Vesti: An In-Memory Computing Processor for Deep Neural Networks Acceleration

Zhewei Jiang, Shihui Yin, *Student Member, IEEE,* Minkyu Kim, *Student Member, IEEE,* Tushar Gupta, Mingoo Seok, *Senior Member, IEEE,* and Jae-sun Seo, *Senior Member, IEEE*

***Abstract*—We present *Vesti*, a Deep Neural Network (DNN) accelerator optimized for energy-constrained hardware platforms such as mobile, wearable, and Internet of Things (IoT) devices. Vesti integrates instances of in-memory computing (IMC) SRAM macros with an ensemble of peripheral digital circuits for dataflow management. The IMC SRAM macros eliminate the data access bottleneck that hinders conventional ASIC implementations performing dot-product computation, while the peripheral circuits improve the macros' parallelism and utilization for practical applications. Vesti supports large-scale DNNs with configurable activation precision, substantially improving chip-level energy-efficiency with favorable accuracy trade-off. The Vesti accelerator is designed and laid out in 65 nm CMOS, demonstrating ultra-low energy consumption of less than <20nJ for MNIST classification and <40μJ for CIFAR-10 classification at 1.0V supply.**

***Index Terms*—In-memory computing, SRAM, deep learning accelerator, deep neural networks, double-buffering.**

I. Introduction

In recent years, works from algorithms to hardware reduced the energy cost of increasingly deeper and largeer DNNs. On the algorithm side, pruning, compression [1], and low-precision techniques [2]–[5] have been investigated with minimal degradation in the classification accuracy. In [2], the weights and activations are binarized to +1 or -1. On the hardware side, many digital accelerators [6]–[8] have been presented to lower the cost of computing DNNs. However, limitations remain. In particular, memory is the biggest energy-efficiency bottleneck, in terms of storing parameters, loading from memory, and moving to processing unit.

To improve this limitation, *in-SRAM computing* was introduced. It performs computation in SRAM hardware without reading out each row of SRAM to a computing unit [10]–[20]. Recently, we demonstrated a new SRAM macro that can perform MAC along the bitline with all rows turned on simultaneously [21]. Titled as *XNOR-SRAM*, it performs XNOR-and-ACcumluate (XAC, replacing MAC in the binary-weight DNN) fast and energy-efficiently, and supports core computing primitives of state-of-the-art DNNs, achieving highly competitive classification accuracy. However, XNOR-SRAM and most in-SRAM computing works only demonstrate a relatively small custom SRAM array at the single-array-level [13], [16], [21].

S. Yin, M. Kim and J. Seo are with School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA (e-mail: jaesun.seo@asu.edu).

Z. Jiang and M. Seok are with Department of Electrical Engineering, Columbia University, New York City, NY, USA.

T. Gupta is with Synopsys, Mountain View, CA, USA.

In this work, we substantially expanded the single-array-level prior XNOR-SRAM design [21] towards a configurable DNN accelerator architecture that integrates 72 XNOR-SRAM arrays with inter-array communication, and supports configurable activation precision. The proposed accelerator supports 3×3 and 1×1 convolutional kernels and up to 256 feature maps in a convolutional layer.

The main contributions of this work are as follows:

- We construct the chip-level DNN accelerator architecture that efficiently loads/maps DNN weights onto many instances of XNOR-SRAM arrays.
- We add peripheral digital logic to support multi-bit activations to the originally binary-DNN only XNOR-SRAM, serving as an effective knob to favorably trade-off energy versus accuracy.
- We employ double-buffering technique with two groups of XNOR-SRAMs to hide the latencies of re-programming SRAM arrays with new DNN weights.
- We evaluate the chip-level energy benefits and remaining bottlenecks of IMC based DNN accelerators.

The remainder of the paper is organized as follows. Sec. II presents the basics of XNOR-SRAM, and discusses practical challenges of designing a chip-level DNN accelerator using many macros. In Sec. III, we describe the microarchitecture of the proposed accelerator, optimal precision study, and methodology to efficiently map DNNs onto XNOR-SRAM arrays. Sec. IV reports experimental results on speed, energy dissipation, and classification accuracy across several workloads. Finally, the paper is concluded in Sec. V.

II. Challenges of Accelerator Design with XNOR-SRAM Macros

In [21], "XNOR-SRAM", a mixed-signal IMC SRAM macro, demonstrates scalability and efficient mapping capability of key computing primitives in binarized neural networks (BNNs) [2], [22], i.e. convolutional and fully-connected layers of Convolutional Neural Networks (CNNs) and Multi-Layer Perceptrons (MLPs). The macro consists of a 256-by-64 custom SRAM array, a row decoder, and a read periphery including a 11-level flash ADC shared across 64 columns. XNOR-SRAM supports computation with binary weights (+1/-1) and binary inputs (+1/-1 or +1/0) as well as ternary inputs (+1/0/-1).

XNOR-SRAM macro was prototyped in 65 nm CMOS, and achieves 81.28 pJ and 178 ns for 64 operations of 256-input XAC at 0.6V. Classification accuracies for MNIST and CIFAR-10 datasets have been evaluated on a single XNOR-SRAM macro in [21]. For MNIST, a MLP consisting of three hidden layers was used. For CIFAR-10, a CNN consisting of six convolutional layers and three fully-connected layers [22] was used. Accumulation of the XNOR-SRAM outputs, max-pooling, and batch normalization were performed in digital simulation with bit precisions of 12, 12, and 10, respectively. The networks computed with XNOR-SRAM achieve 85.7% accuracy for CIFAR-10 and 98.3% for MNIST.

Although XNOR-SRAM and other IMC SRAMs show promising energy-efficiency at the single-array-level, there are several important challenges and missing pieces towards building a chip-level DNN accelerator using these arrays.

Integration of many IMC SRAM arrays: First, to implement an overall DNN accelerator, many IMC SRAM arrays need to be instantiated and integrated together with on-chip communication networks because the capacity of a typical memory array is insufficient to hold enough weights for substantial computation.

ADC overhead and offset cancellation: Second, the ADC design incurs area and power overhead. Furthermore, the performance is sensitive to offset or variability. Calibration and offset compensation thus will incur additional area/power [23].

Post-processing modules: Third, DNNs acceleration using IMC macros still require partial sum accumulation, batch normalization, pooling, non-linear activation, etc. These modules add system-level design complexity and energy consumption.

Activation storage and communication: Fourth, typically the IMC SRAM arrays store the DNN weights, which means that the activations need to be stored in a separate memory and communicated to the IMC SRAM arrays at the right time. It can become the new bottleneck.

Write energy: Finally, as state-of-the-art DNNs are very large, it's not possible to store the entire weights of DNNs in IMC SRAMs. It will be necessary to reload weights as they are used. While IMC SRAMs can have high parallelism, loading new weights still requires row-by-row operation, which incurs long latency and consumes write energy.

In the subsequent sections, we will describe how a number of these key challenges have been addressed and inclusively implemented in the proposed accelerator design, and will report the accelerator evaluation results.

III. Vesti Accelerator Design

A. Microarchitecture Overview

Fig. 1(a) shows the Vesti microarchitecture. Using double-buffering scheme [24]), it consists of two symmetric cores, while one performs in-memory computation, the other load new weights from an on-chip global buffer or DRAM. Each block consists of (1) ensemble of XNOR-SRAM macros that compute convolution and fully-connected layers, and (2) a digital ALU that performs other operations such as batch normalization and max-pooling. In this work, 36×2 XNOR-SRAM macros are employed (Fig. 1) to support representative CNNs for CIFAR-10 dataset [22]. To support larger CNNs, the XNOR-SRAM macros can be time-multiplexed.

The inputs and weights are fetched from off-chip. The inputs are then saved in the activation memory buffer; weights are written into the XNOR-SRAM macros. The 36 XNOR-SRAM macros in each core are divided in 4 groups. The 4 groups share the input activations, performing XAC operations for up to 256 (=64×4) output feature maps in parallel. The 9 XNOR-SRAM macros in each group accept inputs from up to 256 input feature maps when performing 3x3 convolution and up to 2304 (=256×9) inputs in fully-connected matrix vector multiplication. In each cycle, up to 36 256-input XAC operations can be executed in parallel. The outputs of 36 XNOR-SRAM macros are processed by a 256-way digital ALU where ADC output decoding, partial sum accumulation, max-pooling, batch normalization and binary/ReLu activation are performed. The results will be saved back to the activation memory buffer of the other core, which will perform computation for the ensuing layer in a similar fashion.

B. Multi-bit Activation Support

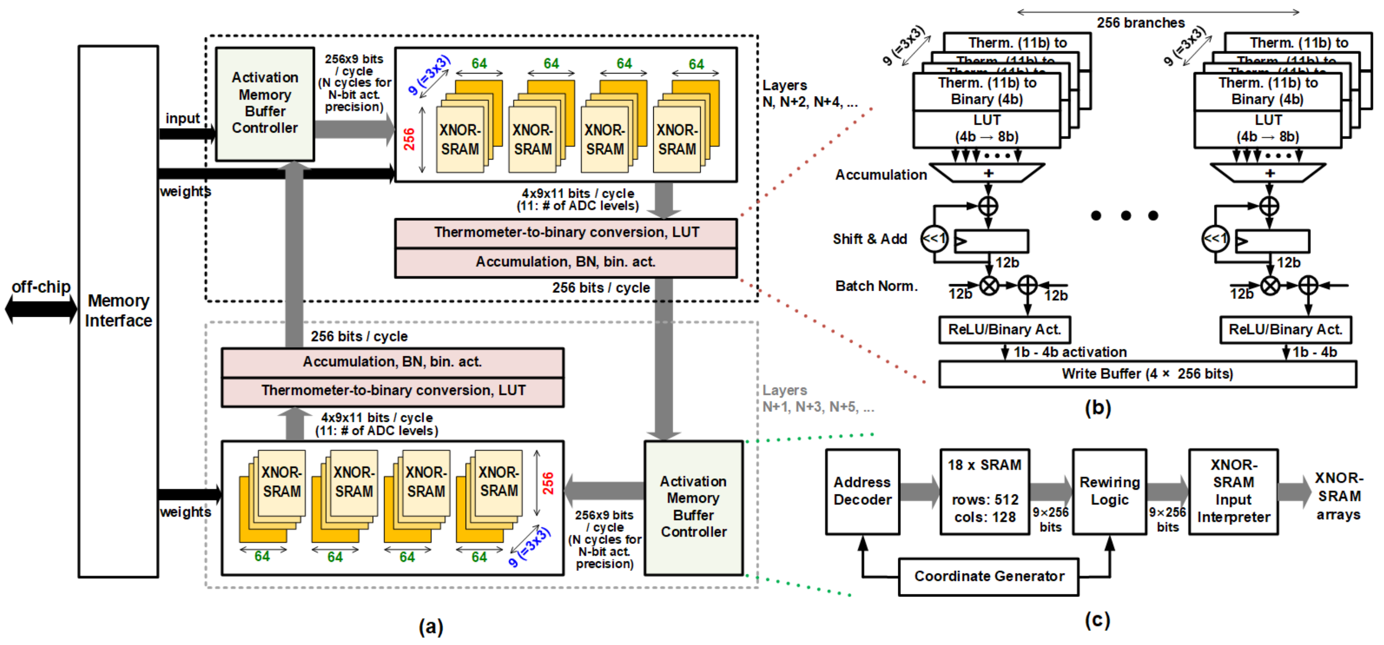
XNOR-SRAM natively supports binary activations and weights, however binary DNNs do not yet reach the same accuracy level of higher precision counterparts in some tasks [2], [22]. Therefore, Vesti support weights with binary precision (+1 or -1) and activations with configurable precision from 1-bit to 4-bit. This precision range is based on algorithmic experiments that we conducted. In particular, we swept (1) several CNN sizes (various numbers of feature maps per layer) for the CIFAR-10 dataset, and (2) activation precision values including binary, ternary, 2-bit, 4-bit, 8-bit, and 32-bit floating point. Accuracy of models using floating-point activation could be reached by employing 3-/4-bit activation with binary weights, while binary activations do show considerable degradation in CNN accuracy for complex tasks.

In the microarchitecture level, the support of the multi-bit inputs is done by performing XAC operation for each bit of input/activation using XNOR-SRAM macros and then shift-and-accumulate the bitwise XAC results in the digital peripheries over multiple cycles (e.g., N cycles for N-bit precision of activations). This is illustrated in Fig. 1(b), together with other digital computations at the periphery. Configurable precision (from 1-bit to 4-bit) for activations can be flexibly supported at the cost of additional clock cycles.

C. Activation Memory

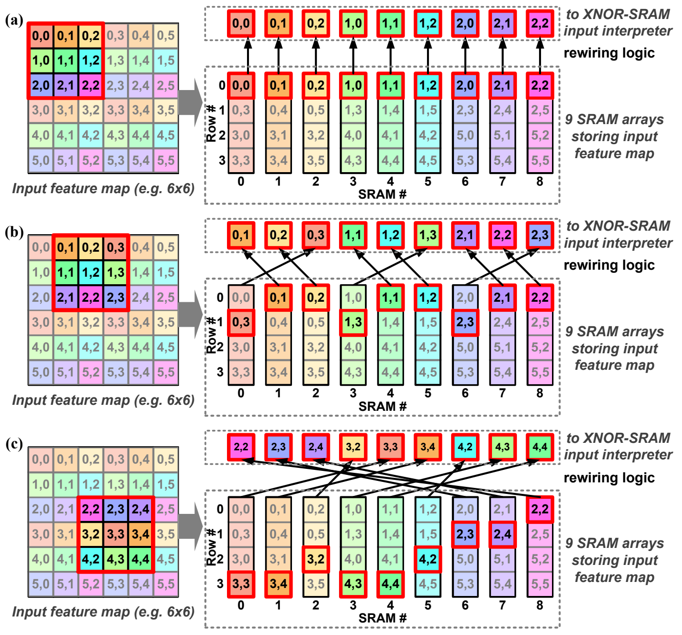
Activation memory in each core stores the input feature maps for that core and the output activations from the other core. We propose a way to store feature maps in the activation memory to exploit data reuse. In particular, we divide the overall activation memory into 9 activation SRAM blocks of 128 rows and 256 columns as shown in Fig. 2. The input feature maps are divided in 3×3 tiles. The input feature maps pixels are grouped and stored in 9 SRAM blocks according to their position in the 3×3 tiles they belong to. In this way, we can read all the 256×3×3 pixels in a single cycle given the fact that any 3×3 patch of input feature maps is now stored in 9 different SRAM blocks. A controller block is designed to generate corresponding addresses for the 9 SRAM blocks.

As shown in Fig. 1(c), the activation memory buffer has four parts: (1) coordinate generator, (2) address decoder, (3)



off-chip
Memory Interface
input
weights
Activation Memory Buffer Controller
256x9 bits / cycle (N cycles for N-bit act. precision)
XNOR-SRAM
64
256
9 (=3x3)
4x9x11 bits / cycle (11: # of ADC levels)
Thermometer-to-binary conversion, LUT
Accumulation, BN, bin. act.
256 bits / cycle
Layers N, N+2, N+4, ...
Layers N+1, N+3, N+5, ...
(a)
256 branches
Therm. (11b) to Binary (4b)
LUT (4b → 8b)
Accumulation
Shift & Add
Batch Norm.
12b
ReLU/Binary Act.
1b - 4b activation
Write Buffer (4 × 256 bits)
(b)
Address Decoder
18 x SRAM rows: 512 cols: 128
9×256 bits
Rewiring Logic
XNOR-SRAM Input Interpreter
XNOR-SRAM arrays
Coordinate Generator
(c)

Fig. 1: (a) Overall microarchitecture of proposed in-memory computing Vesti accelerator. (b) Computations for the thermometer-to-binary conversion, LUT, batch normalization, etc. (c) Block diagram of the activation memory buffer.



(a)
(b)
(c)
Input feature map (e.g. 6x6)
to XNOR-SRAM input interpreter
rewiring logic
9 SRAM arrays storing input feature map
Row #
SRAM #

Fig. 2: Illustration of convolution layer feature map storage and access scheme in 9 independent SRAM arrays. (a) 3x3 window starting from (0, 0); (b) 3x3 window starting from (0, 1); (c) 3x3 window starting from (2, 2).

rewiring logic, and (4) XNOR-SRAM input interpreter module, along with the 18 SRAM blocks.

Coordinate Generator: The coordinate generator block generates the x and y coordinates of the output map pixels sequentially in a row-major order. When the convolutional layer is followed by a pooling layer, the coordinates correspond to each pooling window in a row-major order, easing buffer size requirement for pooling operation. These coordinates will serve as inputs to the address decoder and rewiring logic blocks to different combinations of row addresses and read enable signals for SRAM blocks.

Address Decoder: The address decoder block generates activation memory addresses, including zero padding, for the 9 SRAM blocks according to the output feature map coordinates. When the generated addresses are invalid for the input feature maps, corresponding read enable signals will be inactive and substitute the SRAM output with zero values. Since the feature map size and channel size vary from layer to layer, we further divide each 256-bit-word-length SRAM block in two 128-bit-word-length SRAM blocks. For some layers (e.g., map size = 32×32, channel size = 128), we concatenate these two in depth direction to form a deeper SRAM block with 128-bit word length; for some layers (e.g., map size = 16×16, channel size = 256), we concatenate these two in word direction to form a wider SRAM block with 256-bit word length.

Rewiring logic: As shown in Fig. 2, the SRAM outputs the 3×3 patches are rewired into the correct order to be presented to the XNOR-SRAM macros. There are 9 different rewiring patterns in total, depending on the 3×3 patch row and column offset remainder modulo by 3. Fig. 2 illustrates 3 different patterns with an example of 6×6 feature maps, where the patch row and column offset is (0, 0), (0, 1) and (2, 2), respectively.

XNOR-SRAM input interpreter: Depending on whether we operate the XNOR-SRAM in 1-bit binary activation (+1/-1) or multi-bit activation (+1/0) mode, the XNOR-SRAM input interpreter will generate proper wordline inputs for XNOR-SRAM array from the rewiring logic output.

D. Mapping of Convolution, Fully Connected, and Other Layers

For convolution layers, we propose a mapping scheme where the same location pixels (x, y) from each kernels are stored in

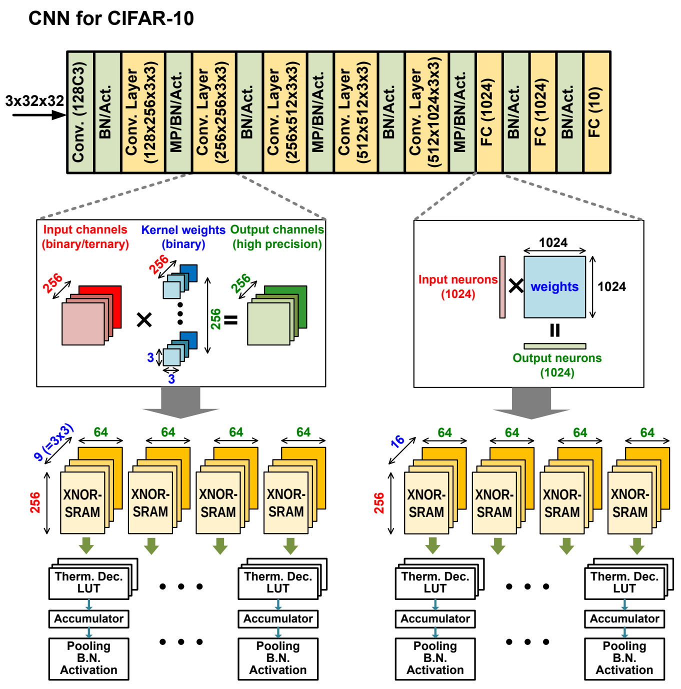


Fig. 3: Mapping convolution layers (left) and fully-connected layers (right) of deep CNNs onto the Vesti accelerator employing XNOR-SRAM macros with in-memory computing.

the same XNOR-SRAM array. Each pixels of a single kernel are stored in different XNOR-SRAM arrays. This is illustrated in Fig. 3 (left). Then, the XAC or bitcount accumulation results will be gathered from these multiple XNOR-SRAM arrays and accumulated together, to obtain the final output activation result. This scheme enables extensive re-use of the activations, with weights being stationary at the XNOR-SRAM arrays.

Using the weight-stationary scheme in the XNOR-SRAM macros, it is straightforward to map fully connected layers of DNNs, where neurons/activations are in vectors and weights are in matrices. This nicely maps to the row drivers for activations and weights stored in the SRAM. For the fully-connected layers whose size is larger than 256x64, we break the large weight matrix into a number of small sub-matrices and accumulate the matrix-vector multiplication results accordingly. This is illustrated in Fig. 3 (right).

We implement other computation modules such as max-pooling, batch normalization, and non-linear activation with all-digital circuits, at the periphery of XNOR-SRAM macros. The computing sequences of these modules are as follows. Once the XAC operations are done inside the XNOR-SRAM array, flash ADC digitizes the voltage in 11 levels, which is then used in a simple look-up table (LUT) to find the quantized bit count. Then, the same columns from each XNOR-SRAM arrays are accumulated to find the final output sum value. Using the trained batch normalization parameters, this final sum value goes through batch normalization and non-linear quantization (thresholding for binary/ternary activations, ReLU for multi-bit activations). Finally when the current layer's computation is completed, the activation outputs will serve as the input activations for the next layer of the DNN/CNN.

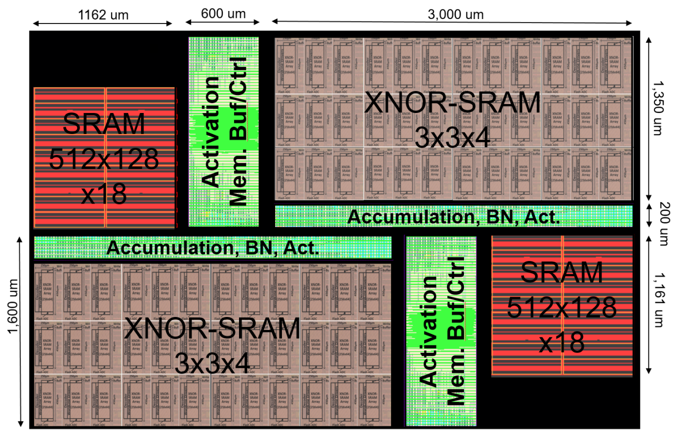


Fig. 4: Including the XNOR-SRAM prototype chip layout, the layout of activation memory buffer/controller, accumulation and batch normalization modules are shown.

IV. Experimental Results

A. Experiment Setup

We characterized the power/energy consumption, throughput and accuracy (for MNIST and CIFAR-10 datasets) of the Vesti accelerator. For all the MAC or XAC operations, the XNOR-SRAM testchip measurements from [21] are used. For all other digital logic and off-the-shelf SRAMs, the power consumption results are simulated with the post-layout netlist with RC parasitics and actual data switching activity information.

We considered MLPs and CNNs for image classification tasks for MNIST and CIFAR-10 datasets, respectively. For MNIST, a MLP with three hidden layers, each with 256 neurons, is used. The CIFAR-10 CNN architecture used in this section is adopted from the CNN reported in [22], consisting of six convolution layers and three fully-connected layers. This represents the network of input-128C3-128C3-MP2-256C3-256C3-MP2-256C3-256C3-MP2-1024FC-1024FC-10FC.

To evaluate the accuracy of binary-weight multi-bit-activation MLPs and CNNs on Vesti, we first obtained a probabilistic model for the XNOR-SRAM XAC and quantization operations from XNOR-SRAM chip measurements. Specifically, we used a total of 656k (513 XAC values$\times$64 columns$\times$20 samples/XAC/column) random test vectors and measured the outputs of the XNOR-SRAM to build XNOR-SRAM's probabilistic model as a function of XAC value. We simulated BNN model accuracy on Vesti by stochastically quantizing XAC partial sums according to the probabilistic model.

B. Area, Energy, Throughput, and Accuracy

In Fig. 4, the placed-and-routed layout of all digital peripheral blocks as well as the 72 XNOR-SRAM arrays are shown. The total area of the Vesti accelerator is 15 mm^2 in the 65 nm CMOS process. Multiple XNOR-SRAM arrays consume 54% of the total area, the activation SRAMs consume 18%, and remainder of the area (28%) is occupied by digital logic and control modules.

For MNIST MLP, we simulated and evaluated the total MLP energy for various activation precisions of 1-3 bits. Since

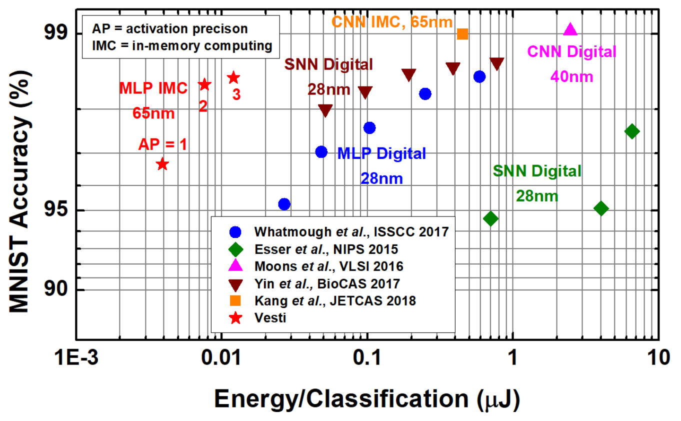


Fig. 5: For MNIST dataset, accuracy vs. energy (per classification) comparison with prior works is shown.

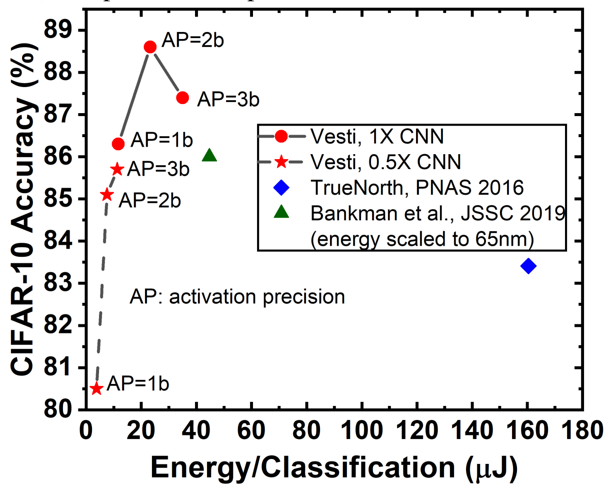


Fig. 6: For CIFAR-10 dataset, accuracy vs. energy (per classification) comparison for variable activation precision with prior works is shown.

activations with N-bit precision consume N cycles to compute using the XNOR-SRAM array, the overall energy roughly increases linearly with the activation precision. To perform a single inference of the MLP using the 1-bit activation precision, the Vesti accelerator consumes 21 cycles. At Vesti's max frequency 0.55-GHz, the throughput of 26M inferences per second is achieved.

For the MNIST dataset, in Fig. 5, we compared Vesti (with 1-/2-/3-bit activation precision) to several prior works [9], [18], [25]–[27] that demonstrated the energy and accuracy for the entire DNN for MNIST dataset. Vesti accelerator results in superior accuracy versus energy trade-offs compared to the state-of-the-art DNN implementations for MNIST dataset.

For CIFAR-10 CNN, we also simulated and evaluated the total CNN energy for various activation precisions of 1-3 bits for two CNN sizes (0.5X refer to half channel size). To perform the single inference of the CIFAR-10 CNN using the 1-bit activation precision, the Vesti accelerator consumes 1,676 cycles. At 0.55-GHz, this marks the throughput of 328K inferences per second. Fig. 6 shows the energy across three activation precision values (from 1-bit to 3-bit) and two CNN sizes (1X and 0.5X CNN). For the CIFAR-10

TABLE I: Comparison with prior works

	[46]	[47]	This Work	[19]	[43]	[44]	[45]	[28]
Technology (nm)	45	28	65	28	45	40	28	65
Model	SNN	Conv	Conv/MLP	MLP	SNN	Conv	SNN	Conv
Voltage Supply (V)	0.6-0.8	0.6-0.8	1	0.6-1.1	0.6-0.8	0.55-1.1	0.9	1
Clock Frequency (MHz)	--	10	550	667	--	204	163	--
CIFAR10 Accuracy (%)	83.41	86.05	88.6	--	--	--	--	--
CIFAR10 Throughput (FPS)	1249	237	328 k	--	--	--	--	--
CIFAR10 Energy/ Prediction (μJ)	163	3.8	23.3	--	--	--	--	--
MNIST Accuracy (%)	--	--	98.5	98.5	99.42	99	98.7	99
MNIST Throughput (FPS)	--	--	8.6 M	--	1 k	13.4 k	91.6 k	70
MNIST Energy/ Prediction (μJ)	--	--	0.012	0.588	108	0.45	0.773	0.45

dataset, in Fig. 6, we show the energy and accuracy trade-offs for variable activation precision using Vesti accelerator, and also compared to the energy and accuracy of the all-digital TrueNorth processor reported in [28] and the mixed-signal binary CNN processor reported in [29] (energy scaled to 65nm). As shown in Fig. 6, Vesti accelerator results in superior accuracy and energy trade-offs for two different CNN sizes (1X and 0.5X) for multiple activation precision schemes for the CIFAR-10 dataset. Comparison with prior works is summarized in Table I.

V. Conclusion

Due to the memory access bottleneck faced by digital NN accelerators, in-memory computing has been gaining significant attention as a solution. However, in many prior IMC works, onlt the memory macro is presented, the overall operation and architecture to implement the DNN accelerator have not been shown or implemented in hardware.

In this work, we expanded the single-array-level prior XNOR-SRAM work [21] towards a configurable DNN accelerator architecture that integrates 72 XNOR-SRAM arrays. The proposed Vesti architecture features (1) methodologies to efficiently load/map weights onto such XNOR-SRAM arrays for convolutional layers and fully-connected layers of DNNs, (2) multi-bit activation memory storage and control, (3) double-buffering technique to hide the latencies of re-programming in-memory computing SRAM arrays, and (4) inter-array communication.

Due to these comprehensive designs, Vesti simultaneously achieves both high accuracy and low energy for representative DNNs that are benchmarked for MNIST and CIFAR-10 datasets. The Vesti accelerator presented in this work feature essential techniques for in-SRAM computing based deep learning processors, which can fit under the stringent power/energy envelopes of mobile, wearable, and IoT devices.

Acknowledgements

This work is in part supported by NSF grant 1652866, and C-BRIC, one of six centers in JUMP, a SRC program sponsored by DARPA.

References

[1] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *International Conference on Learning Representations (ICLR)*, 2016.

[2] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *European Conference on Computer Vision (ECCV)*, 2016.

[3] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *CoRR*, vol. abs/1609.07061, 2016. [Online]. Available: http://arxiv.org/abs/1609.07061

[4] S. Zhou, Z. Ni, X. Zhou, H. Wen, Y. Wu, and Y. Zou, "DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients," *CoRR*, vol. abs/1606.06160, 2016. [Online]. Available: http://arxiv.org/abs/1606.06160

[5] T. Guan, X. Zeng, and M. Seok, "Recursive Binary Neural Network Learning Model with 2-bit/weight Storage Requirement," *CoRR*, vol. abs/1709.05306, 2017. [Online]. Available: https://arxiv.org/abs/1709.05306

[6] 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, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, August 2014.

[7] Y.-H. Chen, T. Krishna, J. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 52, no. 1, pp. 127–138, 2017.

[8] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "ENVISION: A 0.26-to-10TOPS/W Subword-Parallel Dynamic-Voltage-Accuracy-Frequency-Scalable Convolutional Neural Network Processor in 28nm FDSOI," in *IEEE International Solid-State Circuits Conference (ISSCC)*, February 2017, pp. 246–247.

[9] P. N. Whatmough, S. K. Lee, H. Lee, S. Rama, D. Brooks, and G.-Y. Wei, "A 28nm SoC with a 1.2GHz 568nJ/Prediction Sparse Deep-Neural-Network Engine with >0.1 Timing Error Rate Tolerance for IoT Applications," in *IEEE International Solid-State Circuits Conference (ISSCC)*, February 2017, pp. 242–243.

[10] Q. Dong, S. Jeloka, M. Saligane, Y. Kim, M. Kawaminami, A. Harada, S. Miyoshi, D. Blaauw, and D. Sylvester, "A 0.3V VDDmin 4+2T SRAM for Searching and In-Memory Computing Using 55nm DDC Technology," in *IEEE Symposium on VLSI Circuits*, 2017.

[11] M. Kang, S. K. Gonugondla, and N. R. Shanbhag, "A 19.4 nJ/decision 364K decisions/s In-Memory Random Forest Classifier in 6T SRAM Array," in *European Solid-State Circuits Conference (ESSCIRC)*, 2017.

[12] J. Zhang, Z. Wang, and N. Verma, "A Machine-Learning Classifier Implemented in a Standard 6T SRAM Array," in *IEEE Symposium on VLSI Circuits (VLSI)*, June 2016, pp. 1–2.

[13] A. Biswas and A. P. Chandrakasan, "Conv-RAM: An Energy-Efficient SRAM with Embedded Convolution Computation for Low-Power CNN-Based Machine Learning Applications," in *IEEE International Solid-State Circuits Conference (ISSCC)*, February 2018.

[14] S. K. Gonugondla, M. Kang, and N. Shanbhag, "A 42pJ/Decision 3.12TOPS/W Robust In-Memory Machine Learning Classifier with On-Chip Training," in *IEEE International Solid-State Circuits Conference (ISSCC)*, February 2018.

[15] W.-H. Chen, K.-X. Li, W.-Y. Lin, K.-H. Hsu, P.-Y. Li, C.-H. Yang, C.-X. Xue, E.-Y. Yang, Y.-K. Chen, Y.-S. Chang, T.-H. Hsu, Y.-C. King, C.-J. Lin, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, and M.-F. Chang, "A 65nm 1Mb Nonvolatile Computing-in-Memory ReRAM Macro with Sub-16ns Multiply-and-Accumulate for Binary DNN AI Edge Processors," in *IEEE International Solid-State Circuits Conference (ISSCC)*, February 2018.

[16] W.-S. Khwa, J.-J. Chen, J.-F. Li, X. Si, E.-Y. Yang, X. Sun, R. Liu, P.-Y. Chen, Q. Li, S. Yu, and M.-F. Chang, "A 65nm 4Kb Algorithm-Dependent Computing-in-Memory SRAM Unit-Macro with 2.3ns and 55.8TOPS/W Fully Parallel Product-Sum Operation for Binary DNN Edge Processors," in *IEEE International Solid-State Circuits Conference (ISSCC)*, February 2018.

[17] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, "A Mixed-Signal Binarized Convolutional-Neural-Network Accelerator Integrating Dense Weight Storage and Multiplication for Reduced Data Movement," in *IEEE Symposium on VLSI Circuits*, 2018, pp. 141–142.

[18] M. Kang, S. Lim, S. Gonugondla, and N. R. Shanbhag, "An in-memory vlsi architecture for convolutional neural networks," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 8, no. 3, pp. 494–505, Sept 2018.

[19] P. Srivastava, M. Kang, S. K. Gonugondla, S. Lim, J. Choi, V. Adve, N. S. Kim, and N. Shanbhag, "PROMISE: An end-to-end design of a programmable mixed-signal accelerator for machine-learning algorithms," in *IEEE International Symposium on Computer Architecture (ISCA)*, June 2018.

[20] C. Eckert, X. Wang, J. Wang, A. Subramaniyan, R. Iyery, D. Sylvester, D. Blaauw, and R. Das, "Neural Cache: Bit-Serial In-Cache Acceleration of Deep Neural Networks," in *IEEE International Symposium on Computer Architecture (ISCA)*, June 2018.

[21] Z. Jiang, S. Yin, M. Seok, and J. Seo, "XNOR-SRAM: In-Memory Computing SRAM Macro for Binary/Ternary Deep Neural Networks," in *IEEE Symposium on VLSI Technology/Circuits*, 2018, pp. 173–174.

[22] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized Neural Networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 4107–4115.

[23] C. Chen, M. Q. Le, and K. Y. Kim, "A Low Power 6-bit Flash ADC With Reference Voltage and Common-Mode Calibration," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 4, pp. 1041–1046, April 2009.

[24] J. C. Sancho and D. J. Kerbyson, "Analysis of Double Buffering on Two Different Multicore Architectures: Quad-core Opteron and the Cell-BE," in *IEEE International Symposium on Parallel and Distributed Processing*, April 2008, pp. 1–12.

[25] S. K. Esser, R. Appuswamy, P. Merolla, J. V. Arthur, and D. S. Modha, "Backpropagation for energy-efficient neuromorphic computing," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., 2015, pp. 1117–1125.

[26] B. Moons and M. Verhelst, "A 0.3-2.6 TOPS/W precision-scalable processor for real-time large-scale ConvNets," in *2016 IEEE Symposium on VLSI Circuits (VLSI-Circuits)*, June 2016, pp. 1–2.

[27] S. Yin, S. K. Venkataramanaiah, G. K. Chen, R. Krishnamurthy, Y. Cao, C. Chakrabarti, and J. sun Seo, "Algorithm and Hardware Design of Discrete-Time Spiking Neural Networks Based on Back Propagation with Binary Activations," in *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, October 2017.

[28] S. K. Esser, P. A. Merolla, J. V. Arthur, A. S. Cassidy, R. Appuswamy, A. Andreopoulos, D. J. Berg, J. L. McKinstry, T. Melano, D. R. Barch, C. di Nolfo, P. Datta, A. Amir, B. Taba, M. D. Flickner, and D. S. Modha, "Convolutional networks for fast, energy-efficient neuromorphic computing," *Proceedings of the National Academy of Sciences*, vol. 113, no. 41, pp. 11 441–11 446, 2016.

[29] D. Bankman, L. Yang, B. Moons, M. Verhelst, and B. Murmann, "An always-on 3.8 μ j/86chip in 28-nm cmos," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 1, pp. 158–172, Jan 2019.