ARTICLES

Journal of Semiconductors
(2023) 44, 104101
doi: 10.1088/1674-4926/44/10/104101

Forward stagewise regression with multilevel memristor for sparse coding

Chenxu Wu[1, ‡], Yibai Xue[1, ‡], Han Bao[1], Ling Yang[1], Jiancong Li[1], Jing Tian[1], Shengguang Ren[1], Yi Li[1, 2, †], and Xiangshui Miao[1, 2]

[1]School of Integrated Circuits, Huazhong University of Science and Technology, Wuhan 430074, China

[2]Hubei Yangtze Memory Laboratories, Wuhan 430205, China

Abstract: Sparse coding is a prevalent method for image inpainting and feature extraction, which can repair corrupted images or improve data processing efficiency, and has numerous applications in computer vision and signal processing. Recently, several memristor-based in-memory computing systems have been proposed to enhance the efficiency of sparse coding remarkably. However, the variations and low precision of the devices will deteriorate the dictionary, causing inevitable degradation in the accuracy and reliability of the application. In this work, a digital-analog hybrid memristive sparse coding system is proposed utilizing a multilevel Pt/Al_2O_3/AlO_x/W memristor, which employs the forward stagewise regression algorithm: The approximate cosine distance calculation is conducted in the analog part to speed up the computation, followed by high-precision coefficient updates performed in the digital portion. We determine that four states of the aforementioned memristor are sufficient for the processing of natural images. Furthermore, through dynamic adjustment of the mapping ratio, the precision requirement for the digit-to-analog converters can be reduced to 4 bits. Compared to the previous system, our system achieves higher image reconstruction quality of the 38 dB peak-signal-to-noise ratio. Moreover, in the context of image inpainting, images containing 50% missing pixels can be restored with a reconstruction error of 0.0424 root-mean-squared error.

Key words: forward stagewise regression; in-memory computing; memristor; sparse coding

Citation: C X Wu, Y B Xue, H Bao, L Yang, J C Li, J Tian, S G Ren, Y Li, and X S Miao, Forward stagewise regression with multilevel memristor for sparse coding[J]. *J. Semicond.*, 2023, 44(10), 104101. https://doi.org/10.1088/1674-4926/44/10/104101

1. Introduction

Sparse coding, a method that decomposes a signal into a few elements of a dictionary, can uncover semantic information about images[1] and has been applied in image processing tasks such as image inpainting[2] and feature extraction[3–5]. Image inpainting can fill in missing parts of an image and is commonly used to repair aged photos and damaged image files. Feature extraction can reduce the dimensionality of the signal, obtain essential features, and improve the efficiency of data processing. The forward stagewise regression (FSR) is one of the promising algorithms to solve the sparse coding problem and provides competitive results with the commonly used lasso algorithm[6–8]. However, FSR usually involves massive dot product operations to compute the cosine distance, which severely limits the efficiency of the FSR algorithm. Nevertheless, in 2015, RJ. Tibshirani proposed that FSR will outperform the lasso in efficiency when implemented in a highly parallel computing paradigm[7]. Then, for the first time, this work discusses the adoption of memristive systems to accelerate FSR and perform sparse coding. The memristive in-memory computing (IMC) paradigm is an appealing parallel computing approach to perform dot product operations with $O(1)$ time complexity[9–12]. And the IMC has been widely explored to accelerate the processing of applications like the neural network[13], signal process[14], and regression[15]. Recently, sparse coding has also been implemented on the memristor arrays with encouraging performance improvement compared with conventional CMOS-based systems[16–19].

However, these memristive sparse coding systems are chosen to be as efficient as possible, regardless of the degradation of the dictionary and the corresponding detrimental effects on image reconstruction and feature extraction. In their system, a locally competitive algorithm is chosen for implementation. Then, to reduce the time complexity of the algorithm, both the forward and backward iterative operations of the algorithm are accelerated by analog operations. Next, the encoding results will be solely based on the dictionary held on the memristor array. However, the accuracy and reliability of the dictionary are not guaranteed. When the dictionary is stored on the memristor array, it deteriorates into various dictionaries at different terminals due to the accuracy and variability of the memristor[16]. After that, the image cannot be reconstructed by a deterministic dictionary. Ultimately, this inevitably harms image reconstruction and affects the associated image feature extraction[18]. To solve this problem, some computations need to be performed by a digital system to control the accuracy of the algorithm. However, the time complexity of the forward and backward iteration steps of the local competition algorithm is similar, and speeding up only one step does not reduce the time complexity of the entire algorithm. Therefore, to ensure accuracy

Chenxu Wu and Yibai Xue contributed equally to this work and should be considered as co-first authors.

Correspondence to: Y Li, liyi@hust.edu.cn

Received 13 MARCH 2023; Revised 24 MAY 2023.

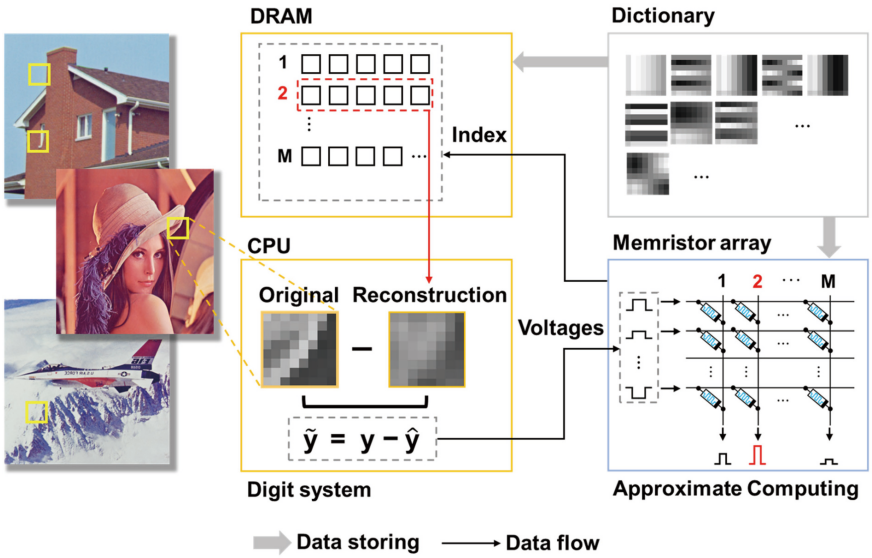

Fig. 1. (Color online) Schematic diagram of the digital-analog hybrid memristive sparse coding system. Input images can be divided into small patches and then represented by a few dictionary elements. The memristor array is used to store the approximate dictionary to calculate the cosine distance between the dictionary elements and the residual vector (image reconstruction error). The digital system then determines the most relevant dictionary element based on the result of the analog calculation, and that element at full precision becomes part of the reconstructed image.

while speeding up the execution of sparse coding, memristive systems should consider digital-analog hybrid operations and alternative algorithm.

In this work, to ensure the quality of sparse coding while accelerating it, we propose a digital-analog hybrid memristive sparse coding system (Fig. 1). In this system, the sparse coding is implemented by forward stagewise regression with mixed digital-analog operations. During the process, an approximate dictionary is mapped to the memristor array to speed up the calculation of the cosine distance between the dictionary elements and the residual vectors, while a full-precision dictionary is stored in the digital system to precisely update the reconstructed image. Thus, by implementing the calculation of the cosine distance on the memristor array, the computational time complexity of FSR can be reduced from $O(m \times n)$ to $O(m + n)$. Besides, the reconstruction is conducted based on the dictionary stored in the digital system, which avoids the dictionary being deteriorated by the nonidealities of the devices. Additionally, to reduce the precision required on the digital-to-analog converters (DACs), a dynamically adjusted data mapping method is proposed. Finally, in the image inpainting task, which is an important application of sparse coding, the proposed method can provide accurate reconstruction on a natural image with 50% lost pixels.

2. Memristor-based FSR

The procedure of sparse coding involves dictionary construction and sparse representation. The constructed dictionary can be either the predefined dictionary or the learned dictionary. The FSR algorithm is utilized for achieving sparse representation. In this section, we illustrate the principle of accelerating the FSR algorithm by memristive IMC. Initially, we introduce the multilevel Pt/Al_2O_3/AlO_x/W memristor, followed by the algorithm of the FSR. The evaluation of the memristor-based FSR is predicated on the performance of the aforementioned multilevel memristors. We then propose data mapping methods and finally demonstrate the digital-analog hybrid system.

2.1. Multilevel Pt/Al_2O_3/AlO_x/W memristor

We fabricated a Pt/Al_2O_3/AlO_x/W stacked memristor. After depositing a Ti adhesion layer on the Si/SiO_2 substrate, we deposited a 100-nm Pt bottom electrode by direct current (dc) magnetron sputtering. Then, through the process of atomic layer deposition, a 3-nm Al_2O_3 layer was formed, followed by a 5-nm AlO_x layer which was deposited via radio frequency magnetron sputtering. Finally, the 100-nm W top electrode was grown by dc magnetron sputtering and patterned by ultraviolet lithography with a size of 50 × 50 μm^2. A sketch of the device structure is given in Fig. 2(a). Fig. 2(b) shows the scanning electron microscope (SEM) image of the devices. As shown in Fig. 2(c), based on the Al 2*p* and O 1*s* X-ray photoelectron spectroscopy (XPS) images in Al_2O_3 and AlO_x layers, it can be observed that the O 1*s* peak in the AlO_x layer exhibits a higher shift in binding energy at 531.10 eV compared to the Al_2O_3 layer, which indicates the higher oxygen vacancy concentration in the AlO_x layer. Homogeneous bilayer structure with oxygen vacancy concentration gradients results in a stable resistive switching behavior of the device. Fig. 2(d) exhibits the 100 consecutive dc *I*–*V* curves of the Pt/Al_2O_3/AlO_x/W memristor. High-resistance-state (HRS) and low-resistance-state (LRS) distributions of 10 devices are highly consistent, as shown in Fig. 2(e). Compared to analogous reported devices, our device has achieved multi-conductance states modulation appropriate for sparse encoding by using the write-verify operation method[20, 21]. A case of tuning the device to a target conductance state of 60 μS by the write-verify method is shown in Fig. 2(f). The tiny step voltage (±20 mV) adopted in the write-verify scheme guarantees high accuracy of conductance modulation. Fig. 2(g) displays the modulation results of the eight high conductance states (HCSs) of the devices, while maintaining variations below 3%. The read distribution of eight target conductance states is illustrated in

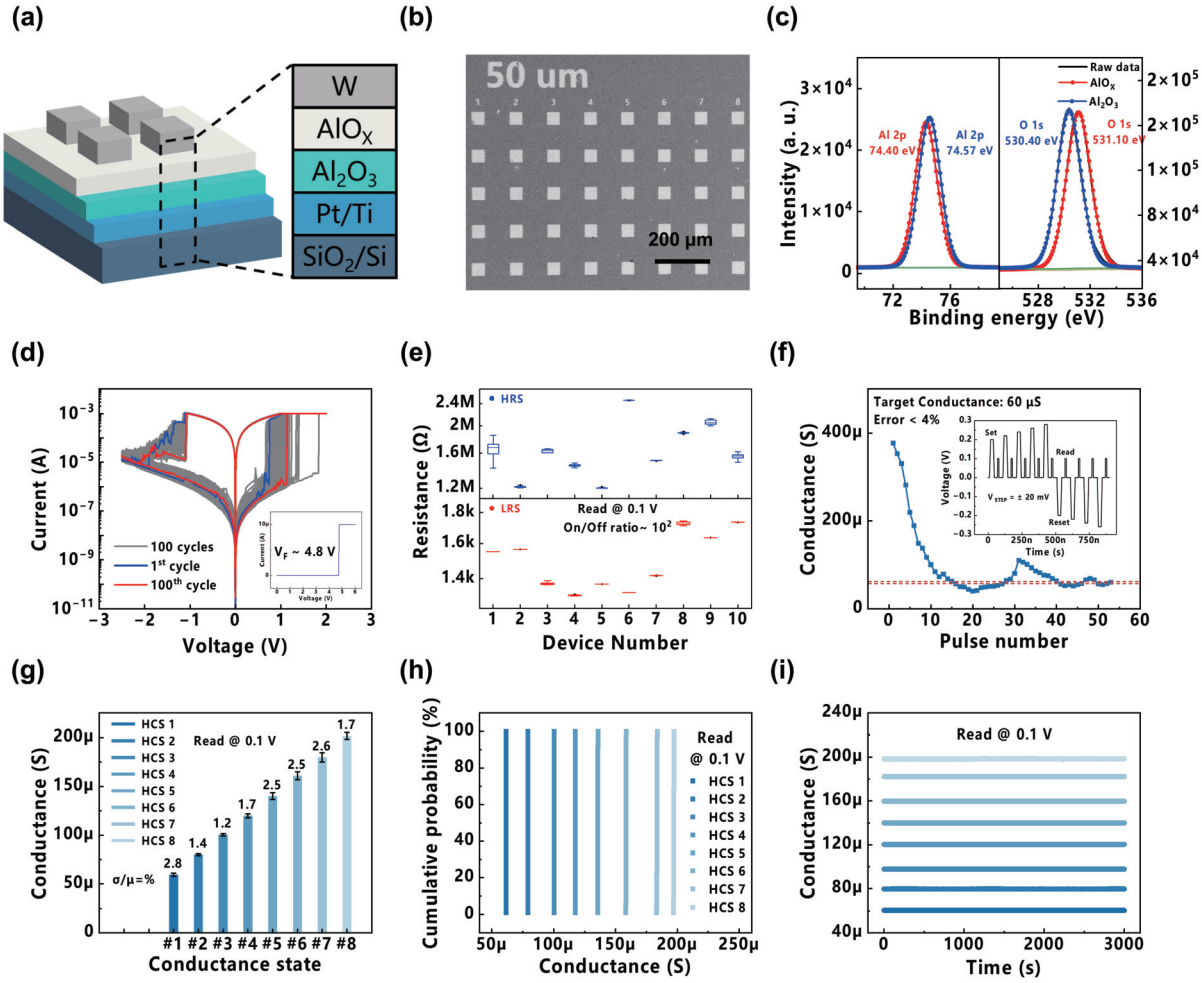

Fig. 2. (Color online) (a) A schematic of the device structure. (b) SEM image of the Pt/Al_2O_3/AlO_x/W memristor. (c) XPS image of Al 2*p* and O 1*s* in the AlO_x and Al_2O_3 layers. (d) 100 consecutive dc *I*–*V* curves with forming voltage about 4.8 V. (e) HRS and LRS distributions for 10 devices. (f) An instance of tuning the device conductance to reach a target conductance state of 60 μS with an error rate < 4% is demonstrated. The inset shows the write-verify method where a step voltage of ± 20 mV is employed. (g) Eight target conductance states are fine-tuned through the write-verify method, with < 4% variations. (h) Stable read distribution of each eight target conductance states at a dc reading voltage of 0.1 V. (i) Retention test over 3000 s of the same eight conductance states mentioned in Fig. 2(h).

Fig. 2(h), which is desirable for the sparse coding model. Fig. 2(i) demonstrates over 3000 s retention results of eight HCSs proving the reliability of the Pt/Al_2O_3/AlO_x/W memristor.

2.2. Algorithm of forward stagewise regression

FSR serves as a kind of L1 norm regression is widely used in processing sparse models[6–8]. In the regression, the n-dimension output vector *y* can be represented by the n-dimension variable vectors x_1, x_2,⋯, x_m as in the following:

$$y = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m, \tag{1}$$

where β stands for the coefficient estimates of variables x_1, x_2,⋯, x_m. To model *y* as a linear combination of x_1, x_2,⋯, x_m, the FSR updates the coefficient estimates $\beta^{(k)}$ by calculating the cosine distance to find the most relevant variable with the current residual y–$\hat{y}$, and then transfer the newly estimated $\hat{y}$ to the next iteration. For the k_{th} iteration, the processes can be expressed as follows:

$$j = \underset{j=1,\cdots,m}{\arg\max} \left\{ \left| x_j^T (y - \hat{y}) \right| \right\}, \tag{2}$$

$$\beta_j^{(k+1)} = \beta_j^{(k)} + \varepsilon \cdot \text{sign}\left\{ x_j^T (y - \hat{y}) \right\}, \tag{3}$$

$$\hat{y} + = \varepsilon \cdot \text{sign}\left\{ x_j^T (y - \hat{y}) \right\} \cdot x_j, \tag{4}$$

where *j* stands for the index of the most related variable, and ε is the step size. Such iteration continues until the repeat time *k* reaches the set maximum or the fitting error $\|y - \hat{y}\|_2$ converges under a certain level. After the convergence, the stagewise estimate will follow the desired sparsity properties[22]:

$$\left\| \beta^{(k)} \right\|_1 \leqslant k\varepsilon \text{ and } \left\| \beta^{(k)} \right\|_0 \leqslant k. \tag{5}$$

The flow chart of the FSR is illustrated in Fig. 3(a), where the calculation of cosine distance dominates in the whole process of the algorithm with the computational complexity of $O(m \times n)$. Therefore, by performing this step on the memristor array in an $O(1)$ manner, the complexity of the FSR could be reduced to $O(m + n)$. At the same time, the computational accuracy of the algorithm can be controlled by the parameter iteration operations performed in the digital system. Theoretically, the execution of the FSR can be accelerated by low-precision memristive calculations while ensuring computational accuracy.

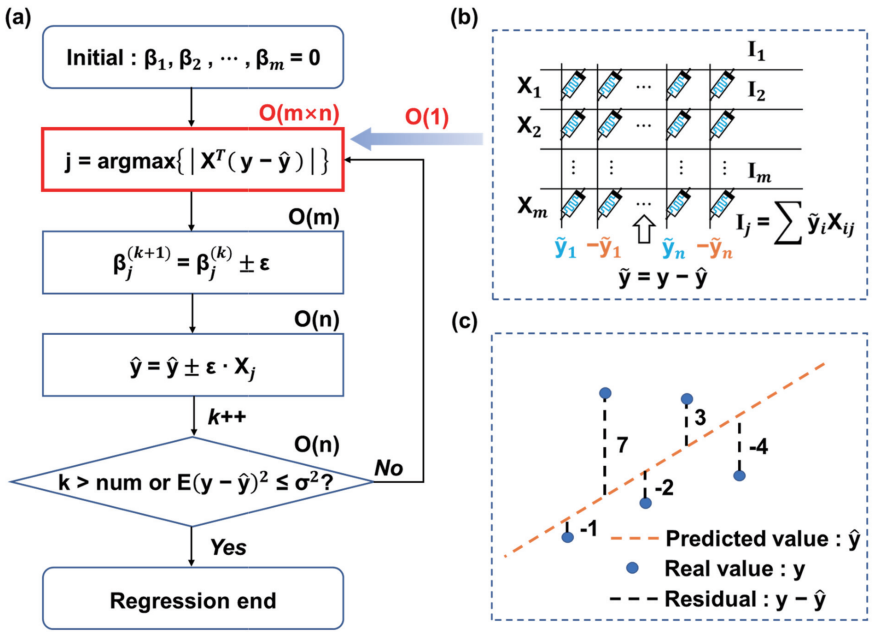

Fig. 3. (Color online) (a) Flow chart of the FSR. The sign of ε is dependent on the cosine similarity between the corresponding variable and residual vector. (b) Calculating the cosine distance between residual vector $y-\hat{y}$ and variables $x_1, x_2,\cdots, x_m$ by the memristor array. Each line of the array stores the values of a variable in the dataset and each element of the variable is represented by the conductance difference of two memristors. (c) Residual vectors mapped to the 4-bit scaling range. During the iteration, the numerical-voltage scaling ratio will be continuously decreased with the shrinking of the residual vector.

2.3. Data-mapping method

For the implementation of FSR on the memristor arrays, the variables x_1, x_2,⋯, x_m and the residual vector $y-\hat{y}$ are mapped as the device conductance and the input voltages, respectively. However, the dilemma of data mapping is that the precision of the data is usually above 8 bits, while the number of available memristor conductance states and the precision of the DAC are usually limited. To reduce the precision requirements of the in-memory sparse coding for conductance and voltage, two optimized data-mapping methods are proposed for the variables and residual vectors, respectively. For the variables, (1) standardize the data separately for each variable vector (z-score standardization)[23], (2) all values are mapped according to the designed precision and the truncation range, and (3) the difference between the two memristors is used to represent the mapped values. For the residual vector, the values of the vector shrink as the regression plane fits the data better through the iterations. Therefore, it is difficult to map the residual vectors to voltages continuously and adequately with a fixed numerical-voltage mapping ratio and low-precision DACs. Accordingly, we propose a dynamically adjusted mapping method (Fig. 3(c)): (1) calculate the average absolute value of the vector elements, (2) take some equally spaced points around 0 to map all the values in the vicinity, where the spacing is dependent on the previously calculated average value, and (3) use the DACs to convert the mapped values into voltages. Moreover, to reduce the computing burden of the digital system, the average value could be updated every few iterations, since it does not vary significantly.

We note that the above mapping method may lead to severe truncation errors at several data points, because it does not use the maximum value as the upper limit of the mapping range. However, large truncation errors occur mainly for outliers, which are usually detrimental to regression analysis[24]. Additionally, the calculation of cosine distances requires only relative values to find the most relevant variables, which can tolerate the mapping error to a certain extent.

2.4. Hybrid digital-analog system

Here, we illustrate the operation process of the proposed hybrid digital-analog system. After the variables x_1, x_2,⋯, x_m are stored on the memristor array in an approximate map-ping manner, the system initializes the regression coefficients $\beta^{(0)}$ and predicted values $\hat{y}$ to zero and then starts iteration. During the iteration process, first, the residual vector $y-\hat{y}$ is converted into a voltage vector according to the aforementioned method and applied to the array. Then, the outputs of the circuit are sampled by analog-digital converters (ADCs), and the results are the approximate cosine distances of the residual vector $y-\hat{y}$ and variables x_1, x_2,⋯, x_m, which can be used to find the most relevant variable to the current residual vector (Fig. 3(b)). Next, the regression coefficient of this variable is updated by adding $\pm\varepsilon$ (the sign depends on the cosine similarity), and the predicted value $\hat{y}$ is updated with that variable in the digital part with full precision. Finally, the updated $\hat{y}$ is transferred to the next iteration. Once the number of iterations reaches the upper limit or the fitting error $\|y-\hat{y}\|_2$ reaches below the threshold, it outputs the regression coefficients $\beta^{(k)}$, completing the variable selection and sparse estimations. In the above procedure, the cosine similarity is computed based on Ohm's law and Kirchhoff's law in the analog circuit, and the parameter iteration operation is based on full-precision computation in the digital system. While low-precision analog circuits are used to perform the most costly computations, full-precision digital operations ensure the computational accuracy of the algorithm. This system takes full advantage of digital-analog hybrid operation and reflects the adaptability of the FSR.

3. Memristor-based FSR for sparse coding

In this section, we demonstrate the use of memristor-based FSR to solve sparse coding tasks. During the process, the variable matrix X is the dictionary, the real value y is the image patch, and the fitting goal is to fit the image patch with the dictionary elements. We will first introduce the algorithm for sparse coding and then perform simulations to verify that the memristor-based FSR can accurately handle sparse coding tasks. The key step, calculating the cosine distance between the residual vector and dictionary elements, is implemented in a modeled memristor array, where the analog properties are simulated on the python platform. Additionally, considering that both the predefined and learned dictionaries are widely used, we employ both dictionaries to perform memristor-based sparse coding and compared the results.

3.1. Sparse coding algorithm

Sparse coding algorithms aim to find a linear combination of a small number of dictionary elements to represent the input signal. It can reduce the dimensionality of high-dimensional data, find essential features of signals, and extract semantic information of graphs. As such, it has a broad range of applications in computer vision and signal processing. Sparse coding usually contains two steps, namely, the construction of a dictionary and the sparse representation. Predefined dictionaries (DCT, Gabor) and learned dictionaries are widely used in this field[5]. The predefined dictionaries can be obtained by sampling the wave[25, 26], while the learned dictionary can be obtained by dictionary learning algorithms such as K-SVD[2]. After the dictionary is obtained, the mathematical description of sparse representation can be viewed as an L0-norm minimization[16]:

$$\min\left\{\sum_{i=1}^{n}\left(y_i-\sum_{j=1}^{m}x_{ij}\beta_j\right)^2\right\}\ subject\ to\ \|\beta\|_0\leqslant q, \qquad (6)$$

where y is the image patch, X is the dictionary, β is the coefficient estimation of the dictionary elements, and q is the number of selected elements. However, since the L0-norm minimization is an NP-hard problem, the L1-norm minimization is a popular choice to replace it in sparse coding, as these two methods have the same effect when the solution is sparse enough[27]. Sparse representation algorithms include lasso, orthogonal matching pursuit, forward stagewise regression, local competition algorithm, etc. Consider the excellent performance of forward stagewise regression and its adaptability to digital-analog hybrid operations. The goal of this work is to use digital-analog hybrid operations to accelerate the execution of forward stagewise regression while ensuring computational accuracy.

3.2. Memristor-based sparse coding

Here we will evaluate the sparse representation performance of the memristor-based FSR with the DCT dictionary and learned dictionary. The over-determined DCT dictionary can be obtained by sampling the cosine wave at different frequencies[25]. The scale of the dictionary is 64×256, which contains 256 image features, with each feature consisting of 64 weights. Limited by the finite conductance states of the memristor, the precision of the dictionary is reduced to 4-bit, and then is mapped as the difference between two memristor arrays. The multiple conductance states in the device model were set to 60, 80, 100, 120, 140, 160, 180, and 200 μS with 4% $(\sigma/\mu)_{write}$ (write variation) and 4% $(\sigma/\mu)_{read}$ (read variation). The distribution of the conductance is shown in Figs. 4(a) and 4(b). Then, Figs. 4(c) and 4(d) demonstrate an example of memristor-based sparse coding, where the step size σ is set to 0.025, and the iteration number is 100. During the process, the number of selected dictionary elements increases with iterations (Fig. 4(e)), which is a forward selection method that allows us to obtain the most important features at the early stage[28]. Finally, all patches will be sparse coding, and the entire image is reconstructed as in Fig. 4(f), showing good image reconstruction quality.

In Figs. 4(g) and 4(h), the sparse coding effects of the memristor-based FSR and the full-precision FSR at different thresholds are compared. The DAC precision is set to 4-bit. Then, at the expense of a little sparsity of selected elements, memristor-based FSR can achieve the same image reconstruction quality as the full-precision one. Furthermore, the image re-construction quality is inversely proportional to the number of selected dictionary elements, and the best image reconstruction quality surpasses that of prior studies[16–19]. This suggests that memristor-based FSR can perform sparse coding with high efficiency and higher quality, and the DCT dictionary can be used for memristor-based sparse coding.

The learned dictionary is obtained by a dictionary learning algorithm trained on natural images. Here, the 64×256 DCT dictionary is selected as the initial dictionary and the K-SVD algorithm[2] is used to learn a dictionary offline without considering the hardware nonidealities. The training set is shown in Fig. 5(a), which does not contain the testing image. Then the offline-learned dictionary is obtained as shown in Figs. 5(b) and 5(c), which has been mapped onto the memristor devices the same as that of the DCT dictionary. To compare the performance of memristor-based FSR using the two dictionaries, the same thresholds are taken to perform sparse coding on the offline-learned dictionary. Then, compared with the DCT dictionary, the offline-learned dictionary achieves a higher upper limit of image reconstruction quality and better sparsity (Figs. 5(d) and 5(e)). This suggests that existing learned dictionaries can be directly mapped to memristor arrays to accelerate sparse coding and achieve better results than when using DCT dictionaries.

3.3. Analysis of hardware nonidealities

To analyze the influence of the various hardware nonidealities on the system, we further conduct simulation analysis. The offline-learned dictionary is selected for the analysis according to the image reconstruction quality, and the threshold is set as in the condition when the reconstruction error (mean square error) is less than 0.0006 to balance the reconstruction quality and sparsity. As can be seen in Fig. 6(a), the 4-bit conductance is sufficient for the sparse coding tasks compared to the higher-precision ones, and 3-bit conductance is also tolerable only when the number of selected elements increases by 6%. This means that memristors with only four states (equivalently 3-bit for differential pairs) can be used to perform sparse coding for natural images, while prior studies usually required over eight states[16–19]. In Fig. 6(b), with 4-bit

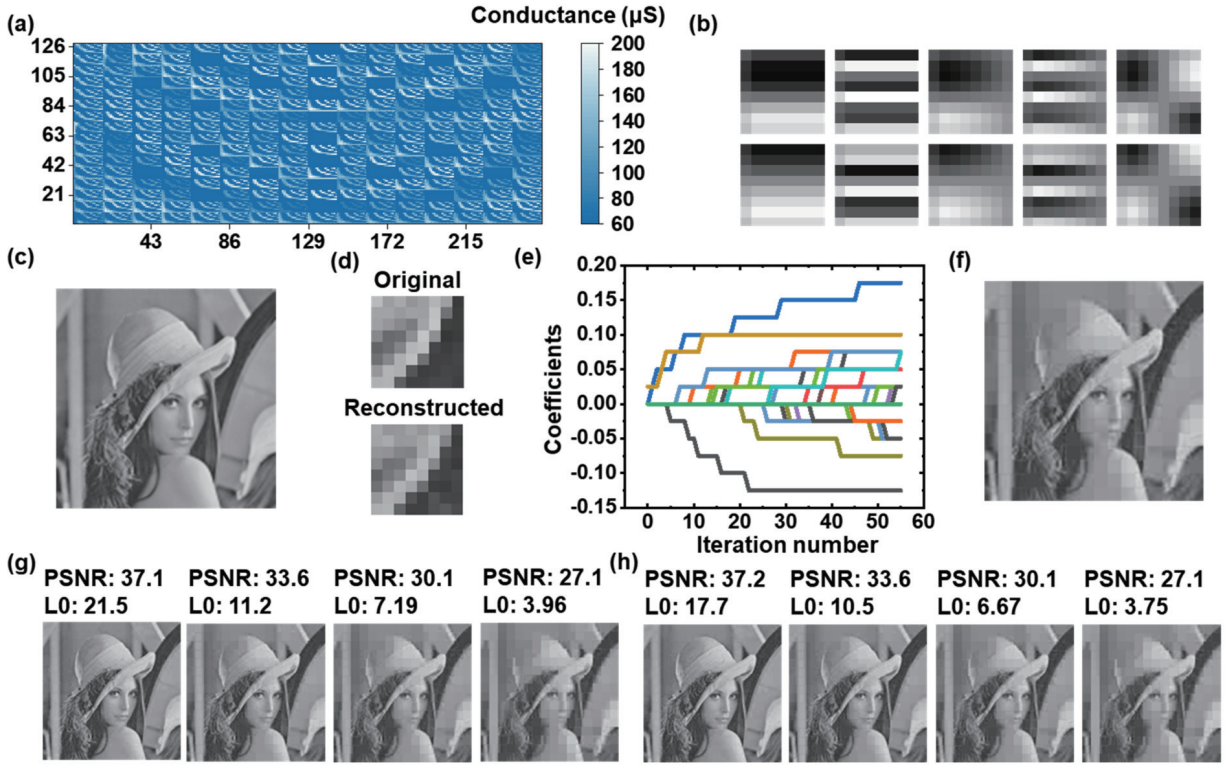

Fig. 4. (Color online) (a) The overdetermined DCT dictionary is mapped to the 128 × 256 memristor array. (b) Examples of the elements of the DCT dictionary. (c) Scheme of the original image (128 × 128). The image is divided into 8 × 8 patches for processing. (d) One patch in (c) to perform sparse coding with consideration of nonideal factors in a real circuit. (e) The dictionary element coefficient update path of (d). (f) Simulated reconstructed picture of (e), with consideration of nonideal factors in a real circuit. (g, h) In the case of adopting the DCT dictionary, the image reconstruction quality and sparsity of FSR under different thresholds (L0 is the average number of selected elements) with respect to (g) memristor-based FSR and (h) full-precision FSR.

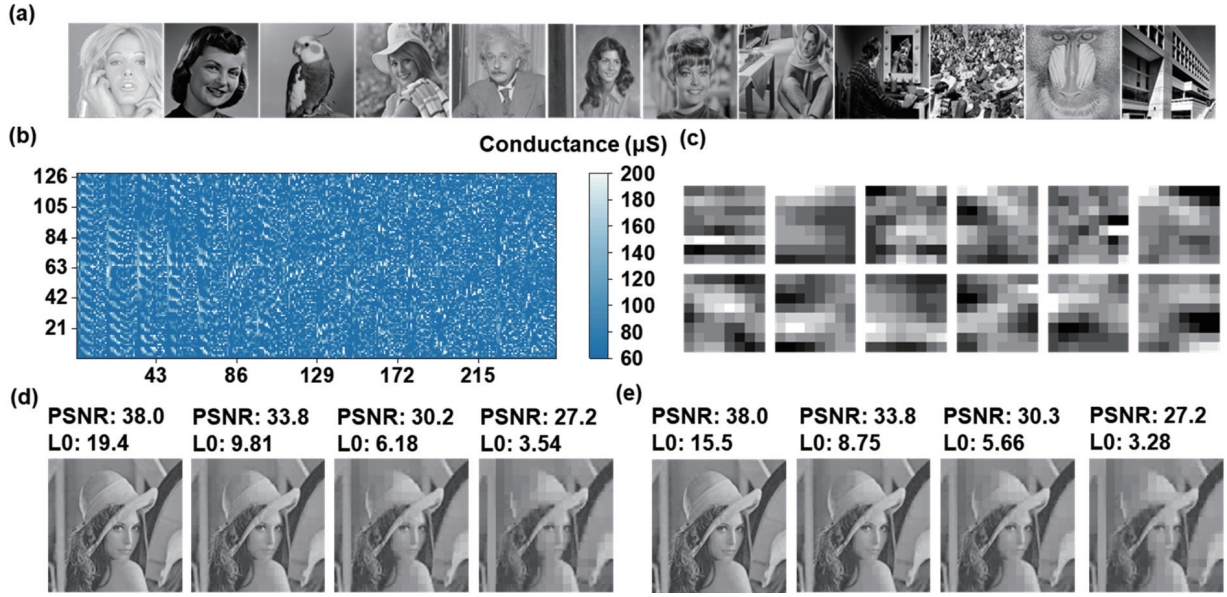

Fig. 5. (Color online) (a) Schemes of the natural pictures used to train the dictionary. (b) The offline-learned dictionary is mapped to 128 × 256 memristor array. (c) Examples of the elements of the learned dictionary. (d, e) In the case of adopting the offline-learned dictionary, the image reconstruction quality and sparsity of FSR under different thresholds with respect to (d) memristor-based FSR and (e) full-precision FSR.

precision DACs, this system can achieve similar reconstruction quality as that of the full precision ones (Fig. 5(e)). Higher precisions can further improve the result, but such improvement is not significant considering the heavy hardware burden of high-precision DACs. Then, under the condition that the voltages and conductance are both 4-bit precision, after a 64-bit MAC operation, the output results should be sampled by 14-bit ADCs to maintain full precisions. However, Fig. 6(c) shows that 10-bit ADCs are enough for this task, because the differences of the conductance obey a normal distribution and the absolute values of the voltage vectors are usually low. Finally, the robustness of this system

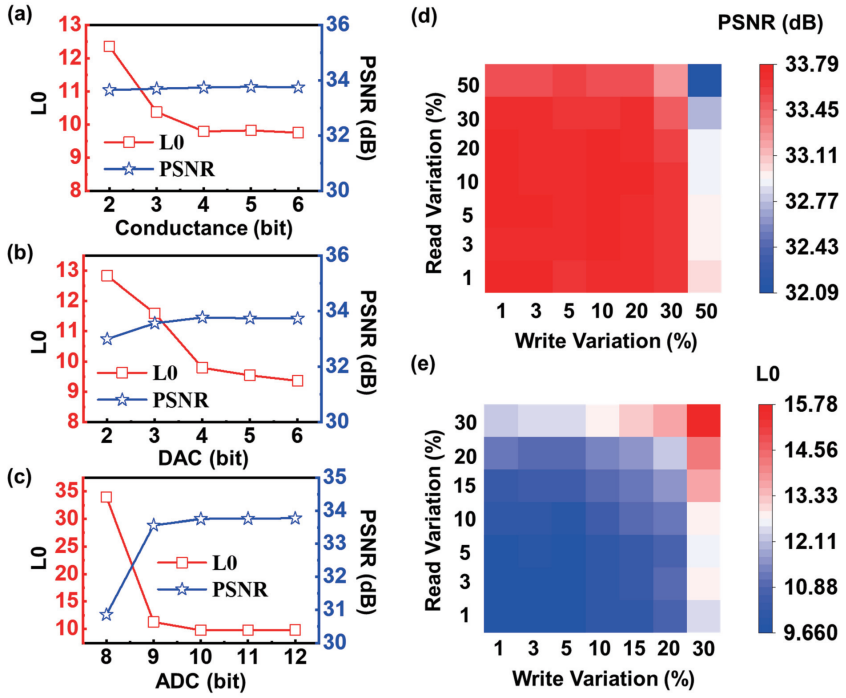

Fig. 6. (Color online) (a) The influence of conductance precision on peak-signal-to-noise ratio (PSNR) and sparsity (L0). (b) The influence of DAC precision on PSNR and sparsity. (c) The influence of ADC precision on PSNR and sparsity. (d) The robustness analysis of PSNR with device variations. (e) The robustness analysis of the sparsity with device variations.

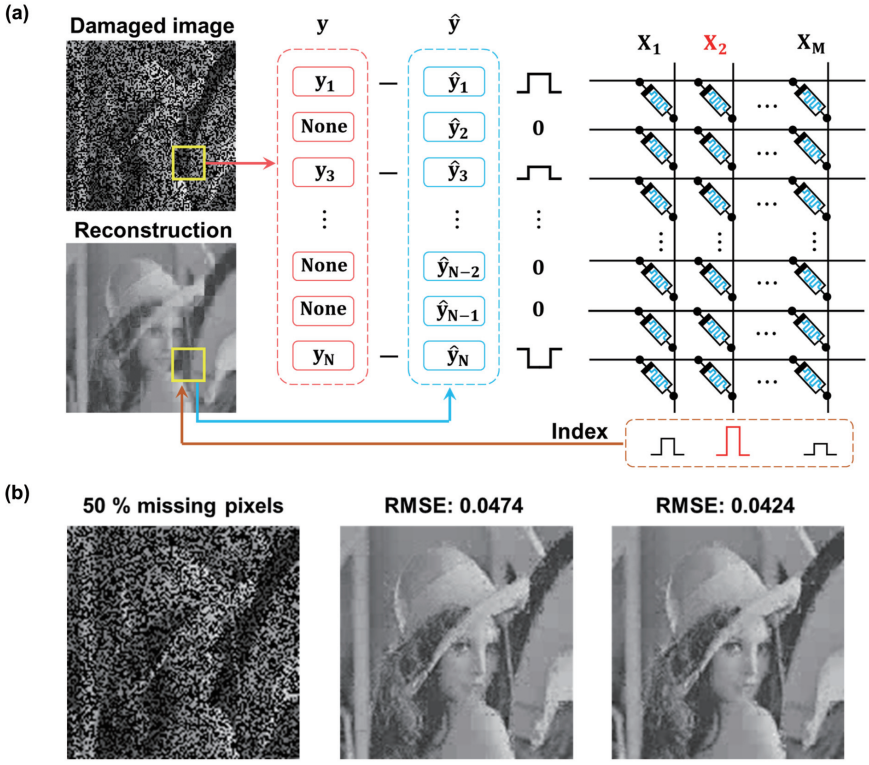

Fig. 7. (Color online) (a) The image inpainting task is performed using memristor-based sparse coding, where the array input voltage is the residual vector of remaining pixels. (b) Image restoration effect based on the DCT dictionary and learned dictionary, the middle one is based on the DCT dictionary and the right one is based on the learned dictionary.

under various device variations is analyzed. As shown in Fig. 6(d), even with 30% $(\sigma/\mu)_{write}$ and 30% $(\sigma/\mu)_{read}$, our system still achieves the same image reconstruction quality with the ideal result. But in Fig. 6(e), the sparsity drops faster, tolerating only 15% $(\sigma/\mu)_{write}$ and 15% $(\sigma/\mu)_{read}$ with the number of selected elements increased by 15%.

3.4. Image inpainting application

The purpose of image inpainting is to restore corrupted images, and it is commonly used to repair aged photos or image files that have been corrupted during data transfer. Sparse coding can capture semantic information from the remaining pixels of a corrupted image to fill in the missing

Table 1. Comparation of memristive sparse coding system.

	PSNR (dB)	L0	Patch size	Compression ratio
IEEE TNNLS (2015)[17]	~24	~40	(10 × 10)	~0.4
Nature Nano (2017)[16]	27.1	15.6	(10 × 10)	0.156
Cognit. Neurodynam (2019)[19]	33.57	/	(4 × 4)	/
This work	**33.8**	**9.81**	(8 × 8)	**0.153**
This work	**38**	**19.4**	(8 × 8)	**0.303**

parts. With the above hardware parameter design considerations, to demonstrate that the memristor-based FSR can be applied to a realistic image processing task, we apply it to complete image inpainting (a basic application of sparse coding[29]). Here, we randomly delete 50% of the pixels in the image and then applied memristor-based FSR to restore their values with the DCT dictionary and the learned dictionary, respectively. In the process, the damaged picture was first split into 8 × 8 patches. Then, in the loop iteration stage, the similarity between the residual vector y-$\hat{y}$ and the dictionary elements are computed and the coefficient of the most relevant dictionary element is updated. Moreover, the image inpainting task only computes the similarity between the residual vectors of the remaining pixels and the dictionary elements (Fig. 7(a)), while sparse coding typically computes the similarity between the residual vectors of all pixels and the dictionary elements. After a certain number of loop iterations, the image is reconstructed based on the coefficients of the dictionary elements to obtain an image filled with missing pixels. Finally, the learned dictionary achieves better image reconstruction than the DCT dictionary (Fig. 7(b)), with a reconstruction error of 0.0424 root-mean-squared error (RMSE). This indicates that the memristor-based FSR can perform the task of image inpainting.

4. Discussion

Finally, we now compare our study with similar works reported in the literature. Can *et al*. have developed a 128 × 64 memristor array for signal and image processing tasks[14]. This in-memory computing system exhibits a significantly higher energy efficiency of 119.7 TOPS/W compared to von Neumann-based chip architectures, highlighting the notable advantages of memristor-based in-memory computing systems in enhancing the efficiency of the image-processing algorithm. Sheridan *et al*. experimentally implemented a memristor-based locally competitive algorithm (LCA) for sparse coding, which theoretically can achieve 16 × improvement in power consumption than the digit CMOS system[16]. Woods *et al*. provided an LCA-like algorithm using a spiking framework, which resulted in a more energy-efficient sparse coding architecture, improving an all-CMOS ASIC with 21 × the throughput while using 99% less energy per input[18]. These systems fully demonstrate that IMC can achieve a sufficiently high energy efficiency at the data scale for sparse coding applications. This work does not outperform them in terms of energy efficiency. But we solved a common problem with their memristive sparse coding systems, the dictionary quality degrades on the memristor array, and images cannot be reconstructed with a definite dictionary. In their systems, the memristor array is employed to store the dictionary. But the variations of the memristor, which may be caused by the various device characteristics and fabrication procedures, will deteriorate one dictionary to various dictionaries in different terminals. Then the above-mentioned problem will arise and critically harm the image reconstruction, which is considered to affect image feature extraction[18]. In contrast, in this work, the memristor array is just used to approximately calculate the cosine distance between the dictionary elements and the residual vector. And the encoding results of different terminals are based on a full-precision dictionary in the digital system, which is not affected by the nonidealities of the devices. Therefore, this memristor-based system can achieve higher image reconstruction quality than the previous studies (Table 1). Specifically, the proposed system achieves the highest image reconstruction quality among memristive sparse coding systems with the same compression ratio (L0/patch size). It is worth noting that this is the first time a low-precision memristive sparse coding system has achieved an image reconstruction quality of 38 dB.

5. Conclusion

In this work, we introduced a digital-analog hybrid memristive sparse coding system (memristor-based FSR), which can reduce the time complexity of the FSR from $O(m \times n)$ to $O(m + n)$ by utilizing memristive IMC. Besides, the image reconstruction quality (PSNR) of memristive sparse coding can be improved to 38 dB. The hardware requirements of the system are low, which only requires moderate memristor conductance states ($\geqslant$ 4), write variation ($\leqslant$ 15%), read variation ($\leqslant$ 15%) and DACs ($\geqslant$ 4 bits). In the image inpainting task, this method has restored the image with 50% lost pixels, and the reconstruction error after a filling is 0.0424 RMSE. These results show that memristor-based FSR can accomplish sparse coding with high efficiency and high quality, and is promising for applications such as image recognition, anomaly detection, etc.

Acknowledgments

This work was supported by the National Key R&D Program of China (Grant No. 2019YFB2205100), and in part by Hubei Key Laboratory of Advanced Memories.

References

[1] Wright J, Ma Y, Mairal J, et al. Sparse representation for computer vision and pattern recognition. Proc IEEE, 2010, 98, 1031

[2] Aharon M, Elad M, Bruckstein A. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans Signal Process, 2006, 54, 4311

[3] Yang M, Zhang L, Yang J, et al. Robust sparse coding for face recognition. Conference on Computer Vision and Pattern Recognition, 2011, 625

[4] Lee H, Battle A, Raina R, et al. Efficient sparse coding algorithms. Proceedings of the 19th International Conference on Neural In-

formation Processing Systems, 2006, 801
[5] Mairal J, Bach F, Ponce J, et al. Online dictionary learning for sparse coding. Proceedings of the 26th annual international conference on machine learning, 2009, 689
[6] Efron B, Hastie T, Johnstone I, et al. Least angle regression. Ann Statist, 2004, 32, 407
[7] Tibshirani R J, Yu B. A general framework for fast stagewise algorithms. J Mach Learn Res, 2015, 16, 2543
[8] Hastie T, Taylor J, Tibshirani R, et al. Forward stagewise regression and the monotone lasso. Electron J Statist, 2007, 1, 1
[9] Wan W E, Kubendran R, Schaefer C, et al. A compute-in-memory chip based on resistive random-access memory. Nature, 2022, 608, 504
[10] Huo Q, Yang Y M, Wang Y M, et al. A computing-in-memory macro based on three-dimensional resistive random-access memory. Nat Electron, 2022, 5, 469
[11] Liu Q, Gao B, Yao P, et al. A fully integrated analog ReRAM based 78.4TOPS/W compute-In-memory chip with fully parallel MAC computing. 2020 IEEE International Solid-State Circuits Conference - (ISSCC), San Francisco, CA, USA, 2020, 500
[12] Wang S C, Li Y, Wang D C, et al. Echo state graph neural networks with analogue random resistive memory arrays. Nat Mach Intell, 2023, 5, 104
[13] Yao P, Wu H Q, Gao B, et al. Fully hardware-implemented memristor convolutional neural network. Nature, 2020, 577, 641
[14] Li C, Hu M, Li Y N, et al. Analogue signal and image processing with large memristor crossbars. Nat Electron, 2018, 1, 52
[15] Sun Z, Pedretti G, Bricalli A, et al. One-step regression and classification with cross-point resistive memory arrays. Sci Adv, 2020, 6, eaay2378
[16] Sheridan P M, Cai F X, Du C, et al. Sparse coding with memristor networks. Nat Nanotechnol, 2017, 12, 784
[17] Sheridan P M, Du C, Lu W D. Feature extraction using memristor networks. IEEE Trans Neural Netw Learn Syst, 2016, 27, 2327
[18] Woods W, Teuscher C. Fast and accurate sparse coding of visual stimuli with a simple, ultralow-energy spiking architecture. IEEE Trans Neural Netw Learn Syst, 2019, 30, 2173
[19] Ji X, Hu X F, Zhou Y, et al. Adaptive sparse coding based on memristive neural network with applications. Cogn Neurodyn, 2019, 13, 475
[20] Huang X D, Li Y, Li H Y, et al. Enhancement of DC/AC resistive switching performance in AlO_x memristor by two-technique bilayer approach. Appl Phys Lett, 2020, 116, 173504.
[21] Huang X D, Li Y, Li H Y, et al. Forming-free, fast, uniform, and high endurance resistive switching from cryogenic to high temperatures in $W/AlO_x/Al_2O_3/Pt$ bilayer memristor. IEEE Electron Device Lett, 2020, 41, 549
[22] Freund R M, Grigas P, Mazumder R. A new perspective on boosting in linear regression via subgradient optimization and relatives. Ann Statist, 2017, 45, 2328
[23] Cheadle C, Vawter M P, Freed W J, et al. Analysis of microarray data using Z score transformation. J Mol Diagn, 2003, 5, 73
[24] Rousseeuw P J, Leroy A M. Robust regression and outlier detection. John wiley & sons, 2005
[25] Yang B, Li S T. Multifocus image fusion and restoration with sparse representation. IEEE Trans Instrum Meas, 2010, 59, 884
[26] Schnass K, Vandergheynst P. Dictionary preconditioning for greedy algorithms. IEEE Trans Signal Process, 2008, 56, 1994
[27] Wright J, Yang A Y, Ganesh A, et al. Robust face recognition via sparse representation. IEEE Trans Pattern Anal Mach Intell, 2009, 31, 210
[28] Blanchet F G, Legendre P, Borcard D. Forward selection of explanatory variables. Ecology, 2008, 89, 2623
[29] Guillemot C, Le Meur O. Image inpainting: Overview and recent advances. IEEE Signal Process Mag, 2014, 31, 127

Chenxu Wu is currently a postgraduate student in School of Integrated Circuits at Huazhong University of Science and Technology. He received his Bachelor degree in Harbin Engineering University in 2019. His research interests mainly focus on in-memory computing.

Yibai Xue is currently a postgraduate student in School of Integrated Circuits at Huazhong University of Science and Technology (HUST). He received his Bachelor degree in HUST in 2021. His research interests mainly focus on metal oxide memristors, as well as nonvolatile memory technology.

Yi Li is currently an associate professor at Huazhong University of Science and Technology (HUST). He received his PhD degree in microelectronics from HUST in 2014. His major research interests focus on memristors and their applications in neuromorphic computing and in-memory computing.