Deep Neural Network Training Accelerator Designs in ASIC and FPGA

Shreyas K. Venkataramanaiah, Shihui Yin, Yu Cao, and Jae-sun Seo
School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA
Email: jaesun.seo@asu.edu

Abstract— **In this invited paper, we present deep neural network (DNN) training accelerator designs in both ASIC and FPGA. The accelerators implements stochastic gradient descent based training algorithm in 16-bit fixed-point precision. A new cyclic weight storage and access scheme enables using the same off-the-shelf SRAMs for non-transpose and transpose operations during feed-forward and feed-backward phases, respectively, of the DNN training process. Including the cyclic weight scheme, the overall DNN training processor is implemented in both 65nm CMOS ASIC and Intel Stratix-10 FPGA hardware. We collectively report the ASIC and FPGA training accelerator results.**

Keywords: on-device training, convolutional neural networks, hardware accelerator, energy efficiency

2020 International SoC Design Conference (ISOCC) | 978-1-7281-8331-2/20/$31.00 ©2020 IEEE | DOI: 10.1109/ISOCC50952.2020.9333063

I. INTRODUCTION

Training of DNNs are typically performed by high-end GPUs, but for edge devices that contain users' sensitive data, performing training on edge devices is preferred due to privacy concerns, as illustrated in Fig. 1. However, it is challenging to execute DNN training on energy/-area-limited edge devices, because the DNN training process with feed-forward (FF), feed-backward (FB), and weight update (WU) phases exhibits a much higher amount of computation and memory storage, compared to execution of DNN inference. One particular challenging aspect is that, DNN training algorithm needs to access the same weights in non-transpose manner for FF phase and in transpose manner in FB phase. Transposable SRAM design [1] incurs large area overhead or requires custom SRAM bitcell design.

To address this, we present a new cyclic weight storage and access scheme that allows using the same off-the-shelf 6T SRAMs for computations with non-transpose/transpose weights during FF/FB phases for both convolution and FC layers of convolutional neural networks (CNNs). Including this cyclic weight storage/access scheme, we implemented the overall CNN training accelerator in both 65nm CMOS ASIC and Intel Stratix-10 FPGA hardware.

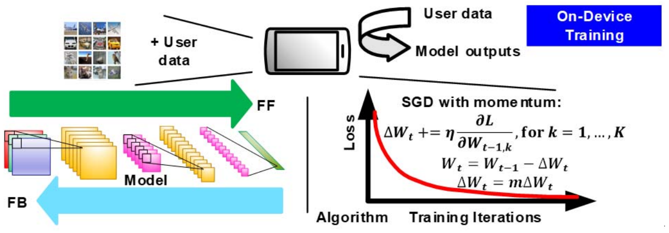
+ User data
User data
Model outputs
On-Device Training
FF
FB
Model
SGD with momentum:
$\Delta W_t += \eta \frac{\partial L}{\partial W_{t-1,k}}$, for $k = 1, \ldots, K$
$W_t = W_{t-1} - \Delta W_t$
$\Delta W_t = m \Delta W_t$
Loss
Algorithm
Training Iterations

Fig. 1. Illustration of DNN training on edge devices with local user data.

Our ASIC CNN training accelerator [2] integrates large on-chip SRAM (>1 MB) that eliminates off-chip weight memory requirement for on-device training of small/medium-size CNNs with up to 131k parameters. The 65-nm prototype chip achieves up to 2.6/0.5 TOPS/W at 0.55V/1.0V supply.

Our FPGA CNN training accelerator [3] is implemented on Intel Stratix-10 GX FPGA by optimized and parameterized custom Verilog modules, and the accelerator is flexible to support various FPGA design parameters. Including off-chip DRAM access and communication, the FPGA training accelerator achieves throughput of up to 479 GOPS and 9.5 GOPS/W at 240MHz for CNNs with 2M parameters.

II. CYCLIC WEIGHT STORAGE/ACCESS SCHEME FOR EFFICIENT DNN TRAINING

To eliminate expensive weight transpose during FB phases, we present a new on-/off-chip weight storage and access scheme that can be used directly with off-the-shelf 6T SRAM arrays from commercial memory compilers. The cyclic weight storage and access scheme is illustrated in Fig. 2, where convolution kernels and fully-connected weights are stored in the form of a circulant matrix using column buffers. Every row of kernel blocks is circularly rotated and stored in the form of a circulant matrix in the single-port column buffers (Fig. 2). In the non-transpose mode, each column buffer shares the same read address, and in transpose mode, each column buffer obtains shifted addresses from the address translator unit. Address

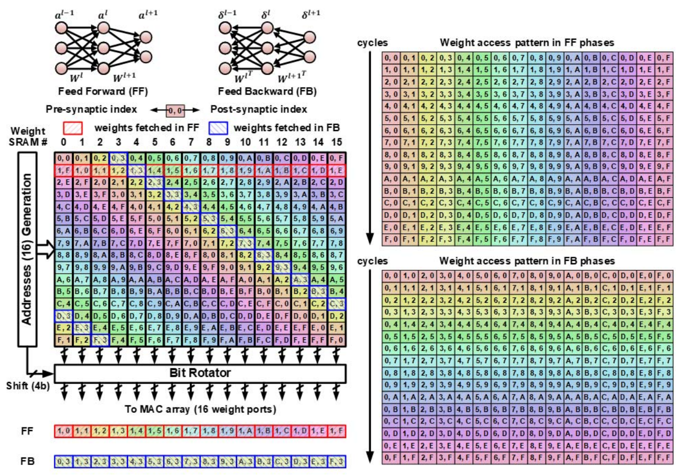

Fig. 2. Cyclic weight storage/access scheme enables using off-the-shelf SRAMs for non-transpose (FF phase) and transpose (FB phase) operations.

translator generates read/write addresses for column buffers for every transposable block. This way, weight data can be read both in non-transpose (for FF phase) and transpose modes (for FB phase) by just changing the read address. In each transposable block, the address vectors and the data are circularly shifted using shift registers.

Compared to the SRAM weight storage for DNN inference (FF only), the cyclic weight scheme supports both FF and FB for DNN training with a small 6.4% area overhead due to the bit rotator and SRAM array breakdown. Transposable SRAM bitcell designs [1] incur >100% overhead due to two additional transistors per SRAM bitcell and non-push rule design.

For the FPGA DNN training accelerator, we also store the weights in off-chip DRAM on the FPGA board in the cyclic transposable format. The transposable weights of each layer are read from DRAM to the old weight buffer. New weights are computed tile-by-tile and written back in transposable format to the new weight buffer. After completing the last tile's computation, the new weights are written back to DRAM.

III. ASIC AND FPGA TRAINING ACCELERATOR RESULTS

A. *ASIC DNN Training Accelerator*

The ASIC DNN training chip is implemented in 65nm CMOS [2] (Fig. 3), with which we trained CNNs for CIFAR-10 and MNIST datasets for 50 epochs. Measurements on training small/medium size CNNs for MNIST/CIFAR-10 datasets demonstrate on-par accuracies compared to the floating-point baselines by GPUs. At 1V/0.55V, the total chip power consumption is 299/10.45 mW (excluding DRAM access and communication) and 294/53 MHz clock frequency, achieving energy efficiency of 0.5/2.60 TOPS/W. The peak throughput at 1.0V is 151 GOPS.

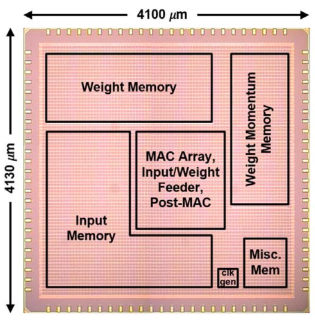

Fig. 3. ASIC chip micrograph.

Fig. 4 shows the MAC array utilization and overall latency breakdown for the case of training the '1X' CNN (16C3-16C3-P-32C3-32C3-P-64C3-64C3-P-FC) for CIFAR-10. The MAC array utilization in FF and FB phases is close to 100% and utilization in WG phases is 56.26%. The relatively low utilization in WG phases is because only 3×3 kernel window out of the 4×4 processing window (in output feature map) of the MAC array is activated.

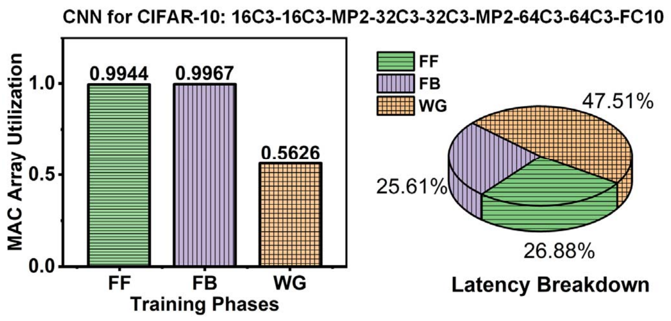

Fig. 4. MAC array utilization (left) and latency breakdown (right) among FF, FB, and WG phases are shown for 6-layer CNN training for CIFAR-10 dataset.

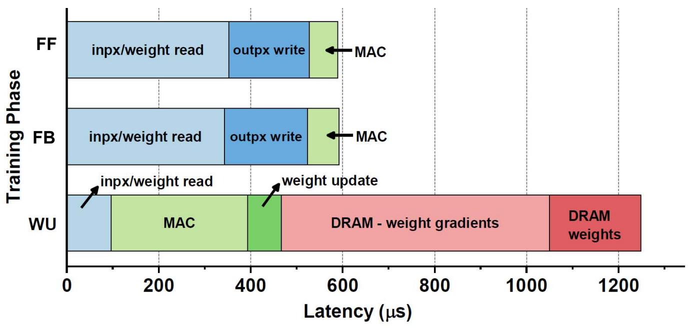

Fig. 4. Latency breakdown of CIFAR-10 4X CNN for FF, FB and WU for the last iteration of a batch.

B. *FPGA DNN training accelerator*

The FPGA DNN training accelerator was designed for Intel Stratix 10 GX FPGA. The development kit FPGA board is equipped with 4Gb DDR3 DRAM with 16.9Gb/s bandwidth. We trained the same '1X' CNN as the ASIC accelerator, as well as 2X and 4X CNNs with more input/output feature maps for each layer, aided by the larger on-chip memory and real-time off-chip DRAM communication. DRAM modules and Intel IPs were used in the testbench adhering to DRAM protocols. Fig. 4 shows the latency breakdown of the 4X CNN during FP/FB/WU phases of training. Weight update layers will have large DRAM access latency due to access of past weight gradients, weights and storing back the updated values. High-speed memory interface such as HBM can largely reduce the DRAM latency. Including the DRAM communication, the peak throughput of the FPGA training accelerator for 4X CNN is 479 GOPS, and estimated to be 718 GOPS without DRAM latency. This is ~5X higher than that of the ASIC design, but the FPGA design has 70-160X more power consumption than the ASIC design.

IV. CONCLUSION AND DISCUSSION

We presented programmable DNN training accelerator designs in ASIC and FPGA in 16-bit fixed-point precision. A dual-read-mode cyclic weight storage scheme is implemented to enable non-transpose/transpose-mode weight access without explicit transpose operations during training. The FPGA design achieves ~5X higher throughput, but the ASIC design achieves much higher energy-efficiency, because the FPGA design has many more compute/memory components and also includes the real-time DRAM communication.

REFERENCES

[1] J. Lee *et al.*, "LNPU: A 25.3TFLOPS/W Sparse Deep-Neural-Network Learning Processor with Fine-Grained Mixed Precision of FP8-FP16," *IEEE International Solid- State Circuits Conference (ISSCC)*, 2019.

[2] J. Seo *et al.*, "45nm CMOS neuromorphic chip with a scalable architecture for learning in networks of spiking neurons," *IEEE Custom Integr. Circuits Conf. (CICC)*, 2011.

[3] S. Yin and J. Seo, "A 2.6 TOPS/W 16-Bit Fixed-Point Convolutional Neural Network Learning Processor in 65-nm CMOS," *IEEE Solid-State Circuits Letters (SSC-L)*, vol. 3, no. 1, pp. 13-16, January 2020.

[4] S. K. Venkataramanaiah *et al.*, "Automatic Compiler Based FPGA Accelerator for CNN Training," *IEEE International Conference on Field-Programmable Logic and Applications (FPL)*, 2019.