hls4ml: deploying deep learning on FPGAs for L1 trigger and Data Acquisition in collider experiments and beyond

Mia Liu
CPAD 2023.
On behalf of the HLS4ML team
From Collisions to Discoveries

CMS Experiment
40MHz collision rate
~1B detector channels

On-detector ASIC compression
~100ns latency

FPGA filter stack
~μs latency

On-prem CPU/GPU filter farm
~100ms latency

Worldwide computing grid
Exabyte-scale datasets

40MHz
Pb/s
10s Tb/s
~5 kHz

High Rate, Volume, Complexity

100ns latency

40MHz
100s kHz

10s Gb/s

Science with Big data: Multi-tier Data Processing
AI/ML: new opportunities for real-time reconstruction

- Tracks and calorimeter clusters
  - Charged particle tracking
  - Raw detector hits
  - Calorimeter clustering
  - Hit-based ML particle-flow reconstruction

- Tracks and calorimeter clusters
  - Track
  - Raw ECAL hit
  - Raw HCAL hit
  - Raw Muon chamber hit

- Particles
  - Charged hadron
  - Photon
  - Neutral hadron
  - Electron
  - Muon

- Higgs? Toponium?
- Dark Matter?

Learning grouping of detector elements
Co-design tool: crucial for prototyping AI at edge solutions

Algorithm hardware co-design for limited computing

Prototype with manageable programming barrier for domain scientists
NN on FPGAs

activation functions

x_n = g_n(W_{n,n-1}x_{n-1} + b_n)

Precomputed, and stored in BRAMs

Multiplications

DSPs

Addition

Logic cells

Multiplications

DSPs

Addition

Logic cells

N_{multiplications} = \sum_{n=2}^{N} L_{n-1} \times L_n

DSPs

Logic cells

Digital Signal Processors (DSPs)

Logic cell:
Flip-flops (FF) and look up tables (LUTs)

Virtex Ultrascale+ VU9P

6800 DSPs
1M LUTs
2M FFs
75 Mb BRAM

16 inputs
64 nodes
activation: ReLU
32 nodes
activation: ReLU
32 nodes
activation: ReLU
5 outputs
activation: SoftMax

Overview

● AWS F1 instances are machines connected directly to a Xilinx Virtex UltraScale+ FPGA (VU9P) using PCI-express

● General application development on AWS done using SDAccel

FPGA diagram

MMACHINE LEARNING & FPGA

Field Programmable Gate Arrays are reprogrammable integrated circuits

Contain many different building blocks ('resources') which are connected together as you desire

Originally popular for prototyping ASICs, but now also for high performance computing

'Computing in space as well as time'

Machine learning algorithms are ubiquitous in HEP

FPGA usage broad across HEP experiments

Centered on DAQ and trigger development

Some early adaptions of ML techniques in trigger [1]

FPGA development becoming more accessible

High Level Synthesis, OpenCL

FPGA interest in industry is growing

Programmable hardware with structures that maps nicely onto ML architectures

Efficient Algorithms

Quantization

Inspired by Phil’s Fast ML for Science workshop
Efficient Algorithms

Pruning

before pruning

after pruning

pruning synapses

pruning neurons
Efficient Algorithms

Quantization       Compression/Pruning

‘Ultimate optimization’ of ‘bits of information’

https://arxiv.org/abs/2102.11289
https://arxiv.org/abs/2304.06745
Efficient Algorithms

Compress it creatively: knowledge distillation. e.g.
Efficient Algorithms

Neural Architecture search
e.g. EfficientNet for image
detection
Efficient Algorithms
Algorithm System Co-design for Your Metrics

Co-design tools enables design-space exploration and convergence on optimal solutions

+ e.g. Radiation Environment: Fault Tolerant

Image source: 1

Image source: 2
HLS4ML

HLS conversion → HLS project

Keras, TensorFlow, PyTorch...

Compressed model

Model

Usual ML software workflow

hls4ml

Co-processing kernel

Custom firmware design

Tune configuration

Precision, rave, pipeline

Vivado HLS

HLS compiler

Mentor
Catapult
Coming soon

ASiC

VivadoAccelerator
for Xilinx
accelerators / SoCs

hls4ml github > 500 downloads per month, 954 stars
HLS4ML in action

- LHC Run 3
- displaced muon ID enables completely new capability; muon momentum regression cuts rate by > 2x for HL-LHC
- Anomaly detection at L1
- Active exploration
  - Applications for Run 3 & HL-LHC from low-level data compression to cluster calibration to high level physics topology selections to anomaly detection.
- See Sridara’s talk on Tuesday.
ML in CMS Phase 2 L1 Trigger

Introduction

- The trigger is a binary classifier.
- Efforts underway to use ML during Run 3 - gaining valuable experience! CICADA, AXOL1TL, TOPOL1TL.
- Machine Learning will be prevalent throughout the Phase 2 System.
- From known projects, object multiplicity, and event rate, conservatively estimate 25 billion ML inferences per second.
- We've developed some sophisticated stuff that has wider applications, here I'll describe some of those.
- We also need to note developments from the wider academic community and industry, I'll highlight some trends.

These have NNs and/or BDTs inside.

Detector hits
Clusters & Tracks
Particles
Event Categorisation

1 bit: keep / discard
ML in CMS Phase 2 L1 Trigger

Also being explored in ATLAS. e.g. talk on CNN for muon identification for Phase 2 Level 0 trigger
HLS4ML: user driven development

- hls4ml 2023 roadmap plans new developments and a regular release schedule
- Q1 release: v0.7.0 & v0.7.1
  - Backend redesign to support multiple compilation targets [395]
  - Documentation updates [710, 744, 774]
  - Efficient network implementations [503, 509, 509]
  - Recurrent neural networks [560, 575]
  - Alveo accelerator FPGA card support [552]
  - Support for Vitis HLS [629]
  - Extension API [528]
- Q2 release: v0.8.0
  - Configuration editor [784]
  - PyTorch parsing improvements [799]
  - Symbolic expressions [660]
  - Optimization API [768, 809]
  - Large streaming CNN [PR Soon]
- Upcoming:
  - QONNX ingestion [591]
  - Catapult HLS
HLS4ML: user driven development

sPHENIX tracking GNN hls4ml synthesis results

- Network inputs: nodes=80, edges=100
  - Input network
    - Can be parallelized to be “nodes” times faster (i.e., 15ns)
  
<table>
<thead>
<tr>
<th>Latency</th>
<th>BRAMs</th>
<th>DSPs</th>
<th>FFs</th>
<th>LUTs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2 us</td>
<td>6.5%</td>
<td>0.3%</td>
<td>5%</td>
<td>7.5%</td>
</tr>
</tbody>
</table>

- Edge network

<table>
<thead>
<tr>
<th>Latency</th>
<th>BRAMs</th>
<th>DSPs</th>
<th>FFs</th>
<th>LUTs</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 us</td>
<td>15%</td>
<td>2%</td>
<td>20%</td>
<td>65%</td>
</tr>
</tbody>
</table>

- Node network (results from HLS synthesis, vivado synthesis OOM’d)
  - Neet to optimize the scatter_add function (expecting ~2us for the net)

<table>
<thead>
<tr>
<th>Latency</th>
<th>BRAMs</th>
<th>DSPs</th>
<th>FFs</th>
<th>LUTs</th>
</tr>
</thead>
<tbody>
<tr>
<td>12 us</td>
<td>42%</td>
<td>7%</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Extremely preliminary - DO NOT TRUST NUMBERS

Support GNN/transformers
Extended operations supported in hls4ml to implement a GNN developed for track reconstruction in the sPhenix trigger

More work needed for CMS applications (100 ns latency)
HLS4ML: user driven development

Symbolic Regression on FPGAs for Fast Machine Learning Inference

Ho Fung Tsoi¹*, Adrian Alan Pol², Vladimir Loncar³,⁴, Ekaterina Govorkova³, Miles Cranmer²,⁵, Sridhara Dasu¹, Peter Elmer², Philip Harris³, Isobel Ojalvo², and Maurizio Pierini⁶

¹University of Wisconsin-Madison, USA
²Princeton University, USA
³Massachusetts Institute of Technology, USA
⁴Institute of Physics Belgrade, Serbia
⁵Flatiron Institute, USA
⁶European Organization for Nuclear Research (CERN), Switzerland

- Backend redesign to support multiple compilation targets
- Documentation updates
- Efficient network implementations
- Recurrent neural networks
- Alveo accelerator FPGA card support
- Support for Vitis HLS
- Extension API

- Q1 release: v0.7.0 & v0.7.1
  - Backend redesign to support multiple compilation targets
  - Documentation updates
  - Efficient network implementations
  - Recurrent neural networks
  - Alveo accelerator FPGA card support
  - Support for Vitis HLS
  - Extension API

- Q2 release: v0.8.0
  - Configuration editor
  - PyTorch parsing improvements
  - Symbolic expressions
  - Optimization API
  - Large streaming CNN [PR Soon]

10X resource reduction

arxiv.org
HLS4ML: collaboration with XILINX FINN

- hls4ml 2023 roadmap plans new developments and a regular release schedule
  - Q1 release: v0.7.0 & v0.7.1
    - Backend redesign to support multiple compilation targets [395]
    - Documentation updates [710, 744, 774]
    - Efficient network implementations [503, 509, 509]
    - Recurrent neural networks [560, 575]
    - Alveo accelerator FPGA card support [552]
    - Support for Vitis HLS [629]
    - Extension API [528]
  - Q2 release: v0.8.0
    - Configuration editor [784]
    - PyTorch parsing improvements [799]
    - Symbolic expressions [660]
    - Optimization API [768, 809]
    - Large streaming CNN [PR Soon]
    - Upcoming:
      - QONNX ingestion [591]
      - Catapult HLS

QONNX: Extension to the ONNX intermediate representation format to represent arbitrary-precision quantized neural networks
HLS4ML On-ASIC

CMS Experiment

40MHz collision rate
~1B detector channels

On-detector ASIC compression
100ns latency

Pb/s 40MHz

FPGA filter stack
~μs latency

10s Pb/s 100s kHz

On-prem CPU/GPU filter farm
~100 ms latency

Worldwide computing grid
Exabyte-scale datasets

‘Pt modules’ for pixels.
Enabled by HLS4ML catapult
See talk by Jieun, and Mathieu
Fast ML for Science

Real-time seizure detection
Talk by W. Lemaire
Fast ML for Science

Particle accelerator controls
Talk by J. Mitrevski
Fast ML for Science

New materials for quantum and energy
Talk by J. Agar
Fast ML for Science

Supernova detection and multi-messenger astronomy

Talk by S.C. Hsu
Fast ML for Science

Qubit readout and control
Talk by J. Campos
Fast ML for Science

See many more at the fast machine learning for science workshop 2023

Qubit readout and control
Talk by J. Campos
5 years research institute composed of Domain Scientists, Computer Scientists and System Experts.

Has been supporting some HLS4ML core development

Interface with consortia to tackle HEP challenges in ML for edge processing?
Summary

HLS4ML is an open-source, community driven product

Assist and Facilitate Fast ML at the edge

Core group and community supporting the tool

HLS4ML core team plans to improve community engagement:

Summer Schools in 2024 and 2025. An open call for applications will be made to undergraduate students at US institutions. The Summer Schools will be organized as a series of hands-on tutorial and demo sessions, a mini-hackathon with hls4ml, and an introductory-level design challenge.

Developer events: all stakeholders of the hls4ml ecosystem (core contributors of hls4ml, invited users of hls4ml, graduate level students who use with hls4ml to facilitate their thesis works, and industry representatives) together.

Continue to improve HLS4ML with improved code generation, tensor operation, integrate with HLS eco-system (scaleHLS, pyLog, MLIR). See talk by Vladimir at Fast ML for Science@ICCAD 2023