SCALENet: A SCalable Low power AccELerator for Real-time Embedded Deep Neural Networks

Colin Shea
University of Maryland, Baltimore County
Baltimore, MD
cshea3@umbc.edu

Adam Page
University of Maryland, Baltimore County
Baltimore, MD
apage2@umbc.edu

Tinoosh Mohsenin
University of Maryland, Baltimore County
Baltimore, MD
tinoosh@umbc.edu

ABSTRACT

As deep learning networks mature and improve classification performance, a significant challenge is their deployment in embedded settings. Modern network typologies, such as convolutional neural networks, can be very deep and impose considerable complexity that is often not feasible in resource bound, real-time systems. Processing of these networks requires high levels of parallelization, maximizing data throughput, and support for different network types, while minimizing power and resource consumption. In response to these requirements, in this paper, we present a low power FPGA based neural network accelerator named SCALENet: a SCalable Low power AccELerator for real-time deep neural Networks. Key features include optimization for power with coarse and fine grain scheduler, implementation flexibility with hardware only or hardware/software co-design, and acceleration for both fully connected and convolutional layers. The experimental results evaluate SCALENet against two different neural network applications: image processing, and biomedical seizure detection. The image processing networks, implemented on SCALENet, trained on the CIFAR-10 and ImageNet datasets with eight different networks, are implemented on an Arty A7 and Zedboard™ FPGA platforms. The highest improvement came with the Inception network on an ImageNet dataset with a throughput of 22x and decrease in energy consumption of 13x compared to the ARM processor implementation. We then implement SCALENet for time series EEG seizure detection using both a Direct Convolution and FFT Convolution method to show its design versatility with a 99.7% reduction in execution time and a 97.9% improvement in energy consumption compared to the ARM. Finally, we demonstrate the ability to achieve parity with or exceed the energy efficiency of NVIDIA GPUs when evaluated against Jetson TK1 with embedded GPU System on Chip (SoC) and with a 4x power savings in a power envelope of 2.07 Watts.

CCS CONCEPTS

• **Computing methodologies** → *Neural networks*; • **Computer systems organization** → *Embedded hardware*; *Real-time system architecture*; • **Hardware** → *Hardware-software codesign*;

GLSVLSI'18, May 23–25, 2018, Chicago, IL, USA

ACM ISBN 978-1-4503-5724-1/18/05…$15.00
https://doi.org/10.1145/3194554.3194601

KEYWORDS

Low power;programmable hardware accelerator; convolutional neural network; machine learning; FPGA; scalable; embedded systems.

ACM Reference Format:
Colin Shea, Adam Page, and Tinoosh Mohsenin. 2018. SCALENet: A SCalable Low power AccELerator for Real-time Embedded Deep Neural Networks. In *Proceedings of Great Lakes Symposium on VLSI 2018 (GLSVLSI'18).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3194554.3194601

1 INTRODUCTION

Convolutional neural networks (CNN) have become the baseline implementation for computer vision and related fields with the ability to obtain an order of magnitude improvement over conventional techniques. As this area of deep learning matures, the challenge to efficiently deploy such networks in embedded settings for a variety of applications. Deployment requires an efficient network architecture implementation that meets the accuracy, power, and latency requirement of a targeted application. Graphics processing units (GPUs), predominately by NVIDIA, have shown the ability to significantly boost efficiency, especially during training, by exploiting the underlying data-level and task-level parallelism that exists within neural networks (NN). While massively outperforming their CPU counterparts, GPUs, including mobile GPUs, have a power envelop that is still too high for a large variety of low-power embedded settings. Current literature has started focusing on developing custom FPGA and ASIC accelerators suited for efficiently deploying pre-trained convolutional neural networks in resource-limited, real-time embedded systems. This work focuses on an architecture that addresses the challenges and characteristics of deploying a variety of Deep Neural Networks (DNN) in the embedded space. Here we propose SCALENet: a SCalable Low power AccELerator for real-time deep neural Networks. The main contributions of this paper include:

- Optimizing for power with coarse and fine grain scheduling of hardware configuration at compilation and runtime.
- Dedicated Hardware Acceleration of Fully Connected (FC) and two different methods, Direct and FFT, to Convolutional Neural Network.
- Implementation flexibility and scalability with hardware or hardware/software co-design.
- Testing SCALENet with a variety of applications.

2 RELATED WORK

In recent years, some publications on CNN accelerators for low-power embedded settings have been proposed. [4] describes a 168 16-bit fixed point processing element (PE) array ASIC with a configurable filter size and filter count. This work provides a series

of optimizations for memory compression, data reuse, and clock gating. For data reuse, the authors approach the problem set with a spatial architecture for parallelization optimization where a filter row and an image row are mapped to a PE. [10] offers a Xilinx High-Level Synthesis defined Deep Neural Network (DNN) accelerator for fully connected layers. The accelerator is based on parameter encoding of weights and a reduction of the number of multipliers. The design targets 32-bit floating point operations and a unique encoding scheme by finding the mean of the positive and negative weights and representing them with a 1-bit substitution and storing them off. In [11] an energy-efficient CNN processor is proposed with an optimized CNN neuron processing engine, a dual-range multiply-accumulate block for low-power convolution operations and an on-chip memory architecture with the scheme to reduce off-chip memory accesses. The chip is fabricated in 65nm 1P8M CMOS technology. In [16], an analytical design scheme using a roofline model is proposed for CNNs. In their work, CNN's computation and memory access are optimized. Then, all possible designs in the roofline model are compared, and the best design for each layer is determined. The CNN accelerator is implemented on a Xilinx Virtex-7 (VX485T) FPGA using Vivado HLS. The authors in [13] also highlight the data-level and task-level parallelism that exists within CNN and how best to exploit the attributes. The authors of [14] evaluate different methodologies for detection of epileptic seizures in EEG signals. These methods include traditional variance-based detection and nonlinear time series analysis such as logistic regression and time-frequency distributions. Their conclusion was that time variance-based analysis, with a fixed threshold, still had the highest overall accuracy. While the work performed in [3], developed a 13-layer deep CNN algorithm to detect normal, preictal, and seizure classes. The customized CNN contained five convolution and max-pooling layers and three fully connected layers. Their results had an accuracy of 88.67%. Most of these previous works have focused on HLS for custom single purpose designs or creating custom application specific integrated circuits (ASIC) with specialized accelerator cores targeted for particular fixed network types. These previous works do not address power, area and the ability to efficiently process different network types without requiring a complete redesign.

3 PROPOSED HARDWARE ACCELERATOR

The high-level diagram of the SCALENet accelerator is depicted in Figure 1. SCALENet's design, as either an all hardware accelerator, or a hardware/software co-design implementation, uses layers as the basic building blocks of neural networks. These layers contain the bulk of the complexity, and this accelerator is designed to operate on a supplied layer configurations. Part of the design, a host application or the accelerator engine, decodes the run-time configuration and is tasked with serializing and dispatching a predefined network topology layer by layer.

Deployment of deep neural networks requires the accelerator to achieve satisfactory throughput while being efficient as possible regarding power and area. To do so, it requires an innovative design that exploits critical requirements of deploying a network architecture including network flexibility, resource consumption, and power minimization. In SCALENet, these design aspects are mapped to

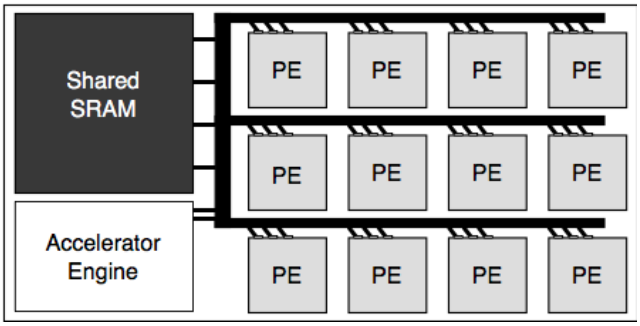

Figure 1: The high-level block diagram of the SCALENet accelerator. The processing engines (PEs) are arranged in a grid configuration with a unified network interface. The network provides two sets of buses that are unidirectional and contain two 16-bit channels. The accelerator engine is used to configure the PEs and perform all of the data marshaling. A shared on-chip memory is used to cache the current layers feature channels along with layer parameters.

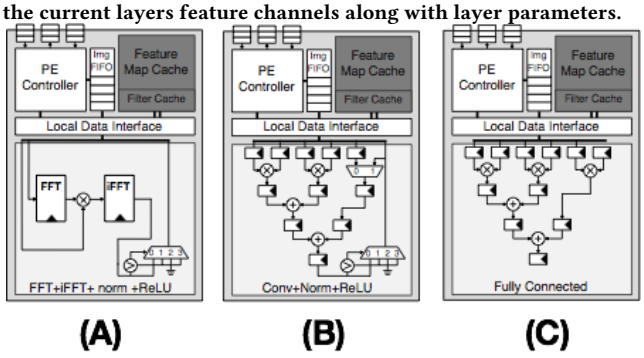

Figure 2: The PEs are instantiated in one of 3 forms, Fast Fourier Transform (FFT) Convolution (A), Direct Convolution (B), or Fully Connected (C), dependent on the coarse scheduler at compilation. In each PE there is a memory dedicated to filter and image caching, a clock enable, and a simple controller for dispatching configuration.

two main architectural choices in the design: the scheduler, and a customizable design feature set.

3.1 Scheduler

The SCALENet scheduler is a two-part algorithm that balances latency, throughput and power consumption through both course grain and fine grain attributes of the system. The schedulers goal is to optimize for latency based on available hardware and software models. The algorithm inputs are the desired throughput, the neural network model to use, the target Xilinx FPGA, and the data precision. The coarse portion of the scheduling algorithm takes the specified model and examines the layers and provides the configuration with the lowest latency for the FPGA, and, if proper, software layer execution. The configuration output informs the second stage with the optimal number of PEs, data memory offsets for implementation, the size of scratch memory, and the optimal number of active PEs per layer.

3.2 Customizable Design

SCALENet, written in C++ within Xilinx VIVADO HLS, allows for customizing each implemented neural network definition. The design enables block-level building and an iterative data path to accommodate different sized datasets without re-compilation. There

are two sets of customization: at compile time and runtime as optimized by the scheduler.

The parameters that are configurable only at compile time are: the maximum number of PEs, the size of a PEs local scratchpad memory, the data precision, and the PE type. The maximum number of PEs is limited by the available FPGA resources, which is a variable of the FPGA family. The design template can be configured to set the scratchpad memory per PE. The memory amount can be increased or decreased depending on the number of PEs and the layer type, CNN or FC.

3.2.1 Flexible Parallelism. In deep neural networks, FC and CNN layers dominate the complexity typically accounting for more than 90% of the computation and memory requirements. Therefore, being capable of supporting and deploying these layers in an efficient methodology is required. SCALENet supports PEs that accelerate both types of layers. For convolution, SCALENet utilizes the *output channel tiling* form of parallelism based on tiling methods [16]. Again, focusing on the convolutional PEs SCALENet supports two different types, direct convolution, and FFT Convolution [1] [2]. To meet latency requirements, the coarse scheduler will choose either a direct or FFT convolution solution. When deploying an all hardware design with direct convolution the scheduler may insert at least one FC PE to accelerate any FC layers as necessary.

3.2.2 Data Precision. The accelerator's design can utilize either fixed-point or floating-point format and varying levels of data precision. The option of a fix-point or floating-point data operation is set at compilation time. The floating-point form has the advantage of supporting dynamic range, underflow/overflow conditions, and a non-uniform scale. This is beneficial for neural networks as layers tend to differ in their dynamic range and filter weights usually follow a normal distribution in which higher precision is needed for smaller values. The downside is that floating-point requires more logic than an equivalent fixed-point implementation. Furthermore, the use of an exponent to support higher dynamic range requires dedicating a subset of bits for the exponent, which leaves fewer bits for the mantissa and hence precision. Figure 3 provides a visual break down of a single PE's resources as a function of the data precision.

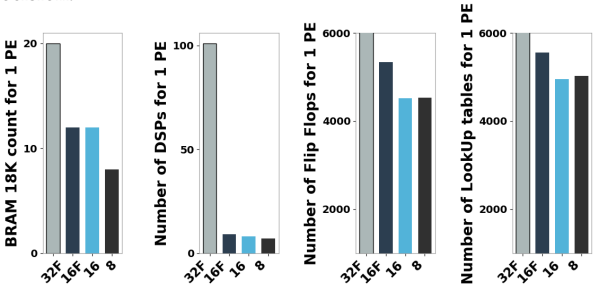

Figure 3: Comparison of floating-point and fixed-point for 32-bit, 16-bit, and 8-bit fixed-point representations regarding required resources for the layer operations associated with direct convolution, max-pooling, and batch normalization. While fixed-point requires fewer resources than the corresponding floating-point, the difference for lower precision, 16-bit Int vs. 16-bit FP becomes minimal.

3.2.3 Clock Gating. Xilinx Vivado HLS does not natively expose user control of the clock enable function for the generated HDL. In SCALENet we explicitly enable the *-clock-enable* option in HLS. This option maps all the clock enables for all clocked resources to be driven by a single top-level enable. SCALENet remaps the generated top level clock enable in HDL to a configurable masking value set by the Acceleration Engine (AE) for each of the PEs. By remapping the control signal, it enables the runtime design to be controlled, by the fine grain scheduler, to turn on and off unused PEs during execution. The clock disablement minimizes power consumption at the expense of latency. With the Xilinx FPGAs, this power savings can range from 1% to 20% [15].

4 CASE STUDIES

4.1 SCALENet Evaluation

A principal aspect of the proposed accelerator to be evaluated is the impact of processing engines on classification throughput, energy, and area utilization. For section 4.2 SCALENet is deployed as a hardware/software co-design with only the convolution, relu, and activation in hardware. In section 4.3 SCALENet is deploy as a hardware only implementation. Figure 4 demonstrates the results of PEs for the four networks trained on the CIFAR dataset. In particular, we see that for a given network increasing the amount of PEs can improve both throughput and energy consumption. Once the throughput begins to saturate, the energy consumption starts negatively impacting the efficiency of the design. Figure 5 highlights the optimal efficiency occurring as a ratio of classification throughput and power consumption (img/sec/watt).

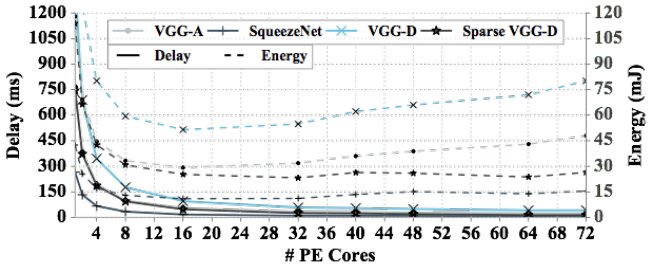

Figure 4: Impact of # of PEs on classification delay and energy consumption when evaluated on four networks: VGG-A, SqueezeNet, VGG-D, and Sparse VGG-D using CIFAR-10.

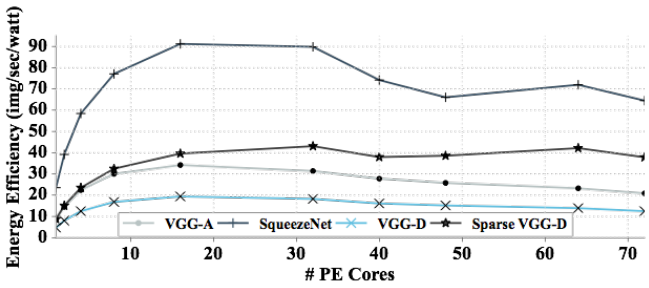

Figure 5: Comparison of energy efficiency as a function of # PEs for networks: VGG-A, SqueezeNet, VGG-D, and Sparse VGG-D. The introduction of subnetworks in SqueezeNet and Sparse VGG-D due to FCP enables obtaining increased efficiency on the ImageNet dataset.

The following two case studies, image classification with CIFAR-10 and ImageNet and seizure detection using EEG data, are used to evaluate SCALENet's use in efficiently deploying convolutional neural networks to resource-bound, real-time embedded systems.

4.2 Image Processing

4.2.1 CIFAR-10 Dataset. The CIFAR dataset consists of 60,000 32x32 color images with ten classes. While the input images are small, achieving good accuracy still requires having reasonably deep neural networks. For this dataset, we target four neural networks including VGG-A, VGG-D, modified SqueezeNet, and Sparse VGG-D [7, 12]. The standard SqueezeNet is designed for the ImageNet dataset with images of size 224x224. We modified the SqueezeNet network for CIFAR-10 by removing a set of convolutional layers and reducing the final pooling layer to cover 4x4 receptive field versus 14x14. Sparse VGG-D is the same network as VGG-D but with three generalized sparsification techniques applied [8]. Figure 6 provides a comparison of the four networks regarding computation, memory, and top-1 test accuracy. Both SqueezeNet and Sparse VGG-D require significantly less computation than VGG-A and VGG-D, relatively, while achieving similar accuracy performance. Sparse VGG-D can reduce computation and memory by 60%, and 93%, respectively. For this dataset, the CPU is a MicroBlaze (μB) on both the Arty A7 FPGA for all networks, but the SqueezeNet is implemented on the Cmod A7.

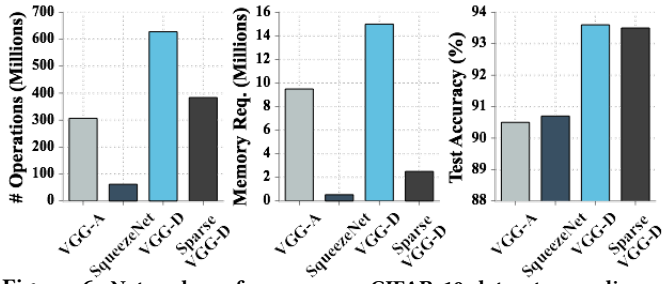

Figure 6: Network performance on CIFAR-10 dataset regarding computation, memory, and top-1 test accuracy. The networks are VGG-A, modified SqueezeNet, VGG-D, and Sparse VGG-D.

4.2.2 ImageNet Dataset. The ImageNet dataset consists of 10 million colored images scaled to 224x224 with 1,000 assigned classes. The specific dataset used comes from the 2012 Large Scale Visual Recognition Challenge (ILSVRC). ILSVRC is a much more difficult dataset and requires a higher complexity network to achieve good accuracy performance. For this dataset, we target four neural networks including AlexNet, Sparse AlexNet, SqueezeNet, and Inception v3. Sparse AlexNet is similar to AlexNet with three sparsification techniques applied in addition to filter factorization and replacing the FC layers with two 3x3 convolutional layers and a spatial pooling layer [8]. Sparse AlexNet, SqueezeNet, and Inception all require substantially less memory than AlexNet by replacing FC layers with convolutional layers. SqueezeNet and Sparse AlexNet, in particular, need 49% and 21% less computation than AlexNet, respectively. Inception requires 635% more computation than AlexNet but can dramatically increase top-5 accuracy by 14%. All datasets utilize the Zedboard's ARM CPU and FPGA fabric.

4.3 Seizure Detection

Using our previous work in time series seizure detection as a basis for EEG detection [9], our architecture is a reduced ResNet-20 with 13 convolution layers and replacement of the last two convolution layers with FC layers. The choice of ResNet is due to our experience in its performance in previous research and its low memory

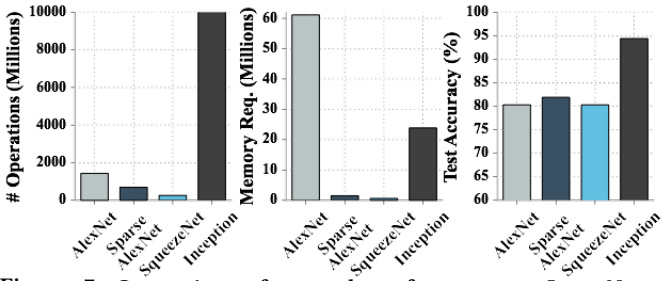

Figure 7: Comparison of network performance on ImageNet dataset in terms of computation, memory, and top-5 test accuracy. The networks include AlexNet, Sparse AlexNet, SqueezeNet, and Inception v3.

requirements for parameters. The input matrix is a fusion of 24 EEG channels from [6] dataset. Here time series samples, 24 consecutive from each channel, are fused to form a 24x24 matrix, and a 3x3 filter is applied.

Each set of fused channels is defined through a sliding window that moves from 1 sample to 24 samples per window, for 24 channels. At the start and end, when there is not enough data to fill the window, samples are set to 0 and shifted in or out. Each matrix of samples passes through the modified network as an image. The constructed model had an accuracy of 86.98%, which is similar in accuracy to [3]. We implemented the ResNet network using both Direct and FFT convolutional PE using SCALENet. The inputs to the SCALENet scheduler where: the modified ResNet model, the latency, the 1 sample every 3-milliseconds (ms) for a 256Hz EEG, a Zynq 7020, and 16-bit FP. The scheduler guided a design for a pure hardware implementation of 32 PEs for the Direct Convolution and 12 FFT Convolution PEs for the FFT based solution and in both cases 2 FC PEs. Due to the latency requirement of 3ms, the scheduler was unable to find a hardware/software co-design that could not meet the deadline of 3ms. Based on these design constraints the resulting solutions produced a decision every 3.05 milliseconds and 2.01 milliseconds, direct and FFT respectively.

5 COMPARISON WITH COTS

5.1 Targeted Platforms and Implementation

To understand the performance of the proposed accelerator, we evaluated SCALENet when running on three FPGA-based SoC platforms for all image processing and seizure detection network architectures. The FPGA platforms are the Zedboard with a Zynq-7000 SoC with dual-core ARM Cortex-A9 and FPGA, the Arty & Cmod A7 containing Artix-7 35T FPGA with Microblaze™ (μB) soft-processors. For each platform, the corresponding CPU is used as a baseline reference. Furthermore, we alternatively target NVIDIAs low-power Jetson TK1 platform containing TK1 SoC with quad-core ARM Cortex-A15 and K1 GPU. Torch, a scientific computing framework with comprehensive support for deep learning algorithms, is used to implement the networks on each platform. While the framework is built on the Lua scripting language, the underlying libraries are developed in C/C++ with support for acceleration on both CPUs and GPUs by utilizing OpenBLAS compiled for the architecture and the necessary CUDA libraries. Using the same framework also enables ensuring consistency across platforms. To use the SCALENet FPGA-based accelerator, a modified version of *SpatialConvolution.lua* is used to call the appropriate accelerator

Database	Network			Host ARM/μB		ARM/μB + SCALENet		Improvement	
	Architecture	Memory (MB)	(GFLOP)	Execution (sec)	Energy (Joules)	Exec (sec)	Energy (Joules)	Exec (%)	Energy (%)
CIFAR	VGG-A	36.22	0.31	0.90	0.85	0.19	0.25	**78.71%**	**70.20%**
	SqueezeNet	2.04	0.06	0.25	0.25	0.05	0.07	**80.14%**	**73.48%**
	VGG-D	57.19	0.63	1.80	1.73	0.30	0.49	**83.48%**	**76.58%**
	Sparse VGG-D	9.56	0.38	0.26	0.26	0.05	0.06	**80.79%**	**75.04%**
ImageNet	AlexNet	233.15	0.80	3.32	5.54	0.39	1.11	**88.28%**	**80.00%**
	Sparse AlexNet	5.76	0.63	0.95	1.61	0.08	0.21	**92.05%**	**86.99%**
	SqueezeNet	4.79	0.40	1.09	1.77	0.07	0.20	**93.42%**	**88.62%**
	Inception	91.05	5.89	16.62	26.76	0.74	2.07	**95.55%**	**92.28%**
Seizure	FFT Conv	0.24	0.4	0.1	0.16	0.00205	0.00328	**99.75%**	**97.9%**
	Direct Conv	0.24	0.4	0.25	0.4	0.00301	0.004816	**99.87%**	**98.8%**

Table 1: Evaluation of SCALENet performance compared to baseline CPU when deploying nine network architectures. All CIFAR processing is based on Arty A7 and MicroBlaze (μB), all ImageNet based on ZedBoard and Cortex-A9. Seizure Detection is all hardware on Zedboard. On average, utilizing the accelerator improves throughput by 10x and energy usage 5.4x

library functions and perform data marshaling. A pre-deployment script is also used to remove all training related details such as gradients in addition to merging sets of convolution, batch normalization, and ReLU layers. We also utilize NVIDIAs CUDA 6.5 framework along with cuDNN v2 library when targeting the K1 GPU of the Jetson TK1.

5.2 Measurement Setup

Each platform is measured in real-time for both the execution time and power consumption required to classify sample data. To accomplish this, we actively record these metrics for a large number of samples and then average to derive the *per classification* performance. For power results, we look to only measure the consumption of both the processor and external DDR memory. While a few platforms include built-in monitoring capabilities, we utilized an external TI INA219 voltage and power IC connected to each system's primary power rails to ensure consistency. For each platform all other peripherals including HDMI, debug circuitry, and additional communication peripherals such as Wi-Fi and Bluetooth were powered off.

5.3 COTS Results

For each of the networks, we performed nine experiments deployed onto the FPGA platforms with and without SCALENet. Networks trained on CIFAR, the Arty platform is using a single SCALENet accelerator. For the ImageNet targeted networks, the Zedboard platform is used and incorporates three PEs of the SCALENet accelerator. For a given layer, the work is split among the three accelerators. Doing so enables avoiding dependencies or starvation conditions among the accelerators. The entire Arty platform containing the μB and SCALENet accelerator has an average measured power consumption of 1.4 Watts when running at 166.67 MHz. The Zedboard containing the dual-core ARM Cortex-A9 and 3 SCALENet PE accelerators have an average measured the power consumption of 2.07 Watts when running at 667.67 MHz. From these results, the proposed SCALENet accelerator can significantly improve classification throughput and energy efficiency when compared to using only the host processor. On average, for all networks evaluated, exploiting the accelerator can dramatically increase classification throughput, up to 10x and decrease total system energy usage by 5.4 for the VGG-A and VGG-D network.

For the modified SqueezeNet trained on CIFAR-10, we targeted the Cmod A7, uses the same FPGA as the Arty, but instead of 256 MB DDR3 memory there is only 512 KB SRAM, and there are no external peripherals. Reduction in memory is possible as SqueezeNet requires a lower number of parameters after compression.

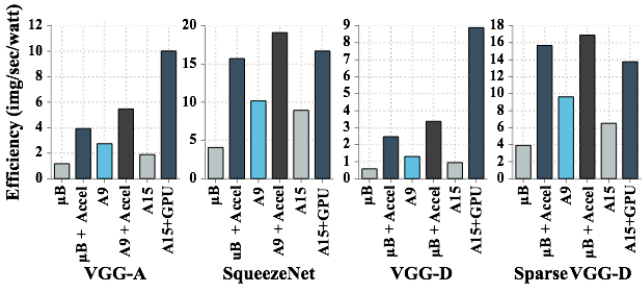

Figure 8: Energy efficiency comparison of platforms with and without accelerator targeted for CIFAR-10 dataset. MicroBlaze™(μB) & μB+Accel are performed using Arty or Cmod A7. While A9 & A9+Accel are performed using Zedboard, and A15 & A15+GPU are using Jetson TK1 platform.

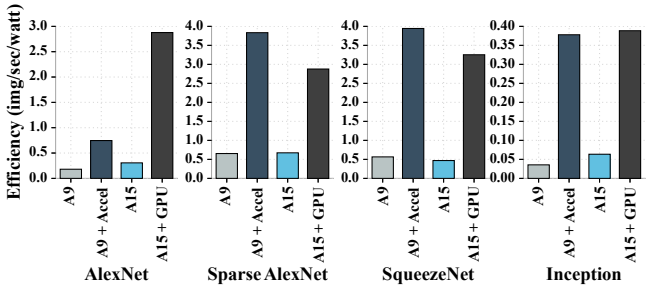

Figure 9: Energy efficiency comparison with and without accelerator targeted for ImageNet dataset. A9 & A9+Accel are using Zedboard, and A15 & A15+GPU are the Jetson TK1 platform.

We also compared SCALENet against NVIDIA's Jetson TK1 mobile GPU platform on all networks. Figure 8 shows the efficiency results of running on the three platforms for the four CIFAR trained networks with and without accelerators. All platforms, there is a

Metrics	[3]	[5]	[16]	**This work**	
Platform	Virtex-5	Zynq	Virtex-7	**Zynq**	Jetson TK1
Precision (bit)	48 Fixed	16 Fixed	322 Float	**16 Fixed**	32 Float
Clock (MHz)	120	150	100	**100**	2320
Network	N/A	N/A	AlexNet	**Inception**	Inception
Network Complexity (GOP)	0.52	0.552	1.33	**15.92**	55.30
Network Performance (GOP/s)	16	25.15	61.62	**1592**	55.30
Total Power (W)	14	8	18.61	**2.07**	12.08
Network Energy Efficiency (GOP/s/W)	1.14	2.90	3.31	**5.70**	4.58

Table 2: Comparison of the proposed accelerator with related works when evaluated on a convolutional network. For this work, measurements are taken when running on Zedboard containing three SCALENet accelerators without sparsification.

large increase in efficiency with the accelerator. The SCALENet accelerator on average can improve efficiency over its host processor by 3x whereas the K1 GPU can increase efficiency by 4.5x over its ARM CPUs. Figure 9 similarly compares the efficiency of the FPGA and GPU platforms when targeting the ImageNet networks. The Zedboard with 3 SCALENet accelerators can achieve an average increase in efficiency of 9x whereas the TK1 accelerated with the GPU can increase efficiency on average by 5x. In Table 2 the last row outlines the power efficiency savings for using the SCALENet architecture, combined with the scheduler, compared to just the ARM processor running torch compiled against the OpenBlast libraries. The execution time improved by 99.7% while the power only improved by 74.10%.

6 COMPARISON WITH EXISTING WORK

Table 2 compares the proposed SCALENet with existing FPGA-based CNN accelerators. Deployed on Zedboard platform with Cortex-A9 running at 667.67 MHz and FPGA running at 100 MHz, the accelerator achieves an energy efficiency of 5.70 GOP/J with total system power of 2.07 W and throughput of 15.92 GOP/s. SCALENet delivers higher energy efficiency for a given network complexity than the previous best accelerators [5] and [16] while targeting a much lower power utilization. Additional gains in efficiency are achieved by exploiting the ability to perform fused convolution, batch normalization, and ReLU.

7 CONCLUSION

In this work, we proposed contributions in two enterprises for deploying convolutional and fully connected neural networks in resource-bound, real-time embedded systems. In the first contribution, we evaluated eight image processing focused networks trained on CIFAR-10 and ImageNet datasets for computation, memory, and accuracy and we assess the use of a modified network for time series EEG seizure detection. In the second contribution, we proposed and evaluated SCALENet: a SCalable Low power Accelerator for real-time deep neural Networks. The accelerator enables deploying a variety of networks through coarse and fine grain configuration as well as acceleration of FC and CNN layers and implementation flexibility for hardware only or hardware and software designs. The proposed accelerator was evaluated in real-time on different FPGA platforms using eight image processing networks and one custom time series network for biomedical seizure detection in EEGs and demonstrated to outperform all other COTS platforms. When accelerating the highest complexity network evaluated, Inception with 11.78 GOP, on the Zedboard platform with a dual-core ARM Cortex-A9 running at 667.67 MHz and FPGA running at 100 MHz, the accelerator achieves an energy efficiency of 5.70 GOP/s/W with a total system power of 2.07 W and throughput of 15.92 GOP/s. SCALENet produces higher energy efficiency than the prior best accelerators as well as Jetson TK1 while targeting a power profile that is more than 4x lower than the Jetson TK1. In the case of time series seizure detection, our SCALENet architecture shows an execution time and power improvement of 99.75% and 97.9% respectively compared to software-only implementations.

REFERENCES

[1] T. Abtahi, A. Kulkarni, and T. Mohsenin. 2017. Accelerating convolutional neural network with FFT on tiny cores. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. 1–4. https://doi.org/10.1109/ISCAS.2017.8050588

[2] T. Abtahi, C. Shea, A. Kulkarni, and T. Mohsenin. 2018. Accelerating Convolutional Neural Network with FFT on Embedded Hardware. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (2018).

[3] U. Rajendra Acharya et al. 2017. *Computers in Biology and Medicine* (2017). https://doi.org/10.1016/j.compbiomed.2017.09.017

[4] Y. Chen et al. 2016. 14.5 Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. In *ISSCC*.

[5] Vinayak Gokhale et al. 2014. A 240 G-ops/s mobile coprocessor for deep neural networks. In *CVPRW*.

[6] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. 2000 (June 13). PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. *Circulation* 101, 23 (2000 (June 13)), e215–e220. Circulation Electronic Pages: http://circ.ahajournals.org/content/101/23/e215.full PMID:1085218; doi: 10.1161/01.CIR.101.23.e215.

[7] Forrest N. Iandola et al. 2016. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. *CoRR* (2016).

[8] Adam Page, Ali Jafari, Colin Shea, and Tinoosh Mohsenin. 2017. SPARCNet: A Hardware Accelerator for Efficient Deployment of Sparse Convolutional Networks. *J. Emerg. Technol. Comput. Syst.* 13, 3, Article 31 (May 2017), 32 pages. https://doi.org/10.1145/3005448

[9] A. Page, C. Shea, and T. Mohsenin. 2016. Wearable Seizure Detection using Convolutional Neural Networks with Transfer Learning. In *ISCAS*.

[10] Mohammad Samragh et al. 2017. Customizing Neural Networks for Efficient FPGA Implementation. In *FCCM*. IEEE.

[11] Jaehyeong Sim et al. 2016. 14.6 A 1.42 TOPS/W deep convolutional neural network recognition processor for intelligent IoE systems. In *ISSCC*. IEEE.

[12] Karen Simonyan et al. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* (2014).

[13] L. Song et al. 2016. C-Brain: A deep learning accelerator that tames the diversity of CNNs through adaptive data-level parallelization. In *DAC*.

[14] Abdulhamit Subasi et al. [n. d.]. Classification of EEG signals using neural network and logistic regression. *Computer Methods and Programs in Biomedicine* ([n. d.]).

[15] Xilinx. 2011. Power Methodology Guide. (2011).

[16] Chen Zhang et al. 2015. Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks. In *FPGA (FPGA '15)*. ACM.