Energy Efficient RRAM Spiking Neural Network for Real Time Classification

Yu Wang[1], Tianqi Tang[1], Lixue Xia[1], Boxun Li[1], Peng Gu[1], Hai Li[2], Yuan Xie[3], Huazhong Yang[1]
[1] Dept. of E.E., Tsinghua National Laboratory for Information Science and Technology (TNList),
Centre for Brain Inspired Computing Research (CBICR), Tsinghua University, Beijing, China
[2]Dept. of E.C.E., University of Pittsburgh, Pittsburgh, USA
[3]Dept. of E.C.E., University of California at Santa Barbara, California, USA
e-mail: yu-wang@mail.tsinghua.edu.cn

ABSTRACT

Inspired by the human brain's function and efficiency, neuromorphic computing offers a promising solution for a wide set of tasks, ranging from brain machine interfaces to real-time classification. The spiking neural network (SNN), which encodes and processes information with bionic spikes, is an emerging neuromorphic model with great potential to drastically promote the performance and efficiency of computing systems. However, an energy efficient hardware implementation and the difficulty of training the model significantly limit the application of the spiking neural network. In this work, we address these issues by building an SNN-based energy efficient system for real time classification with metal-oxide resistive switching random-access memory (RRAM) devices. We implement different training algorithms of SNN, including Spiking Time Dependent Plasticity (STDP) and Neural Sampling method. Our RRAM SNN systems for these two training algorithms show good power efficiency and recognition performance on realtime classification tasks, such as the MNIST digit recognition. Finally, we propose a possible direction to further improve the classification accuracy by boosting multiple SNNs.

1. INTRODUCTION

The era of Big Data brings new chances and new challenges in many fields especially for the applications needing real-time data processing such as the EEG classification, tracking, etc[1, 2]. These applications demonstrate huge demands for more powerful platforms with higher processing speed, lower energy consumption, and more intelligent mining algorithms. However, the classic "scaling down" method is approaching the limit, making it more and more difficult for CMOS-based computing systems to achieve considerable improvements from the device scaling [3]. Moreover, 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. Therefore, innovation in both device technology and computing architecture is required to overcome these challenges.

The spiking neural network (SNN) is an emerging model which encodes and processes information with sparse time-encoded neural signals in parallel [4]. As a bio-inspired architecture abstracted from actual neural system, SNN not only provides a promising solution to deal with cognitive tasks, such as the object detection and speech recognition, but also inspires new computational paradigms beyond the von Neumann architecture and boolean logics, which can drastically promote the performance and efficiency of computing systems [5, 6].

However, an energy efficient hardware implementation and the difficulty of training the model remain as two important impediments that limit the application of the spiking neural network.

On the one hand, we need an applicable computing platform to utilize the potential ability of SNN. IBM proposes a neurosynaptic core named TrueNorth [7]. To mimic the ultra-low-power processing of brain, TrueNorth uses several approaches to reduce the power consumption. Specifically, TrueNorth uses digital messages between neurons to reduce the communication overhead and event-driven strategy to further save the energy computation [6]. However, the CMOS based implementation still has some limitations that are hard to avoid, while some RRAM's inherent advantages can overcome these difficulties. First, on-chip SRAM, where the synapse information is stored, is a kind of volatile memory with considerable leakage power, while RRAM is non-volatile with very low leakage power [8]. Another limitation is that TrueNorth may still needs adders to provide the addition operation of neuron function, but RRAM crossbar can do the addition, or the matrix-vector multiplication, with ultra-high energy efficiency by naturally combing the computation and memory together [9, 10, 11]. Consequently, RRAM shows potential on implementing low-power spiking neural network.

On the other hand, from the perspective of algorithm, the efficient training of SNN and mapping a trained SNN onto neuromorphic hardware presents unique challenges. Recent work of SNN mainly focuses on increasing the scalability and level of realism in neural simulation by modeling and simulating thousands to billions of neurons in biological real time [12, 13]. These techniques provide promising tools to study the brain but few of them support practical cognitive applications, such as the handwritten digit recognition. Even TrueNorth [10] uses seven kinds of applications to ver-

GLSVLSI'15, May 20–22, 2015, Pittsburgh, PA, USA.

http://dx.doi.org/10.1145/2742060.2743756 .

ify its performance, but the training and mapping methods for spike-oriented network are not discussed in detail. In other words, the mapping problem and efficient training method for SNN, especially for the real-world applications, to achieve an acceptable cognitive performance is severely demanded.

These two problems are always coupled together and only by overcoming these two challenges can we actually utilize the full power of SNN for realtime data processing applications. In this paper, we discuss these two problems with the RRAM based system architecture and two different offline training algorithms of SNN. We use the MNIST digit recognition task[14] as an application example for the realtime classification. The goal of this paper is to design a RRAM-based SNN system with higher classification accuracy and to analyze its strengths and weaknesses compared with other possible implementations.

The main contribution of this paper includes:

1. We compare different training algorithms of spiking neural networks for practical cognitive tasks, including the unsupervised Spike Timing Dependent Plasticity (STDP), and the supervised method, i.e, the Neural Sampling learning method. For STDP, we can NOT provide an acceptable cognitive performance, while the performance of Neural Sampling method is comparable to ANN systems for the MNIST digit recognition task.

2. We propose an RRAM-based implementation of different architectures of spiking neural networks. The RRAM implementation mainly includes an RRAM crossbar array working as network synapses, an analog design of the spiking neuron, an input encoding scheme, and an mapping algorithm to configure the RRAM-based spiking neural network.

3. We compare the power efficiency and recognition performance of SNN and the RRAM-based artificial neural network (ANN). The experiment results show that ANN can outperform SNN on the recognition accuracy, while SNN usually requires less power consumption. Based on these results, we discuss the possibility of using boosting methods, which combine some weak SNN learners together, to further enhance the recognition accuracy for real-world application.

The rest of this paper is organized as follows: Section II introduces the background for SNN and RRAM-based hardware architecture. We propose two kinds of RRAM-based SNN architectures with two different training algorithms and discuss the potential possibility to boost the recognition accuracy by combining multiple SNNs together in Section III. Finally, Section IV concludes the paper and proposes some future directions for RRAM-based SNN systems.

2. BACKGROUND

The spiking neural network consists of layers of spiking neurons connected by weighted synapses as shown in Fig. 1 [10]. The input data, such as images, will be encoded into the spike trains and then sent into the network. The output of the network can be treated as another representation of the input data, e.g., the corresponding classification results of the input images. According to Fig. 1 , the two most important operations in SNN are (1) the nonlinear function made by spiking neurons; (2) the matrix-vector multiplication based on the synapse weights.

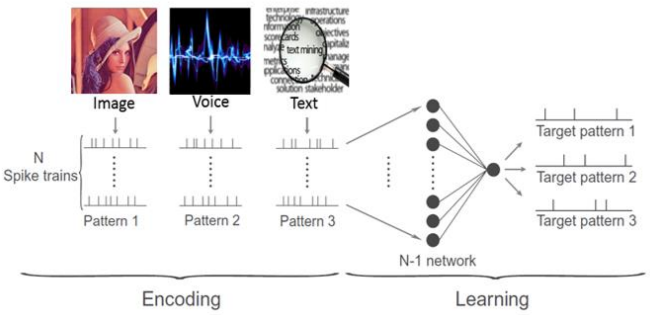

Figure 1: Spiking Neural Network System Flow. [10]

The spiking neural network, especially the two most important operations can be realized by either digital or analog circuits. Different from the digital implementation like TrueNorth, an analog implementation based on CMOS analog neuron and RRAM crossbar is introduced in our design. For the nonlinear function, specific circuits for functions like sigmoid and Leaky Integrate-and-Fire (LIF) have been proposed [15] and used in our design. Thanks to the RRAM crossbar structure, it is possible to get several orders of energy efficiency gain for the matrix-vector multiplication operation for weighted synapse computation [9]. We briefly show how to implement (1) spiking neuron unit; (2) weight matrix crossbar in the following two subsections.

2.1 Analog Spiking Neuron Unit

LIF neuron unit has an energy efficient implementation in the analog mode and the details can be found in [15]. Here we just give the diagram in Fig. 2: The capacitor C_{mem} works as an integrator to accumulate the crossbar output current I_{oj}, which passes through a mirror current source (T_1, T_2). The resistor R_{mem} functions as the leaky path. T_3 works as a digital switch which controls the reset path: Once the voltage on V_{mem} exceeds the threshold V_{th}, the output of the comparator would be set to high voltage level, the flip-flop would send a pulse to the next crossbar and at the same time, the reset path is conducted, which means the voltage on C_{mem} is reset to V_{reset}. Such design[15] is quite power efficiency with the average power consumption of ∼300nW. However, it may still bring considerable overhead in very large SNN system design since there are 14 transistors in one complete neuron unit.

2.2 RRAM based Crossbar for synapse weight computation

The RRAM device is a passive two-port element with variable resistance states (an example of 2D filament model of

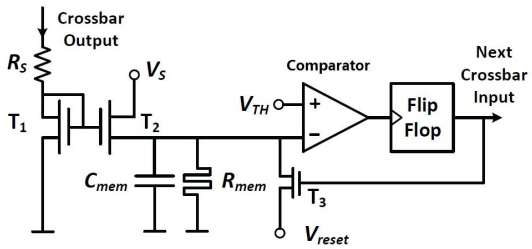

Figure 2: The Circuit Diagram of Analog Spiking Neuron.

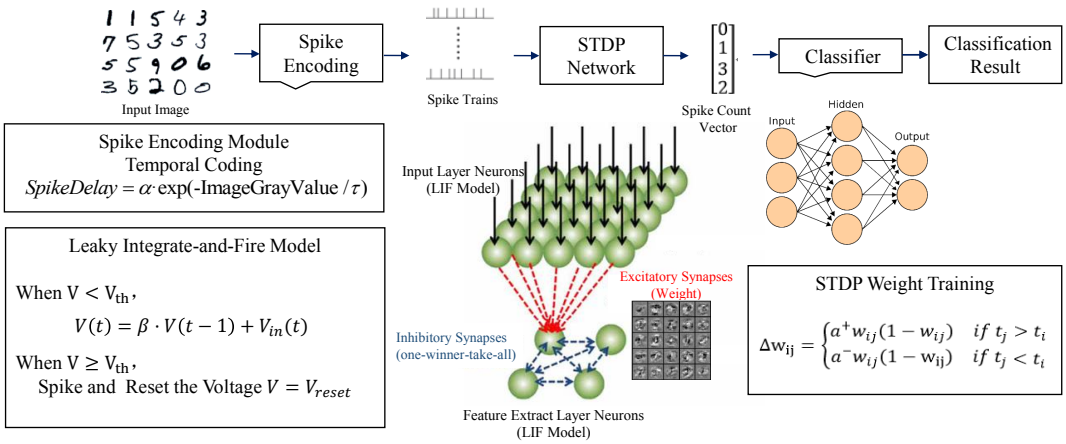

Figure 4: System Structure of Unsupervised Feature Extraction + Supervised Classifier: 2-layer STDP based SNN + 3-layer ANN. [10]

the HfO_x based RRAM is shown in Fig. 3(a)). The most attractive feature of RRAM devices is that they can be used to build resistive cross-point structure, which is also known as the RRAM crossbar array (Fig. 3(b)). Compared with other non-volatile memories like flash, the RRAM crossbar array can naturally transfer the weighted combination of input signals to output voltages and realize the matrix-vector multiplication efficiently by reducing the computation complexity from $O(n^2)$ to O(1).

As shown in Fig. 3(b), the relationship between the input voltage vector ($\vec{V_i}$) and output voltage vector ($\vec{V_o}$) can be expressed as follows [16]:

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

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

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

The continuous variable resistance states of RRAM devices enable a wide range of weight matrices that can be represented by the crossbar. The precision of RRAM crossbar based computation may be limited by Non-Ideal factors, such as process variations, IR drop[17], drifting of RRAM resistance[18], etc. However, SNN only requires low precision of single synaptic value, meanwhile the binary input and LIF operation also alleviate the precision requirement of matrix vector multiplication. Therefore, the RRAM crossbar array is a promising solution to realize matrix-vector multiplication for synapse weight computation in neural networks.

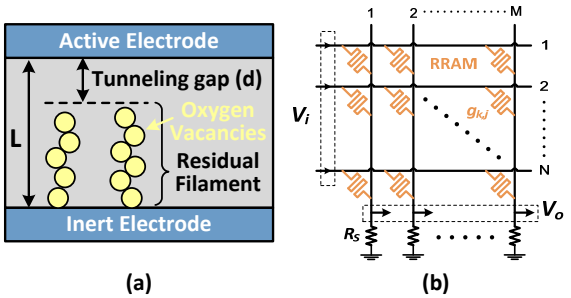

Figure 3: (a). Physical model of the HfO_x based RRAM. (b). Structure of the RRAM Crossbar Array. [11]

3. RRAM-BASED SPIKING LEARNING SYSTEM

For a SNN system used for realtime classification applications, an offline training scheme is needed to decide the weights of the neural networks, i.e. coefficients in the crossbar matrix. To our best knowledge, there are two kinds of SNN training methods to build up classification systems: (1) Unsupervised SNN training method, for example, Spike Timing Dependent Plasticity (STDP), is first introduced for extracting features; then the supervised classifier is introduced to finish the classification task. (2) First train an equivalent ANN using the gradient-based method, then transfer ANN to SNN and map SNN to the RRAM-based system for real-world applications. Both offline SNN systems are implemented with RRAM crossbar arrays [10, 11], and the details are shown in the following subsections.

3.1 Unsupervised Feature Extraction + Supervised Classifier

As an unsupervised method, STDP is mainly used for feature extraction. We can not build a complete classification system only based on STDP. A classifier is usually required for practical recognition tasks. Therefore, when mapping the system onto hardware, just as shown in Fig. 4, a five-layer neural network system is introduced: a two-layer spiking based neural network and a three-layer artificial neural network.

The first two layer SNN is trained using an unsupervised learning rule: Spike Timing Dependent Plasticity (STDP) [19], which updates the synaptic weights according to relative spiking time of pre- and post-synaptic neurons. The learning rate is decided by the time interval: the closer the distance between pre- and post-synaptic spikes, the larger the learning rate. The weight updating direction is decided by which neuron spikes first: for the excitatory neuron, if

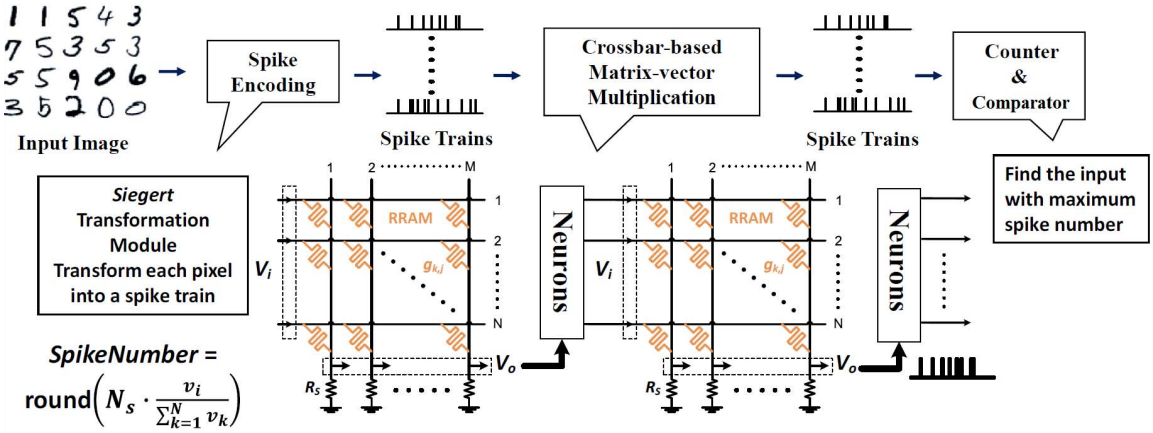

Figure 5: System Structure: Transferring ANN to SNN – Neural Sampling Method. [11]

the post-synaptic neuron spikes later, the synapse will be strengthened; otherwise, it will be weakened. When every synaptic weight no longer changes or is set to 0/1, the learning process is finished.

There is a converting module between the two layer SNN and 3-layer ANN to convert the spike trains into the spiking count vectors. Then the spike count vectors are sent into the following layers of the network (the 3-layer ANN). We use a 3-layer ANN as a classifier to process the features extracted from the input data by the previous 2-layer SNN. We use the CMOS analog neuron in Section II for the LIF neuron; and the RRAM crossbar for synaptic computation in both 2-layer SNN (vector addition) and 3-layer ANN (matrix vector multiplication).

An experiment is made on MNIST digit recognition task to evaluate the RRAM-based SNN system performance. The training algorithm is implemented on the CPU platform where LIF neurons are used in the first two layers and the sigmoid neurons are used in the last three layers. For the testing process (forward propagation of neural networks), we use circuit level simulation where the weight matrix is mapped to RRAM-based crossbar. Since the input images are 28×28 sized 256-level gray images. The five-layer spiking neural network system has five layers of neurons in all and the experiment result with the network size of "784×100SNN + 100×50×10ANN" shows the recognition accuracy of 91.5% on CPU platform and 90% on RRAM-based crossbar model (circuit simulation result). The performance is a little worse than that of the three-layer ANN sized "784×100×10" with the recognition accuracy of 94.3% on CPU platform and 92% on RRAM-based crossbar model (circuit simulation result). Moreover, when we change the supervised classifier into a two-layer SVM and make the system of "784×50 STDP + 50×10 SVM", the recognition accuracy on CPU platform only reaches 90% while the traditional "784×50 PCA + 50×10 SVM" gets the accuracy of 94%. As PCA is usually the baseline for evaluating the performance of feature extraction, STDP does **NOT** demonstrate itself as an exciting method for MNIST digit recognition tasks.

We observe that ANN consumes more power than SNN when ANN use similar or even smaller number of neurons, when both ANN and SNN use RRAM crossbar to implement matrix vector multiplication. Here we only consider the crossbar and the neuron power consumption. For example, the proposed "784×100 SNN + 100×50×10 ANN" consumes 327.36mW on RRAM while the power consumption increases to 2273.60mW when we directly use "784×100×10 ANN". The power (energy) saving of SNN comparing to ANN mainly comes from the different coding scheme. The input voltage of SNN can be binary since it transforms the numeric information into the temporal domain, so there is no need for SNN to hold a large voltage range to represent multiple input states as implemented in ANN. ANN needs input voltages of 0.9V, but SNN can work with much lower voltage supply (0.1V). Furthermore, binary coding in SNN can avoid the usage of large number of AD/DA on the input and output interfaces. The AD/DA power consumption, which is not considered here, consumes a considerable large portion of total power consumption in the RRAM based NN systems[20].

3.2 Transferring ANN to SNN – Neural Sampling Method

The Neural Sampling method provides a way to transfer ANN to SNN, thus offering a useful training scheme on classification tasks. An equivalent transformation is made between the nonlinear function (named Sigert function, which is similar to sigmoid function) of ANN and the Leaky Integrate-and-Fire (LIF) neuron of SNN. Therefore, it is possible to first train the ANN made up of the stacked Restricted Boltzmann Machine (RBM) structure using Contrastive Divergence (CD) method. In this way, a satisfying recognition accuracy of ANN can be first achieved. And then, the spike-based stacked RBM network with the same synaptic weight matrices can also be implemented for the classification tasks. The system structure is shown in Fig. 5 [11].

Since spike trains propagate in the spiking neural network, original input $x = [x_1, \cdots, x_N]$ should be mapped to spike trains $X(t) = [X_1(t), \cdots, X_N(t)]$ before running the test samples where $X_i(t)$ is a binary train with only two states 0/1. For the i^{th} input channel, the spike train is made of N_t spike pulses with each pulse width T_0, which implies that the spike train lasts for the length of time $N_t \cdot T_0$. Suppose the spike number of all input channels during the given time $N_t \cdot T_0$ is N_s, then the spike count N_i of the i^{th} channel is allocated as:

$$N_i = \sum_{k=0}^{N_t-1} X_i(kT_0) = \text{round}(N_s \cdot \frac{v_i}{\sum_{k=1}^{N} v_k}) \qquad (3)$$

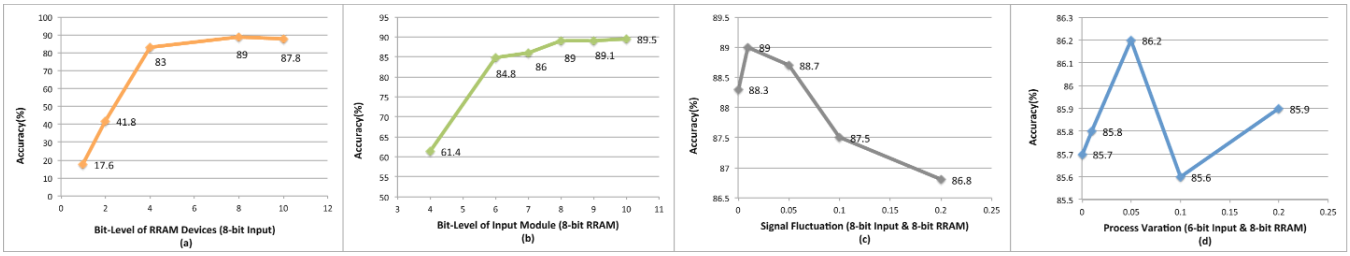

Figure 6: Recognition accuracy under (a). different bit-level of RRAM devices (b). different bit-level of input module, and (c) different degrees of input signal fluctuation (d) different degrees of process variation of RRAM devices. [11]

which implies

$$\frac{N_i}{N_s} = \frac{v_i}{\sum_{i=1}^{N} v_i} \tag{4}$$

Then the N_i spikes of the i^{th} channel is randomly set on the N_t time intervals. For an ideal mapping, we would like to have $N_i << N_t$ to keep the spike sparsity on the time dimension. However, for the speed efficiency, we would like the running time $N_t \cdot T_0$ to be short. Here, T_0 is defined by the physical clock, i.e. the clock of the pulse generator. which implies that we can only optimize N_t directly. Here, we define the bit level of the input as

$$\log(\frac{N_t}{\text{mean}(N_i)}) \tag{5}$$

which evaluates the tradeoff between time efficiency and the accuracy performance.

We train the SNN with the size of $784 \times 500 \times 500 \times 10$. And the parameters are shown in Table 1. The experiment results show that the recognition accuracy of MNIST dataset is 95.4% on the CPU platform and 91.2% on the ideal RRAM-based hardware implementation. The recognition performance decreases about 4% because it is impossible to satisfy with $N_t << N_s$ on the RRAM platform.

We show the results for recognition under different bit level quantization of input signal and RRAM devices, together with RRAM process variation and input signal fluctuation. The simulation results in Fig. 6 (a) show that a 8-bit RRAM device is able to realize a recognition accuracy of nearly 90%. The simulation results in Fig 6 (b) shows that the input signal above 6-bit level achieves satisfying recognition accuracy (>85%). Based on the 8-bit RRAM result, different levels of signal fluctuation are added on the 8-bit input signal. The result shown in Fig 6 (c) demonstrates that the performance of accuracy just decreases ~3% given 20% variation. Fig. 6 (d) shows that when RRAM device is configured to 8-bit level with the 6-bit level input, the performance does not decrease under 20% process variation. The sparsity of the spike train leads to the system robustness, making it insensitive to the input fluctuation and process variation.

Table 1: Important Parameters of the SNN System

Network Size	$784 \times 500 \times 500 \times 10$
Number of Input Spike (N_s)	2000
Number of Pulse Interval (N_t)	128
Input pulse Voltage (V)	1V
The Pulse Width (T_0)	1ns

The power consumption of the system is mainly contributed by three parts: the crossbar, the comparator and the $R_{mem}C_{mem}$ leaky path. The simulation results show that the power consumption is about 3.5 mW on average. However, it takes $N_t = 128$ cycles with the physical clock $T_0 = 1ns$. Though input conversion from numeral values to spike trains leads to about 100X clock rate decrease, the system is able to **complete the recognition task in real time** (~1μs/sample) thanks to the short latency of RRAM device.

3.3 Discussion on How to Boost the Accuracy of SNN

The experiment results in the above subsections show that the recognition accuracy will decay after transferring an ANN to a SNN. However, due to the ultra-high integration density of the RRAM devices and the 0/1 based interfaces of SNN, SNN tends to consume much less circuit area and power compared with ANN. This result inspires us that we may integrate multiple SNN with the same or even less circuit area and power consumption of ANN, and combine these SNNs together to boost the accuracy and robustness of the SNN system.

Previously, we have proposed an ensemble method to boost the accuracy of RRAM-based ANN systems [20], named SAAB (Serial Array Adaptive Boosting), which is inspired by the AdaBoost method [21]. The basic idea of AdaBoost, which is also its major advantage, is to train a series of learners, such as ANNs or SNNs, sequentially, and every time we train a new learner, we try to "force" the new learner to pay more attention to the "hard" samples incorrectly classified by previous trained learners in the training set. The proposed technique can improve the accuracy of ANN by up to 13.05% on average and ensure the system performance under noisy conditions in approximate computation applications.

SAAB boost the computation accuracy at the cost of consuming more power and circuit area. As SNN usually consumes much less area and power compared with the ANN, there is a chance to integrate multiple SNNs under the same circuit resource limitation of ANN. And these SNNs can be boosted together by the similar idea of SAAB. However, the inherent attributions of SNN systems should be considered when designing the boosting algorithm. According to our observation, there are two types of errors in the SNN-based classification tasks: (i). a "traditional" type: more than one neuron in the output layer spikes and the neuron spiking the most is not the target neuron; and (ii). a "special" type of SNN: no neuron in the output layer spikes; It is interesting to observe that most of the wrong trials are the "special" type and it can be reduced slightly when increasing the input

spike counts. We regard such samples as "the difficult classifying cases". When seeking for the possibility to make up the performance loss after transferring ANN to SNN with a boosting-based method, this problem should be considered.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we analyzed the possibility of designing RRAM based SNN system for the realtime classification. Two different SNN systems based on different training algorithms are shown, including the training methods and hardware architecture. We also compare the energy efficiency and accuracy for MNIST application of both systems. Finally, we discuss the possibility of how to boost the accuracy of SNN.

For future work, besides the boosting algorithm for multiple SNNs, we propose the following research directions: (1) Design the basic core structure for large SNN "processor" based on RRAM crossbars. For the two proposed SNN systems in this paper, we design it following the design style of fixed accelerators without considering the scalability or reconfigurability for bigger or different problems. (2) Design on-line learning RRAM based SNN system. To achieve intelligent systems, we need the system to learn by itself. How to use the inherent learning feature of RRAM to build up on-line learning computing system for real world application remains an interesting problem.

Acknowledgment

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

5. REFERENCES

[1] C. Guger, H. Ramoser, and G. Pfurtscheller, "Real-time eeg analysis with subject-specific spatial patterns for a brain-computer interface (bci)," *Rehabilitation Engineering, IEEE Transactions on*, vol. 8, no. 4, pp. 447–456, 2000.

[2] G. A. Carpenter, S. Grossberg, and J. H. Reynolds, "Artmap: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network," *Neural networks*, vol. 4, no. 5, pp. 565–588, 1991.

[3] H. Esmaeilzadeh, E. Blem, R. St Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *ISCA*, 2011, pp. 365–376.

[4] E. Neftci, S. Das, B. Pedroni, K. Kreutz-Delgado, and G. Cauwenberghs, "Event-driven contrastive divergence for spiking neuromorphic systems," *Frontiers in neuroscience*, vol. 7, 2013.

[5] T. Masquelier and S. J. Thorpe, "Unsupervised learning of visual features through spike timing dependent plasticity," *PLoS computational biology*, vol. 3, no. 2, p. e31, 2007.

[6] S. K. Esser, A. Andreopoulos, R. Appuswamy, P. Datta, D. Barch, A. Amir, J. Arthur, A. Cassidy, M. Flickner, P. Merolla *et al.*, "Cognitive computing systems: Algorithms and applications for networks of neurosynaptic cores," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–10.

[7] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[8] B. Govoreanu, G. Kar, Y. Chen, V. Paraschiv, S. Kubicek, A. Fantini, I. Radu, L. Goux, S. Clima, R. Degraeve *et al.*, "10× 10nm 2 hf/hfo x crossbar resistive ram with excellent performance, reliability and low-energy operation," in *Electron Devices Meeting (IEDM), 2011 IEEE International*. IEEE, 2011, pp. 31–6.

[9] B. Li, Y. Shan, M. Hu, Y. Wang, Y. Chen, and H. Yang, "Memristor-based approximated computation," in *ISLPED*, 2013, pp. 242–247.

[10] T. Tang, R. Luo, B. Li, H. Li, Y. Wang, and H. Yang, "Energy efficient spiking neural network design with rram devices," in *Integrated Circuits (ISIC), 2014 14th International Symposium on*. IEEE, 2014, pp. 268–271.

[11] T. Tang, L. Xia, B. Li, R. Luo, Y. Wang, Y. Chen, and H. Yang, "Spiking neural network with rram : Can we use it for real-world application?" in *DATE*, 2015.

[12] E. Painkras, L. A. Plana, J. Garside, S. Temple, F. Galluppi, C. Patterson, D. R. Lester, A. D. Brown, and S. B. Furber, "Spinnaker: A 1-w 18-core system-on-chip for massively-parallel neural network simulation," *Solid-State Circuits, IEEE Journal of*, vol. 48, no. 8, pp. 1943–1953, 2013.

[13] R. Wang, T. J. Hamilton, J. Tapson, and A. van Schaik, "An fpga design framework for large-scale spiking neural networks," in *ISCAS*, 2014, pp. 457–460.

[14] Y. LeCun and C. Cortes, "The mnist database of handwritten digits," 1998.

[15] G. Indiveri, "A low-power adaptive integrate-and-fire neuron circuit," in *ISCAS (4)*, 2003, pp. 820–823.

[16] M. Hu, H. Li, Q. Wu, and G. S. Rose, "Hardware realization of bsb recall function using memristor crossbar arrays," in *DAC*, 2012, pp. 498–503.

[17] P. Gu, B. Li, T. Tang, S. Yu, Y. Wang, and H. Yang, "Technological exploration of rram crossbar array for matrix-vector multiplication," in *ASP-DAC*, 2015.

[18] B. Li, Y. Wang, Y. Chen, H. H. Li, and H. Yang, "Ice: inline calibration for memristor crossbar-based computing engine," in *DATE*, 2014.

[19] S. Song, K. D. Miller, and L. F. Abbott, "Competitive hebbian learning through spike-timing-dependent synaptic plasticity," *Nature neuroscience*, vol. 3, no. 9, pp. 919–926, 2000.

[20] B. Li, L. Xia, P. Gu, Y. Wang, and H. Yang, "Merging the interface: Power, area and accuracy co-optimization for rram crossbar-based mixed-signal computing system," in *DAC*, 2015.

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