Merging the Interface: Power, Area and Accuracy Co-optimization for RRAM Crossbar-based Mixed-Signal Computing System

Boxun Li, Lixue Xia, Peng Gu, Yu Wang, Huazhong Yang
Dept. of E.E., Tsinghua National Laboratory for Information Science and Technology (TNList),
Centre for Brain Inspired Computing Research (CBICR), Tsinghua University, Beijing, China
e-mail: yu-wang@mail.tsinghua.edu.cn

ABSTRACT

The invention of resistive-switching random access memory (RRAM) devices and RRAM crossbar-based computing system (RCS) demonstrate a promising solution for better performance and power efficiency. The interfaces between analog and digital units, especially AD/DAs, take up most of the area and power consumption of RCS and are always the bottleneck of mixed-signal computing systems. In this work, we propose a novel architecture, *MEI*, to minimize the overhead of AD/DA by *MErging the Interface* into the RRAM crossbar. An optional ensemble method, the *Serial Array Adaptive Boosting (SAAB)*, is also introduced to take advantage of the area and power saved by MEI and boost the accuracy and robustness of RCS. On top of these two methods, a design space exploration is proposed to achieve trade-offs among accuracy, area, and power consumption. Experimental results on 6 diverse benchmarks demonstrate that, compared with the traditional architecture with AD/DAs, MEI is able to save 54.63%∼86.14% area and reduce 61.82%∼86.80% power consumption under quality guarantees; and SAAB can further improve the accuracy by 5.76% on average and ensure the system performance under noisy conditions.

1. INTRODUCTION

Power Efficiency has become a major concern in modern computing system design [1]. However, as the scaling down of traditional CMOS technique is approaching the physical limit, it is becoming more and more difficult for contemporary digital systems to achieve substantial gains of power efficiency in the predictably scaled technology node. Meanwhile, the memory bandwidth required by high-performance CPUs has also increased beyond what conventional memory architectures can efficiently provide, leading to an ever-increasing "memory wall" challenge to the efficiency of von Neumann architecture [2]. Consequently, there is a growing research interest of exploring emerging nano-devices and new computing architectures to satisfy the continuously growing requirement of power efficiency [3].

In recent years, the innovation of resistive-switching random access memory (RRAM) devices and RRAM Crossbar-based computing System (RCS) provides a promising solution to significantly boost the performance and efficiency of computing systems. The ultra-high integration density enables RRAM devices to support large amounts of signal connections within a small footprint [4]. More importantly, the RRAM crossbar structure provides an innovative alternative to the von Neumann architecture by changing the architecture to combine computation and memory together and naturally realize the matrix-vector multiplication [5]. For example, an approximate computing framework based on the RCS has demonstrated hundreds of times of power efficiency gains compared with the CPU [6].

The interfaces between digital and analog units are always the key consideration of RRAM crossbar-based computing systems. The RRAM crossbar realizes matrix operations in analog, and analog-to-digital and digital-to-analog converters (AD/DAs) are usually required in the mixed-signal system to bridge the digital part and the RRAM-based analog data processing unit. However, compared with the high density and efficiency of RRAM crossbar, AD/DAs not only take up most of the chip area, but also consume much more power than RRAM devices and other analog peripheral circuits. Consequently, the overhead of conversions significantly limits the potential efficiency gains of RCS [7, 8].

In this paper, we present a novel solution to minimize the overhead of AD/DAs by merging the interface into the RRAM crossbar structure. The technique is based on the idea that we can try to make an RCS directly learn the relationship between the binary arrays, which represent the input and output digital data, instead of the analog value converted by AD/DAs. The main contributions of this work include:

- We present *MEI* (MErging the Interface), a novel architecture for reducing the interface overhead by integrating the AD/DA into the RRAM crossbar array. Experimental results of 6 benchmarks show that MEI is able to provide comparable accuracy and reduce up to 86.14% area and 86.80% power consumption of RCS. To the best of our knowledge, this is the first work that proposes to use the RRAM crossbar itself to eliminate the interface problem.
- We argue that the area and power consumption saved by MEI can be used to boost the accuracy and robustness. And we introduce *SAAB* (Serial Array Adaptive Boosting), an ensemble method customized for RCS, which is able to achieve up to 13.05% further improvement of accuracy, and can simultaneously increase the system robustness to different non-ideal factors.
- Because MEI is aimed to reduce the area and power consumption, while SAAB boosts the accuracy at the cost of consuming more energy and area, we propose a flow to explore the design space and achieve trade-offs among accuracy, area, and power consumption.

The rest of this paper is organized as follows. In Section 2, we introduce the background knowledge and a motivation example. Then Section 3 presents the proposed techniques of MEI and SAAB. A design space exploration is proposed in Section 4, and experimental results are provided in Section 5. Finally, Section 6 concludes this work.

DAC '15, June 7-11 2015, San Francisco, CA

http://dx.doi.org/10.1145/2744769.2744870 .

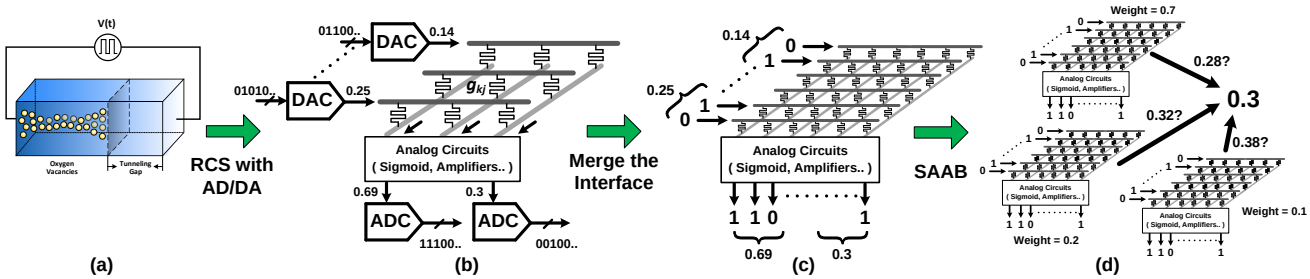

Figure 1: (a). Physical model of RRAM device. (b). RRAM crossbar-based computing system (RCS). (c). Basic idea of merging the interface (MEI). (d). Basic idea of serial array adaptive boosting (SAAB).

2. PRELIMINARIES AND MOTIVATION

2.1 RRAM Device Characteristics and RRAM Crossbar-based Computing System

An RRAM device is a passive two-port element with variable resistance states. Fig. 1(a) illustrates a 3D filament model of the HfO_x-based RRAM device [9]. Theoretically, the resistance of an RRAM device can be changed to arbitrary state within a specific range.

The RRAM devices can be used to build the crossbar structure as shown in Fig. 1(b). The relationship between the input voltage "vector" ($\vec{V_i}$) and output voltage "vector" ($\vec{V_o}$) can be expressed as follows [10]:

$$V_{o,j} = \sum_k c_{k,j} \cdot V_{i,k} \tag{1}$$

where k (k = 1,2,..,N) and j (j = 1,2,..,M) are the indexes of input and output ports of the crossbar. The parameter $c_{k,j}$ can be represented by the conductivity of the RRAM device ($g_{k,j}$) and the load resistor (g_s) as:

$$c_{k,j} = \frac{g_{k,j}}{g_s + \sum_{l=1}^{N} g_{k,l}} \tag{2}$$

Therefore, the RRAM crossbar array is able to perform analog matrix-vector multiplication and the parameters of the matrix depend on the RRAM resistance states.

With the RRAM crossbar structure, an RRAM crossbar-based computing system (RCS) can be implemented by realizing analog artificial neural networks (ANNs) [8]. Generally speaking, an ANN processes the data by executing the following operations layer by layer:

$$\vec{y_j} = f(W_{ij} \cdot \vec{x_i} + \vec{b_i}) \tag{3}$$

where $\vec{x_i}$ and $\vec{y_j}$ represent the data in the i^{th} and j^{th} layer of the network. W_{ij} is the weight matrix between *Layer* i and *Layer* j. $f(x)$ is a nonlinear activation function, e.g., the sigmoid function. It can be seen that the basic operations of an ANN are the matrix-vector multiplication and the nonlinear activation function, which can be implemented with RRAM crossbar structures and analog circuits, respectively [8]. An ANN can learn the relationship between the input and output data automatically, which makes the RCS an efficient and powerful tool to accomplish a wide range of tasks [11]. For example, an RCS can be configured as a power efficient approximate computing system by learning to fit complex numerical functions [6, 8].

2.2 Motivation

The RCS achieves significant power efficiency gains by taking advantage of the RRAM crossbar structure. As the RCS processes data in analog, an interface, such as AD/DA, is usually required to connect the RRAM crossbar-based analog accelerator to digital systems. However, compared with the high density and efficiency of RRAM analog units, the interface is drastically area and power consuming.

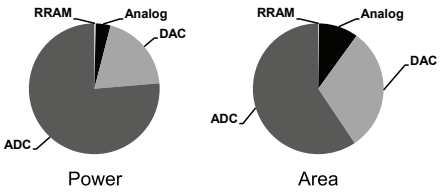

Figure 2: Normalized power and area consumption for a 2×8×2 RCS with 8-bit accuracy for robotics.

For example, Fig. 2 demonstrates the power and area consumption breakdowns for a 2×8×2 RCS[1] which can be used in robotics [7]. The data for power and area estimation are referred to Ref. [7, 12, 13, 14] after we comprehensively analyze the speed, power, and area of state-of-the-art devices. The results demonstrate that AD/DAs contribute to a significant portion (>85%) of the area and power consumption of RCS, while RRAM devices only account for ∼1% proportion of the whole system. Consequently, the potential efficiency gains of RCS are significantly limited by the interface overhead. This observation motivates us to reduce the overhead of AD/DA in RRAM crossbar-based computing systems.

3. THE PROPOSED METHOD

3.1 MEI: MErging the Interface

Fig. 1(c) demonstrates the basic idea of MEI. The technique is based on the idea that, instead of using an RCS to approximate the function between the analog value converted by AD/DA, we can try to make it directly learn the relationship between the binary 0/1 arrays which represent the input and output digital data. For example, for a traditional $2 \times 8 \times 2$ RCS equipped with 8-bit AD/DAs as described in Section 2.2, MEI will directly set $2 \times 8 = 16$ ports in both the input and output layers of RCS. Digital 0/1 signals, instead of the analog signals converted by DA, will be set to the 16 input ports in parallel, and MEI will directly calculate the corresponding 16 digital outputs. Therefore, MEI is able to directly connect to digital systems without AD/DA.

An important difference between the proposed architecture and the original RCS with AD/DAs is that, after all the input and output ports are exposed, they will be *independent* with each other and we can treat each port *differently* to increase the performance and efficiency of MEI.

To be specific, we propose to pay more attention to the ports that represent the most significant bits (MSB) of a binary number. If we can reduce of error rate of MSBs, we may significantly decrease the error rate of the whole system. This technique is realized by carefully modifying the loss function of the training algorithm. As described in Section 2.1, an RCS realizes different tasks by realizing an RRAM-based ANN. The training process of an ANN can be described as adjust-

[1] An $I \times H \times O$ RCS represents that the RCS consists of a 3-layer ANN with I nodes in the input layer, H nodes in the hidden layer and O nodes in the output layer.

ing the network weights (W_{ij} in Eq. (3)) to minimize the difference between the target and actual outputs by solving the following optimization problem [15]:

$$\min \sum_n \sum_p [t_p(n) - o_p(n)]^2 \quad (4)$$

where n represents the index of the training sample and p is the port index of the output "vector". t_p is the target output of the network, e.g., the label of the input training sample. And o_p is the actual output of the network by executing a series of Eq. (3).

In order to suppress the error rate of the MSB and increase the accuracy of MEI, we revise the loss function of the training optimization problem as follows:

$$\min \sum_n \sum_p [w_p \cdot (t_p(n) - o_p(n))]^2 \quad (5)$$

where w_p is the weight of each output port. We set larger weights to the MSBs while the least significant bits (LSBs) will be given smaller weights. For example, we exponentially increase the weight of each bit and set the MSB and LSB weights in an 8-bit output array to 2^0 and 2^{-7}, respectively. By using the revised loss function, an error from the MSB will lead to a much lager penalty than the error of LSB. So the training process will put more effort to suppress the MSB error and the accuracy of MEI will be improved.

Fig. 3 demonstrates a comparison of different architecture performance. We use a $1 \times N \times 1$ RCS to perform approximate computing by fitting the calculation of $f(x) = \exp(-x^2)$ [6, 7]. We randomly generate 10,000 samples within the range of (0, 1) to train the RCS and test it with another 1,000 samples. It can be seen that the revised training algorithm not only helps significantly improve the accuracy of the proposed architecture, but also can even achieve better performance than the traditional RCS with AD/DAs.

A major side effect of MEI is that the number of input and output ports and the crossbar size will increase massively. However, due to the ultra-high integration density and efficiency of RRAM devices, which can be more than hundreds of times better than AD/DA, MEI still significantly reduces the area and power consumption of RRAM crossbar-based computing systems. A minor problem is that the outputs of the proposed architecture are continuous analog signals. We use flip-flop buffers or analog comparators (to work as 1-bit ADCs) to convert them to discrete binary digital signals.

3.2 SAAB: Serial Array Adaptive Boosting

With MEI, we can save the area and power consumption of AD/DAs. We argue that we can use these saved resources to integrate more RRAM devices and analog peripheral circuits and further boost the accuracy and robustness of the RCS. The first step is to choose a proper method. The experimental results of MEI provide us with the following observations:

- As shown in Fig. 3, although the proposed architecture may perform better than the traditional RCS with AD/DAs, it may also require a larger hidden layer to support more output ports and achieve a good result.
- It can be also observed that there's a bottleneck of the RCS performance. The accuracy begin to stall after we increase the size of the hidden layer to a certain value.
- The RRAM devices may suffer from different non-ideal factors [10], such as the signal fluctuation and the process variation. The robustness of an RCS is important for physical realization.

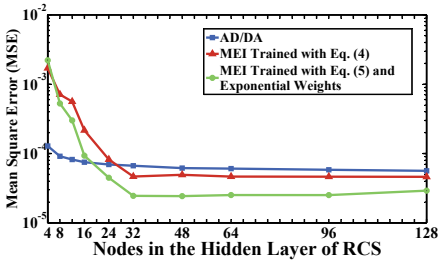

Figure 3: Comparison of the performance of different architectures when approximating $f(x) = \exp(-x^2)$.

Algorithm 1: Serial Array Adaptive Boosting

Input: Training Samples: $X = \{(\vec{x}_1, \vec{y}_1), ..., (\vec{x}_N, \vec{y}_N)\}$,
Bits for Comparison: B_C, Boost Times: K, Non-Ideal Factors: $\vec{\sigma}$
Output: Trained RCSs $\{R_1, ..., R_K\}$ with weights $\{\alpha_i, ..., \alpha_K\}$

1 Initialize weights of training samples $w_n = 1/N$;
2 **for** $k = 1 \rightarrow K$ **do**
3 Normalize the distribution of samples: $p_n = w_n / \sum_n w_n$;
4 Generate training samples s_k with X and distribution p_n;
5 Train the RCS R_k with s_k;
6 Evaluate the error rate of R_k with Non-Ideal Factors $\vec{\sigma}$: $\varepsilon_k = \sum_n p_n \cdot [R_k(\vec{x}_n, \vec{\sigma})^{B_C} \neq \vec{y}_k^{B_C}]^*$;
7 Calculate the weight of R_k: $\alpha_k = \frac{1}{2} \ln[\frac{1-\varepsilon}{\varepsilon}]$;
8 Update weights of training samples:

$$w_n = w_n \times \begin{cases} e^{-\alpha_k} & \text{if the first B bits of } R_k(\vec{x}_n, \vec{\sigma}) \text{ are correct} \\ e^{\alpha_k} & \text{else} \end{cases}$$

9 **end**
10 Output: $h(\vec{x}) = \arg\max_{\vec{y} \in Y} \sum_k \alpha_k \cdot [R_k(\vec{x}) = \vec{y}]$;
11 * $[R_k(\vec{x}_n, \vec{\sigma})^B \neq \vec{y}_k^B]$ represents the operation of comparing the most significant B bits of $R_k(\vec{x}_n, \vec{\sigma})$ and $\vec{y}_k$.

Based on the above observation, the ensemble method becomes a promising solution to boost the accuracy and robustness of the RCS. Compared with the traditional redundancy method, an ensemble method uses a series of learning machines (learners) with different parameters to provides better results. We propose SAAB, which uses Serial Array to Adaptive Boost the performance of an RRAM crossbar based computing system as shown in Fig. 1(d).

To be specific, SAAB is inspired by the AdaBoost method [16]. The basic idea of AdaBoost, which is also its major advantage, is to train a series of learners sequentially, and every time we train a new learner, the method will increase the weights of examples that are incorrectly classified by previous trained learners and force the new learner to focus on these "hard" examples with wrong answers in the training set. Compared with the original AdaBoost, SAAB is customized to MEI by relaxing the error calculation, focusing on the MSBs and introducing the impact of non-ideal factors.

Algorithm 1 demonstrates the basic flow of SAAB. We maintain a distribution (p_n) of training samples according to their weights (w_n), which reflect the "hardness" of a sample, i.e., a larger weight will represent that the corresponding sample is more likely to be misclassified by previous learners. Each time we need to train a new RCS, we use this distribution to generate a customized training data (s_k), where the "hard" examples that are incorrectly classified by previous learners will have a greater proportion in the training set to make the new learner (R_k) pay more attention to these samples ($Line\ 4-5$). After the training process of a learner finishes, the algorithm will test the learner's performance (ε_k), calculate its contribution to the system (α_k), and update the weights w_n according to the training results ($Line\ 6-8$).

In order to enhance the robustness of RCS, in $Line\ 6$, we also introduce the non-ideal factors when evaluating the performance of a trained RCS to find out the "sensitive" samples as well as the "hard" ones under noisy conditions. Moreover,we relax the error calculation by only comparing the most significant B_C bits, e.g., the first 4-6 bits in an 8-bit array, of

the RCS in *Line* 6. Otherwise, most of the training samples will be either sensitive or "hard" in the algorithm, and the performance of SAAB may significantly decrease.

Finally, SAAB will provide a balanced output by a weighted voting of different RCSs as described in *Line* 10. Compared with the training process, in which a series of RCSs are configured in sequence, each trained RCS can predict the output of a given input in parallel. And the weighted voting operation of the outputs of different RCSs can be executed by the digital system directly connected to these RRAM crossbar-based computing systems.

4. DESIGN SPACE EXPLORATION

4.1 Area and Power Estimation

The area and power consumption are both mainly determined by the architecture and the size of ANN in an RCS. For example, the area of a traditional $I \times H \times O$ RCS with AD/DAs can be estimated as follows:

$$A_{\text{org}} \approx I \cdot A_{\text{DA}} + O \cdot A_{\text{AD}} + H \cdot A_{\text{P}} + 2(I+O)H \cdot A_{\text{R}} \quad (6)$$

where A_{DA}, A_{AD}, A_{P}, A_{R} are the circuit size for a cell of DAC, ADC, analog peripheral circuits and RRAM devices, respectively. The area of RRAM devices are doubled because two crossbar are required to represent a matrix with both positive and negative parameters as described in Section 2.1.

And for the proposed $I' \times H' \times O'$ RCS with MEI and B-bit accuracy, the area estimation should be modified to:

$$A_{\text{MEI}} \approx H' \cdot A_{\text{P}} + B \cdot 2(I'+O')H' \cdot A_{\text{R}} \quad (7)$$

Moreover, Eq. (6) & (7) can also be used to evaluate the power consumption by replacing the area parameters, such as A_{AD} and A_{DA}, with parameters for power estimation.

4.2 Accuracy and Robustness Evaluation

As RCS is based on the analog realization of ANN, the accuracy and robustness of an RCS are usually both determined by the scale of the system [15]. Therefore, we discuss them together when exploring the design space.

As discussed in Section 3.2, there are two methods to scale up an RCS which may result in accuracy and robustness improvement of the system: 1). increasing the scale of a single RCS; and 2). combining several RCSs together with SAAB. Because the dimensions of input and output data are usually determined by the application, the size of the hidden layer in an RCS, and the number of RCSs combined with SAAB, become the two major parameters that can be configured to boost accuracy and robustness in the design space.

It should be noted that, it will be very difficult, if not impossible, to directly predict that SAAB will achieve better accuracy or robustness to non-ideal factors than the increasing-hidden-layer method. Therefore, we keep both the methods when exploring the design space.

4.3 Exploring the Design Space

Because MEI is aimed to reduce the area and power consumption of an RCS, while SAAB boost the accuracy at the cost of consuming more power and area, there will be a trade-off among power, area, and accuracy in an RCS equipped with MEI and SAAB. Therefore, we propose a design space exploration flow to help convert a traditional RCS to the proposed architecture with MEI and achieve better trade-offs.

Algorithm 2 demonstrates the technical flow for exploring the design space. As shown in *Line* 1, for each specific application, the first step is to determine a proper hidden layer size of a single RCS. Inspired by the results shown in Fig. 3, we search a proper hidden layer size by gradually increasing the size (with linear or exponential searching steps) until the absolute change rate of the error rate or accuracy becomes

Algorithm 2: Design Space Exploration

```
   Input: Training and Testing Samples: X and T;
   Initial RCS Size: I × H_i × O;
   Required Bit-Length: B_r; Non-Ideal Factors: σ⃗;
   Error Rate Requirement: ε; Robustness Requirement: γ;
   Output: Trained RCS R with a size of B_in I × H × B_out O;
1  Search a proper Hidden Layer Size H from H_i for the RCS;
2  Calculate the Maximum SAAB Number K_max with H and
   Eq. (9);
3  Train the RCS R_1 with a size of B_r I × H × B_r O with X;
4  Test the error rate ε_s and robustness γ_s of R_1 with T and σ⃗;
5  if ε_s < ε && γ_s > γ then
6  |  R ← R_1;
7  end
8  else
9  |  Calculate α_1, the weight of R_1, with σ⃗ as Algorithm 1;
10 |  K ← 1;
11 |  while ε_s > ε || γ_s < γ do
12 |  |  K++;
13 |  |  if K > K_max then
14 |  |  |  Return Mission Impossible;
15 |  |  end
16 |  |  Train R_K and α_K as Algorithm 1;
17 |  |  Test the error rate ε_s and robustness γ_s of the ensemble
   |  |  of {R_1, ..., R_K} with weights {α_1, ..., α_K} with T and σ⃗;
18 |  |  Train an RCS R_0 with a size of B_r I × HK × B_r;
19 |  |  Compare the error rate and robustness of R_0 and the
   |  |  ensemble of {R_1, ..., R_K}, and set H and R according to
   |  |  the better one;
20 |  end
21 end
22 Prune the least significant bits in the input and output layer of R
   to B_in and B_out;
23 Return R;
```

lower than a certain value (e.g. 5%). The absolute change rate of the i^{th} training result can be defined as follows:

$$\eta = |\frac{\epsilon_i - \epsilon_{i-1}}{\epsilon_{i-1}}| \quad (8)$$

where ϵ can be error rate, mean square error (MSE) or any other index that can be used to evaluate the performance of a trained RCS.

After a proper hidden layer size is determined, the maximum number of RCSs that can be used for SAAB can be estimated to reduce the design space as described in *Line* 2. In our design, both the circuit area and power consumption of an RCS with MEI should not be larger than the original architecture with AD/DA. Therefore, the maximum SAAB number will be bounded by the following expression:

$$K_{\text{SAAB}\max} = \min\{\frac{A_{\text{org}}}{A_{\text{MEI}}}, \frac{P_{\text{org}}}{P_{\text{MEI}}}\} \quad (9)$$

where $K_{\text{SAAB}\max}$ is the maximum number of RCSs can be used in SAAB. P_{org} and P_{MEI} are the power consumption estimation for the original and proposed architectures, respectively, and they can be estimated as Eq. (6) & (7).

After achieving the basic RCS scale and the maximum SAAB number, the algorithm will begin to explore the design space by gradually adding new learner to the organization of RCSs with SAAB until the requirements of accuracy and robustness (such as the error rate under noisy conditions) are both satisfied (*Line* $13-17$).

As discussed in Section 4.2, we retain both SAAB and the increasing-hidden-layer method to enhance the robustness of an RCS. Therefore, in *Line* $18-19$, the algorithm will also train a single RCS with the same hidden layer size of K trained RCSs with SAAB, and then compare their performance. The algorithm will select a better one as the final output candidate. In addition, as an $I \times HK \times O$ RCS can save $2(K-1)O$ RRAM devices and $(K-1)O$ analog peripheral circuits in the output ports compared with K RCSs with a size of $I \times H \times O$, the increasing-hidden-layer method will be preferred if the performance of these two methods are similar.

Table 1: Benchmark Description and Results

Name	Type	Digital/ AD/DA Topology	Pruned MEI Topology	MSE Digital ANN	MSE AD/DA RCS	MSE MEI RCS	Error Metric	Error Digital ANN	Error AD/DA RCS	Error MEI RCS	Area Saved	Power Saved
FFT	Signal Processing	1×8×2	(1·7)×16×(2·8)	0.0046	0.0071	0.0052	Average Relative Error	6.03%	10.72%	8.87%	74.24%	87.23%
Inversek2j	Robotics	2×8×2	(2·8)×32×(2·8)	0.0038	0.0053	0.0067	Average Relative Error	6.57%	9.07%	10.45%	54.63%	73.73%
Jmeint	3D Gaming	18×48×2	(18·6)×64×(2·1)	0.0117	0.0258	0.0262	Miss Rate	7.19%	9.50%	9.96%	69.67%	61.82%
JPEG	Compression	64×16×64	(64·6)×64×(64·7)	0.0081	0.0153	0.0142	Image Diff	6.89%	11.44%	9.73%	86.14%	79.58%
K-Means	Machine Learning	6×20×1	(6·6)×32×(1·8)	0.0052	0.0081	0.0094	Image Diff	3.59%	7.59%	8.13%	67.00%	70.25%
Sobel	Image Processing	9×8×1	(9·6)×16×(1·1)	0.0024	0.0028	0.0026	Image Diff	3.71%	4.00%	3.77%	85.99%	86.80%

A special technique in the algorithm is that we propose to prune the least significant bits (LSBs) of a trained RCS to further reduce the area and power consumption. Traditional AD/DAs require time and energy to convert analog signals from/to the LSBs of the output/input binary number. This operation is fixed in the AD/DA architecture but may contribute little to the performance of RCS. However, thanks to MEI, each individual bit of the interface is exposed independently and this gives us a chance to easily remove the bits of little importance. Therefore, we propose to prune the LSBs of the input and output ports of MEI and this technique is demonstrated in *Line* 22.

To be specific, for the input ports, we treat all groups of the input ports the same and keep reducing the LSBs of each group together until the performance becomes worse than the requirement. For example, an original RCS may require 3 analog input ports. And there will be 3 groups of 8 input ports, i.e., 24 input ports in total, for a proposed architecture with 8-bit accuracy. We will try to remove the ports for the least significant 1, 2, ... bits of each group simultaneously, test the pruned architecture's performance, and finally reach the minimum size of the input layer. The pruning of the output layer is much easier and will be executed after the size of the input layer is determined. The algorithm will first compare the accuracy of the LSBs and the performance, such as the mean square error (MSE), of the trained network, and then try prune to the LSBs whose weights are much smaller than the RCS error. For example, the LSB of an 8-bit fixed-point binary number may account for a value of 2^{-8}, and we can try to remove it once the MSE of the RCS is $\sim 2^{-10}$ or larger. It should be noted that the algorithm only prunes the LSBs of a given bit-length (B_r), and we set the bit-length to the same as AD/DA in this work.

Finally, by combing the above steps, we can convert a traditional RCS to MEI and SAAB, and achieve trade-offs among accuracy, area, power consumption and even robustness.

5. EXPERIMENTAL RESULTS

5.1 Experimental Setup

In the experiment, 6 different benchmarks from a wide range of applications are used to evaluate performance of the proposed method. The benchmarks are the same as that described in Ref. [1, 7] and they are used to test the performance of an analog neural processing unit. The data for the area and power estimation of analog peripheral circuits RRAM devices and are taken from Ref. [7, 12, 13, 14] as discussed in Section 2.2. For the accuracy and robustness emulation, an RRAM device model packed in Verilog-A [9] is used to build up the SPICE-level crossbar array. We choose the 90nm technology node to build the interconnection of the crossbar array and reduce the impact of IR drop [17]. The configuration of RCS is referred to Ref. [6, 7, 8], where both the accuracy of AD/DA and the basic bit-length requirement of MEI (B_r) are both set to 8-bit.

5.2 Results of MEI and SAAB

Table 1 summarizes the results of different methods. In order to reflect the performance of MEI, SAAB is not used in this section. The 'Digital' method refers to an ideal ANN performed by CPU with 32-bit floating-point numbers, and the 'AD/DA' method represents the traditional RRAM crossbar-based computing system with 8-bit AD/DAs as interfaces. For the 'Pruned MEI Topology', a $(D \cdot B)$ value represents that there are D groups of B ports in the input/output layer, and each group stands for the most significant B bits of a binary number.

Compared with the traditional architecture, the proposed technique significantly reduces more than half of the area and power consumption in all the 6 benchmarks. It can be seen that our method can achieve approximate, or even better, error rate compared with the traditional architecture. Moreover, although the required RRAM devices and analog peripheral circuits both increased in the proposed architecture, this overhead is still well compensated by the high density and efficiency of RRAM devices compared with the AD/DA interface. More specifically, the proposed merging the interface method demonstrates to benefit more to the application with a larger ratio of the interface size to the hidden layer size, such as the 'JPEG' and 'Sobel'.

On the contrary, for the application like 'Inversek2j' and 'Jmeint', where more hidden nodes are required in the RCS, the gains of the proposed method may decrease. Finally, the topology results of MEI, except the 'Inversek2j', demonstrate that the LSBs in both input and output ports of many applications can be pruned to further reduce the area and power consumption, which verifies the feasibility and effectiveness of the proposed design space exploration flow.

Fig. 4 illustrates the comparison of different methods[2]. MEI cannot achieve better performance for all benchmarks. It seems that MEI will be more suitable for the application

[2]In SAAB method, we boost the system performance with the Maximum SAAB Number. For example, the area and power saved in the 'JPEG' benchmark are 86.14% and 79.58%, and we use 4 RCSs in SAAB according to Eq. (9).

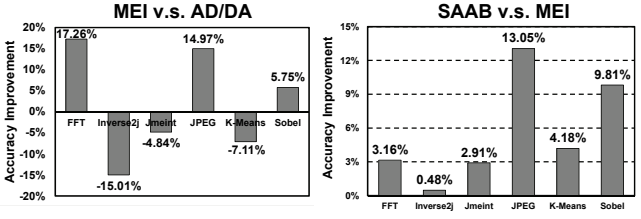

Figure 4: Comparison of different methods.

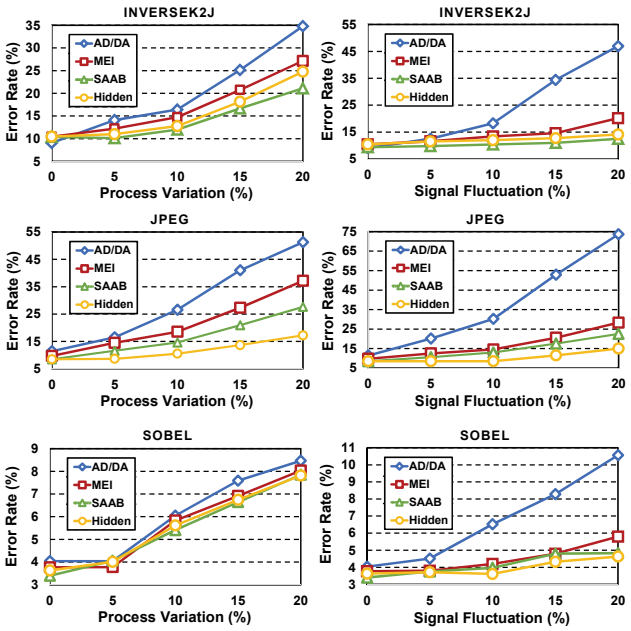

Figure 5: System performance under different noisy conditions.

where the output changes more "slowly" with the input, especially for the LSBs, like 'JPEG'. And for the application like 'Inversek2j' in which many LSBs in the output results change sensitively with the input data, the relationship between the input and output 0/1 binary array may be much more complex than that between the converted analog value. It will be more difficult for MEI to approximate the relationship better than the traditional architecture with AD/DA. In that case, the performance of MEI may be worse than 'AD/DA'. However, although the accuracy may decrease, the performance of MEI is still within the acceptable range and may be compensated by increasing the bit requirement of MEI to from 8 10, 12 or a higher level. Moreover, compared with MEI, SAAB is able to further boost the accuracy of all the benchmarks with an average improvement of 5.76%.

5.3 Impact of Non-Ideal Factors

We also evaluate the impact of different non-ideal factors to the proposed architecture and compare it with previous methods. In this paper, we mainly focus on two major non-ideal factors in the RRAM crossbar-based computing systems: the process variation (PV) and the signal fluctuation (SF) [10]. The process variation reflects the degree of the RRAM device deviating from the required resistance state, and the signal fluctuation represents the impact of noise to the electrical signal, such as the input signal. To fully demonstrate the impact of the these non-ideal factors, the lognormal distribution is used to generate variations of different levels. Under each noisy condition, we evaluate the system performance 1,000 times and statistically analyze the average result. The simulation results[3] are demonstrated in Fig. 5.

It can be seen that both SAAB and the increasing-hidden-layer method can improve the robustness of the RCS to non-ideal factors. But as discussed in Section 3.2, it is difficult to predict which method will perform better than the other in each specific application. For example, SAAB benefit more to the 'Inversek2j' benchmark while the increasing-hidden-layer method is more suitable for the 'JPEG'. And in 'Sobel', they perform almost the same. This result motivates us to keep both methods in the design space exploration (*Line* $18-19$ in Algorithm 2). In addition, as MEI only requires discrete inputs of 0/1 signals, the proposed architecture with MEI demonstrates much better robustness to the signal fluctuation than the traditional method with AD/DA. Such results suggest that the MEI architecture will be more easier to be physically realized.

[3]We evaluate all the 6 benchmarks and 3 group of results are presented in this paper as they are enough to reflect all the simulation results.

6. CONCLUSION AND FUTURE WORK

The RRAM crossbar-based computing system (RCS) provides a promising solution to boost performance and power efficiency. In this paper, we propose a novel architecture, *MEI*, as well as an ensemble method, *SAAB*, to minimize the overhead of AD/DA and further improve the system performance. A design space exploration is introduced to help convert a traditional RCS to the proposed architecture and achieve trade-offs among accuracy, area, and power consumption. We envision that MEI and SAAB can be generally applied to a broad range of applications and a large number of followup work, such as reducing the IR drop for a larger RCS under smaller technology node, are needed for this emerging architecture. In addition, we set the basic bit-length of MEI according to AD/DA, e.g. 8-bit, in order to convert a traditional RCS to the proposed architecture. In future work, we may directly use higher bit-level or even floating-point format in MEI to further improve the system performance.

Acknowledgment

This work was supported by 973 Project 2013CB329000, National Natural Science Foundation of China (No. 61373026), Brain Inspired Computing Research, Tsinghua University (20141080934), Tsinghua University Initiative Scientific Research Program, the Importation and Development of High-Caliber Talents Project of Beijing Municipal Institutions, and Huawei Technologies.

7. REFERENCES

[1] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Neural acceleration for general-purpose approximate programs," in *MICRO*, 2012, pp. 449–460.

[2] S. A. McKee, "Reflections on the memory wall," in *Proceedings of the 1st conference on Computing frontiers*. ACM, 2004.

[3] B. Liu *et al.*, "Reduction and ir-drop compensations techniques for reliable neuromorphic computing systems," in *ICCAD*, 2014.

[4] C. Xu, X. Dong, N. P. Jouppi, and Y. Xie, "Design implications of memristor-based rram cross-point structures," in *DATE*, 2011.

[5] M. Versace and B. Chandler, "Moneta: a mind made from memristors," *IEEE Spectrum*, vol. 47, no. 12, pp. 30–37, 2010.

[6] B. Li, Y. Shan *et al.*, "Memristor-based approximated computation," in *ISLPED*, 2013, pp. 242–247.

[7] R. St. Amant, A. Yazdanbakhsh *et al.*, "General-purpose code acceleration with limited-precision analog computation," in *ISCA*, 2014, pp. 505–516.

[8] X. Liu, M. Mao *et al.*, "A heterogeneous computing system with memristor-based neuromorphic accelerators," in *IEEE High Performance Extreme Computing (HPEC)*, 2014.

[9] S. Yu, B. Gao *et al.*, "A low energy oxide-based electronic synaptic device for neuromorphic visual systems with tolerance to device variation," *Advanced Materials*, 2013.

[10] M. Hu, H. Li *et al.*, "Hardware realization of bsb recall function using memristor crossbar arrays," in *DAC*, 2012, pp. 498–503.

[11] B. Li, Y. Wang *et al.*, "Ice: inline calibration for memristor crossbar-based computing engine," in *DATE*, 2014, pp. 184–187.

[12] Y. Deng, H.-Y. Chen *et al.*, "Design and optimization methodology for 3d rram arrays," in *IEEE International Electron Devices Meeting (IEDM)*, 2013, pp. 9–11.

[13] W.-H. Tseng and P.-C. Chiu, "A 960ms/s dac with 80db sfdr in 20nm cmos for multi-mode baseband wireless transmitter," in *VLSI Circuits Digest of Technical Papers, 2014 Symposium on*. IEEE, 2014, pp. 1–2.

[14] J. Proesel, G. Keskin *et al.*, "An 8-bit 1.5 gs/s flash adc using post-manufacturing statistical selection," in *CICC*, 2010.

[15] C. M. Bishop *et al.*, "Neural networks for pattern recognition," 1995.

[16] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*. CRC Press, 2012.

[17] ITRS, "International technology roadmap for semiconductors," 2013.