CNNWire: Boosting Convolutional Neural Network with Winograd on ReRAM based Accelerators

Jilan Lin, Shuangchen Li, Xing Hu, Lei Deng, Yuan Xie
{jilan,shuangchenli,xinghu,leideng,yuanxie}@ucsb.edu
University of California, Santa Barbara

ABSTRACT

Resistive random access memory (ReRAM) demonstrates the great potential of in-memory processing for neural network (NN) acceleration. However, since the convolutional neural network (CNN) is widely known as compute-bound, current ReRAM-based accelerators are not able to support CNN efficiently. In this paper, we for the first time propose the **CNN** accelerator with **Wi**nograd's convolution on **Re**RAM (CNNWire), which minimizes the multiplications to enable fast and efficient CNN inference. We realize the convolution with Winograd Processing Element (WPE) based on convolutional tiles. Interconnections between WPEs are designed aiming to improve the data reuse. Finally, we introduce the full mapping flow to implement the Winograd convolution The results show that CNMWire gains 3.85× energy efficiency boosting and 3.24× speedup on average among different CNN benchmarks, compared with traditional GEMM based mapping.

CCS CONCEPTS

• **Computing methodologies → Machine learning**; • **Hardware → Application specific integrated circuits**;

KEYWORDS

CNN, ReRAM, Winograd Convolution

ACM Reference format:
Jilan Lin, Shuangchen Li, Xing Hu, Lei Deng, Yuan Xie. 2019. CNNWire: Boosting Convolutional Neural Network with Winograd on ReRAM based Accelerators. In *Proceedings of Great Lakes Symposium on VLSI 2019, Tysons Corner, VA, USA, May 9–11, 2019 (GLSVLSI '19),* 4 pages.
https://doi.org/10.1145/3299874.3318018

1 INTRODUCTION

The inference for neural network (NN) is both time- and energy-consuming. Recently, the Resisitive Random Access Memory (ReRAM) becomes attractive to researchers by showing the ability to perform matrix-vector multiplication (MVM) within memory arrays.

ReRAM demonstrates two main advantages when conducting NN acceleration. First, ReRAM conducts very fast MVM. When the data are stored in the ReRAM crossbar array, we can get the results within a single cycle. Second, ReRAM provides a promising non von-Neumann architecture. Since the matrix data are stored in the memory array, the data movement for weights is not needed any more, and thus we archive the in-situ computation and save the time and energy cost for data fetching.

However, as the most common used NNs in computer vision, CNNs are compute-bound and not friendly to ReRAM devices. The reason is that convolution requires much of the execution time and frequent input reusing for every weight kernel, but currently it is difficult to leverage the data reusing in analog field with analog caching. As studies have shown digital/analog interfaces occupy the majority of the energy consumption in ReRAM based analog computing [5], this limits the ReRAM-based CNN accelerators.

In this paper, we for the first time propose Winograd convolution based CNNs acceleration on ReRAM. Winograd convolution is a fast algorithm [8], which has already been applied in the cuDNN [1]. Leveraging the transforming for inputs and filters, Winograd algorithm turns the convolution into matrix operation and makes the least times of multiplication. We analyze the data flow for the algorithm and design the full architecture with mapping scheme. As dealing with tiled convolution, we realize Winograd Processing Element (WPE) for every tile, which conducts the transform in digital field and leverages the channelized accumulating in ReRAM crossbars. Further, to make better use of the data locality and reusing, interconnections are designed for fast data transfer between WPEs. The main contributions of this work are as follows:

- We boost the CNN with a novel architecture enabling **Wi**nograd algorithm on **Re**RAM devices (CNNWire). We design Winograd Processing Elements (WPEs) for tiles in convolution. We propose buffer organization and interconnections between WPEs to enhance the data reusing in Winograd tiles.
- We proposed a full mapping scheme to implement convolutional layers in CNNWire with transforming the element-wise multiplication into channelized matrix multiplication.
- We evaluate our architecture by comparison with traditional GEMM mapping and overlapped mapping, which shows about 3.85× energy boosting and 3.24× speedup.

2 PRELIMINARY AND MOTIVATION

In this section, we will introduce ReRAM, CNN and Winograd algorithm. Then we present our motivation in detail.

2.1 RRAM and RRAM Computing System

ReRAM stores information with resistive cells. The ReRAM crossbar structure performs analog MVM with complexity of $O(1)$. Equ. 1 expresses the relationship between input voltage vector (V_{in}) and the output voltage vector (V_{out}) among cell conductance g.

$$v_{out}^{j} = \sum_{i=1}^{N} v_{in}^{i} g_{i,j} \tag{1}$$

2.2 CNN and Winograd Convolution

A convolution layer can be expressed with Eq. 2:

$$d^{l+1}_{x,y,c_{l+1}} = f(\sum_{c_l=0}^{C_l-1}\sum_{r=0}^{R-1}\sum_{s_l=0}^{S-1} d^l_{x+r,y+s,c_l} \times w^l_{r,s,c_l,c_{l+1}}) \quad (2)$$

Here C is the number of feature maps with size $H^l \times W^l$, and l denotes the current layer. The size of a kernel is $R \times S$. w^l is the weight and d^l is the input data. f is a non-linear function.

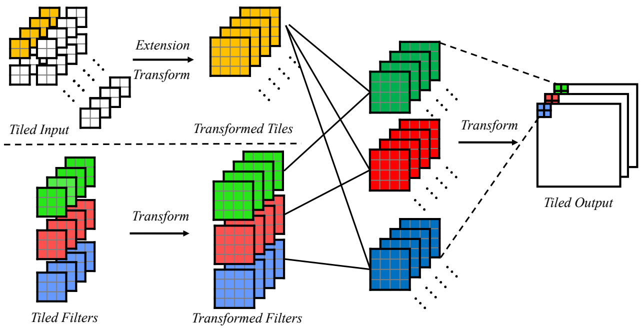

Figure 1: An illustration of Winograd convolution.

Fig. 1 gives an illustration of Winograd convolution. It first tiles the input feature maps (IFMs) into 2×2 pieces. Second, both the tiles and kernels are transformed into the same dimension. Third, we conduct element-wise multiplication, and then transform back to the convolution results. The process can be formulated as Eq. 3:

$$y = A^T[(Gw_lG^T) \odot (B^Td_lB)]A \quad (3)$$

The A, G, B are three transform matrices. With 4×4 tiles and convolutional kernel of 3×3, the transform matrix A and B for input/output can be as simple as shown in Eq. 4:

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & -1 \\ 0 & -1 \end{bmatrix}, B^T = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix} \quad (4)$$

2.3 Motivation

Previous work presented two mappings for convolution. The first one is GEneral Matrix Multiplication (GEMM) based mapping [7]. As shown in Fig. 2(a), it unfolds the 3-D convolution and computes as matrix multiplication. An alternative is to directly do the convolution with overlapped weights mapping scheme [10], as shown in

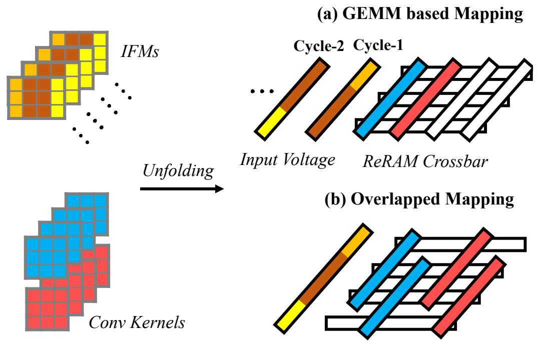

Figure 2: Two common used mappings. (a) GEMM based mapping. (b) Overlapped mapping.

Fig. 2(b). Kernels get duplicated in the crossbar array, and a longer vector of input pixels can directly does the convolution in 1 cycle.

GEMM based convolution requires iterative computing to traverse the IFM, which makes CNN quite compute-intensive. Besides, we need two cycles of DA converting and the data reuse cannot be fully exploited. Fig. ref{fig:moti gives the comparison between the energy/time consumption and parameter amount. We find that the convolutional layers dominate the execution time and energy for CNNs. For VGG-16, about 10% convolution parameters occupy more than 90% execution time and energy for the total network.

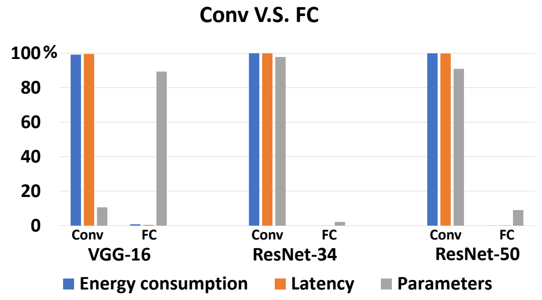

Figure 3: The energy/time consumption of convolutional and FC layers. The data are shown in percentage.

Overlapped mapping reduces the DA interfaces and is able to increase the throughput arbitrarily by duplicating more kernels. However, this obviously requires more memory capacity. Also, it introduces redundancy in crossbar as we see some ReRAM cell not used. Fig. 4 demonstrates the trend of memory utilization rate when increasing the duplication times, with mapping the conv3-128 layer in VGG-16. For the crossbars with size 64×64, the utilization rate keeps going down, and with only 3 times of duplication, more than 40% of the memory is idle.

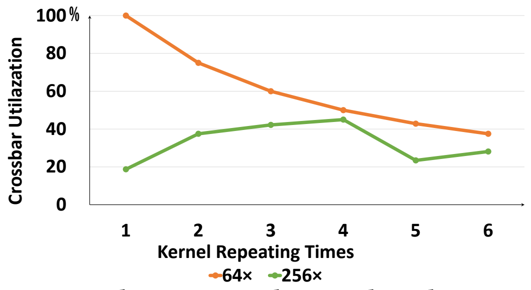

Figure 4: The ReRAM utilization degrading curve.

3 CNNWIRE ARCHITECTURE

In this section, we first introduce the full architectures of CNNWire, and then the mapping for Winograd algorithm.

3.1 Overview of the architecture

Fig. 5 shows the overview of CNNWire. Within a bank, the key components are Winograd Processing Elements (WPEs). WPE aims to process a channelized tile in Winograd algorithm. The transformed convolutional kernels are stored in the ReRAM array. The input/output bank buffers are composed with SRAM. A controller mainly decides the compute modes (Winograd based convolution, GEMM based convolution, FC, etc.) in WPEs. Finally, we enable the interconnections between WPEs to make use of the data reuse.

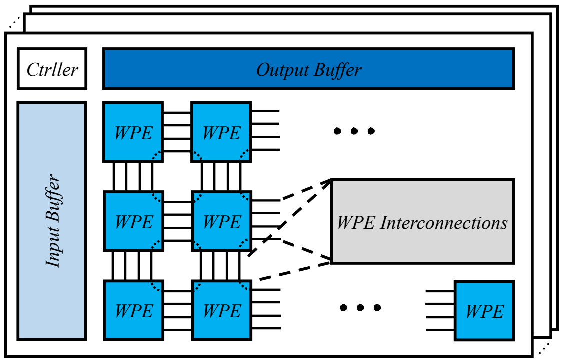

Figure 5: The overview of the whole architecture.

3.2 Winograd Processing Element

As shown in Fig. 7, WPE processes tiled IFMs (shown in orange) and gets the convolution results (shown as colored tiles). WPE consists of Winograd transform modules in Equ. 4, Matrix Vector (MV) multipliers, functional units, and input/output buffers. The tiled IFMs are buffered with registers, which will get transformed with WTM_B array and sent to the ReRAM based MV multiplier. WTM_A array deals with outputs transform.

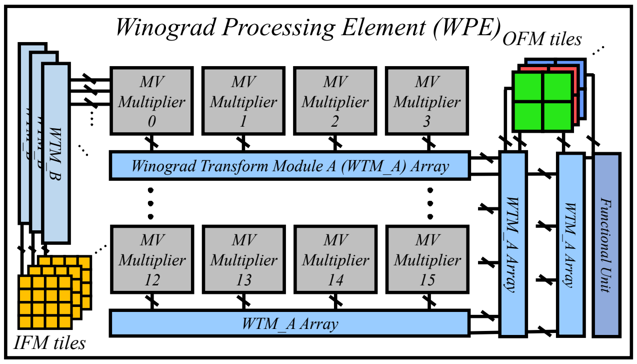

Figure 6: A illustration of WPE design.

Winograd Transform Module (WTM) are designed to complete the transform process for input and output tiles. According to Eq. 4, only 2/4 adders and NOT gates are needed for a 4-input vector. Besides, several modules are organized into arrays, which is for better leveraging the data locality.

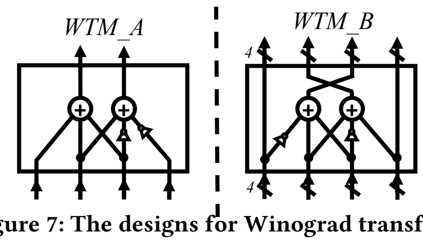

Figure 7: The designs for Winograd transform.

MV multipliers are realized by ReRAM crossbars along with shifting and adding trees, with complement format storing in ReRAM. We decide single-bit ReRAM for a higher reliability. No two pieces of crossbars are required anymore for positive and negative values. After the computations complete in crossbars, the results will shift certain number of bits according to the slice index and add together.

Functional unit is for activation and pooling operations.We only enable ReLU in the module.

3.3 Buffer and Communication Optimization

ReRAM devices are known for the energy-efficient reading but not writing [9]. Therefore, it is not suitable for activation buffer needing frequent data update. In CNNWire, we use SRAM for global bank buffer. On the other hand, regesiter files compose the buffer within WPE. We design interconnections between WPEs to reduce the data reloading from the global buffer. As shown in the Fig. 8, the input buffer in WPE is partitioned in 4 blocks, with 1 directly reading from global buffer, and 3 sharing with adjacent WPEs. This is because 1/4 of the extended tile is unique but 3/4 is shared.

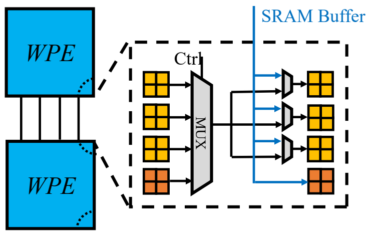

Figure 8: The WPE interconnections.

3.4 Winograd Mapping

Fig. 9 illustrates the mapping for Winograd convolution. First, IFMs are partitioned into $2 \times 2 \times C_l$ tiles as shown in orange. Then they are extended into $4 \times 4 \times C_l$ with adjacent pixels. Second, within the WPE, the data get transformed with WTM_B into $4 \times 4 \times C_l$. Third, the $3 \times 3 \times C_l$ filters, as shown in three colors, already get pre-transformed and stored in the crossbars. The element-wise matrix multiplication are accumulated across channels. As shown in the Fig. 9, deeper colors represent the first vector mapped to crossbar column, while lighter ones represent the 16th vector. In total 16 sets of ReRAM crossbars are needed. Finally, the MVM results are three vectors with length of 16. We fold them into 4×4 tiles and transform them back to 2×2, which are convolutional results.

4 EVALUATION

In this section, we evaluate the performance, energy results and area overhead of CNNWire.

4.1 Experiment Setup

We evaluate CNNWire with simulation. ReRAM and SRAM are simulated with NVSim [2] and CACTI 6.0 [6], and digital designs are synthesized with Synopsys DC with 45nm technology. DA/AD are referred to [3, 4]. ReRAM capacity has been fixed at 64MB per bank with 8 banks. We set 1KB registers for WPE buffer, and 4MB SRAM buffer per bank. System clock is 200MHz. We compare with GEMM based mapping and overlapped mapping The benchmarks are VGG-16 and ResNet serie. For 1×1 convolution and fully-connected layers, all the three mappings get to be the same.

4.2 Performance

As shown in Fig. 10, CNNWire improves the inference by 3.24× on average compared with GEMM based mapping, and outperforms the overlapped mapping (OM) by 1.22× on average - the reason for a relatively small speedup is that the overlapped mapping could maximize the throughput with arbitrary duplication.

For ResNet-50 and VGG-16, the speedups are lower than ResNet-18/34. The reason for ResNet-50 traces back to the huge amount of 1×1 convolutions, since there are 2 of 3 1×1 convolutional layers within every residual block. For VGG-16, the convolution layer 1

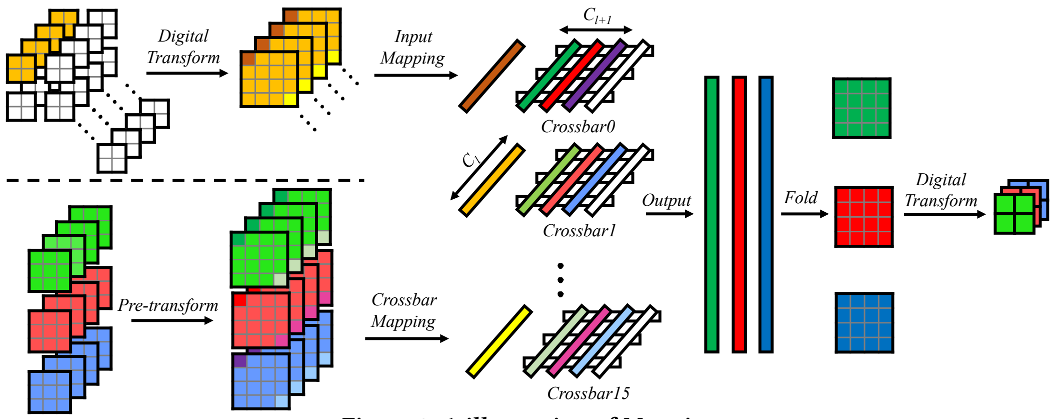

Figure 9: A illustration of Mapping.

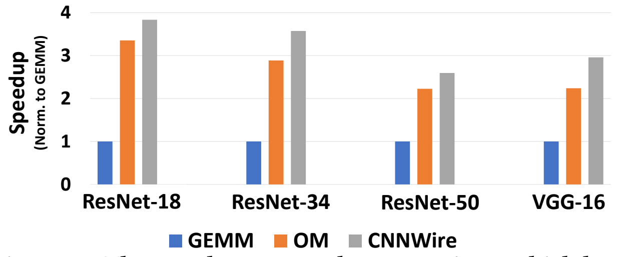

Figure 10: The speedup among three mappings, which have been normalized to GEMM based mapping.

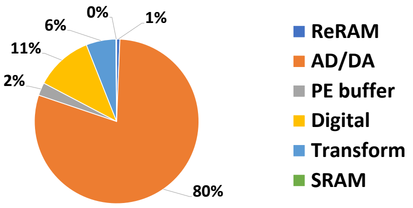

Figure 12: The area breakdown of CNNWire.

needs kernel size of 7 × 7 by GEMM with the largest IFMs, leading to huge amount of inefficiency.

4.3 Energy

Fig. 11 gives energy results of CNNWire. On average 3.85× energy efficiency boosting is observed compared with GEMM, while 1.88× on everage compared with overlapped mapping. It is still found that 1 × 1 convolutions degrades the energy performance in ResNet-50.

As shown in the energy breakdown, AD/DA interfaces occupy most of the energy consumption for ReRAM computing. CNNWire reduces a considerable amount of AD/DA energy with efficient Winograd convolution. Besides, the cost of buffering has also been reduced notably with the implementation of WPE interconnections.

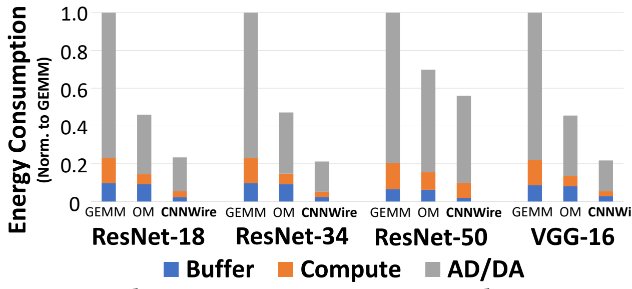

Figure 11: The energy consumption among three mappings with normalization to GEMM based mapping.

4.4 Area

The area breakdown of CNNWire is shown in Fig. 12. The AD/DA take over 80% area, with the ReRAM crossbar array being a really small part. The digital part occupies the second largest area. Our introduced Winograd transform circuits bring 6% area overhead, which is relatively small.

5 CONCLUSIONS

In this paper, we prensent CNNWire, a **CNN** accelerator with **Wi**nograd's convolution on **Re**RAM (CNNWire) to tackle the inefficiency in traditional mapping. We realize the convolution with Winograd Processing Element (WPE) dealing with convolution tiles.Interconnections between WPEs are designed aiming to improve the data reuse. Finally, we introduce the mapping flow to implement the Winograd convolution. The results show 3.85× energy efficiency boosting and 3.24× speedup on average among different CNN benchmarks, compared with traditional GEMM mapping.

ACKNOWLEDGMENTS

This work was supported in part by CRISP, one of six centers in JUMP, a SRC program sponsored by DARPA, NSF grant CCF 1740352 and SRC nCORE NC-2766-A, and NSF 1500848, 1719160, 1725447, and 1730309.

REFERENCES

[1] Sharan Chetlur et al. 2014. cuDNN: Efficient Primitives for Deep Learning. *CoRR* abs/1410.0759 (2014).
[2] Xiangyu Dong et al. 2012. NVSIM: A circuit-level performance, energy, and area model for emerging nonvolatile memory. *IEEE TCAD* 31, 7 (2012), 994–1007.
[3] Brian P Ginsburg and Anantha P Chandrakasan. 2007. 500-MS/s 5-bit ADC in 65-nm CMOS with split capacitor array DAC. *IEEE Journal of Solid-State Circuits* 42, 4 (2007), 739–747.
[4] Hyeok-Ki Hong et al. 2012. A 7b 1GS/s 7.2 mW nonbinary 2b/cycle SAR ADC with register-to-DAC direct control. In *CICC*. IEEE, 1–4.
[5] Boxun Li et al. 2015. Merging the interface: Power, area and accuracy co-optimization for rram crossbar-based mixed-signal computing system. In *DAC*. IEEE, 1–6.
[6] Naveen Muralimanohar, Rajeev Balasubramonian, and Norman P Jouppi. 2009. CACTI 6.0: A tool to model large caches. *HP laboratories* (2009), 22–31.
[7] Linghao Song et al. 2017. PipeLayer: A pipelined ReRAM-based accelerator for deep learning. In *HPCA*. 541–552.
[8] Shmuel Winograd. 1980. *Arithmetic complexity of computations*. Vol. 33. Siam.
[9] H-S Philip Wong et al. 2012. Metal–oxide RRAM. *Proc. IEEE* 100, 6 (2012), 1951–1970.
[10] Zhenhua Zhu et al. 2018. Mixed Size Crossbar based RRAM CNN Accelerator with Overlapped Mapping Method. In *ICCAD*.