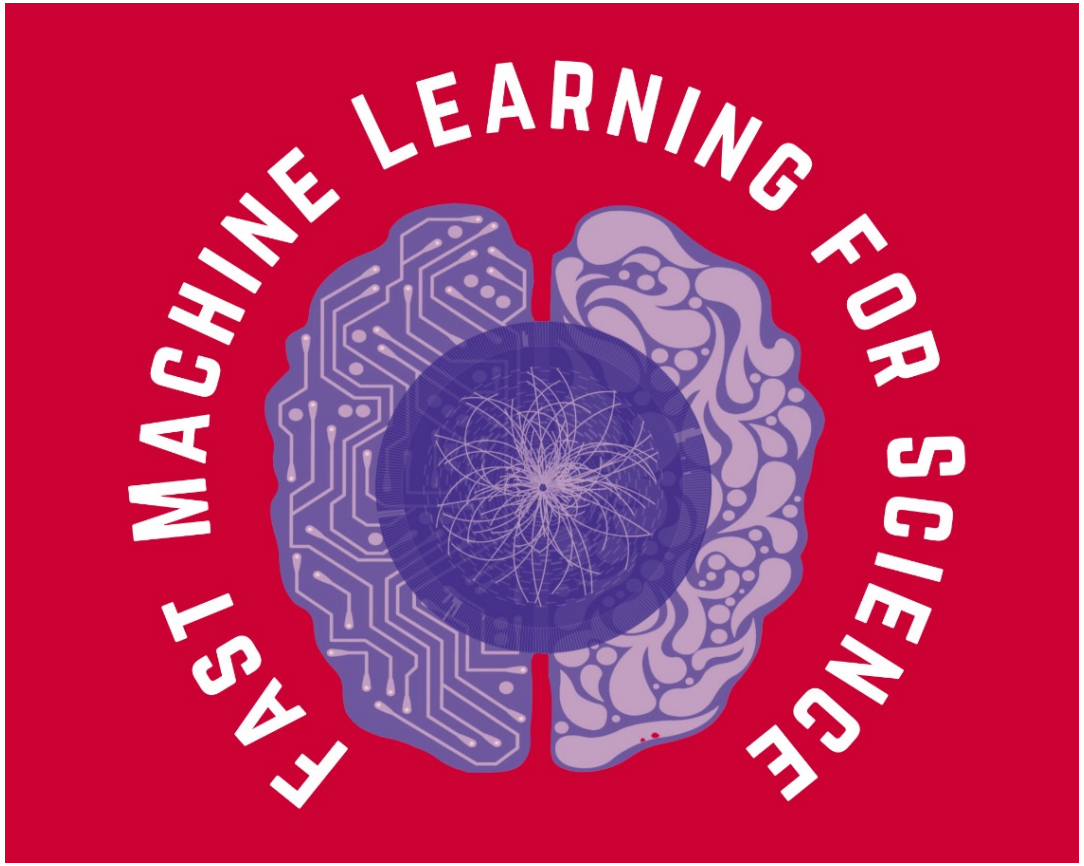
Fast ML at the LHC

Allison Deiana



(From dedicated workshop at SMU)

"Fast" Machine Learning

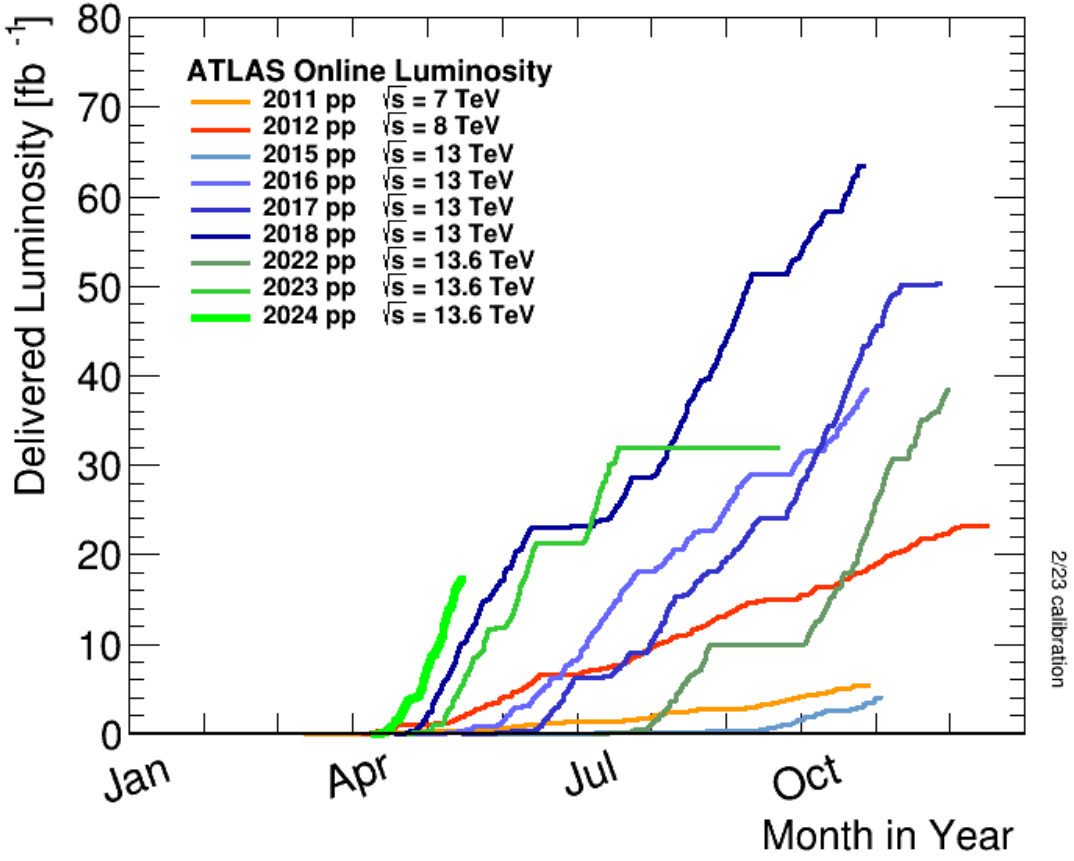
- Machine Learning has had a **major impact** on how high energy particle physics research is conducted.
  - Typical use involves separation of a signal class from a background class, using multiple inputs for discrimination. (BDT, NN, etc.)

- What is "**fast**" machine learning?
  - Accelerating the speed of ML algorithms.
  - Classifying information in real-time, taking into account constraints in **latency**, **bandwidth**, and **throughput**.
  - Often requires specific hardware.

Fast ML Uses for ATLAS at the LHC

- The ATLAS detector at the LHC collects a massive amount of data, and this will only increase in the high-luminosity era (2029+). Algorithms can:
  - Be computationally expensive
  - Not be comfortably scalable
  - Require more time than is available

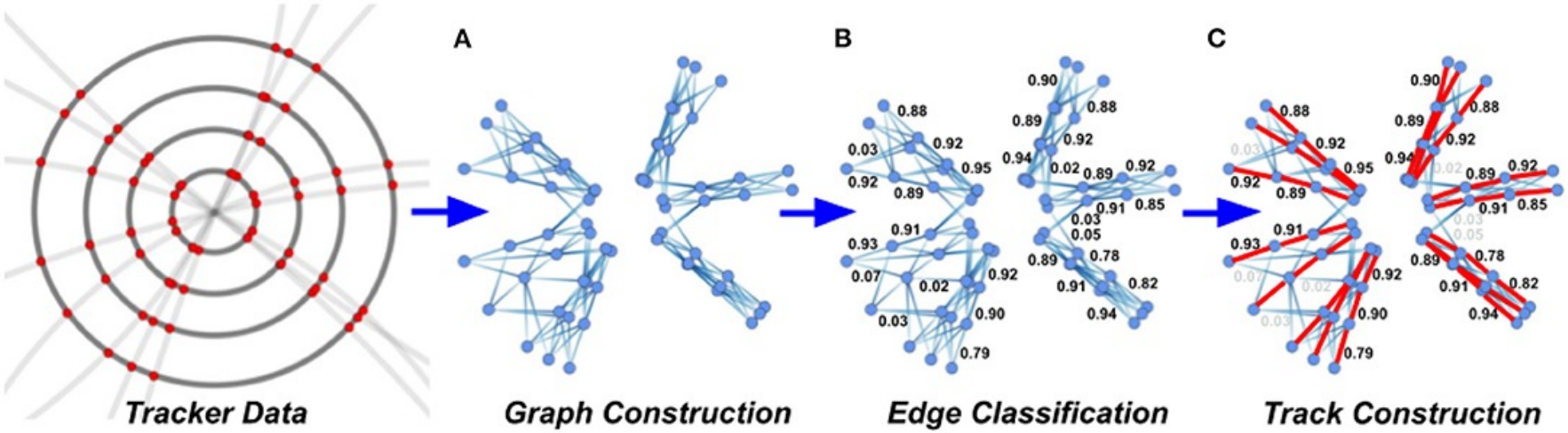
- Acceleration of ML algorithms is one way to address these limitations moving forward. Applications are being explored for:
  - Event reconstruction
  - Event simulation
  - Real-time analysis at 40 MHz
  - On-detector applications



https://twiki.cern.ch/twiki/bin/view/AtlasPublic/LuminosityPublicResultsRun3

Event Reconstruction

- Reconstructing proton-proton collision events involves challenging pattern-recognition tasks, given the high number of secondary particles and the high granularity of the detector.
- Traditional algorithms scale poorly to high-luminosity conditions. Some examples include reconstructing tracks of charged particles and energy reconstruction of high-granularity calorimeters.
- Many studies are currently underway that make use of graphical neural networks (GNNs) and heterogenous computing (CPU+GPU or CPU+FPGA) to improve the speed of the algorithm.



Fast Machine Learning Applications for Science

Event Simulation

- Simulation of events is necessary for performing searches and measurements in particle physics.
  - Simulation is used to model as-yet-unobserved signals as well as many major backgrounds.
  - MC methods are used to simulate the interactions, and then GEANT4 is used to simulate the response of the detector.

- Given the increasingly large ATLAS dataset, simulation is becoming very expensive.
  - Currently require over half of the experiments computing resources, expected to increase with HL-LHC.
  - Many efforts to develop ML methods to take over specific computationally intensive tasks, like modeling of EM showers, jet reconstruction, or matrix element calculations.

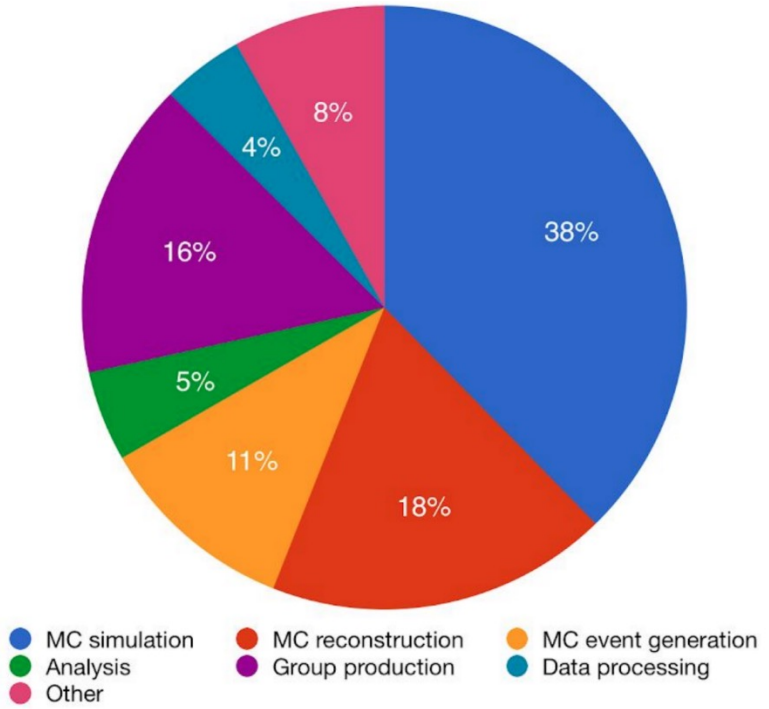
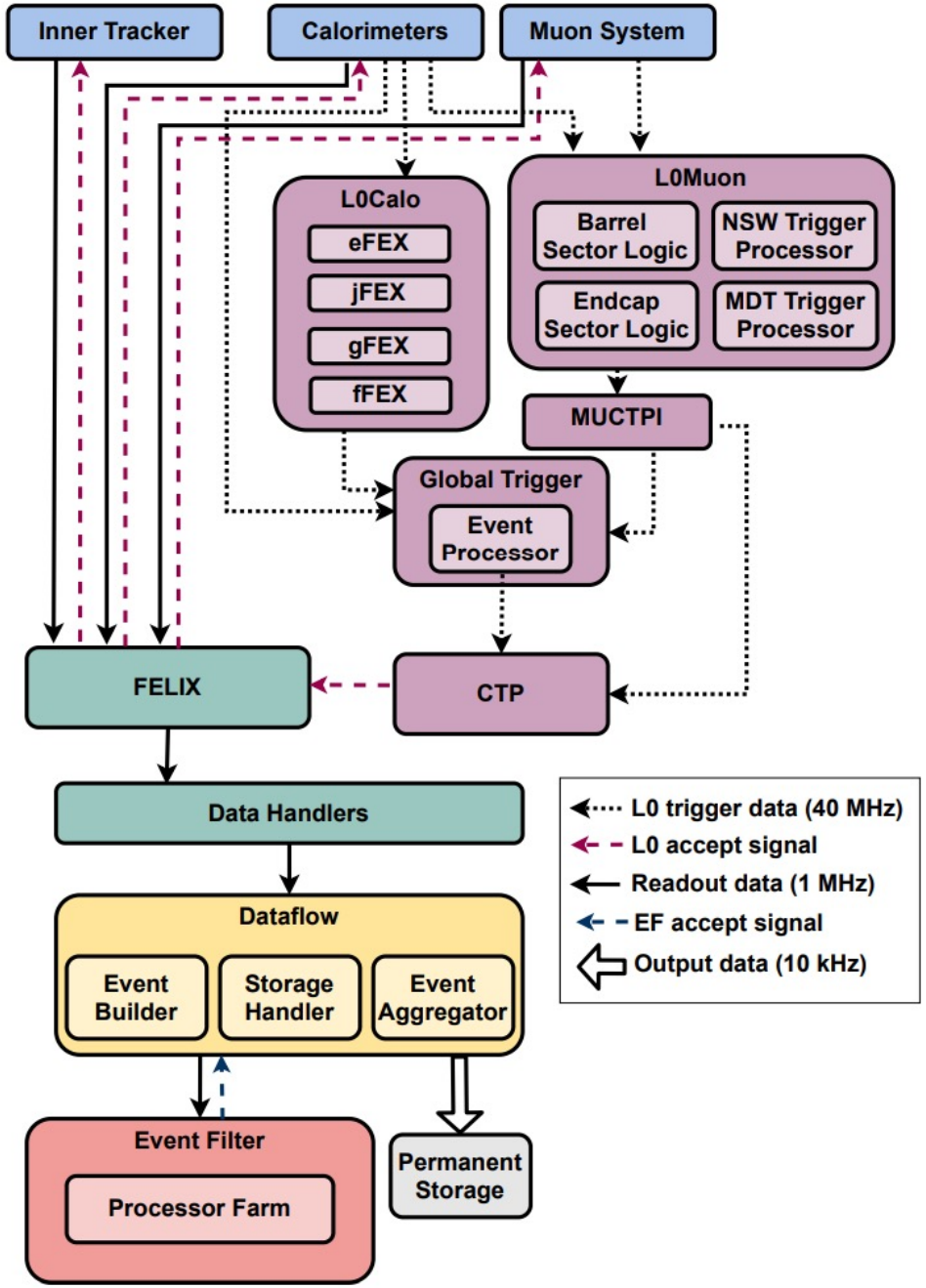


Figure 1: ATLAS CPU hours used by various activities in 2018

ATLAS HL-LHC Computing Conceptual Design Report

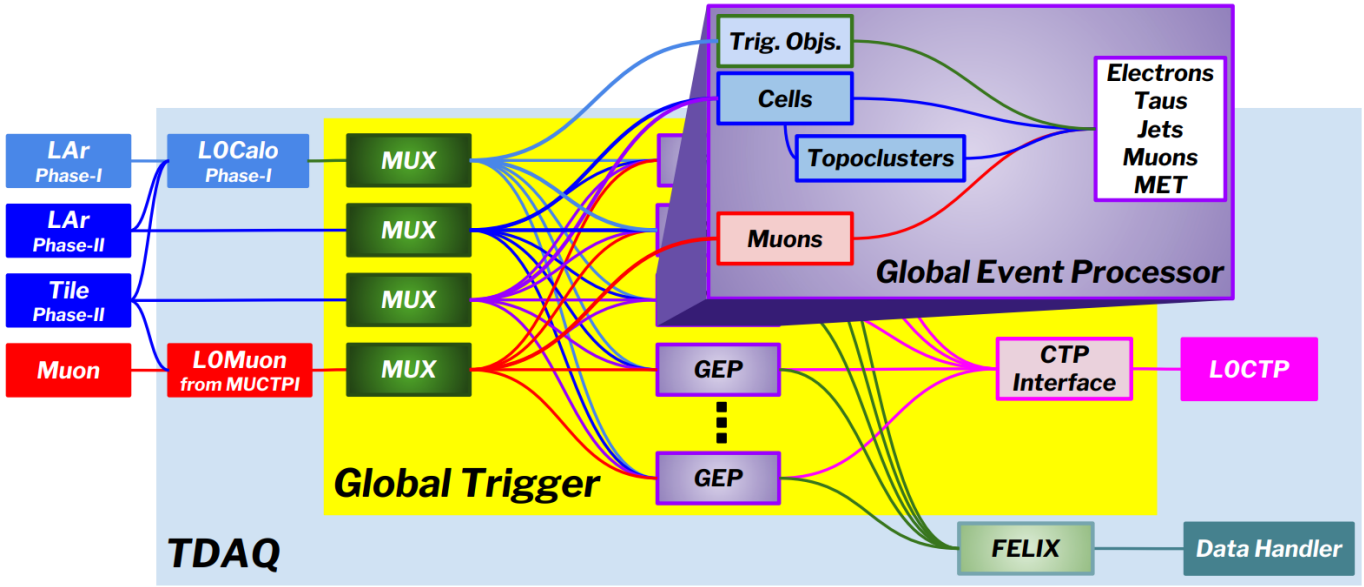


Real-Time Analysis

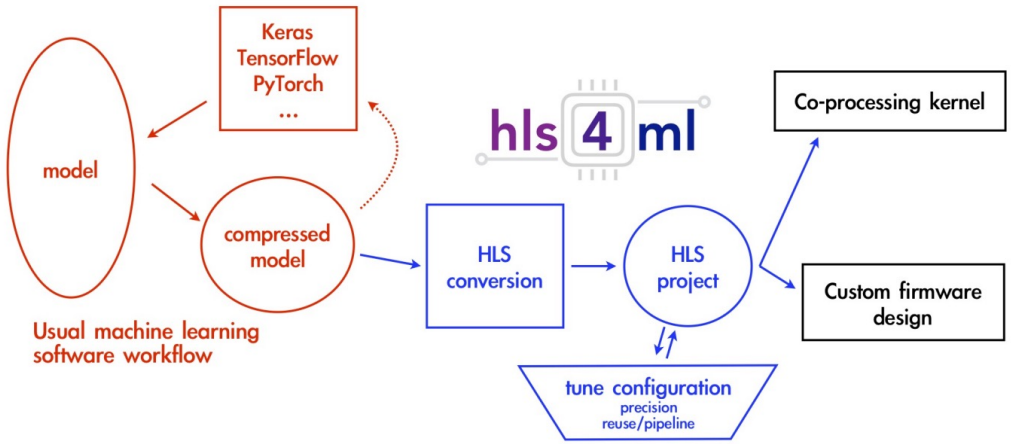
- Collisions occur in the ATLAS experiment at a rate of 40 MHz. It is impossible to record every collision event.
- Instead, ATLAS has a **trigger system** designed to save physically interesting events.
- This trigger system has **two steps** (for HL-LHC):
  - **Level-0 (L0):** Reduction of event rates from 40 MHz → 1 MHz, hardware trigger
  - **High-Level Trigger (HLT):** Reduction of event rates from 1 MHz → 10 kHz, software trigger.
- Already reducing to 2.5% of the event rate at L0!
  - These events must be processed very quickly, O(μs), so complex algorithms not used
- Fast ML is being investigated for applications at L0, in the Global Trigger.

Fast ML for the Global Trigger

- The goal for the Global Trigger is to bring offline-level criteria to the L0 system.
- The system has 48 units accepting data in round robin style, allowing for around a 1.2 μs latency per algorithm.
- It is all implemented on a Field Programmable Gate Array (FPGA), and each algorithm can only use a few percent of the FPGAs available resources.
- Data is pipelined to different algorithm units in a modular approach.
- Goal is to have baseline algorithms which may be improved by ML.
- Complication: any ML algorithm must be programmable onto an FPGA.



Tools for Fast ML: High-Level Synthesis



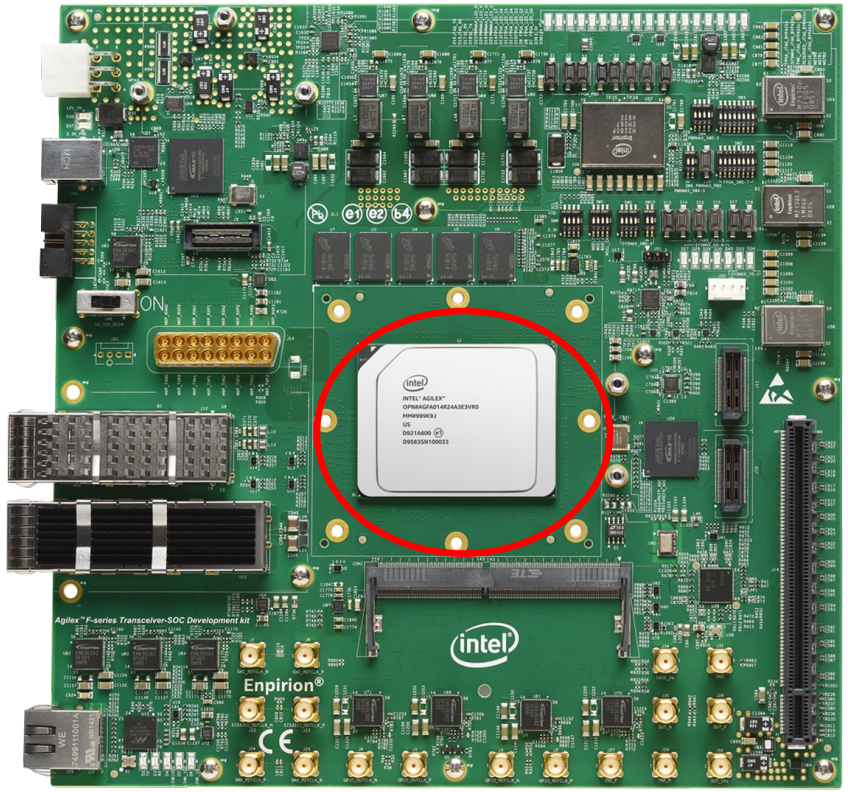
Example of hls4ml workflow

Designing for Field-Programmable Gate Arrays (FPGAs) involves additional complications

- FPGAs are programmed using hardware description languages (HDLs - Verilog and VHDL), not object-oriented software languages like python or C++.
- Most physicists involved ML development are not well-versed in HDL.
  - It is not an easy transition, HDL is fundamentally different from software languages.
- Tools for "high-level synthesis" are often utilized to allow development in software and translation to HDL.
- hls4ml is one such open source tool.
- Other similar tools include fwXmachina and Conifer.

FPGAs: Definition of Terms

- One thing that makes FPGAs useful is that they can be reprogrammed repeatedly, as opposed to other devices where programming is set on fabrication.
- Main vendors: **Intel** and **Xilinx** – similar performance, but each company has its own suite of design software (Intel – Quartus, Xilinx – Vivado).
- Capabilities and available resources for FPGAs continually advance year-by-year.
- Some Basic FPGA Resources:
  - LUT: Look-up Tables
  - DSP: Digital Signal Processors
  - FF: Flip-flops (basic storage element)
  - BRAM: Block Random Access Memory (storing data)



Intel Agilex Development Kit

Some Uses Explored in Global Trigger

- **VBF classifier**
  - To select events with a vector-boson fusion topology, relative to multi-jet backgrounds
  - fwXmachina, BDT
- **Missing Transverse Momentum Regression**
  - Reconstruction of missing transverse momentum, expected from neutrinos
  - fwXmachina, BDT
- **B-tagging algorithm**
  - Multi-class output to define particle origin of jets
  - hls4ml, DNN
- **Quark/gluon jet algorithm**
  - Jet images in eta-phi space are used to distinguish between jets initiated by a quark and a gluon.
  - hls4ml, CNN

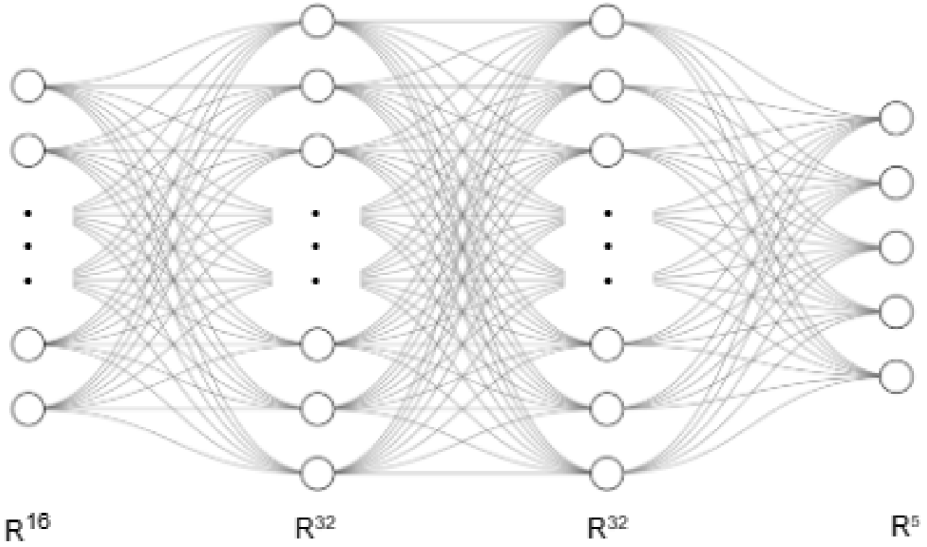


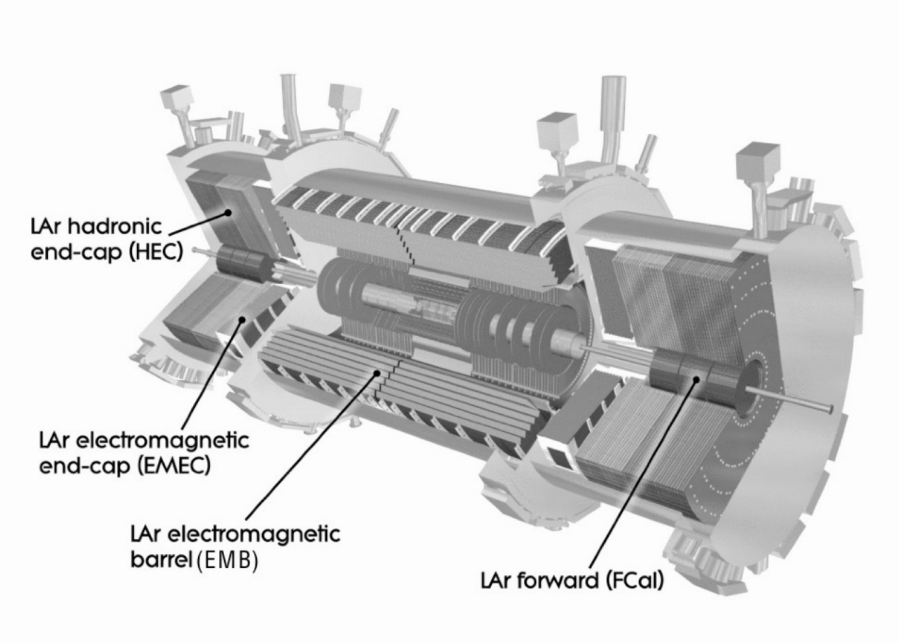
Diagram of B-tagging algorithm

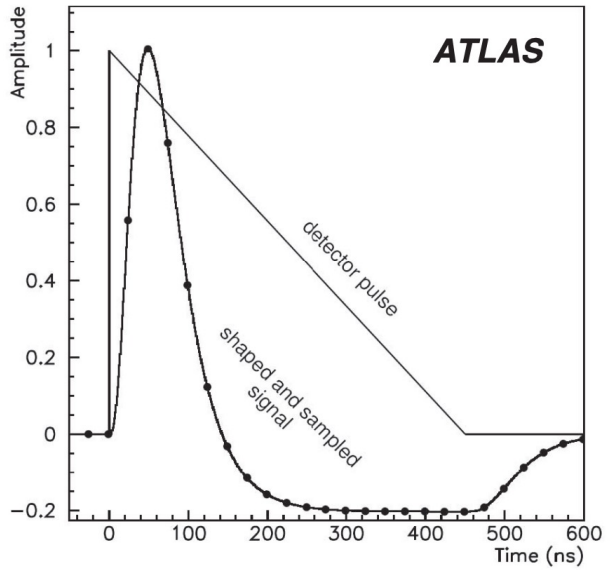
	VBF classifier	MET regression	B-tagging DNN	q/g CNN
Tool	`fwX` (Depth = 4)	`fwX` (Depth = 6)	`hls4ml`	`hls4ml`
Clock	320 MHz	320 MHz	200 MHz	200 MHz
Latency	22 ns	34 ns	50 ns	1.2 us
LUT	0.23%	0.30%	4.6%	0.63%
DSP	0.029%	0.0%	9.1%	4.5%
FF	0.025%	0.084%	0.41%	0.20%
BRAM	2.2%	0.56%	0.83%	0.42%

10.1088/1748-0221/19/05/P05031

Fast ML for the LAr Calorimeter

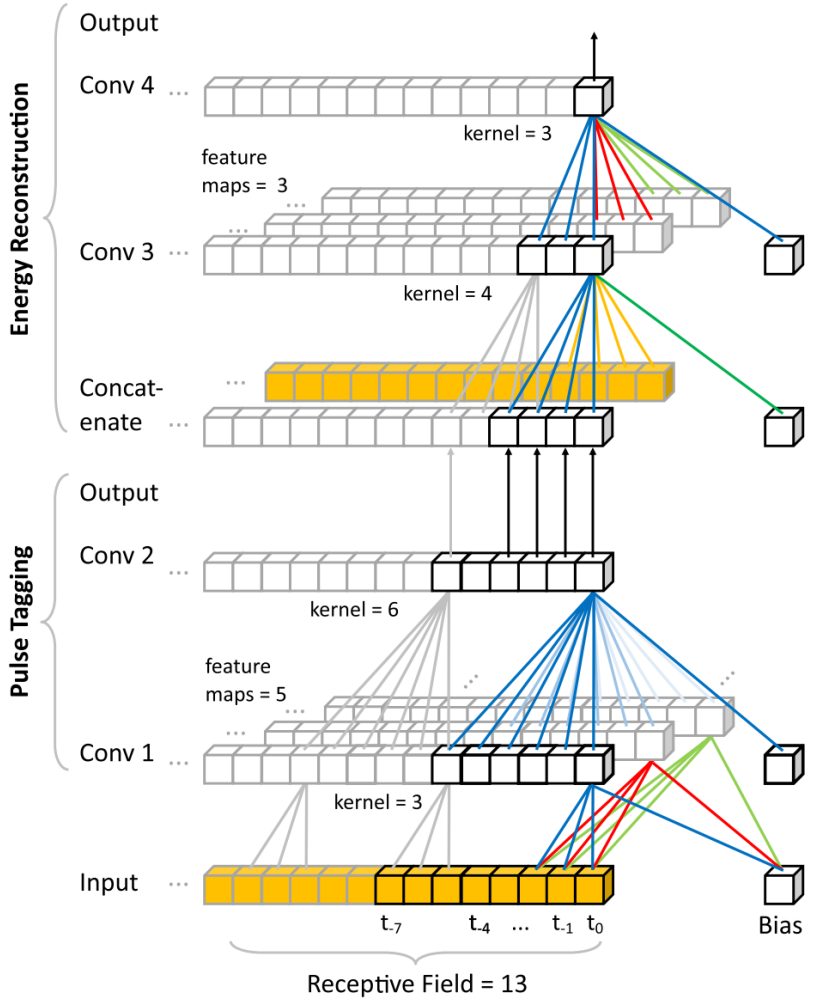
- At the HL-LHC, we expect up to 200 simultaneous collisions per bunch crossing (every 25 ns)
  - Calorimeter pulse signal has a long tail.
- Important information is peak time and amplitude – bipolar pulse shaping performed by an CR – RC^2 analog circuit, so the peak time can be extracted from multiple sampling points.
- Currently, "Optimal Filtering" technique is used to take samples from the digitized pulse and evaluate energy and peak time.
- This assumes a perfect signal, with no overlap – 'out of time pileup' complicates this.





Fast ML for the LAr Calorimeter

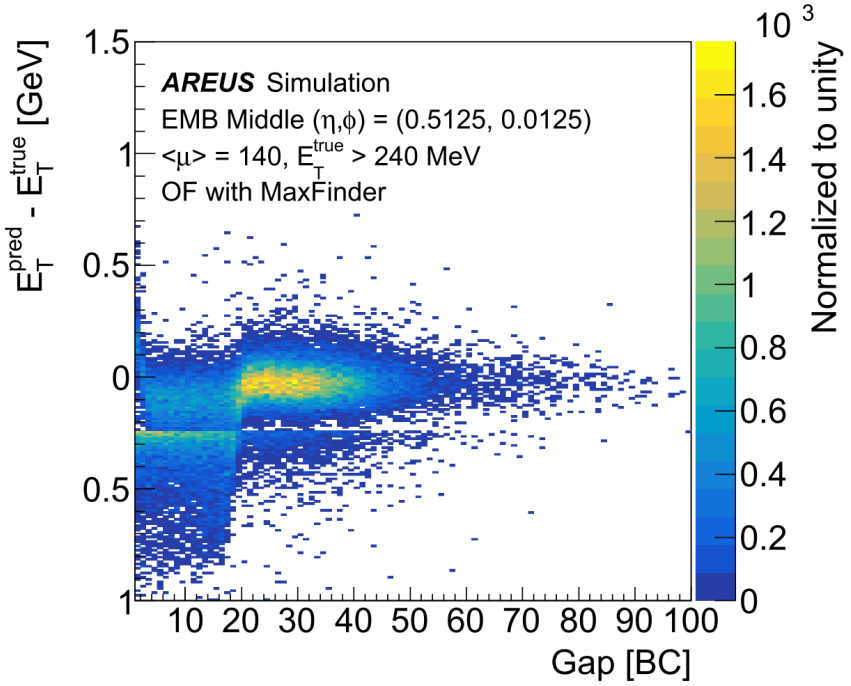
- Machine learning explored to improve performance for overlapping signals.
- In this case, strategy is directly designing neural networks in VHDL, rather than using high-level synthesis.
- Several approaches have been studied, I show here the results for Convolutional Neural Networks (CNNs).
- Example shown for 4-layer CNN:
  - **Input:** Calorimeter energy deposits
  - **Pulse Tagging:** Identifies significant energy deposits over background
  - **Energy Reconstruction:** Detection probability and sample sequence input, reconstructs energy.
  - **Output**: Reconstructed energy.

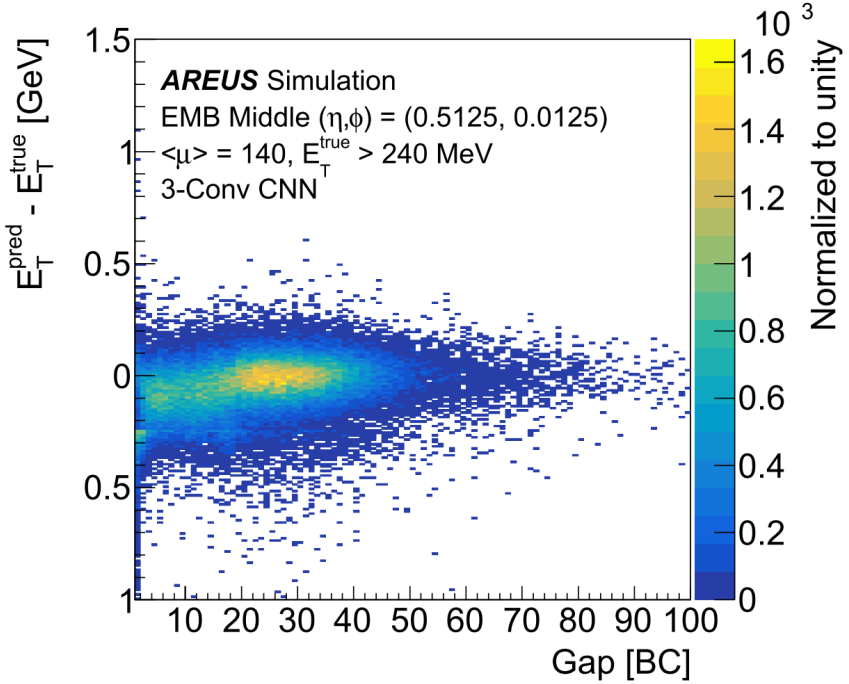


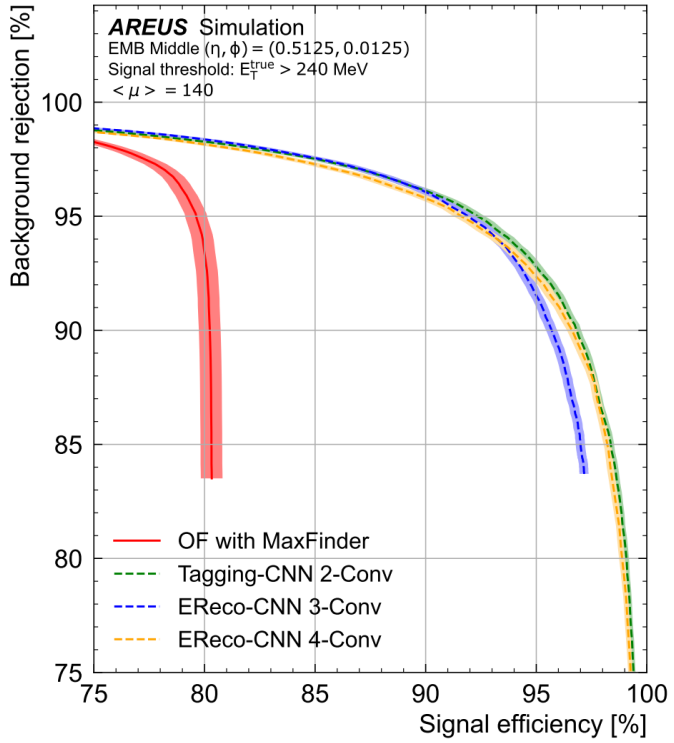
(For this slide and next) Georges Aad et al. Artificial Neural Networks on FPGAs for Real-Time Energy Reconstruction of the ATLAS Lar Calorimeters. In: Computing and Software for Big Science 5.1 (Oct. 2021) DOI: 10.1007/s41781-021-00066-y. URL: https://doi.org/10.1007/s41781-021-00066-y.

Fast ML for the LAr Calorimeter

- **Pulse Tagging**: Use of CNN shows a clear improvement over the Optimal Filter method.
- **Energy Reconstruction**: Use of a CNN shows clear improvement over Optimal Filer method for small ‘gaps’, where pulses have fewer bunch-crossings in between.
- **Design Constraints:** Currently NN implementation fits within FPGA resource requirements and 150 ns latency requirement.







“On-Detector” Uses

- Up until now, we have spoken of “off-detector” electronics, which receive information from the detectors at the LHC (e.g. ATLAS, CMS…) but are not physically located on the detectors themselves.
- Detectors at the LHC are very complex, and part of the job of “on-detector” electronics is to **compress event data for transmission** to off-detector electronics.
  - These electronics are subject to radiation
  - They are typically low-power, radiation hard ASICs (Application-Specific Integrated Circuits)
  - Simple algorithms are used to sacrifice as little physics information as possible.
- **Use of fast ML on ASICs** has been explored by CMS, for instance, to improve readout granularity on the High-Granularity Endcap Calorimeter.
- Unlike FPGAs, ASICs cannot be completely reprogrammed – the firmware would need to be finalized prior to fabrication and could not be changed.
  - Could change NN weights by implementing them as configurable registers.

This is an area that might be of interest for the future!

Summary

- There are many aspects of experimental research at the LHC that can benefit from “fast” machine learning.

- What I have highlighted today are examples of R&D and use of fast ML at LHC experiments, not a comprehensive look at all work that is being performed.

- I encourage people to explore the Fast Machine Learning Lab and associated annual international workshops for a broad look at the use of ML in high energy particle physics as well as many other science domains.

Thank you for your attention! Any questions?

Sources

- Deiana, et al. Applications and Techniques for Fast Machine Learning in Science, Frontiers in Big Data, Vol. 5 (2022), DOI=10.3389/fdata.2022.787421 https://www.frontiersin.org/articles/10.3389/fdata.2022.787421
- Jiang, et al. Machine learning evaluation in the Global Event Processor FPGA for the ATLAS trigger upgrade, Journal of Instrumentation, Vol. 19, 5 (2024), DOI= 10.1088/1748-0221/19/05/P05031, https://dx.doi.org/10.1088/1748-0221/19/05/P05031
- Georges Aad et al. Artificial Neural Networks on FPGAs for Real-Time Energy Reconstruction of the ATLAS Lar Calorimeters. In: Computing and Software for Big Science 5.1 (Oct. 2021) DOI: 10.1007/s41781-021-00066-y. URL: https://doi.org/10.1007/s41781-021-00066-y
- ATLAS HL-LHC Computing Conceptual Design Report, CERN-LHCC-2020-015, https://cds.cern.ch/record/2729668
- Intel, Agilex FPGA: https://www.intel.com/content/www/us/en/products/details/fpga/development-kits/agilex/si-agf014.html