A 0.23mW Heterogeneous Deep-Learning Processor Supporting Dynamic Execution of Conditional Neural Networks

Hsi-Shou Wu*, Zhengya Zhang*, Marios Papaefthymiou*†
*Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109
†Department of Computer Science, University of California, Irvine, CA 92697
Email: {hsiwu, zhengya}@umich.edu, marios@ics.uci.edu

***Abstract*—A deep-learning processor is presented for achieving ultra-low-power operation in mobile applications. Using a heterogeneous architecture that includes a low-power always-on front-end and a selectively-enabled high-performance back-end, the processor dynamically adjusts computational resources at runtime to support conditional execution in neural networks and meet performance targets with increased energy efficiency. Featuring a reconfigurable datapath and a memory architecture optimized for energy efficiency, the processor supports multi-level dynamic activation of neural network segments, performing object detection tasks with 5.3× lower energy consumption in comparison with a static baseline design. Fabricated in 40nm CMOS, the processor test-chip dissipates 0.23mW at 5.3 fps. It demonstrates energy scalability up to 28.6 TOPS/W and can be configured to run a variety of workloads, including severely-power-constrained ones such as always-on monitoring in mobile applications.**

I. Introduction

Deep neural networks (DNNs) are essential elements of Artificial Intelligence (AI) systems. With AI capabilities increasingly moving from data centers to embedded and mobile platforms, there is growing need for energy-efficient DNN hardware designs capable of supporting always-on operation while meeting stringent power constraints.

Various techniques have been proposed for reducing power consumption in DNN processors. Numerical quantization is a relatively simple and effective such approach. Early work in this area has focused on configurable hardware that supports variable bit precision to trade off accuracy for lower power consumption [1] [2]. BinaryNet further quantizes weights and activation to +1 or -1, simplifying multiplication to XNOR and leveraging mixed-signal circuits to yield highly efficient designs [3]. The work described in [4] leverages the sparsity of neural networks obtained by network pruning without sacrificing classification accuracy. Zero-valued weights and activations are skipped, reducing workload and data bandwidth to increase energy efficiency.

Recently, researchers have proposed a new class of neural networks that can be dynamically adjusted during inference to adapt to input data. Specifically, [5] introduces a conditional classifier on the feedforward path to obtain classification early without completing execution of the entire network. The work in [6] proposes a methodology for augmenting neural networks using control edges to determine inference paths dynamically. In principle, conditional execution can be an effective approach for increasing power efficiency of DNN computations above and beyond what can be achieved by other techniques such as quantization and pruning. In practice, however, power savings may be difficult to realize, or they may come at unacceptable performance overheads. Specifically, conditional branches introduce additional latency to the computation. Moreover, conditional classifiers may yield irregular memory access patterns, resulting in energy-inefficient memory operation. Processor architectures for conditionally-executing DNNs have thus remained elusive.

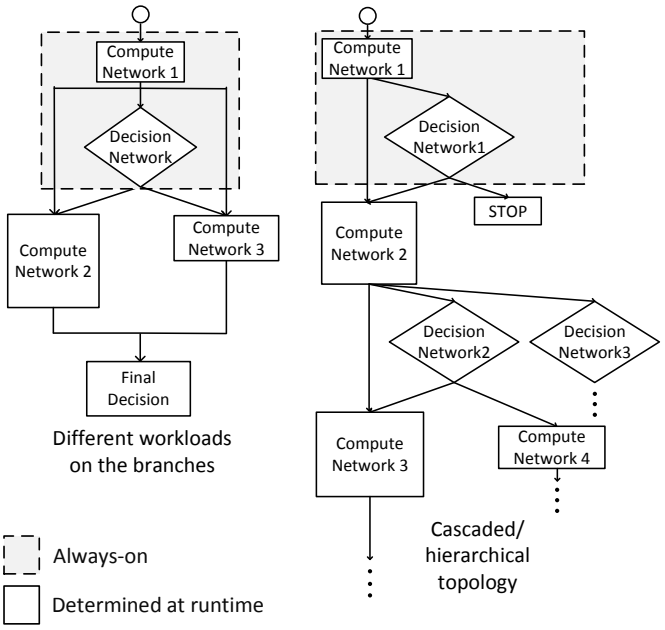

Fig. 1: Conditional Deep Neural Network.

This paper presents a heterogeneous processor that relies on an energy-efficient front-end and a high-performance back-end to support conditionally-executing DNNs with high performance and low power consumption through voltage and frequency scaling. A key feature of this processor is a *dynamic execution unit* that can be reconfigured dynamically at runtime to optimize performance and power efficiency. A real-time controller dispatches jobs to the execution unit during runtime reconfiguration, reducing conditional branch overheads.

978-1-5386-5404-0/18/$31.00 ©2018 IEEE

The data memory architecture supports burst-mode access and fine-grain gating to maximize energy efficiency under irregular access patterns. Fabricated in a 40nm CMOS process, the test-chip achieves 5.3× higher energy efficiency than a baseline static design, with an average power of 0.23mW at 5.3fps for the LFW face recognition dataset and maximum achievable efficiency of 28.6 TOPS/W.

II. Heterogeneous Architecture

A high-level view of conditional DNNs is given in Fig. 1. These networks are derived by augmenting conventional neural networks, called *compute nets*, with small neural nets, called *decision nets*, which adjust the operation of the original DNN. Decision nets extract additional features from intermediate layer outputs and decide whether to proceed with the inference task using the entire compute nets, branch to a small compute subnet or terminate.

The main characteristic of conditional DNNs is that initially a few header compute nets and decision nets perform course classification, pruning unnecessary paths in the early stages. The remaining layers then perform detailed fine-grain feature extraction for final classification. The header compute nets and decision nets determine the high-level run-time pattern of the DNN. Although they are relatively light-weight in terms of complexity, they must be implemented with high efficiency, as they are always active.

The processor architecture for conditional DNNs is shown in Fig. 2. The header compute nets and decision nets, both of which consist of convolution (Conv) or fully-connected (FC) layers, are assigned to a low-power front-end. A high-performance back-end is used only for processing fine-grain computationally demanding tasks. By running the front-end at low voltage and heavily duty-cycling the operation of the back-end, the processor is capable of operating with high energy efficiency.

The front-end unit is an always-on subsystem, continuously monitoring the processors input. It is equipped with a 3×8 array of single-instruction multiple-data (SIMD) multiply-accumulate (MAC) units for RGB raw image input processing and Conv and FC computations. A 34KB static memory is used for storing/framing continuous input data, weights, and temporary features. The front-end automatically adjusts the workload of the back-end at runtime. For instance, in an object recognition scenario, if the target object is absent from view, the front-end executes light workloads and then idles. When an object of interest appears, it wakes up the back-end to perform the more computationally intensive tasks for correct recognition results.

The back-end is a high-performance unit for accelerating compute nets and decision nets. Instead of making dedicated compute net accelerators and decision net accelerators, which would result in hardware underutilization due to the dynamic nature of the conditional DNNs, we design the back-end using Conv cores and FC cores that can be configured to implement compute nets, decision nets, or a mix of both. These cores comprise a finer-grain dynamic execution unit (DEU) that is

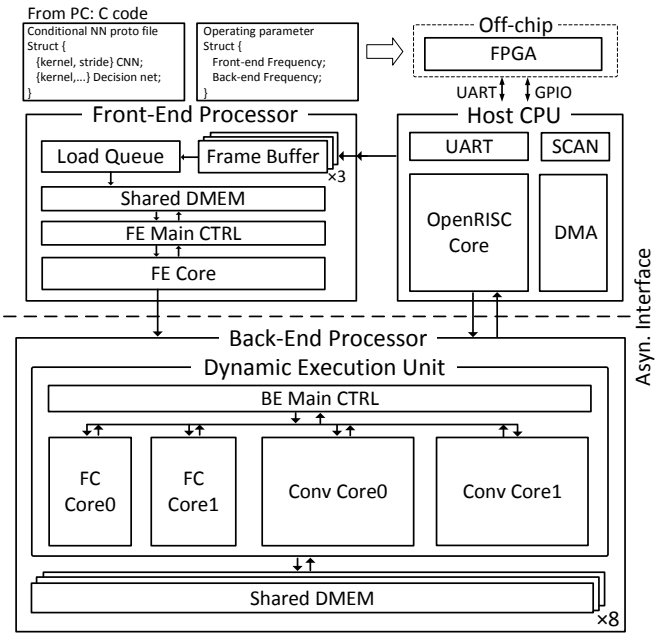

Fig. 2: Heterogeneous processor architecture.

designed to ensure full hardware utilization while providing the execution flexibility for gaining performance or reducing power by taking advantage of timing slacks. The back-end includes two Conv cores, each with a 16×16 SIMD MAC array, and two FC cores, each with a 4×13 SIMD MAC array. A 240KB static memory is used to store features from the front-end, convolutional kernels, FC layer weights, and temporary features results. Once the back-end wakes up, it executes the fine-grain subtree inference shown in Fig. 1. The configurable dynamic execution units are therefore designed to efficiently execute compute and decision nets at the same time, supporting a variety of object search modes with minimum energy cost.

To meet application latency requirements (1s in surveillance systems, 30ms for real-time applications), the front-end and the back-end run at or above 10MHz and 150 MHz, respectively. An OpenRISC processor serves as the host CPU to control the operation of the two cores during runtime, sending compiled network configurations and instructions to the two cores, and receiving data from them. It also controls the direct memory access (DMA) for on-chip data movement. At external host processor (FPGA) is used to program the OpenRISC processor through a UART interface and move image data from an off-chip DRAM onto the chip.

III. Dynamic Execution Unit

A key feature of the proposed processor is the DEU in the back-end, shown in Fig. 3. The DEU is designed to support a variety of workloads to meet dynamic inference requirements. The conditional DNN configurations are stored in the CPU, which maps them on the DEU part-by-part in runtime. The CPU monitors the decision net outputs and dynamically maps the next part on the DEU by selecting an appropriate number

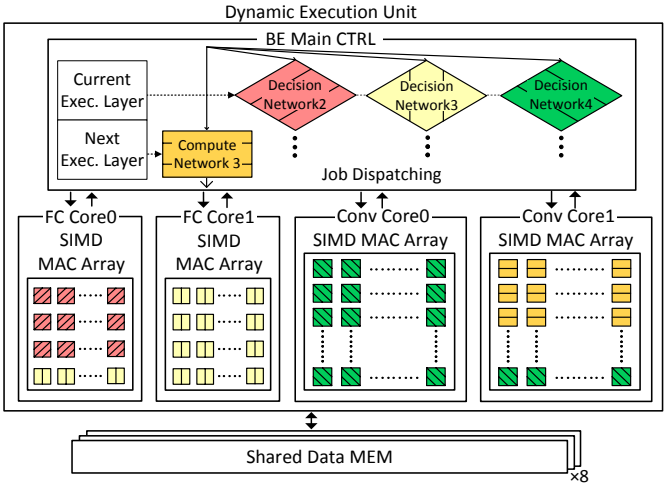

Fig. 3: Dynamic Execution Unit (DEU) in Back-End.

of Conv cores and FC cores, configuring the cores, and setting the memory address space needed by the cores.

There is flexibility in configuring the DEU. If a higher performance is desired, decision nets and the next compute nets can be both mapped onto the DEU. The next compute nets are speculatively executed without waiting for the decision nets' output, allowing the possibility of a lower latency and faster processing speed. The lower processing latency can also be exploited to lower the supply voltage for reducing power consumption. When multiple decision nets and/or compute nets need to be executed, they can be mapped on the DEU in a pipelined fashion: when one net reaches the FC core, the Conv core is freed up to accommodate the next net. More importantly, the DEU makes it possible to perform load balancing. While an FC core is busy, the Conv cores can speculatively execute the next compute net or decision net. Load balancing also ensures the full utilization of the hardware to reach the highest possible efficiency.

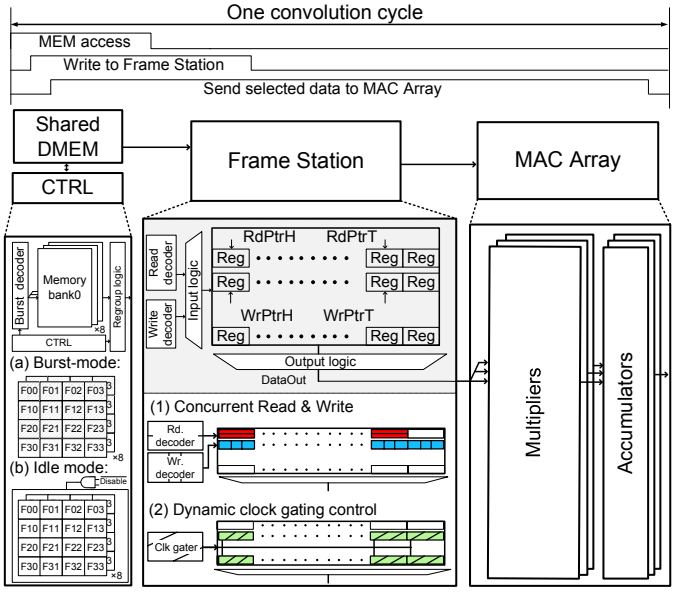

Fig. 4: Burst-mode memory access and frame station.

IV. Memory Architecture

To support irregular data access patterns during decision net execution, the back-end relies on a shared local data memory with 8 banks and 240b-wide I/O for fast data sharing. A custom-designed register file, called *frame station*, stores the memory data, enabling efficient data reuse. Due to their spatial locality, Conv/FC computations can be performed with notably lower memory bandwidth by reusing data in the frame station. Each Conv/FC core has a dedicated 1.5KB frame station tailored to accommodate the data transfer pattern of that core.

The data memory and the frame station support burst-mode operation, as shown in Fig. 4 for a Conv core. The data used in several consecutive cycles is fetched from the data memory in a single access, and the data memory is duty-cycled to save power. Compared with a 64b-wide I/O memory with no burst mode, burst-mode access reduces power by 29%, reducing the overall power consumption of the memory cell, word line, and sense amplifier during memory access, and saving power through duty-cycled operation.

Unlike a typical FIFO, which keeps data moving to feed the compute pipelines, the frame station keeps data stationary to eliminate data movement costs. Read/write head/tail pointers are used to keep track of the range of data currently being referenced and processed. If zero operands are detected, the corresponding MACs are skipped. The registers outside the selected range are clock-gated at bit-level to further save power. By keeping data stationary and activating only a portion of its access logic, the frame station reduces dynamic power by 54% compared with a FIFO implementation in simulation. The frame station reduces memory accesses by 67% with the smallest convolutional window (3×3, stride=1). Reductions are even higher when kernel size and overlap increase.

V. Experimental Results

The chip has been fabricated in a 40nm CMOS 1P10M technology. Fig. 5 shows measured power across a range of operating clock frequencies. Front-end power is as low as 0.19mW at 0.53V, 11MHz. The back-end achieves a peak clock rate of 500MHz at 0.9V, consuming 114mW.

To assess energy savings achieved by dynamic execution, we obtain the baseline power of the test-chip's back-end with the DEU disabled. We also train conditional DNNs using

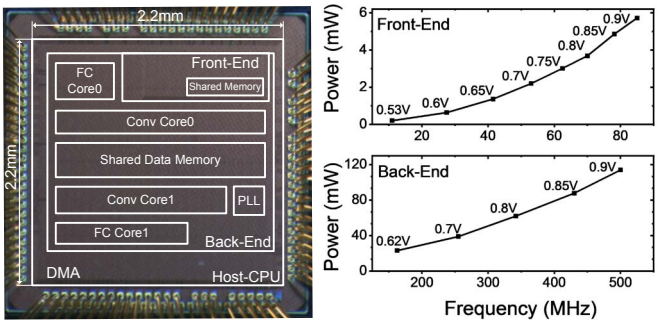

Fig. 5: Chip micrograph and measured power consumption versus operating frequency.

reinforcement learning with different rewards, thus obtaining different sets of weights that yield different decision net outcomes and trade off energy consumption for recognition accuracy. The graphs in Fig. 6 show efficiency results of the test-chip on the LFW face dataset and the CIFAR-10 dataset for face and object recognition. For the LFW data in Fig. 6(a), energy savings scale with relatively low loss in accuracy. Energy consumption decreases by 2.7× with 97.2% accuracy rate (1.4% loss) when mapping all DNNs to the back-end with the DEU enabled for dynamic execution. Energy consumption decreases by 5.3× with the front- and back-end both activated. Using different sets of weights, the chip operates at various power/accuracy points. Energy consumption is reduced by 7.1× when accuracy rate decreases to 91.5%.

Fig. 6(b) shows the results with the CIFAR-10 dataset. In this experiment, we trained and tested a conditional DNN using ResNet as the baseline to show applicability to advanced benchmarks. Energy savings are 2.5× with 91.3% accuracy, and 4.9× with 86% accuracy using dynamic execution.

Table I compares the test-chip with previously published designs. Our test-chip consumes an average of 0.23mW at 5.3fps on the LFW dataset. Using dynamic execution, it achieves 2.49 TOPS and demonstrates a competitive energy scalability of 4.7-28.6 TOPS/W. By comparison, with the same dataset and accuracy, the processor with analog-digital front-end face detector and digital back-end CNN face recognition in [7] consumes 0.62mW on average at 1fps. Moreover, the digital front-end in our chip provides support for different applications and energy scalability under various conditions. Compared with [1] [2], which combine voltage-frequency scaling and quantization with optimized configurable hardware, our test-chip achieves higher energy efficiency without sacrificing bit precision.

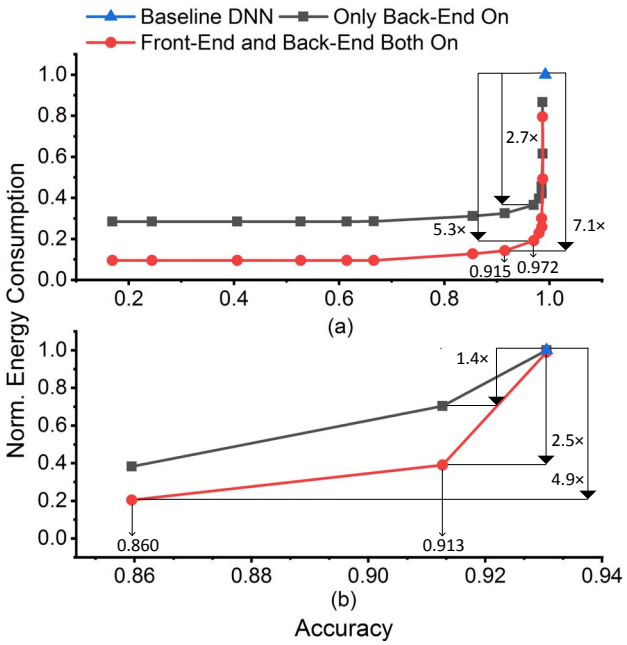

Fig. 6: Normalized energy consumption versus accuracy for (a) LFW and (b) CIFAR-10 data sets.

TABLE I. PERFORMANCE SUMMARY AND COMPARISON

	This Work		ISSCC'17 [1]	ISSCC'17 [7]	ISSCC'18 [2]
	Front-End	Back-End			
Technique	HPDE[1]		DVAFS	HHFD(FD) + DM-CNNP/ SF-CONV (FR)	UNPU
Technology	40nm		28nm	65nm	65nm
Die Area	4.84mm²		1.87mm²	27mm²	16mm²
SRAM Size	34KB	240KB	144KB	N/A	256KB
Max Frequency	85MHz	500MHz	200MHz	100MHz	200MHz
Power	0.23mW		7.5mW	0.62mW	3.2mW
Peak Performance (TOPS)	2.49 (8b)		0.408 (4b)	0.072 (16b)	7.372 (1b)
Efficiency (TOPS/W)	4.7-28.6 (8b)		0.53 (16b) 10 (4b)	2.1 (16b)	3.08 (16b) 11.6 (4b) 50.6 (1b)

[1]Heterogeneous Processor with Dynamic Execution

VI. CONCLUSION

This paper presents a heterogeneous low-power processor that combines a low-power front-end with a high-performance back-end to support conditionally-executing DNNs. The proposed architecture is compatible with previous approaches to DNN processor design, such as bit-precision and sparsity optimization, enabling additional energy savings above and beyond what is achievable by those approaches. The back-end can be configured in runtime to support the most efficient execution of dynamic DNNs through speculative execution of compute nets and overlapped execution of compute nets and decision nets. On the LFW dataset, the chip achieves 5.3× lower energy with 0.23mW average power and 1.4% accuracy loss. With energy efficiency in the 4.7-28.6 TOPS/W range, it can adjust its operation to support edge devices under various operating conditions.

ACKNOWLEDGMENT

This research was supported in part by NSF under Grant No. IIS-1539011 and gifts from Intel and Broadcom Foundation.

REFERENCES

[1] B. Moons *et al.*, "14.5 Envision: A 0.26-to-10TOPS/W subword-parallel dynamic-voltage-accuracy-frequency-scalable Convolutional Neural Network processor in 28nm FDSOI," *ISSCC*, pp. 246–247, Feb 2017.

[2] J. Lee *et al.*, "UNPU: A 50.6TOPS/W unified deep neural network accelerator with 1b-to-16b fully-variable weight bit-precision," *ISSCC*, pp. 218–220, Feb 2018.

[3] D. Bankman *et al.*, "An always-on 3.8μJ/86% CIFAR-10 mixed-signal binary CNN processor with all memory on chip in 28nm CMOS," *ISSCC*, pp. 222–224, Feb 2018.

[4] A. Parashar *et al.*, "SCNN: An accelerator for compressed-sparse convolutional neural networks," *ISCA*, pp. 27–40, June 2017.

[5] S. Teerapittayanon *et al.*, "BranchyNet: Fast inference via early exiting from deep neural networks," *ICPR*, pp. 2464–2469, Dec 2016.

[6] L. Liu and J. Deng, "Dynamic Deep Neural Networks: Optimizing Accuracy-Efficiency Trade-offs by Selective Execution," *AAAI*, 2018.

[7] K. Bong *et al.*, "14.6 A 0.62mW ultra-low-power convolutional-neural-network face-recognition processor and a CIS integrated with always-on haar-like face detector," *ISSCC*, pp. 248–249, Feb 2017.