Accelerating Deep Convolutional Neural Networks Using Specialized Hardware

Kalin Ovtcharov, Olatunji Ruwase, Joo-Young Kim, Jeremy Fowers, Karin Strauss, Eric S. Chung
Microsoft Research
2/22/2015

Abstract

Recent breakthroughs in the development of multi-layer convolutional neural networks have led to state-of-the-art improvements in the accuracy of non-trivial recognition tasks such as large-category image classification and automatic speech recognition [1]. These many-layered neural networks are large, complex, and require substantial computing resources to train and evaluate [2]. Unfortunately, these demands come at an inopportune moment due to the recent slowing of gains in commodity processor performance.

Hardware specialization in the form of GPGPUs, FPGAs, and ASICs[1] offers a promising path towards major leaps in processing capability while achieving high energy efficiency. To harness specialization, an effort is underway at Microsoft to accelerate Deep Convolutional Neural Networks (CNN) using servers augmented with FPGAs—similar to the hardware that is being integrated into some of Microsoft's datacenters [3]. Initial efforts to implement a single-node CNN accelerator on a mid-range FPGA show significant promise, resulting in respectable performance relative to prior FPGA designs and high-end GPGPUs, at a fraction of the power. In the future, combining multiple FPGAs over a low-latency communication fabric offers further opportunity to train and evaluate models of unprecedented size and quality.

Background

State-of-the-art deep convolutional neural networks are typically organized into alternating convolutional and max-pooling neural network layers followed by a number of dense, fully-connected layers—as illustrated in the well-known topology by Krizhevsky et al. in Figure 1 [1]. Each 3D volume represents an input to a layer, and is transformed into a new 3D volume feeding the subsequent layer. In the example below, there are five convolutional layers, three max-pooling layers, and three fully-connected layers.

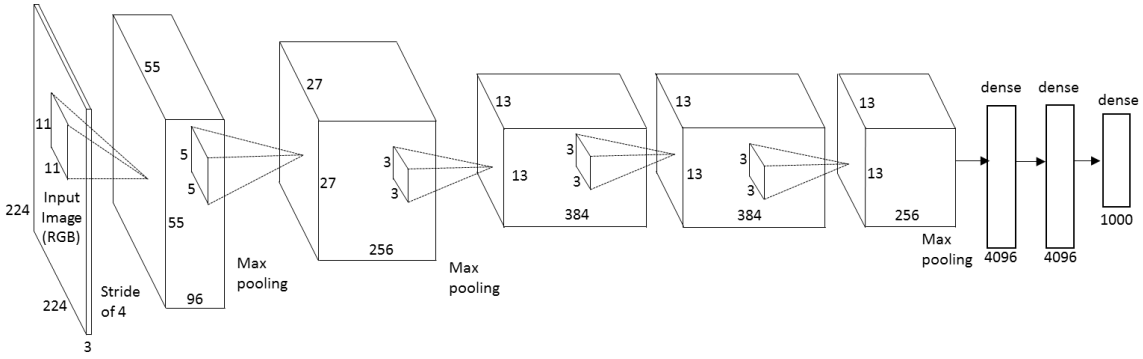

Figure 1. Example of Deep Convolutional Neural Network for Image Classification. Image source: [1].

[1] General Purpose Computing on Graphics Processing Units, Field Programmable Gate Arrays, Application-Specific Integrated Circuits.

In this paper, we primarily discuss the problem of 3D convolution, although other operations such as pooling and fully-connected layers are also targeted. Figure 2 illustrates the basic pattern of 3D convolution. A 3D input volume of dimensions N x N x D is convolved with H kernels of dimension k x k x D and stride S. Each 3D kernel is shifted in a sliding-window-like fashion (with a shift offset defined by parameter S) across the input volume. During each shift, every weight belonging to the 3D kernel is multiplied and added with every pair-wise input element from the overlapping region of the 3D input volume. After convolution, an optional pooling operation (defined by parameters p and s) is used to subsample the convolved output by sliding a 2D window across the 3D convolved output and selecting the maximum (or average) value over the window.

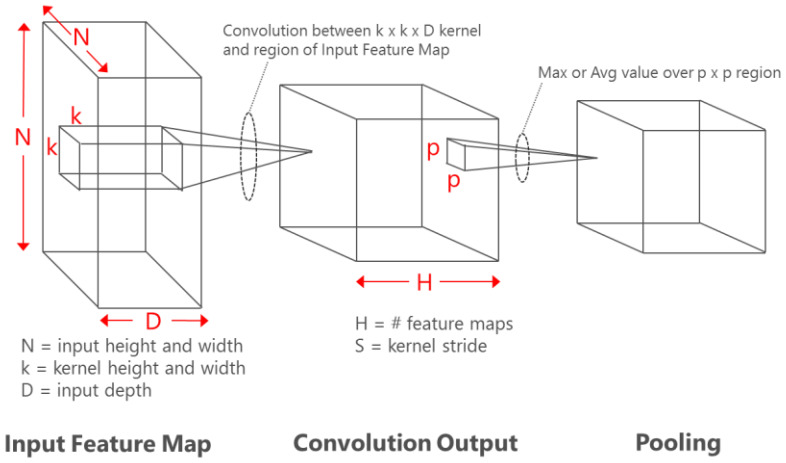

Figure 2. Pattern of 3D convolution and pooling.

Accelerating Deep Convolutional Neural Networks in the Datacenter

In 2014, Microsoft announced the Catapult project, which successfully demonstrated an effort to accelerate Bing Ranking by a factor of nearly 2X using FPGAs in the datacenter [3]. Leveraging the infrastructure pioneered in Catapult, our team at Microsoft Research has developed a high-throughput Convolutional Neural Network FPGA accelerator that achieves excellent performance while consuming a small fraction of server power.

Figure 3 gives a high-level view of the CNN FPGA accelerator designed to efficiently compute forward propagation of convolutional layers. The key features of this design are: (1) a software configurable engine that can support multiple layer configurations at run-time (without requiring hardware re-compilation), (2) an efficient data buffering scheme and on-chip re-distribution network that minimizes traffic to off-chip memory, and (3) a spatially distributed array of processing elements (PEs) that can be scaled easily up to thousands of units.

In normal operation, the CNN accelerator is capable of accepting an input image and processing multiple convolutional layers in succession. During the initial layer, input image pixels are streamed on-chip from local DRAM, then stored into a multi-banked input buffer. These inputs are then streamed into multiple PE arrays, which perform independent dot-product operations in the 3D convolution step. A top level controller orchestrates the sequencing, addressing, and delivery of data to each of the PE arrays. Finally,

accumulated results are sent to a specialized network-on-chip, which re-circulates the computed output layer to the input buffers for the next round of layer computation[2].

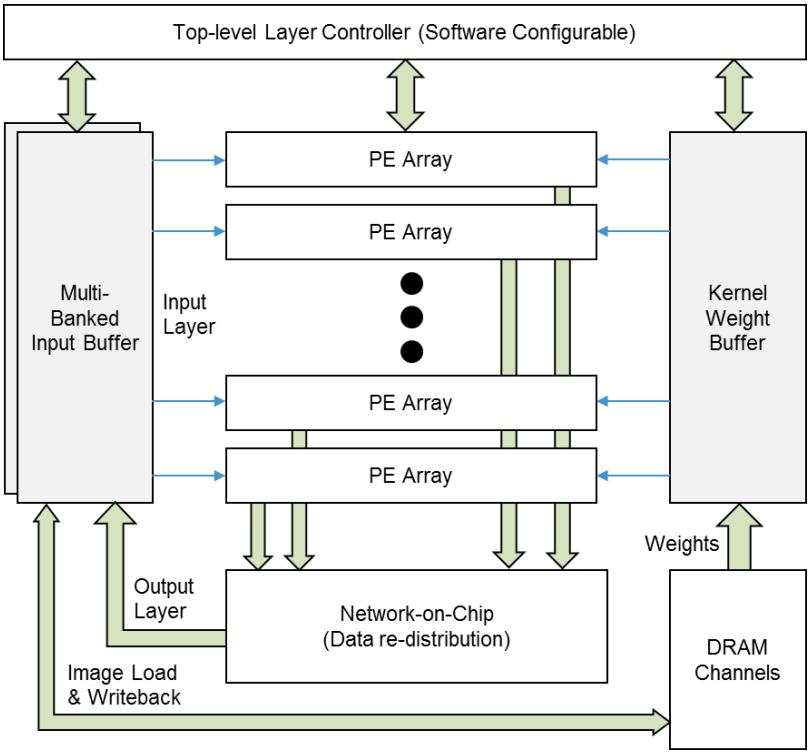

Figure 3. Top-Level Architecture of the Convolutional Neural Network Accelerator.

The accelerator highlighted in Figure 3 targets a dual-socket Xeon server equipped with a Catapult FPGA card, which includes a mid-range Stratix V D5 FPGA and 8GB of DDR3-1333 [3]. Each FPGA card supports up to 8GB/s of bandwidth over PCIe 3x8 and up to 21.3 GB/s of bandwidth to local DRAM. More specifications of the hardware are described in the original Catapult paper [3].

Table 1 shows the throughput of image classification (forward propagation only) using well-known models such as CIFAR-10 based on cuda-convnet [4], and ImageNet-1K based on Krishevsky et al [1]. We further evaluate the largest and most challenging model available to us, the ImageNet 22,000-category deep convolutional neural network trained using Project ADAM at Microsoft [2].

In general, our current Catapult server equipped with a mid-range Stratix V D5 FPGA achieves competitive processing throughput relative to recently published state-of-the-art FPGA solutions [5] and Caffe+cuDNN running on high-end GPGPUs [6]. It is worth noting that the GPGPU solutions require up to 235W of power to operate [7], making them impractical to deploy at scale in our power-constrained datacenters. In contrast, the FPGA solution consumes no more than 25W of power, incurring a less than 10% overhead in overall power consumption of the server. Also, our design achieves nearly 3X speedup relative to the most recently published work on accelerating CNNs using a Virtex 7 485T FPGA [5].

[2] Although not shown in Figure 3, additional logic is present to handle pooling and rectified linear operations.

	CIFAR-10 [4]	ImageNet 1K [1]	ImageNet 22K [2]	Max Device Power
Catapult Server + Stratix V D5 **[3]**	2318 images/s	134 images/sec	91 images/sec	25W
Catapult Server + Arria 10 GX1150 **[8]**	-	~233 images/sec (projected)	~158 images/sec (projected)	~25W (projected)
Best prior CNN on Virtex 7 485T **[5]**	-	46 images/sec[3]	-	-
Caffe+cuDNN on Tesla K20 **[6]**	-	376 images/sec	-	235W
Caffe+cuDNN on Tesla K40 **[6]**	-	500-824 images/sec[4]	-	235W

Table 1: Comparison of Image Classification Throughput and Power.

Our CNN accelerator is parameterizable and can be scaled to newer and faster FPGAs with minimal effort. Our team is currently mapping the design to Altera's new Arria 10 FPGA, which offers dedicated support for floating-point operations, and can deliver over 1 TFLOPS with very high energy efficiency [8]. Table 1 lists our conservative projection in expected gains once the design is adapted to Arria 10.

In conclusion, this paper described an investigation to accelerate deep convolutional neural networks using FPGAs. Initial results are promising and have shown that hardware specialization can achieve high levels of performance at low power consumption. In the future, we anticipate further significant gains when mapping our design to newer FPGAs such as the Arria 10 and Stratix 10, and when combining a large number of FPGAs together to parallelize both evaluation and training.

Acknowledgments

We would like to thank Doug Burger and the Catapult team for their feedback and support in this project. We thank Altera for their partnership and providing us with FPGA boards, tools, and support. We thank Trishul Chilimbi and the ADAM team for their support and providing us with the ImageNet-22K model.

References

[1] A. Krizhevsky, I. Sutskever and G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, Advances in Neural Information Processing Systems, 2012.

[2] T. Chilimbi, Y. Suzue, J. Apacible and K. Kalyanaraman, Project Adam: Building an Efficient and Scalable Deep Learning Training System, 11th USENIX Symposium on Operating Systems Design and Implementation, 2014.

[3] A. Putnam, et al., A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services, International Symposium on Computer Architecture, 2014.

[4] *https://code.google.com/p/cuda-convnet/.*

[5] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao and J. Cong, *Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks,* FPGA'2015, 2015.

[6] *http://caffe.berkeleyvision.org/performance_hardware.html.*

[7] *http://www.nvidia.com/content/PDF/kepler/Tesla-K40-Active-Board-Spec-BD-06949-001_v03.pdf.*

[8] *http://www.altera.com/literature/hb/arria-10/a10_overview.pdf.*

[3] Computed based on Table 7 from [5].

[4] Requires disabling ECC and setting clock boost to 875MHz.