry of China, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: wangxiaoping@hust.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2021.3136355.

Digital Object Identifier 10.1109/TCSI.2021.3136355

in the base model in [22]) and complex calculations require computing systems with 'big data' transmission capabilities and large computing units. However, it is usually difficult to achieve the above requirements in the von Neumann systems due to limited resources and memory wall. Memristor can perform in-memory computing, and the memristor crossbar can speed up the vector-matrix multiplication. Therefore, the memristor-based TN circuit will be a promising way to solve the above problems. A recent work proposed by Yang *et al*. [31] gives an implementation for TN based on ReRAM and has achieved remarkable calculating efficiency and good results in WMT 2016 Translation Task through matrix decomposition technique, however, the use of ADCs is still dependent. That is to say, the full-circuit implementation of TN is still absent.

In this work, a full-circuit implementation of the TN based on memristor is proposed. It involves four modules: a memristor crossbar module, an analog signal memory module, function circuit modules and a timing signal generation module, which are respectively responsible for weight representation and calculation acceleration, intermediate result storage, signal transformation and operation scheduling. These modules cooperate to carry out all computations of the TN in the form of analog signal. The main contributions of this work are as follows.

1) Based on memristor, a full-circuit implementation of the TN is proposed. Thanks to the in-memory computing characteristic of the memristor, the storage of weight and vector-matrix multiplication are combined in the memristor crossbar module to carry out computation efficiently and avoid the separation of weight representation and computing.

2) A capacitor-based analog signal memory is exploited to store intermediate results in a near-memory mode, thus circumventing analog-to-digital conversion and data transmission.

3) Five modular circuits with the functions of Softmax, Layer Normalization, Relu, Residual and Multiply-add are designed, which are vital components for various neural networks. Combined with the designed analog memory, all calculations can be performed in the form of analog signals without the use of ADCs and DCAs.

4) The correctness of the proposed circuit is verified by characters recognition experiments. In addition, the performances of the circuit are carried out, and the results show that the circuit has the advantages of anti-noise, low power consumption and small area overhead. The non-idealities of the memristors and the comparison of energy consumption with GPU are also analyzed based on the MNIST handwritten digital recognition task.

The rest of this paper is organized as follows. Section II introduces the classical Transformer block and the memristor model. Section III illustrates the overall architecture and the circuit level design. Section IV shows the character image recognition experiments based on the proposed TN memristive circuit. Section V discusses the performance of the proposed circuit, including signal retention rate of analog memory, area overhead, power consumption, stability and non-ideality analysis. Section VI concludes this paper and discusses further works. Note that the simulations of the circuit are completed in PSPICE except that non-idealities analysis is carried out based on MATLAB.

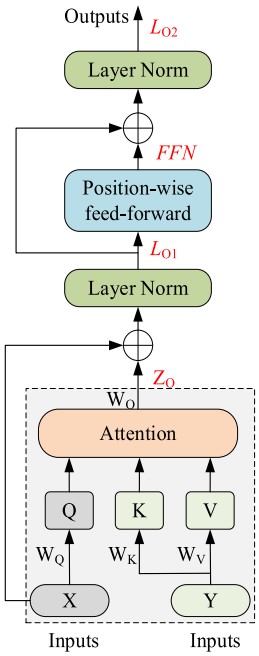


Fig. 1. The computing flow chart of the basic block of Transformer.

II. The Transformer Network and Memristor Model

A. The Classical Transformer Block

Transformer consists of two parts: encoder and decoder. Both encoder and decoder are composed of a stack of several identical blocks. Each block of the encoder has an attention sub-layer and a feed-forward sub-layer. Compared with the encoder, the decoder has an additional cross-attention sub-layer insert in the front of each layer to receive and fuse the information from the encoder. Therefore, the attention sub-layer and feed-forward sub-layer can be considered as basic components to make up any TNs. This basic block involves two processes: attention calculation and forward calculation. Fig. 1 shows the computing flow chart. The details are as follows.

Firstly, calculate the cross-attention (or self-attention).

It is assumed that X and Y are two different (or the same) sequence features.

1) Use matrixes W_Q, W_K, W_V to map X and Y to other feature spaces $Q = XW_Q$, $K = YW_K$, $V = YW_V$. Here, the row vectors of Q, K, V are called as queries, keys, values respectively. The dimension of the queries and keys is d, and the dimension of values is d_v.

2) Calculate the Scaled Dot-Product Attention as

$$atten(Q, K, V) = softmax(\frac{QK^{\mathsf{T}}}{\sqrt{d}})V, \tag{1}$$

where $1/\sqrt{d}$ is a scaling factor. The superscript T means transpose operation. Take $W = W_Q W_K^{\mathsf{T}}$, then $QK^{\mathsf{T}} = XWY^{\mathsf{T}}$. the

Eq. (1) could be written as

$$atten(Q,K,V) = softmax(\frac{XWY^{\mathsf{T}}}{\sqrt{d}})V. \tag{2}$$

Here, using W instead of W_Q and W_K can reduce the use of electronic components. The $softmax(\cdot)$ function takes a row vector as input and normalizes it into a probability distribution vector as output.

3) In order to set up the Residual layer, use W_O to map $atten$ and make the output dimension the same as X.

$$Z_O = atten \cdot W_O, \tag{3}$$

where $W_O \in R^{d_v \times n_X}$.

4) Set up the Residual layer and Layer Normalization layer successively. The output is

$$L_{O1} = LN(X + Z_O), \tag{4}$$

where $LN(\cdot)$ means that a Layer Normalization operation is performed.

$$LN(X) = \delta_0 \frac{X - \bar{\mu}}{\delta} + \mu, \tag{5}$$

where δ_0 and μ are constants. Usually take $\delta_0 = 1$ and $\mu = 0$. $\bar{\mu}$ and δ are calculated as

$$\begin{cases} \bar{\mu} = \dfrac{1}{N}\sum_{i=1}^{N} x_i, \\ \delta = \sqrt{\dfrac{1}{N}\sum_{i=1}^{N}(x_i - \bar{\mu})^2 + \epsilon}, \end{cases} \tag{6}$$

where x_i is the ith component of X. ϵ is a tiny decimal to prevent division by 0.

Secondly, perform feed-forward network calculations:

1) Perform the position-wise feed-forward, which involves two linear transformations and a ReLU activation in between.

$$FFN(L_{O1}) = max(0, L_{O1}W_1 + b_1)W_2 + b_2. \tag{7}$$

2) Set up the Residual layer and the Layer Normalization layer in sequence again. The output L_{O2} is

$$L_{O2} = LN(FFN + L_{O1}). \tag{8}$$

Generally, multi-head attention has more abundant representation subspaces compared with single-head attention. In that case, the mapping matrices W_Q, W_K and W_V will be divided as $W_Q = [W_Q^1, \ldots, W_Q^n]$, $W_K = [W_K^1, \ldots, W_K^n]$, $W_V = [W_V^1, \ldots, W_V^n]$. Then the attention $atten$ can be obtained by concatenating each head's attention h_i together.

$$atten(Q,K,V) = [h_1, \ldots, h_n], \tag{9}$$

where

$$h_i = softmax(\frac{XW^iY^{\mathsf{T}}}{\sqrt{d_k}})YW_V^i, \tag{10}$$

and $W^i = W_Q^i(W_K^i)^{\mathsf{T}}$. Here $d_k = d/n$, and n is the number of head.

TABLE I
PARAMETER SETTING OF AIST-BASED MEMRISTOR IN THIS WORK

$R_{on}(\Omega)$	1K	$R_{off}(\Omega)$	10K
$D(nm)$	3	$i_0(A)$	$1e-3$
$i_{off}(A)$	$1e-5$	$i_{on}(A)$	1
$V_{T-}(V)$	-0.37	$V_{T+}(V)$	0.19
p	10	$\mu_v(m^2s^{-1}\Omega^{-1})$	$1.6e-12$

B. The Memristor Model

For mimicing memristor in software, many models have been proposed based on the characteristics or experimental data, such as TEAM model [32], VTEAM model [33], AIST-based memristor model [34], PCMO-based memristor model [35], Memristor model based on TaO_x [36] and so on. A comparison of these models is given in [37], [38]. In this work, an AIST-based model [39] with a voltage threshold property is adopted, which is built based on experimental data. The differential of state variable is defined as

$$\frac{dw(t)}{dt} = \begin{cases} \mu_v \dfrac{R_{on}}{D}\dfrac{i_{off}}{i(t) - i_0} f(w(t)), & 0 < V_{T+} < v(t), \\ 0, & V_{T-} \le v(t) \le V_{T+}, \\ \mu_v \dfrac{R_{on}}{D}\dfrac{i(t)}{i_{on}} f(w(t)), & v(t) < V_{T-} < 0, \end{cases} \tag{11}$$

$$f(w(t)) = 1 - (\frac{2w(t)}{D} - 1)^{2p}. \tag{12}$$

Here, i_0 is constant. i_{off} and i_{on} are the average currents at the highest resistance (R_{off}) and the lowest resistance (R_{on}) respectively. μ_v indicates the average ionic mobility. V_{T-} and V_{T+} represent negative and positive voltage thresholds severally. The specific values of the parameters in this work are shown in Table I.

III. TRANSFORMER NETWORK CIRCUIT DESIGN BASED ON MEMRISTOR

A. Overall Circuit Architecture

Similar to the basic block is the fundamental element of TNs, the block circuit is the cornerstone to construct various TNs circuits. For the convenience of discussion, this section takes the single-head attention block as an example to explain the implementation architecture of the TNs circuit based on memristor. Fig. 2 shows the vital modules and processes.

In the block circuit, four kinds of modules are involved. Firstly, there are five memristor crossbar modules applied to represent five weight matrices W, W_V, W_O, W_1 & b_1, W_2 & $b2$. Secondly, the analog signal memories (gray boxes) marked by ①-④ are applied to store the intermediate results. Thirdly, five function circuits are exploited to realize Softmax, Layer Normalization, ReLU, Multiply-add and Residual. Finally, the timing signal generation module is employed to schedule all operations. The calculation process of the circuit is briefly described as follows.

1) First of all, the input matrix X and Y are converted into voltage signals. Each signal is connected to a CMOS

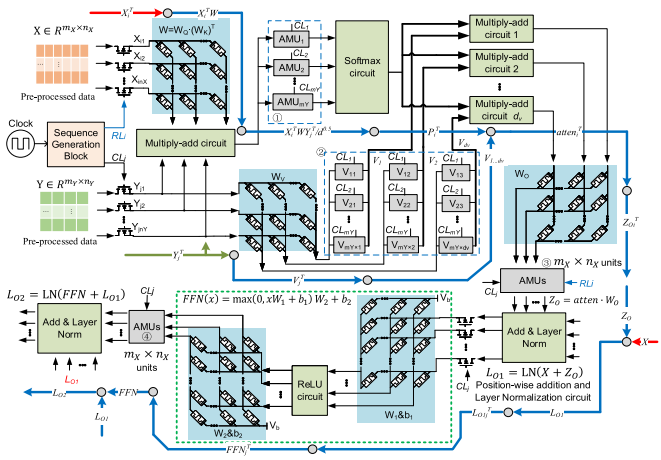


Fig. 2. The circuit implementation architecture of single-head attention block. Three thicker signal lines with different colors (red, green, blue) and tags are used to show the signal routing. The gray circles between the signal lines represent the operation of the circuit closest to it. X, Y are the matrices of input signals whose row signal X_i^T and Y_j^T are gated by RL_i and CL_j, respectively. When RL_i and CL_j come, the correlations, marked as $X_i^T W Y_j^T/\sqrt{d}$, will be calculated via the memristor crossbar W and a multiply-add circuit, and then be stored in analog signal memory ①. On the other hand, Y_j^T is also input into the memristor crossbar W_V to obtain the jth row of V. After m_Y cycles of CL, the correlations and V will be ready, and the ith row of $atten$, marked as $atten_i^T$, can be obtained through the softmax circuit and d_V multiply-add circuits. Then $atten_i^T$ is fed to memristor crossbar W_o and the outputs are stored in analog signal memory ③. After m_X cycles of RL, Z_o is ready and L_{o1} can be carried out. As CL_j continues to arrive, the rows of L_{o1} will be fed in FFN circuit, and the outputs will be stored in analog signal memory ④. Once the last row of L_{o1} is gated, the last normalization circuit will output the final results L_{o2} via FFN circuit.

switch whose gates in the same row are connected together and controlled by a timing signal. Thus, if the timing signal is effective, the corresponding row signal will be applied to subsequent circuits. To calculate the correlation matrix $(XWY^{\mathsf{T}})/\sqrt{d}$, each row of X should be applied one by one into the memristor crossbar circuit determined by W, and the output will be further delivered to the Multiply-add module whose scaling factor is $\sqrt{d}$. At the same time, the rows of Y are also gated sequentially and input to the Multiply-add module. That is, when X_i^{T} and Y_j^{T} are gated, the correlation score $(XWY^{\mathsf{T}})_{ij}/\sqrt{d}$ will be obtained. Here, X_i^{T} and Y_j^{T} represent the ith row of X and jth row of Y which are controlled by timing signal RL_i $(i = 1, \ldots, m_X)$ and CL_j $(j = 1, \ldots m_Y)$, respectively. The correlation scores are stored in analog signal memory ① under the control of CL_j. The number of cells in memory ① is m_Y. After the rows of Y are gated for one round, the cells will be all refreshed.

2) Synchronously, the rows of Y, which are gated in the above step, are also applied to the memristor crossbar circuit determined by W_V, and the output $V = YW_V$ is saved in the analog signal memory marked by ② which has $m_Y \times d_v$ cells. Here, d_v is the number of columns of W_V. In memory ②, the cells in the same row are controlled by the same CL_j, and the ones in the same column are connected to the same output of the memristor crossbar.

3) Once the units in memory ① are all updated, the Softmax circuit will output a set of valid probability-distributed signals P_i^{T}. It is used to weight the rows of V stored in the memory ② by d_V Multiply-add modules arranged in parallel to accelerate the calculations. The outputs of these Multiply-add modules make up the $atten_i^{\mathsf{T}}$ and directly input to the memristor crossbar circuit determined by Wo. The output $(Z_O)_i^{\mathsf{T}} = atten_i^{\mathsf{T}} W_O$ is saved in analog memory ③ under the control of CLs and RLs, where the subscript i indicates that the ith row of X is gated at present. When the rows of X and Y are all gated, Z_O is ready.

4) The Residual circuit will output a valid result when Z_O is ready. The subsequent Layer Normalization circuit outputs the normalized signals marked as L_{O1} in Fig. 2.

5) In order to perform the feed-forward network (FFN) calculation, the rows of L_{O1} will also be applied to the FFN circuit one by one under the control of CL_j. The FFN circuit is composed of three modules framed by the green box in Fig. 2 and its outputs are also stored in analog memory ④ under the control of CL_j.

6) Finally, the Residual circuit and the Layer Normalization circuit are employed again to process signal FFN just like step 4), and the output signal L_{O2} can be as an input matrix signal of the next block or other circuits (such as classification circuits).

Besides, we should replace W and W_V with n (the number of heads) sub-circuits in the case of multi-head attention, according to Eqs. (9) and (10). The number of cells in memory ② does not change, but the number of units in memory ① is increased by n times. In addition, the number of Softmax module and Multiply-add module should also be increased by n times for the parallel calculation. Finally, the outputs by each head are packed together to form multi-heads $atten$, which has the same dimension as the single-head case. The subsequent operations are the same too.

B. Memristor Crossbar Module

TNs have huge weight parameters and matrix calculations. For example, the base model in [22] has 65×10^6 weights. Here, memristor crossbar module is adopted to represent weights and perform an efficient vector-matrix multiplication based on Ohm's law and Kirchhoff's law. The schematic diagram is shown in Fig. 3. The basic unit consists of a 1T1M (one CMOS and one memristor) structure. The operational amplifier (OP-AMP) A_1 provides a bias voltage V_A (marked in red) to realize negative weight. V_b is a constant voltage to obtain a bias current for the bias circuit. OP-AMP A_2 converts current to voltage. When WLs are valid, the output voltage of the kth column is

$$V_k = -I_k \cdot R_{fo}, \quad k = 1, \ldots, n. \tag{13}$$

According to Kirchhoff's current law, I_k is

$$I_k = V_b \cdot (G_{bk} - G_s) + \sum_{i=1}^{q} X_i \cdot (G_{ik} - G_s), \quad k = 1, \ldots, n. \tag{14}$$

Here, G_{bk} $(G_{bk} = 1/M_{bk})$ denotes the memristor conductance of M_{bk} in ith column of the bias circuit, and G_s $(G_s = 1/R_s)$

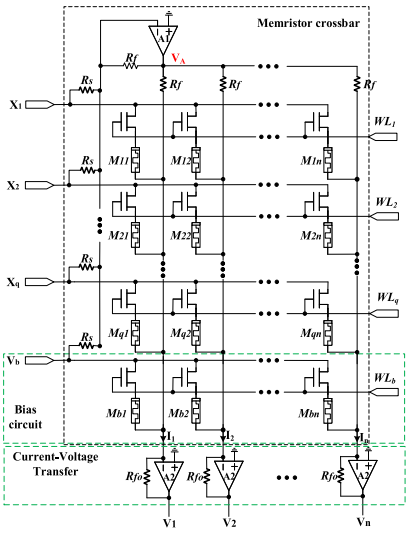


Fig. 3. Schematic diagram of memristor crossbar circuit.

is the conductance of R_s. X_i is the input voltage of the row, G_{ik} ($G_{ik} = 1/M_{ik}$) represents the memristor conductance at the crossing position of the ith row and the kth column. V_k is rewritten as

$$V_k = V_b \cdot W_{bk} + \sum_{i=1}^{q} X_i \cdot W_{ik}, \tag{15}$$

where $W_{ik} = R_{fo} \cdot (G_s - G_{ik})$, $W_{bk} = R_{fo} \cdot (G_s - G_{bk})$, for $i = 1, \ldots q$ and $k = 1, \ldots n$. The output $\boldsymbol{V} \in R^{n\times 1}$ can be expressed in matrix term as

$$\boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}. \tag{16}$$

Here, $\boldsymbol{X} = [X_1, \ldots, X_q, V_b]^{\mathsf{T}}$, $\boldsymbol{W} = \begin{bmatrix} W_M \\ W_b \end{bmatrix}$, element $W_M(i,k)$ equals to W_{ik}, and $\boldsymbol{W_b} = [W_{b1}, \ldots, W_{bn}]$.

C. Analog Signal Memory Module

The memristor crossbar circuit can realize a fast Vectors-Matrix multiplication. But for the matrix multiplication, the row signals of the input matrix needs to be applied one by one, and the corresponding results should be saved in sequence at the same time. In fact, this requirement also exists in CNNs' circuits based on memristors because the convolution kernels need to slide step by step on the image. A common practice is to use ADCs sampling the calculation results into digital memory chips and then output them into subsequent circuits through DACs [19], [40]–[42]. However, These methods not only increase the complex auxiliary circuit and power consumption but also introduce quantization errors and transmission time-lag. In fact, if the signal does not have to be stored for a long time, it is possible to use capacitors to save the information. As shown in Fig. 4, the unit is composed of a capacitor and an OP-AMP. The OP-AMP is in form of a voltage follower, which can increase the discharge resistance of the capacitor and improve the load capacity. The CL_m ($m = 1, \ldots, M$) and RL_n ($n = 1, \ldots, N$) are the control signals.

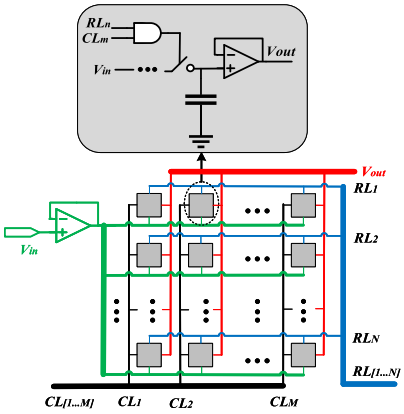


Fig. 4. Schematic diagram of analog signal memory.

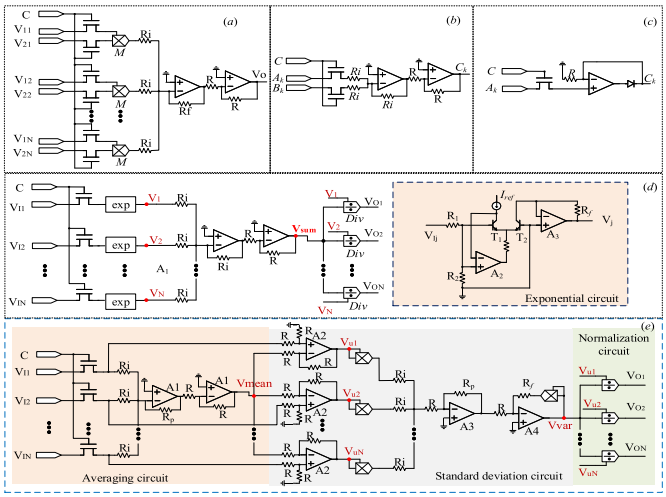


Fig. 5. Schematic of function circuit modules. a) Multiply-add circuit; b) Residual circuit; c) ReLU circuit; d) Softmax circuit and exponential circuit (yellow block); e) Layer Normalization circuit. Nodes with the same label are connected together.

D. Function Circuit Module

In function circuit module, there are five function circuits are designed, including a) Multiply-add circuit. b) Residual circuit. c) ReLU circuit. d) Softmax circuit. e) Layer Normalization circuit.

a) As shown in Fig. 5(a), the Multiply-add circuit is mainly composed of multipliers and a summation circuit. The output of the multiplier is proportional to the product of two input voltages. That is, $V_{out} = K V_{in1} V_{in2}$, where $K = 1V^{-1}$. When the signal C is effective, the output voltage is

$$V_o = \sum_{i=1}^{N} \frac{R_f}{R_i} K V_{1i} V_{2i}. \tag{17}$$

Here, V_{1i} and V_{2i} ($i = 1, \ldots, N$), are two sets of input. Let R_f equal to R_i, which is a standard Multiply-add module. While $R_f/R_i = 1/\sqrt{d}$, that is the scaling factor in Eq. (1).

b) Fig. 5(b) shows the schematic diagram of the kth sub-circuit of the Residual circuit. It is a simple summation circuit, and its output voltage is determined by

$$C_k = A_k + B_k. \tag{18}$$

c) The ReLU circuit performs nonlinear mapping on input signal. The kth sub-circuit diagram is shown in Fig. 5(c). If voltage A_k is less than or equal to 0, the output voltage of the OP-AMP will cut off the diode. While the input voltage is greater than 0, the diode and OP-AMP form a voltage follower so that the output voltage $C_k = A_k$. That is

$$C_k = max(A_k, 0). \tag{19}$$

d) The softmax circuit transforms the inputs into the probability distribution. As shown in Fig. 5(d), the circuit mainly includes the exponential circuit (yellow block), summation circuit and division circuit. In exponential circuit, the transistor T_2 and T_1 form a complementary symmetrical structure, and the collector current of transistor T_1 is exponentially related to the voltage between its base and emitter. The output voltage is

$$V_j = k_1 e^{k_2 V_{Ij}}, \tag{20}$$

where $k_1 = R_f I_{ref}$, $k_2 = -\frac{R_2}{U_T(R_1+R_2)}$, U_T is a voltage related to temperature. The jth output voltage V_{oj} is

$$\begin{aligned} V_{oj} &= \frac{V_j}{V_{sum}} \\ &= \frac{e^{k_2 V_{Ij}}}{\sum_{i=1}^{N} e^{k_2 V_{Ii}}}. \end{aligned} \tag{21}$$

Let $R_2 \gg R_1$, and perform an appropriate voltage magnification (such as $-U_T$) to V_{Ij}, the function of Softmax is implemented.

e) The layer normalization circuit includes three parts: averaging circuit, standard deviation circuit and normalization circuit. The averaging circuit is a proportional summing circuit. The standard deviation circuit is made up of subtraction sub-circuit, summation sub-circuit and square root sub-circuit. The voltages of nodes V_{mean}, $V_{ui}(i\text{th})$ and V_{var} are

$$V_{mean} = \frac{R_p}{R_i} \sum_{i=1}^{N} V_{Ii}, \tag{22}$$

$$V_{ui} = V_{Ii} - V_{mean}, \tag{23}$$

$$V_{var} = \sqrt{\frac{R_f}{R} \cdot \frac{R_p}{R_i} \sum_{i=1}^{N} V_{ui}^2}. \tag{24}$$

Here, if $R_p = R_i/N$, then $R_p/R_i = 1/N$, it is just the normalization factor. The coefficient $\sqrt{\frac{R_f}{R}}$ can be used to realize δ_0 in Eq. (5), usually $R_f = R$. So the ith output voltage V_{Oi} is

$$V_{Oi} = (V_{Ii} - V_{mean})/V_{var}. \tag{25}$$

E. Timing Signal Generation Module

Matrix multiplication is the main operation in the TNs. Using memristor crossbar circuit to realize the matrix multiplication needs timing signals to schedule the input matrix and save the calculated results in sequence. In the proposed circuit, two sets of timing signals are exploited, which are RLs and CLs. The number of channels are m_X and m_Y, respectively.

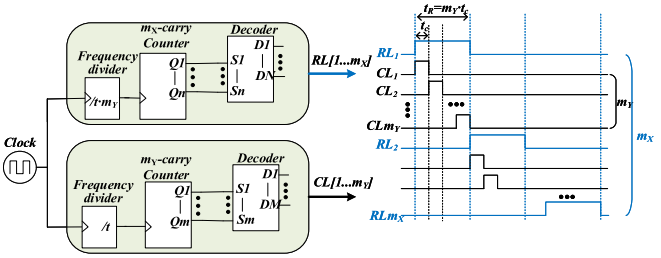


Fig. 6. Timing signal generation module.

The relationship between their period is $T_{RL} = m_Y \cdot T_{CL}$, and the effective voltage duration time meets $t_{RL} = m_Y \cdot t_{CL}$. Fig. 6 shows the circuit diagram and signals diagram. The left figure is the circuit diagram, which includes two generators to generate RLs (above) and CLs (below), respectively. For the RLs (CLs) signals, the clock frequency is reduced by $m_Y \times t$ (t) times by the frequency divider firstly, and the output signals then are counted by the m_X (m_Y)-number counter. At last, the decoder decodes the counting result into m_X and m_Y channels timing signals as shown in blue (black) on the right side.

IV. Experiment and Simulation

Generally, a complex TN can be considered as a stack of basic blocks, and the number of stacked blocks is related to the complexity of the task. Similar to the Vision Transformer (ViT) [25] for image recognition, this work designs a TN with a single block to recognize 3×3 character images and builts the corresponding circuit in PSPICE to verify the effectiveness. Fig. 7 shows the network, which includes a basic transformer block and a classification layer. Although the representation subspace of single-head attention is not so rich compared to multi-head attention, it is complete to describe the attention mechanism. The entire network is firstly built on the PyTorch platform and trained with three character images 'z,' 'v' and 'n.' Then, download the trained weights into the corresponding circuit built in the PSPICE for simulation.

A. Network Training

Specifically, W, W_V, and V_O in the attention block are all 3×3 matrices, W_1 & b_1 is a $(3+1) \times 6$ matrix, and W_2 & b_2 is a $(6+1) \times 3$ matrix. The final classification output layer is defined as a 3×3 fully connected layer. To reduce the dimension and integrate the information, a mean layer is added after the block to calculate the mean value of the row vector.

The data set consists of three character images which are 'z,' 'v,' 'n' with a size of 3×3, and the pixel value is normalized to $[0, 1]$. In order to improve the robustness of network, white noise with an intensity of 0.2 is added to diversify the samples, and 5000 samples are generated for each character as the training set. For the testing set, the white noise with an intensity of 0.2 and 0.4 is added to generate two sets of samples, that is, each of them contains 3000. Fig. 8 shows the examples of them. The row of the image is treated the same way as token similar to ViT [25], and the positional encodings generated by the sine and cosine functions [22] are added to each image. With cross entropy loss, the network is trained by stochastic gradient descent method. Fig. 9 shows

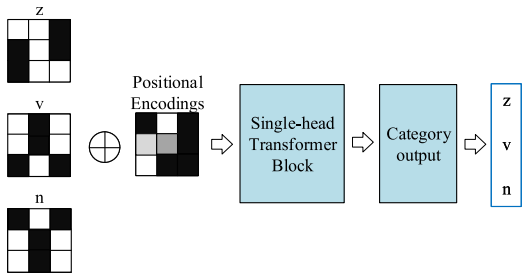


Fig. 7. The Transformer network for 3 × 3 character image recognition.

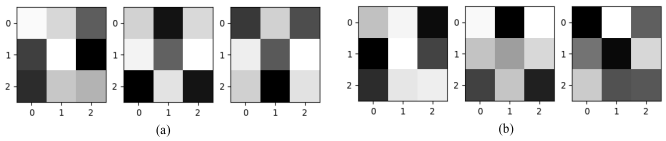


Fig. 8. The sample examples with different intensities of white noise. (a) and (b) correspond to the noise intensity of 0.2 and 0.4 respectively.

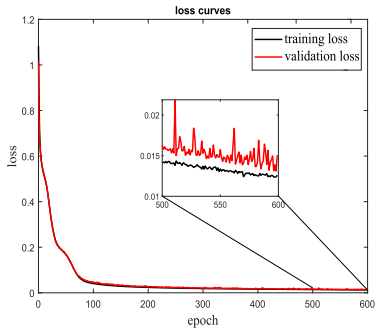


Fig. 9. Training loss curves.

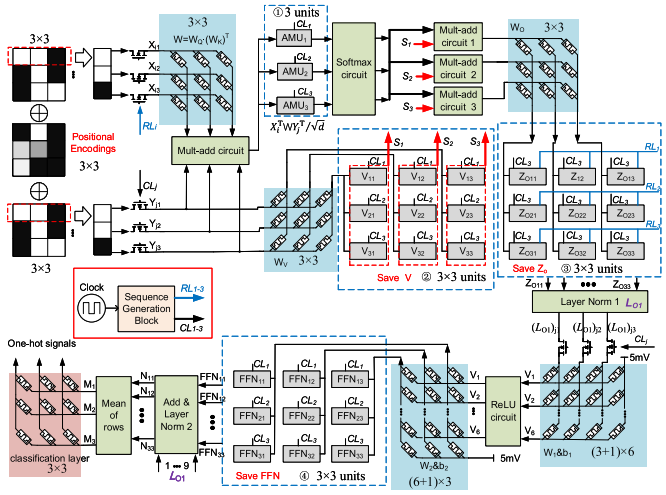


Fig. 10. The schematic diagram of the 3 × 3 character image recognization.

the training curves, which indicates that the network converges well after 600 epochs. The recognition rates of the two testing sets are 100% and 97.27% for 0.2 and 0.4 intensity white noise, respectively. It means that the network can solve the character recognition task well.

B. Circuit Settings

The entire circuit of the network is shown in Fig. 10. The classification layer is also built based on a memristor crossbar circuit. Before simulating, the parameters of the circuit are configured as follows.

First of all, the trained matrix parameters are downloaded and written into the corresponding memristor crossbar circuits according to Eq. (15). Taking the resistance range of memristor into account, the resistances are taken as $R_s = 1.8K\Omega$, $R_{fo} = 6.6K\Omega$, and the bias voltage is $V_b = 5mV$.

Considering the memristor voltage threshold, each image pixel is converted into a millivolt signal through Eq. (26), and the same with the positional encodings. The voltage signals of the three characters and the positional encodings are listed in Tables II–V. The input matrix X and Y are connected to the identical image added positional encoding.

$$V_x = 5x \times 10^{-3} V. \tag{26}$$

where x is the pixel, and V_x indicates the converted voltage.

For the timing control signals, the period of the clock signal is configured to 10μs, so the period of the timing signal RLs and CLs are 90μs and 30μs, respectively. Their effective level duration is 30μs and 10μs. In this way, a total of 12 CLs pulses (120μs) are required to complete all calculations, of which 90μs is consumed to calculate attention and 30μs is for calculating the feed-forward network. The timing waveforms of RLs and CLs are shown in Fig. 11.

Lastly, the number of cells in memory ① is 3, and the number of units of the memories ②, ③, and ④ are all 3 × 3. Multiply-add circuit 1-3 are used for parallel processing.

C. Circuit Simulation and Result Analysis

In order to facilitate the simulation, the control signal C in each function module is always valid. Under the scheduling

TABLE II
VOLTAGE SIGNALS OF 'Z'(mV)

5	5	0
0	5	0
0	5	5

TABLE III
VOLTAGE SIGNALS OF 'V'(mV)

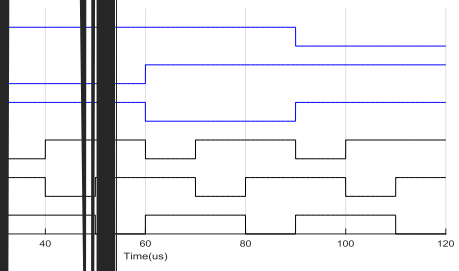
5	0	5
5	0	5
0	5	0

TABLE IV
VOLTAGE SIGNALS OF 'N'(mV)

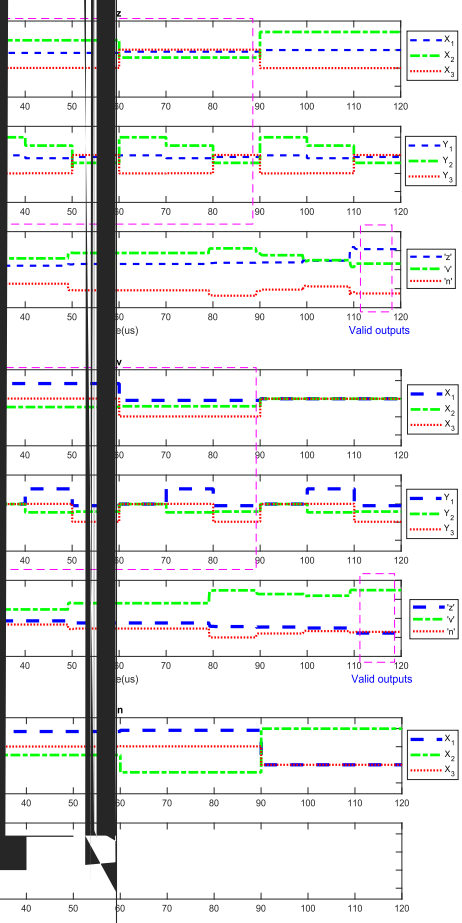
0	5	0
5	0	5
5	0	5

TABLE V
VOLTAGE SIGNALS OF POSITION ENCODING (mV)

0.00	5.00	0.00
4.21	2.70	0.05
4.55	-2.08	0.10



aveforms of *RL*s and *CL*s.



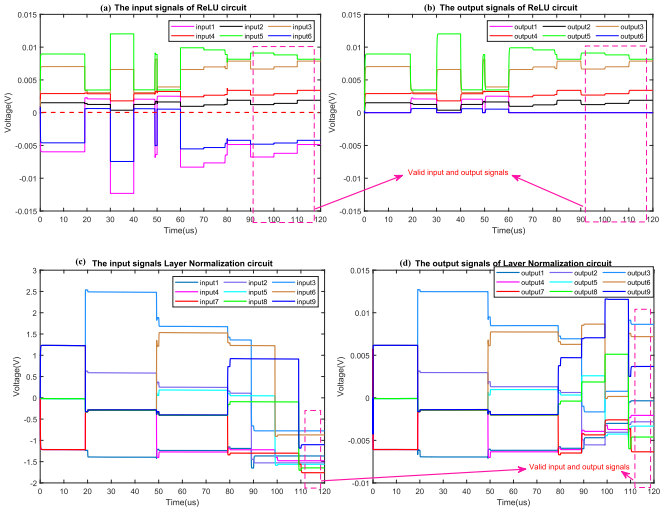


Fig. 15. The input and output signals of ReLU circuit and Layer Normalization circuit. The signals with the same color mean that they are the corresponding channels in the circuit. (a)-(b) ReLU circuit's signals. (c)-(d) Layer Normalization circuit's signals.

V. Circuit Performance Analysis

This section analyzes the factors that affect the signal retention rate of the analog signal memory, and discusses the area overhead, power consumption, anti-noise, non-ideality of the circuit.

A. The Signal Retention Rate Analysis

The analog memory allows the signal to be stored directly in analog form. However, the core component to hold information is a capacitor, which usually faces the problem of information weakening over time. As shown in Fig. 4, the capacitance and the discharge resistance are the main factors of information attenuation. Firstly, taking the discharge resistance as a fixed value $10^8\Omega$, the relationship between the information retention rate and the capacitance is revealed in Fig. 16. The larger the capacitance is, the more information can be retained, but the larger capacitance may also be cause a larger circuit delay. While the capacitance is 1nF, the information retention rate reaches 99.88% at 120μs. This is an enough precision to get correct result in our experiments. So 1nF is taken in the proposed circuit. Here, the information retention rate is defined as

$$Retention\_rate = \frac{V_t}{V_0} \times 100\%, \tag{27}$$

where V_0 and V_t are the voltage at the initial time and time t respectively.

Moreover, in order to discuss the effect of discharge resistance on information retention rate, we take the capacitance as a fixed value of $1nF$. Since the input resistance of the OP-AMP at the output port can be considered as the discharge resistance, Fig. 17 reveals the tendency of the information retention rate with the resistance. Similarly, the larger the resistance, the better the information is kept. When the input resistance is 100MΩ, the information retention rate can still reach 99.88% after 120μs. Therefore, the input resistance is set to 100MΩ in our circuit.

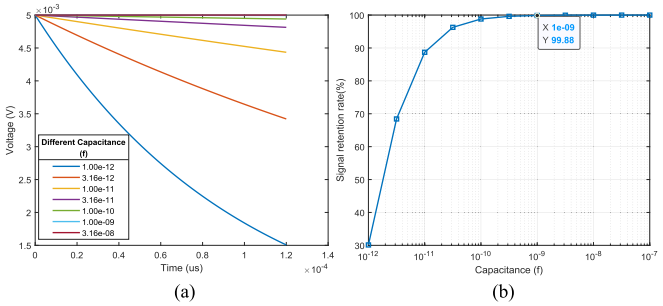


Fig. 16. The influence of capacitance on signal preservation. (a) the attenuation curves of signal voltage with time (initial voltage is 5mV) under different capacitances. (b) the relationship between the signal retention rate and the capacitance at 120μs.

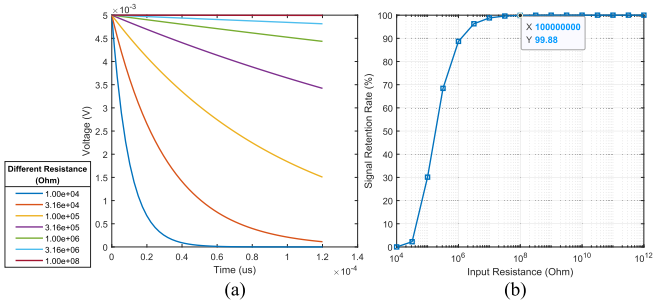


Fig. 17. The effect of discharge resistance on the signal preservation. (a) the decay curves of information (voltage) with time under different discharge resistance. (b) the relationship between the discharge resistance and the signal retention rate at 120μs.

B. Area Overhead and Power Consumption Analysis

In the proposed circuit, the memristor crossbar circuits are responsible for preserving the parameters of the network. Its scale and number correspond to the network parameters one-to-one. But the analog signal memory and functional circuit modules do not need to save any network parameters, they work only in its calculation period. So if the calculations are properly scheduled, these modules can be multiplexed in different blocks or even the same block. Such will greatly reduce the number of these modules. As a result, the total area overhead will mainly be determined by the memristor crossbar circuits.

For a memristor crossbar circuit with m rows and n columns, the number of memristors, resistors, and MOSFETs are listed in Table VI. In this experiment circuit, a total of six memristor crossbar circuits are used, the number of memristors, resistors, MOSFETs are estimated as 81, 50 and 89. Generally, in integrated circuits, the dimensions of memristors, resistors, and MOSFETs are all dozens of nanometers [43], [44]. Therefore, we can estimate that the maximum area overhead is about $2.2 \times 10^4 nm^2$ (The area of each component is regarded as $100nm^2$). Furthermore, taking the base model of transformer network with 65×10^6 parameters [22] as an example, the total area is about $1.3 \times 10^4 \mu m^2$. It is relatively small and conducive to further integration and application to various terminals.

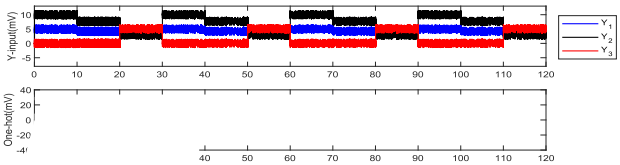
Since the matrix multiplication is the main calculation in the TNs, the memristor crossbar circuits has also become the

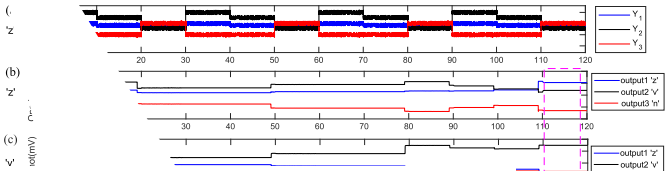
TABLE VI

THE NUMBER OF MEMRISTORS, RESISTORS, AND MOSFETS USED IN A MEMRISTOR CROSSBAR CIRCUIT WITH m ROWS AND n COLUMNS

Number of Memristors	Number of Resistors	Number of MOSFETs [1]
mn	$m+n+1$	$mn+8$

[1] An operational amplifier generally has 8 MOSFETs.

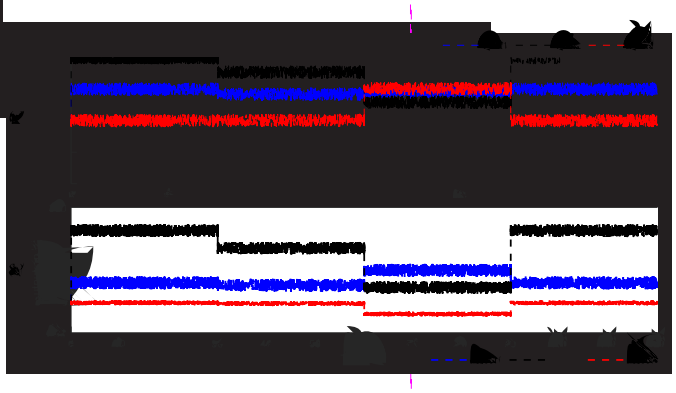




signals cor z, v, n.

major sou n. Besides, the input
is limited ling methods. Tal
minimum r $= 1K\Omega$ and n
the turn-on series with r
(about 20m of a mem

about 3×675 /
$25nW = 5.625\mu$ value.



C. Stability Analy

To analyze the s the circuit, w.
noise with differer o the input ima
signals. Fig. 18 sho nals and the corre-
sponding one-hot ou gh the input signals
are added some noise it can still recognize
the input characters c icates that the circuit
has good anti-noise ab Fig. 19 further reveals
the proposed circuit ha ability, where the noise
with an intensity of 0.3

As shown in Figs. 18 vident noise is observed
in the output signals. Ou that the analog memory
doesn't latch the signal sing edge of the control
signal arrives. Another im ason may be that the core
component of analog mem capacitor, which can form
a natural low-pass filter wi equivalent resistance of the
input circuit and filter out th se to a certain extent. Fig. 20
ne denoising ability e analog memory. In addition,
etwork itself has the abi y of anti-noise. The memristor
ossbar circuit can map the i put signals to another subspace
where the signals can be better distinguished. As shown in

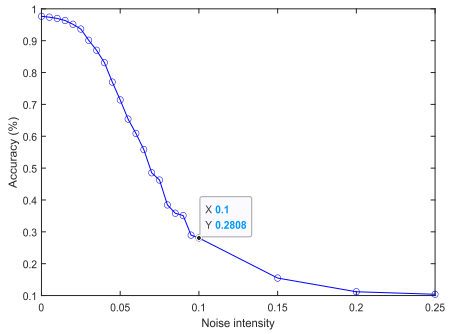


Fig. 22. Performance of the TN under different noise intensities is added to its weights, and each point represents the average of 30 experiments.

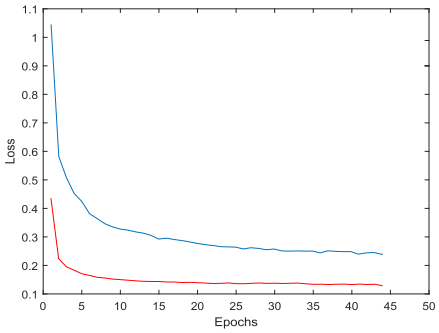


Fig. 23. Training curves of the TN with two different update noise intensity (blue: 0.1, red: 0.01).

digital (the bicubic interpolation is used to scale the original images from 28×28 to 8×8 pixels) recognition task, and its architecture is similar to the 3×3 character recognition network mentioned above. In the network, W_Q, W_K, and W_V are all 16×32 matrices and are divided into four heads, $W_O \& W_2 \in R^{32\times16}$, $W_1 \in R^{16\times32}$, and the classification layer outputs ten classes. The simulation results in MATLAB show that the accuracy of 97.65% achieved by TN. It is better than 95.5% obtained by a fully connected network with the ideal crossbar in Reference [45], which means that the attention mechanism is helpful.

Considering that the performance of TN circuit will be greatly degraded in practice due to the non-idealities factors of the memristor. A certain intensity of Gaussian noise is added to the memristance to mimic the non-idealities of memristors, and the noise is defined as

$$noise = \delta * (G_{max} - G_{min}) * r, \tag{28}$$

where δ indicates the noise intensity, and r is a random number generated from standard normal distribution. G_{max} and G_{min} correspond to $1/R_{on}$ and $1/R_{off}$, respectively. Similar to Monte Carlo simulation, random numbers are generated according to the specified distribution as the conductivity drift caused by the non-idealities, and the average of 30 simulations is taken as the final accuracy. As shown in Fig. 22, with the noise intensity increasing, the circuit performance decreases gradually, which means that the non-idealities of memristors have an obvious impact on the TN circuit performance. In practice, the influence of the non-idealities of memristors can be effectively mitigated by weight quantization and in-situ training. Therefore, a modified in-situ training method based on [15] is developed in which the memconductance errors caused by programming pulse, wire resistance, and memristor non-idealities are all considered in the online learning process (more details can be found in Reference [15]). Accordingly, the MATLAB simulation results show that the TN circuit can still achieve an accuracy of 91.75% although the update operation of memconductance suffers from noise with an intensity of 0.1. Here, the noise is the same as Eq. 28 and is served as the memconductance drift caused by memristor non-idealities. This is a significant improvement over the performance shown in Fig. 22 where the accuracy has dropped to about 28.08% under the same noise. Fig. 23 shows the training cross-entropy loss curves under the noise intensity of 0.1 and 0.01. As can be seen, the lower the noise intensity is, the faster the convergence speed and the better state is. It means that the more non-ideal the memristor is, the more difficult it is to train, and the worse the final performance. These simulations can further provide a reference for circuit applications.

E. Discussion

Generally, the full-circuit implementation is a severe challenge for neural networks due to the large number of matrix multiplication operations, intermediate values storing, various function modules realizing, operations scheduling and so on. For the intermediate values storing, a common practice is to save the intermediate signals into digital memories through sampling. It usually needs more auxiliary circuits and power consumption and is easy to introduce quantization error. In this work, we suggest an analog memory to save the intermediate values, which can store analog signals directly and has obvious anti-noise ability. Moreover, function modules are important elements for neural network circuits. TNs have many complex function modules, which increases the challenge of circuit design and signal processing compared with other networks. In the proposed circuit, the function modules are realized by analog circuits which makes the circuit can complete all operations with analog signal. In addition, it is another point worth deserving attention that the circuit is implemented in full-circuit form, which means that the whole circuit can carry out all operations automatically with itself control circuits. Table VII gives the comparison of this work with other similar methods.

Furthermore, an energy consumption comparison between the proposed method and GPU platform is given by performing the same MNIST handwritten digital recognition task in the same TN-based module with both platforms. Firstly, 10,000 8×8 pixels images are input into the trained network, which runs on Pytorch-based GPU (NVIDIA TITAN RTX) platform, to perform forward calculation, and the time and power consumption are recorded at the same time. The total time is about $163.23ms$, and the power consumption is 61.8W, which is estimated by subtracting the average power in idle time from the case in operation. Hence the energy consumed by an image is $61.8W \times 163.23ms/10,000 = 1.009mJ$. Secondly, due to each 8×8 image being divided into 4×16 signal matrix, the energy consumed by an image calculation can be estimated

TABLE VII
COMPARISON OF THE PROPOSED TN CIRCUIT AND OTHER NEURAL NETWORK CIRCUITS BASED ON MEMRISTOR

Refs.	Storage method of intermediate value	Full-analog calculation	Anti-noise	Full-circuit implementation
CNNs [19]	Digital memory	No	No	No
CNNs [47]	Digital memory	No	No	No
Blaze Block [48]	Digital memory	No	No	No
FCNs [49]	Saved in Workspace	No	No	No
CNN and LSTM [49]	No report	No	No	No
This work	Analog memory	Yes	Yes	Yes

by using the method given in Section V-B. That is, the power consumption can be estimated to be $239.2uW$. Considering that $200us$ will be used in the proposed circuit to complete a forward calculation with 4×16 inputs, so that the energy consumed by each image is $239.2uW \times 200us = 47.84nJ$. Therefore, without considering other auxiliary circuits, this work will reduce the energy by 21,154 times, but about 12 times more time.

VI. CONCLUSION

In this work, a full-circuit implementation of the TN is proposed based on memristor, which mainly consists of a memristor crossbar module, an analog memory module, five function circuit modules and a timing signal generation module. In the proposed circuit, due to the weight matrices are physically mapped on the resistance of the memristor crossbar modules directly, the vector-matrix multiplication of the TN is carried out in a single step according to Ohm's law and Kirchhoff's law in the terms of energy-area efficiency. In addition, the intermediate result can be stored directly with the help of an analog memory module in near-memory mode instead of stored in digital memory through sampling. In this way, the hardware overhead and power consumption can be reduced and the quantitative error can also be avoided as well as the 'big data' transmission. Moreover, five modular function circuits are designed to achieve different transformations in the forms of analog signals. Finally, a timing signal generation module is designed to schedule the operations of the circuits mentioned above.

The character recognition experiments are carried out in PSPICE based on the proposed TN circuit. The simulation results indicate that the proposed circuit can complete the task correctly and has strong robustness. In addition, the results of performance analysis show that the circuit has the advantages of low power consumption, small area overhead and strong anti-noise ability. These good features are conducive to integrating it into various terminals. Finally, the non-idealities of the memristors and the comparison of the energy consumption with the GPU are also analyzed based on the MNIST handwritten digital recognition task, which can provide a reference for circuit applications. The designed circuit still have room for future optimization. Many OP-AMPs are used in the function circuit modules, reducing the number will further reduce power consumption. Future work will focus on implementing large-scale TN circuit to deal with more complex tasks such as multi-modal (vision, language, text) sentiment analysis and applying it to robotic systems or IoT terminals.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[3] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[4] N. Ma, X. Zhang, H. T. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 116–131.

[5] C. Gao, D. Neil, E. Ceolini, S.-C. Liu, and T. Delbruck, "DeltaRNN: A power-efficient recurrent neural network accelerator," in *Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays*, Feb. 2018, pp. 21–30.

[6] Y. Shen, M. Ferdman, and P. Milder, "Maximizing CNN accelerator efficiency through resource partitioning," in *Proc. 44th Annu. Int. Symp. Comput. Archit.*, Jun. 2017, pp. 535–547.

[7] S. Lu, M. Wang, S. Liang, J. Lin, and Z. Wang, "Hardware accelerator for multi-head attention and position-wise feed-forward in the transformer," 2020, *arXiv:2009.08605*.

[8] Y. Zhang *et al.*, "A system hierarchy for brain-inspired computing," *Nature*, vol. 586, no. 7829, pp. 378–384, Oct. 2020.

[9] Y. Zhang *et al.*, "Brain-inspired computing with memristors: Challenges in devices, circuits, and systems," *Appl. Phys. Rev.*, vol. 7, no. 1, Mar. 2020, Art. no. 011308.

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

[11] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, pp. 80–83, May 2008.

[12] L. Camuñas-Mesa, B. Linares-Barranco, and T. Serrano-Gotarredona, "Neuromorphic spiking neural networks and their memristor-CMOS hardware implementations," *Materials*, vol. 12, no. 17, p. 2745, Aug. 2019.

[13] M. E. Fouda, F. Kurdahi, A. Eltawil, and E. Neftci, "Spiking neural networks for inference and learning: A memristor-based design perspective," 2019, *arXiv:1909.01771*.

[14] F. M. Bayat, M. Prezioso, B. Chakrabarti, H. Nili, I. Kataeva, and D. Strukov, "Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits," *Nature Commun.*, vol. 9, no. 1, pp. 1–7, Dec. 2018.

[15] C. Li *et al.*, "Efficient and self-adaptive *in-situ* learning in multilayer memristor neural networks," *Nature Commun.*, vol. 9, no. 1, pp. 1–8, Dec. 2018.

[16] Y. Zhang, X. Wang, and E. G. Friedman, "Memristor-based circuit design for multilayer neural networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 2, pp. 677–686, Feb. 2017.

[17] S. G. Hu *et al.*, "Associative memory realized by a reconfigurable memristive Hopfield neural network," *Nature Commun.*, vol. 6, no. 1, pp. 1–8, Nov. 2015.

[18] S. Zhang, J. Zheng, X. Wang, Z. Zeng, and S. He, "Initial offset boosting coexisting attractors in memristive multi-double-scroll Hopfield neural network," *Nonlinear Dyn.*, vol. 102, no. 4, pp. 2821–2841, Dec. 2020.
[19] C. Yakopcic, M. Z. Alom, and T. M. Taha, "Memristor crossbar deep network implementation based on a convolutional neural network," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 963–970.
[20] C. Yakopcic, M. Z. Alom, and T. M. Taha, "Extremely parallel memristor crossbar architecture for convolutional neural network implementation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 1696–1703.
[21] C. Li *et al.*, "Long short-term memory networks in memristor crossbar arrays," *Nature Mach. Intell.*, vol. 1, pp. 49–57, Jan. 2019.
[22] A. Vaswani *et al.*, "Attention is all you need," 2017, *arXiv:1706.03762*.
[23] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.
[24] Z. Lin *et al.*, "A structured self-attentive sentence embedding," 2017, *arXiv:1703.03130*.
[25] A. Dosovitskiy *et al.*, "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.
[26] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang, "VisualBERT: A simple and performant baseline for vision and language," 2019, *arXiv:1908.03557*.
[27] Y.-H.-H. Tsai, S. Bai, P. P. Liang, J. Z. Kolter, L.-P. Morency, and R. Salakhutdinov, "Multimodal transformer for unaligned multimodal language sequences," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 6558–6569.
[28] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, "Neural speech synthesis with transformer network," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, no. 1, 2019, pp. 6706–6713.
[29] J. Yu, J. Li, Z. Yu, and Q. Huang, "Multimodal transformer with multi-view visual representation for image captioning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 12, pp. 4467–4480, Dec. 2020.
[30] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis.*, Aug. 2020, pp. 213–229.
[31] X. Yang, B. Yan, H. Li, and Y. Chen, "ReTransformer: ReRAM-based processing-in-memory architecture for transformer acceleration," in *Proc. 39th Int. Conf. Comput.-Aided Design*, Nov. 2020, pp. 1–9.
[32] S. Kvatinsky, E. G. Friedman, A. Kolodny, and U. C. Weiser, "TEAM: Threshold adaptive memristor model," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 1, pp. 211–221, Jan. 2013.
[33] S. Kvatinsky, M. Ramadan, E. G. Friedman, and A. Kolodny, "VTEAM: A general model for voltage-controlled memristors," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 62, no. 8, pp. 786–790, Aug. 2015.
[34] Y. Li *et al.*, "Activity-dependent synaptic plasticity of a chalcogenide electronic synapse for neuromorphic systems," *Sci. Rep.*, vol. 4, no. 1, pp. 1–7, May 2015.
[35] A. M. Sheri, H. Hwang, M. Jeon, and B.-G. Lee, "Neuromorphic character recognition system with two PCMO memristors as a synapse," *IEEE Trans. Ind. Electron.*, vol. 61, no. 6, pp. 2933–2941, Jun. 2014.
[36] S. Choi, P. Sheridan, and W. D. Lu, "Data clustering using memristor networks," *Sci. Rep.*, vol. 5, no. 1, pp. 1–10, Sep. 2015.
[37] A. Ascoli, F. Corinto, V. Senger, and R. Tetzlaff, "Memristor model comparison," *IEEE Circuits Syst. Mag.*, vol. 13, no. 2, pp. 89–105, 2nd Quart., 2013.
[38] Z. Wang, X. Wang, Z. Lu, W. Wu, and Z. Zeng, "The design of memristive circuit for affective multi-associative learning," *IEEE Trans. Biomed. Circuits Syst.*, vol. 14, no. 2, pp. 173–185, Apr. 2020.
[39] Y. Zhang, X. Wang, Y. Li, and E. G. Friedman, "Memristive model for synaptic circuits," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 64, no. 7, pp. 767–771, Jul. 2017.
[40] P. Yao *et al.*, "Fully hardware-implemented memristor convolutional neural network," *Nature*, vol. 577, no. 7792, pp. 641–646, 2020.
[41] P. Chi *et al.*, "Prime: A novel processing-in-memory architecture for neural network computation in ReRAM-based main memory," *ACM SIGARCH Comput. Archit. News*, vol. 44, no. 3, pp. 27–39, Jun. 2016.
[42] A. Shafiee *et al.*, "ISAAC: A convolutional neural network accelerator with *in-situ* analog arithmetic in crossbars," *ACM SIGARCH Comput. Archit. News*, vol. 44, no. 3, pp. 14–26, 2016.
[43] Z. Wang, Q. Hong, and X. Wang, "Memristive circuit design of emotional generation and evolution based on skin-like sensory processor," *IEEE Trans. Biomed. Circuits Syst.*, vol. 13, no. 4, pp. 631–644, Aug. 2019.
[44] J. H. Yoon *et al.*, "An artificial nociceptor based on a diffusive memristor," *Nature Commun.*, vol. 9, no. 1, p. 417, Dec. 2018.
[45] T. V. Nguyen, J. An, and K.-S. Min, "Memristor-CMOS hybrid neuron circuit with nonideal-effect correction related to parasitic resistance for binary-memristor-crossbar neural networks," *Micromachines*, vol. 12, no. 7, p. 791, Jul. 2021.
[46] D. Dang, J. Dass, and R. Mahapatra, "ConvLight: A convolutional accelerator with memristor integrated photonic computing," in *Proc. IEEE 24th Int. Conf. High Perform. Comput. (HiPC)*, Dec. 2017, pp. 114–123.
[47] H. Ran *et al.*, "Memristor-based edge computing of blaze block for image recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Dec. 29, 2021, doi: 10.1109/TNNLS.2020.3045029.
[48] J. Chen *et al.*, "An efficient memristor-based circuit implementation of squeeze-and-excitation fully convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 6, 2021, doi: 10.1109/TNNLS.2020.3044047.

Chao Yang received the B.S. and M.S. degrees from the College of Electronic Engineering, Guangxi Normal University, Guilin, China, in 2012 and 2015, respectively. He is currently pursuing the Ph.D. degree with the School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China.

His current research interests include memristive systems and circuits, artificial neural networks, and brain-like intelligent computing circuit based on memristor.

Xiaoping Wang (Senior Member, IEEE) received the B.S. and M.S. degrees in automation from Chongqing University, Chongqing, China, in 1997 and 2000, respectively, and the Ph.D. degree in systems engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2003.

Since 2011, she has been a Professor with the School of Artificial Intelligence and Automation, Huazhong University of Science and Technology. Her current research interests include memristors and its applications to memory storage, modeling, and simulation.

Zhigang Zeng (Fellow, IEEE) received the Ph.D. degree in systems analysis and integration from the Huazhong University of Science and Technology, Wuhan, China, in 2003.

He is currently a Professor with the School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, and also with the Key Laboratory of Image Processing and Intelligent Control of Education Ministry of China, Huazhong University of Science and Technology. He has authored or coauthored more than 200 international journal papers. His current research interests include the theory of functional differential equations and differential equations with discontinuous right-hand sides, and their applications to dynamics of neural networks, memristive systems, and control systems.

Dr. Zeng has been a member of the Editorial Boards of *Neural Networks* since 2012, *Cognitive Computation* since 2010, and *Applied Soft Computing* since 2013. He was an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS from 2010 to 2011. He has been an Associate Editor of the IEEE TRANSACTIONS ON CYBERNETICS since 2014 and the IEEE TRANSACTIONS ON FUZZY SYSTEMS since 2016.