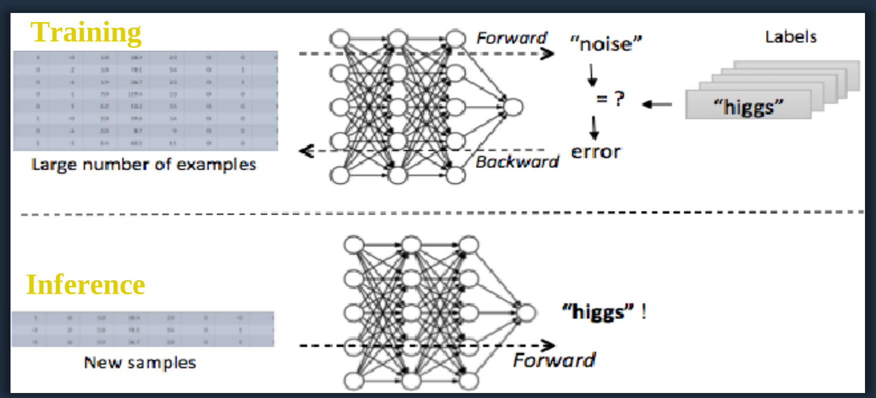
Simplified Computing Framework for FPGA-Accelerated Workloads

David Ojika, Varad Ghate, Herman Lam, Ann Gordon-Ross, Bhavesh Patel, Roland Kunz

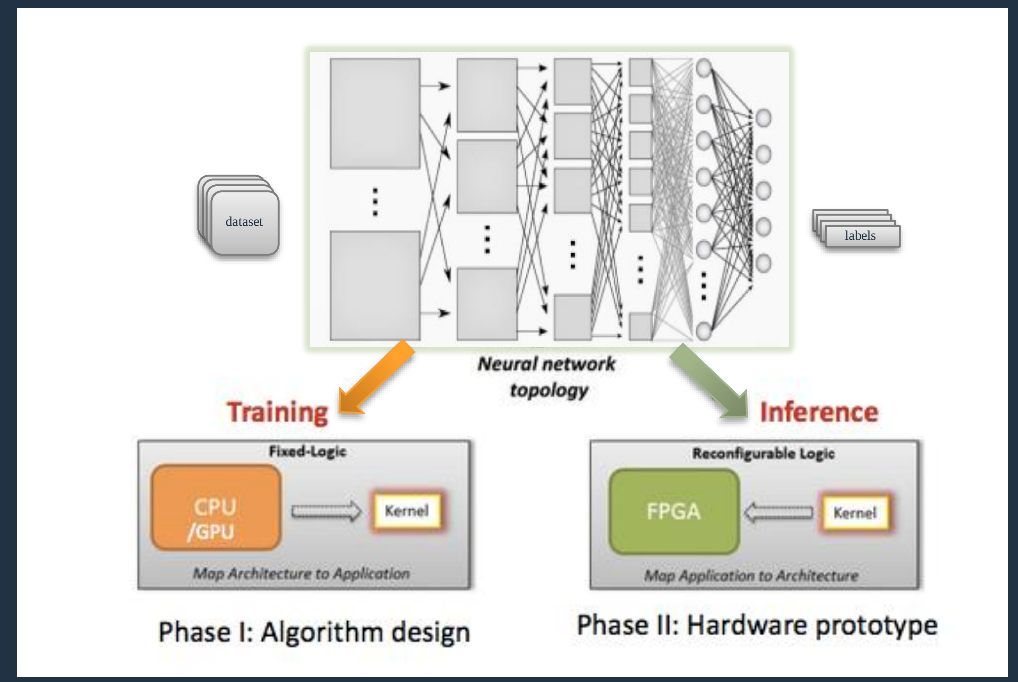
Introduction

- While the *training* task of **deep neural network (DNNs)** has been largely explored on GPUs and many-core processors for improving speedup, the *inference* task of DNNs on **field-programmable gate array (FPGAs)** remains a highly growing area of interest - for example, in real-time or streaming applications like image recognition



- FPGAs, a set of reconfigurable hardware logic, provide accelerated performance and energy efficiency for specific applications such as DNNs, however it is quite difficult to develop optimized FPGA accelerators for such applications while deploying them at scale in datacenters
- This work is an effort to derive a scheme for the efficient deployment and scaling of DNN models on FPGAs for scientific workloads

Methods



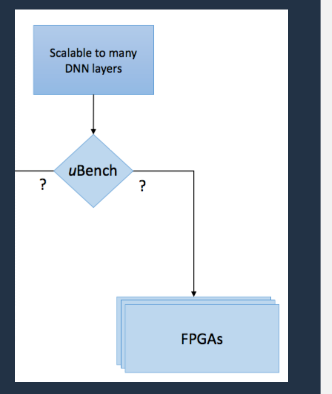
Deploy rapidly trained model to FPGA

FPGA Development

- High-level synthesis (HLS): Use OpenCL™ to build compute kernels for faster development versus register transfer logic (RTL) method
- Component-based design (CBD): Optimize specific compute kernels (e.g. matrix multiplication) by using vendor-specific extensions (e.g. loop pipelining, etc.)

Performance-Guided Scale-Out

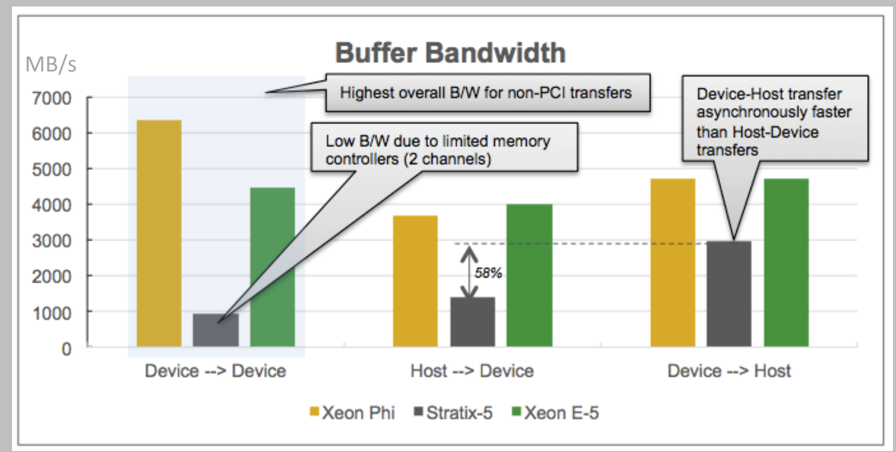
- Derive relationship between platform's theoretical performance versus microbenmarking performance
  - Several issues need to be addressed:
    - Arithmetic throughput (kernel performance)
    - Memory sub-system
    - OpenCL runtime overheads
    - End-to-end latency (via *interconnection network*)
    - Resource utilization and power

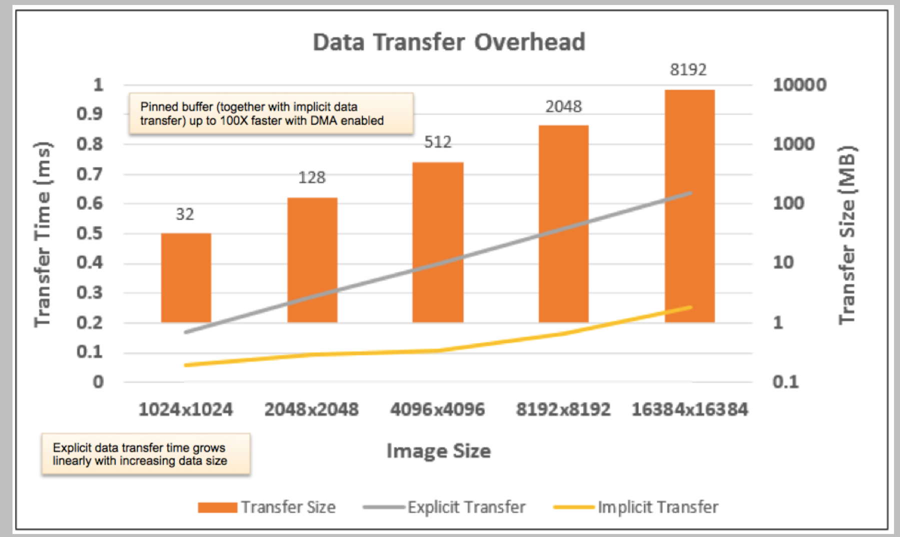


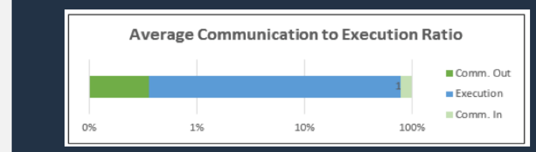
Use developed scheme to guide scale-out to neighbor FPGA(s)

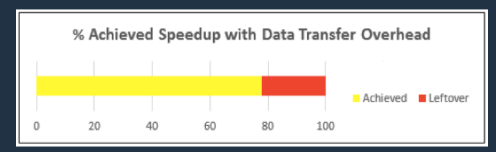
Benchmarking

Intranode Communication

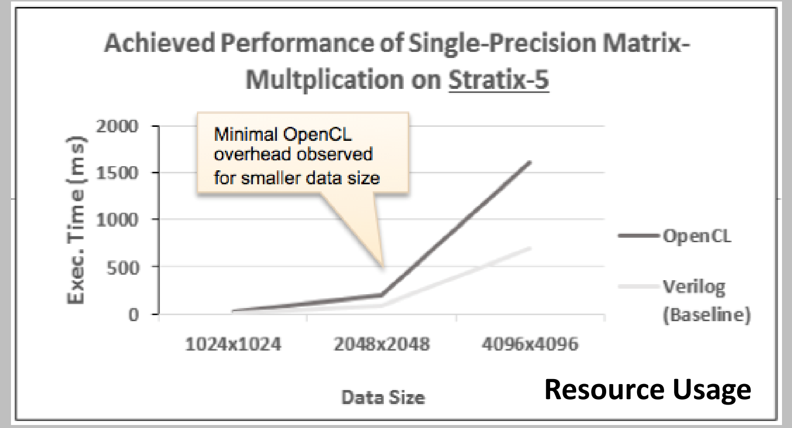


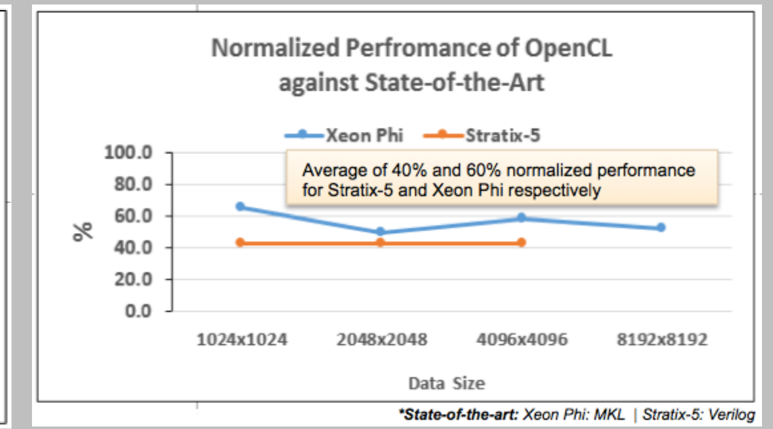






Computation

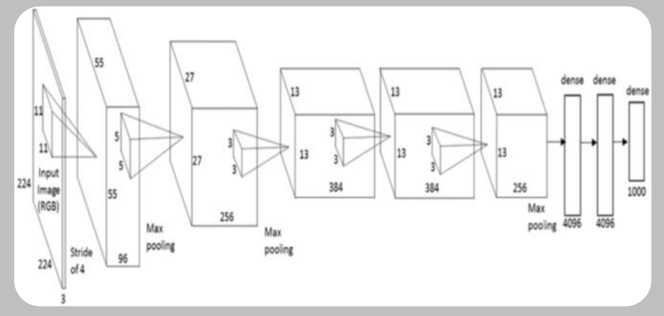




Resource Usage

Porting of AlexNet Model (Single FPGA)

Model Topology



Arria-10

- Batch size: 16
- Kernel runtime: 421.8 ms.
- Avg. process time per batch: 26.4 ms.
- Total floats read: 61,063,552
- Bytes of image read: 618, KB

Resource Usage (Arria-10)

Resource	Usage (%)
Logic utilization	48
ALUTs	26
Logic registers	23
Memory blocks	34
DSP blocks	20

Summary

- Preliminary results of image recognition on field-programmable gate arrays (FPGAs) indicate that FPGAs are efficient for DNN inference
- FPGA microbenchmarking (memory, throughput, etc.) can provide good hindsight towards performance tuning; 10G network latency not a performance bottleneck – but further experiments are required
- Data transfer overhead (over PCIe) can be amortized by choosing appropriately sized buffers. 1024x1024 matrices indicate most efficient performance on Stratix-5 FPGA
- OpenCL™ programming model and CBD present opportunities for making FPGAS more widely accessible to traditional software developers; less design effort at minimal performance degradation [1]
- Various FPGA interconnection networks present opportunities for exploring a variety of neural network architectures, both current and emerging
- Emerging FPGA devices (with floating-point units) are also important for accelerating several DNN computations within the *node* level

Future Directions

- Algorithmic: research and development of a parallel programming model for graph-based partitioning of compute nodes
- FPGA-as-a-service: FPGA microservices for hyperscaling of accelerated services with fault-tolerance
- Deployment model: containers and Kubernetes orchestration for datacenter-wide deployment
- New set of devices
  - PAC / Stratix-10 FPGA

Discussions

Memory Bandwidth

- PCIe data transfer mechanism with OpenCL can either improve or hurt performance
  - Choice of *transfer size*, *implicit* vs. *explicit* transfers
  - Implicit transfer of less than 1024x1024 matrices yields best results

RTL vs. OpenCL

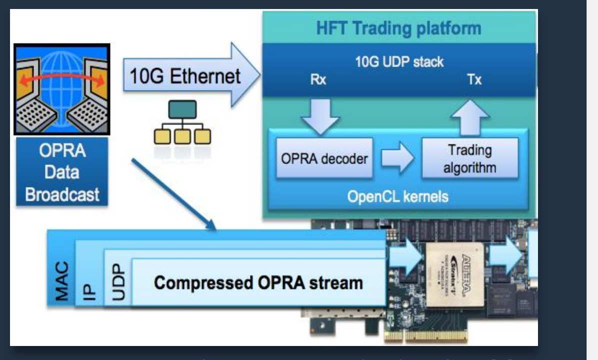
- Matrix-Mult: Execution time of RTL implementation much better than OpenCL – but only for large metrices; smaller matrices show comparable performance
- Much lower design flow and design effort with OpenCL

OpenCL Initialization Overhead

	OpenCL Setup Overhead (ms)	Kernel Setup Overhead (ms)	Total Overhead (ms)
Stratix-5	**1250**	**230**	**1480**
Xeon Phi	1240	220	1460
Xeon E-5	190	30	220

Network Latency

- < 0.5 uS
- Ethernet header removed by OpenCL
- Requires OpenCL board support package



OpenCL-based high-frequency trading (HFT) platform [5]

AlexNet Model on Arria-10 FPGA

- Convolution kernel most compute intensive (~ 89% of kernel computation time)
- OpenCL-based DNN kernels portable to multiple vendor platforms (Xilinx, Intel)
- *Parallelization*: room for improvement across multiple FPGAs (both model and data parallelism)

Interconnection Network

- Variety of interconnection networks for *multi-node FPGA architectures*
- Workload partitioning scheme depends on underlying *interconnection network and size of nodes*

Ring
Tens of nodes

Torus
Hundreds of nodes

Switch
Thousands of nodes

References

[1] D. Ojika, P. Majcher, W. Neubauer, S. Subhaschandra and D. Acosta, "SWiF: A Simplified Workload-Centric Framework for FPGA-Based Computing," 2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), Napa, CA, 2017

[2] A. M. Caulfield et al., "A cloud-scale acceleration architecture," 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), Taipei, 2016.

[3] U. Aydonat, S. O'Connell, D. Capalija, A. C. Ling, and G. R. Chiu. "An OpenCL™ Deep Learning Accelerator on Arria 10," in Proc. FPGA 2017.

[4] Dong Wang, Jiangjing An and Ke Xu, "PipeCNN: An OpenCL-Based FPGA Accelerator for Large-Scale Convolution Neuron Networks", https://arxiv.org/abs/1611.02450, 2016.

[5] "OPRA FAST Parser Design Example", www.altera.com

23rd Conference on High Energy and Nuclear Physics - CHEP 2018
Sofia, Bulgaria.