Computing-in-Memory with SRAM and RRAM for Binary Neural Networks

Xiaoyu Sun[1], Rui Liu[1], Xiaochen Peng[1], and Shimeng Yu[2]

[1]Arizona State University, Tempe, AZ 85281, USA
[2]Georgia Institute of Technology, Atlanta, GA 30332, USA Email: shimeng.yu@ece.gatech.edu

Abstract – **Recent advances in deep learning have shown that Binary Neural Network (BNN) is able to provide a satisfying accuracy on various image datasets with a significant reduction in computation and memory cost. With both weights and activations binarized to +1 or -1 in BNNs, the high-precision multiply-and-accumulate (MAC) operations can be replaced by XNOR and bit-counting operations. In this work, we present two computing-in-memory (CIM) architectures with parallelized weighted-sum operation for accelerating the inference of BNN: 1) parallel XNOR-SRAM, where a customized 8T-SRAM cell is used as a synapse; 2) parallel XNOR-RRAM, where a customized bit-cell consisting of 2T2R cells is used as a synapse. For large-scale weight matrices in neural networks, the array partition is necessary, where multi-level sense amplifiers (MLSAs) are employed as the intermediate interface for accumulating partial weighted sums. We explore various design options with different sub-array sizes and sensing bit-levels. Simulation results with 65nm CMOS PDK and RRAM models show that the system with 128×128 sub-array size and 3-bit MLSA can achieve 87.46% for an inspired VGG-like network on CIFAR-10 dataset, showing less than 1% degradation compared to the ideal software accuracy. The estimated energy-efficiency of XNOR-SRAM and XNOR-RRAM shows ~30X improvement compared to the corresponding conventional SRAM and RRAM architectures with sequential row-by-row read-out.**

1. Introduction

Deep learning has shown great performance in various intelligent applications including computer vision and speech recognition. However, the high demands on memory storage capacity and computational capability make it challenging to implement the state-of-the-art deep neural networks (DNNs) on resource-limited platforms such as mobile or sensory devices. For example, ResNet-50 [1] has 25.5M parameters and requires 3.9G high precision multiply-and-accumulate (MAC) operations to classify one image and these numbers become higher for even deeper networks. Therefore, it is prohibitive to directly implement the entire DNNs on-chip, and the intensive data movements between on-chip processor and off-chip memory (e.g., DRAM) is the bottleneck of the system performance and energy efficiency. Various techniques such as network pruning [2] and fixed-point precision [3] were proposed to reduce the energy and area cost of the storage. Recently, it is demonstrated that the weight and neuron precision can be aggressively reduced to 1-bit in Binary Neural Networks (BNNs) [4, 5], which are still able to achieve a reasonable classification accuracy on representative image datasets (e.g., MNIST, CIFAR-10, and ImageNet). In these BNNs, both the weights and neuron activations are binarized to +1/-1, thus 1) the memory storage requirement is dramatically reduced; 2) computational resources are significantly reduced as high-precision MAC operations are replaced by XNOR and bit-counting operations. Therefore, BNNs provide a promising solution for on-chip implementation of DNNs.

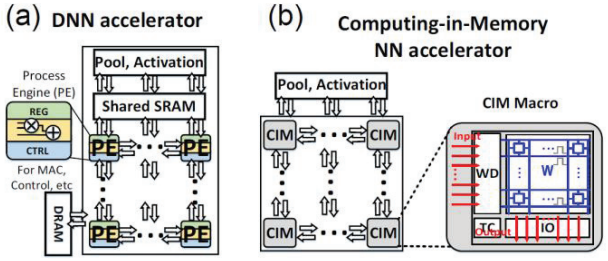

Fig. 1. (a) The conventional deep neural network accelerator where the processing element (PE) arrays exploit parallelized computation but with inefficient row-by-row access to the weights stored in shared buffers (i.e. SRAM). (b) The diagram of CIM architecture where the input vectors activate multiple rows and the dot-product output is obtained as column voltage or current. (Adapted from [14])

In the CMOS based DNN accelerators, SRAM buffer is commonly utilized to store the synaptic weights; however, the extensive computation such as MAC is performed using other digital logic circuits, e.g. multiplier and adders in the processing element (PE) [6, 7], as shown in Fig. 1(a). To improve the data utilization efficiency, parallelized computation is exploited across multiple PE arrays but still with inefficient row-by-row access to the weights stored in the shared SRAM buffer. To overcome this problem, it is more attractive to integrate the computation into the memory array itself, namely computing-in-memory (CIM) as shown in Fig. 1(b). In CIM, the vector-matrix multiplication is done in a parallel fashion where the input vectors activate multiple rows and the dot-product is obtained as column voltage or current. For instance, the standard 6T SRAM array is proposed to perform MAC operation in parallel [8]. In [8], the word line (WL) of SRAM is driven by an analog voltage representing a 5-bit input vector using a digital-to-analog converter (DAC), while the weights are stored in the bit-cells in a binary format (i.e. +1/-1). However, the analog WL voltage induces non-linear bit line (BL) discharge current and extra circuitry is

978-1-5386-4441-6/18/$31.00 ©2018 IEEE

required for the linearization. Meanwhile, researchers have also proposed using emerging non-volatile memories (eNVMs) with much less area (than SRAM at the same technology node) such as resistive random access memory (RRAM) [9] and phase change memory (PCM) [10] to implement synaptic weights, where the computations are naturally performed in a true crossbar array or the pseudo-crossbar array with 1-transistor-1-resistor (1T1R) [11]. The prior work in [12] experimentally demonstrated BNNs (a two-layer perceptron) on a 16Mb RRAM macro chip with row-by-row sequential read-out of binary RRAM cells, showing ~96.5% accuracy on MNIST dataset. To get rid of the row-by-row sequential read-out, one could allow fully-parallel read-out by activating all the word lines (WLs) simultaneously for the weighted sum operation.

Theoretically, the binary activation in BNNs could allow using 1-bit sense amplifiers (SAs) instead of analog-to-digital converters (ADCs) to serve as the binary neuron. However, due to the intrinsic offset of the SAs introduced by process variation, the sensing failure becomes intensified when the column current increases [13] when multiple WLs are activated in the parallel read-out scheme. This may substantially degrade the classification accuracy as the threshold of binary neuron in the hardware may differ from the ideal value in algorithms, leading to a constraint on the column length or the array size. To overcome this design challenge, array partition must be adopted to split a large-scale matrix into multiple small sub-arrays. In this way, ADC-like multi-level sense amplifiers (MLSAs) are exploited to generate the partial sums of sub-arrays, which are eventually added up by an adder tree to be the final sum for binary activation.

In this paper, we review two CIM architectures with parallel weighted-sum operation for accelerating the inference of BNNs: 1) parallel XNOR-SRAM [14], where a customized 8T-SRAM cell is used as a synapse; 2) parallel XNOR-RRAM [15], where a customized bit-cell consisting of 2T2R is used as a synapse.

2. CIM Architctures Based on SRAM and RRAM

2.1 Binary Neural Networks

In a BNN, both the weights and neuron activations are binarized to -1 or +1. Therefore, multiplications between activations and weights can be simplified as XNOR operations and accumulation of the products is equivalent to bit-counting operation. In this work, we trained BNNs using the algorithm proposed in [4] on the Theano platform. An inspired VGG-like convolutional neural network (CNN) with 6 convolution layers and 3 fully-connected layers was trained for evaluations on CIFAR-10 dataset. The corresponding classification accuracy with floating point precision and binary precision is 89.98% and 88.34%, respectively. Such a minor degradation has also been observed in state-of-the-art BNNs.

2.2 8T-SRAM based Synaptic Array

We propose a customized 8T-SRAM bit-cell design as shown in Fig. 2(a) for parallel XNOR-SRAM, which has two complementary WLs and two pairs of pass gates (PGs). The first pair of PGs controlled by WL connects Q and QB to BL and BLB, respectively. In contrast, the second pair of PGs controlled by WLB connects Q and QB to BLB and BL,

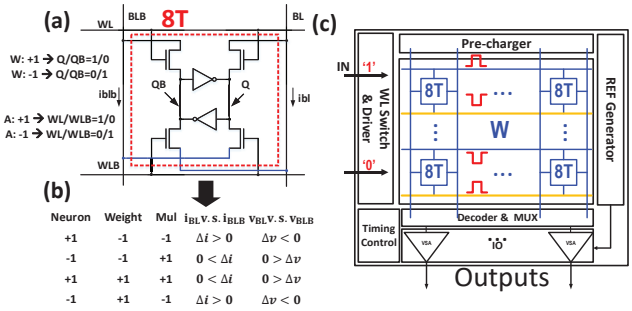

Fig. 2. (a) Schematic of the customized 8T-SRAM bit-cell design and bitwise XNOR operation of both binary neuron and weight in +1/-1. (b) The truth table and operation principles. (c) Diagram of proposed parallel XNOR-SRAM architecture. (Adapted from [14])

Table I. Discharged voltages from BL and BLB

	W	In	Mul	V_{BL}	V_{BLB}
	+1	+1	+1	0	Δv
	-1	+1	-1	Δv	0
	-1	+1	-1	Δv	0
	+1	-1	-1	Δv	0
	-1	+1	-1	Δv	0
	-1	-1	+1	0	Δv
Sum	N/A	N/A	-2	4Δv	2Δv

respectively. This design is different from the conventional 8T-SRAM that aims to improve the static noise margin [16], where two added transistor connected in series: 1) one gate is connected to one storage node and the other gate is controlled by a read WL; 2) one terminal of the two serial transistors is connected to ground and the other terminal is attached to the BL in order to decouple the read path and write path. In our 8T-SRAM design, the synaptic weight is stored in Q and QB similarly as in 6T-SRAM. However, different from 6T SRAM, the input binary neuron of an 8T-SRAM is represented by a complimentary WLs. The two WLs (WL, WLB) in a bit-cell are in a complementary state of (1, 0) to represent neuron +1 and (0, 1) to represent neuron -1. In the proposed 8T-SRAM bit-cell, neuron and weight are both non-zero. As a result, the multiplication result is non-zero. Therefore, there is always a non-zero voltage difference between BL and BLB, effectively performing the XNOR operation. Fig. 2 (b) summarizes the possible combination patterns of binary neuron and weight, and the resulting BL discharge current (Δi) or voltage (Δv) in 8T-SRAM for BNNs. Fig. 2 (c) shows the diagram of the proposed parallel synaptic array architectures with 8T-SRAM for XNOR-SRAM. Instead of a normal decoder, a WL switch matrix is employed to activate multiple WLs simultaneously according to the input neuron vector to enable the parallel read-out. Note that each bit in the input vector is encoded to a pair of complementary signals to enable one WL and disable the other. In the parallel access design, the currents from multiple rows along the same column contribute together to discharge the bit line (BL or BLB). Thus, the total discharged voltage from BL or BLB depends on how many cells along the columns are discharging. As illustrated in the example in

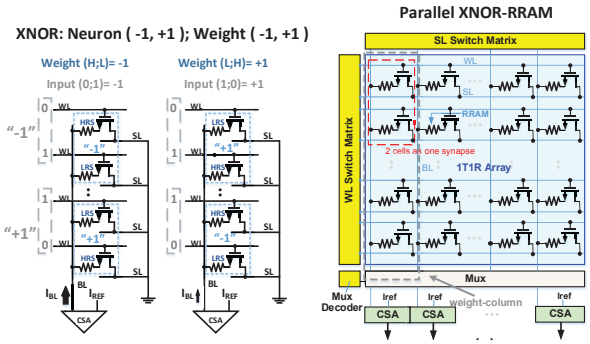

Fig. 3. (a) The customized RRAM bit-cell design for XNOR-RRAM. (b) Diagram of proposed parallel XNOR-RRAM architecture. (Adapted from [15])

Table I, the total voltages discharged from BL and BLB are 4Δv and 2Δv, resulting in a voltage difference -2Δv. The analog voltage difference between BL and BLB can be used to determine the weighted sum. The parallel XNOR-SRAM architecture leverages the analog voltage discharge along the column to effectively realize the MAC operation.

2.3 RRAM based Synaptic Array

Fig. 3 (a) shows the proposed bit-cell design for parallel XNOR-RRAM implementation. For each synaptic weight, -1 is represented by two cells where the top one is in high resistance state (HRS) and the bottom one is in low resistance state (LRS). The reversed pattern is used for +1. For the WL input pattern, two adjacent WLs for each weight-cell are in complimentary state where (0, 1) represents -1 and (1, 0) represents +1. In this way, the value of the current that flows through each weight-cell during read-out is dependent on the combination of WL input pattern and bit-cell pattern. For example, when input vector is -1, for the cell of weight -1, the cell in the activated row is in LRS, leading to a large cell current, which can be regarded as a bit-wise XNOR output of "+1". For the cell of weight +1, the cell in the activated row is in HRS, leading to a small cell current, which can be regarded as a bit-wise XNOR output of "-1". When multiple WLs are activated in parallel, the LRS-cells will dominate the total bit line current (I_{BL}) if the on/off ratio of RRAM is sufficiently large. Consequently, I_{BL} will be proportional to the bit-counting results equivalent to the number of LRS-cells along the column. For example, 50% "+1" and 50% "-1" will lead to a final weighted sum of 0. Assuming the column length of the sub-array is 64, the sum of 0 can be mapped to the I_{BL} = 32 activated LRS-cells. Therefore, the reference current (I_{REF}) for the current sense amplifier (CSA) could be set to 32 LRS-cells' current for the binary neuron activation. If I_{BL} is smaller than I_{REF} that generates a CSA output "-1", it represents that there are more "-1" than "+1" along the column, and vice versa.

For the parallel XNOR-RRAM design in Fig. 3(b), similar as parallel XNOR-SRAM design, instead of a normal decoder, a WL switch matrix is designed to activate multiple WLs simultaneously depending on the input vectors to enable the parallel read-out operation. The parallel XNOR-RRAM architecture leverages the analog current summation along the column to effectively realize the MAC operation.

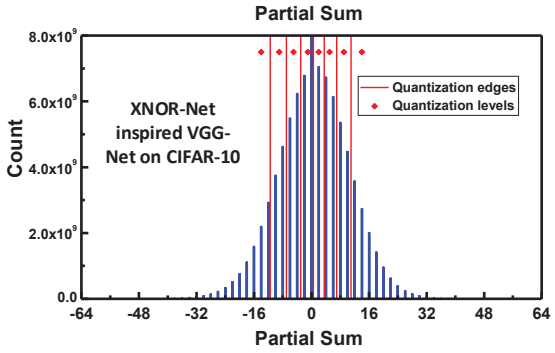

Fig. 4. Distribution of partial sums of XNOR values of the CNN on CIFAR-10. Sub-arrays are assumed to be 64×64. Red lines are 7 nonlinear quantization edges (or references) and red diamonds indicate 8 quantization levels. (Adapted from [14])

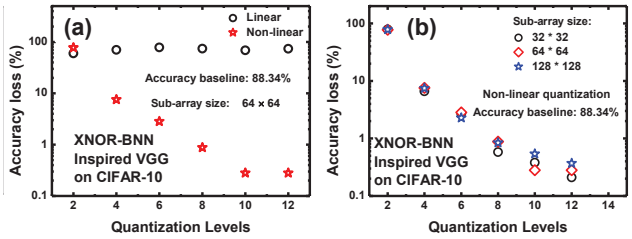

Fig. 5. (a) The accuracy degradation as a function of quantization levels for linear and non-linear quantization for inspired VGG-like network on CIFAR-10. (b) The accuracy degradation as a function of quantization levels for different sub-array size. (Adapted from [14])

2.4 Array Partition for Implementing Arbitrary Network Size

At the system level, the proposed XNOR-SRAM array and XNOR-RRAM array can be treated similarly as one PE to perform MAC operations for a certain matrix size. In this section, we analyze different design options with array partition. Firstly, the size of the sub-array is a key design parameter that may affect the classification accuracy and the cost of system. After the matrix splitting, each small matrix needs to generate a partial sum, which will be added up to obtain the final sum for binary activation. Thus, the precision of the partial sum may affect the value of the final sum and then influence the classification accuracy. As a result, ADC-like MLSAs are employed to generate partial sums with fixed-point precision (larger than 1-bit). To minimize the quantization error of the partial sums, we propose to perform nonlinear quantization where quantization edges (or references) are determined via Lloyd-Max algorithm [17] according to the distribution of the partial sums. The idea is to make the quantization edges denser in the center of the distribution thus each quantization level has roughly the same number of partial sums. Fig. 4 presents the distribution of partial sums for the VGG-like network on CIFAR-10 with 7 quantization edges and 8 quantization levels. Due to the reduced quantization error, nonlinear quantization achieves significantly better accuracy than linear quantization given the same number of quantization levels. As shown in Fig. 5(a), the inspired VGG-like network achieves an accuracy degradation of only 0.88% with nonlinear quantization and of 74.07% with linear quantization with 8 quantization levels. The system with larger

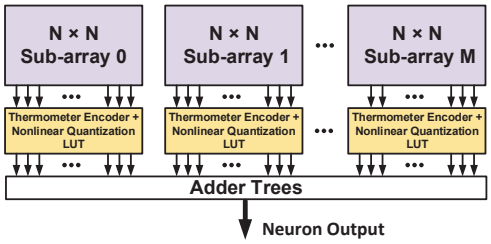

Fig. 6. Generic system diagram for implementing one layer with arbitrary size in a network. (Adapted from [15])

sub-array size will cause a slightly larger accuracy degradation than a system with a smaller sub-array size for 8 quantization levels as shown in Fig. 5(b). We will use 8 quantization levels (3-bit MLSA) in the design.

A generic system diagram that implements one BNN layer is shown in Fig. 6. MLSAs in a sub-array take the quantization edges as references and generate digital outputs, which then go through a thermometer-to-binary (TM2B) encoder and a look-up table (LUT) to be converted to the corresponding quantization values as partial sums. Adder trees sum up the partial sums to be the final weighted sum, which then goes through the binary activation to generate the neuron output.

3. Benchmark Results

We employed a circuit-level macro model NeuroSim [18] that can be used to estimate the area, latency, and energy consumption of hardware accelerators. The hierarchy of the simulator consists of different levels of abstraction from the memory cell parameters and transistor technology parameters, to the gate-level sub-circuit modules, and then to the array architecture. In this section, we compared conventional row-by-row sequential SRAM, parallel XNOR-SRAM, row-by-row sequential RRAM, and parallel XNOR-RRAM architectures with various options on sub-array size while MLSA bitwidth is fixed as 3-bit for implementing a 512×512 weight matrix at 65nm technology node. Table II summarizes the results of each case. The energy efficiency of XNOR-SRAM and XNOR-RRAM with 128×128 sub-array size could achieve 21.26 TOPS/W and 122.35 TOPS/W, respectively, showing ~30X improvement compared to corresponding sequential SRAM/RRAM architectures. However, the area overhead of XNOR-SRAM and XNOR-RRAM architectures significantly increases due to more peripheral circuits such as MLSAs and adder trees, especially when the sub-array size is small (e.g. 64×64). In addition, RRAM based architectures outperform SRAM based ones on area and energy-efficiency due to smaller cell size and lower read energy.

TABLE II. Comparison between different architectures for implementing a 512×512 weight matrix at 65 nm

Architecture	Area (mm^2)	Latency (ns)	TOPS/W
Sequential SRAM	1.74	88.55	0.71
XNOR-SRAM 64×64	3.64	7.28	18.91
XNOR-SRAM 128×128	1.99	5.19	21.26
Sequential RRAM	0.14	5036.28	4.23
XNOR-RRAM 64×64	0.36	15.20	98.24
XNOR-RRAM 128×128	0.20	15.40	122.35

4. Conclusion

In this paper, we reviewed two CIM architectures, namely, XNOR-SRAM and XNOR-RRAM, with customized bit-cell designs and parallel weighted-sum operation for accelerating the inference of BNNs. We investigated the MLSA's bitwidth for accumulating the partial sums from sub-arrays, and then explored design trade-offs between different sub-array sizes and MLSA bitwidth. Benchmark results show that the proposed parallel XNOR-SRAM and XNOR-RRAM improve the energy efficiency by ~30X compared to the corresponding conventional SRAM and RRAM architectures with sequential row-by-row read-out. Compared to the XNOR-SRAM, XNOR-RRAM has further benefits such as ~10X smaller area, and ~5.7X improvement in energy efficiency.

ACKNOWLEDGMENT

This work is in part supported by NSF-CCF-1552687, NSF/SRC E2CDA under grant NSF-CCF-1740225 and SRC Contract 2018-NC-2762, and ASCENT, one of the six SRC/DARPA JUMP Centers.

REFERENCES

[1] K. He, et al., "Deep residual learning for image recognition," in *IEEE CVPR*, 2016.

[2] S. Han, et al., "Deep compression: compressing deep neural network with pruning, trained quantization and huffman coding," in *ICLR*, 2016.

[3] S. Gupta, et al., "Deep learning with limited numerical precision," in *ICML*, 2015.

[4] M. Courbariaux, et al., "Binarized neural network: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv: 1602.02830*, 2016.

[5] M. Rastegari, et al., "XNOR-Net: ImageNet classification using binary convolutional neural networks," *arXiv: 1603.05279*, 2016.

[6] Y.-H. Chen, et al., "Eyeriss: an energy-efficient reconfigurable accelerator for deep convolutional neural networks," in *IEEE ISSCC*, 2016.

[7] N. P. Jouppi, et al., "In-datacenter performance analysis of a Tensor Processing Unit," in *ACM/IEEE ISCA*, 2017.

[8] J. Zhang, et al., "A machine-learning classifier implemented in a standard 6T SRAM array," in *Symp. VLSI Circuits*, 2016.

[9] P. Yao, et al., "Face classification using electronic synapses," *Nature Communications*, 8, 15199, 2017.

[10] G. W. Burr, et al., "Experimental demonstration and tolerancing of a large-scale neural network (165,000 synapses), using phase-change memory as the synaptic weight element," in *IEEE IEDM*, 2014.

[11] S. Yu, "Neuro-inspired computing with emerging non-volatile memory," *Proc. IEEE*, vol. 106, no. 2, pp. 260-285, 2018.

[12] S. Yu, et al., "Binary neural network with 16 Mb RRAM macro chip for classification and online training," in *IEEE IEDM*, 2016.

[13] C.-P. Lo, et al., "Embedded 2Mb ReRAM macro with 2.6 ns read access time using dynamic-trip-point-mismatch sampling current-mode sense amplifier for IoE applications," in *IEEE Symp. VLSI Circuits*, 2017.

[14] R. Liu, et al., "Parallelizing SRAM arrays with customized bit-cell for binary neural networks," in *ACM/IEEE DAC*, 2018.

[15] X. Sun, et al., "XNOR-RRAM: A scalable and parallel resistive synaptic architecture for binary neural networks," in *ACM/IEEE DATE*, 2018.

[16] L. Chang, et al., "Stable SRAM cell design for the 32 nm node and beyond," in *IEEE Symp. VLSI Circuits*, 2005.

[17] J. Max, "Quantizing for minimum distortion," *IRE Transactions on Information Theory*, 6(1), 7-12, 1960.

[18] P.-Y. Chen, et al., "NeuroSim+: An integrated device-to-algorithm framework for benchmarking synaptic devices and array architectures," in *IEEE IEDM*, 2017.