The CMS Detector: Phase 0

**BRIL**
- Pixels Tracker
- ECAL
- HCAL
- Solenoid
- Steel Yoke
- Muons

**Silicon Tracker**
- Pixels (100 x 150 μm²)
  - ~1m²  ~66M channels
- Microstrips (80-180μm)
  - ~200m²  ~9.6M channels

**BRIL**
- Luminosity Telescope: ~200k Si pixels (100 x 150 μm²)
- Beam Monitors: 80 diamond sensors, 40 quartz counters

**Crystal Electromagnetic Calorimeter (ECAL)**
- ~76k scintillating PbWO₄ crystals

**Preshower**
- Silicon strips (6cm x 2mm)
  - ~16m²  ~137k channels

**Steel Return Yoke**
- ~13000 tonnes

**Superconducting Solenoid**
- Niobium-titanium coil carrying ~18000 A

**Hadron Calorimeter (HCAL)**
- Brass + plastic scintillator
  - ~7k channels

**Forward Calorimeter**
- Steel + quartz fibres
  - ~2k channels

**Muon Chambers**
- Barrel: 250 Drift Tube & 480 Resistive Plate Chambers
- Endcaps: 473 Cathode Strip & 432 Resistive Plate Chambers

---

**Total weight**: 14000 tonnes
**Overall diameter**: 15.0 m
**Overall length**: 28.7 m
**Magnetic field**: 3.8 T

---

LPC Topic of the Week

Kevin Pedro
The CMS Detector: Phase 2

Phase 2 Upgrade

1947M

BRIL
Pixels
Tracker
ECAL
HCAL
Solenoid
Steel Yoke
Muons

STEEL RETURN YOKE
~13000 tonnes

SUPERCONDUCTING
Solenoid
Nickel-titanium coil
carrying ~16000 A

HADRON CALORIMETER (HCAL)
Brass + plastic scintillator
~7k channels

CRYSTAL ELECTROMAGNETIC
CALORIMETER (ECAL)
~76k scintillating PbWO₄ crystals

PRESHOWER
Silicon strips (6cm x 2mm)
~16m² ~137k channels

FORWARD CALORIMETER
Steel + quartz fibres
~2k channels

Total weight : 14000 tonnes
Overall diameter : 15.0 m
Overall length : 28.7 m
Magnetic field : 3.8 T

Phase 2 Upgrade

Kevin Pedro
CMS Computing Challenges

Energy frontier: HL-LHC

• 10× data vs. Run 2/3 → exabytes

• 200PU (vs. ~30PU in Run 2)

• CMS:
  o 15× increase in pixel channels
  o 65× increase in calorimeter channels
  o (similar for ATLAS)
Neutrino Computing Challenges

Intensity frontier: DUNE
- Largest liquid argon detector ever designed
- ~1M channels, 1 ms integration time w/ MHz sampling
→ 30+ petabytes/year

CPU needs for particle physics will increase by more than an order of magnitude in the next decade
• CMS does extensive resource projections for HL-LHC
• Expected needs for 2027 (HL-LHC startup):
  o 1,400,000 CPU cores
  o 2.2 exabytes disk, 3 exabytes tape
• Usually project 20%/year “technology improvement” from Moore’s law
  ➢ Still a shortfall of $2–5\times$
• More realistic: 10%/year
  ➢ Shortfall of $4–10\times$
• Try to *thrive*, not just survive

**HSF Community White Paper**

*arXiv:1712.06982*

More at [CMSOfflineComputingResults](https://www.cern.ch/)
The Computing Landscape

- Transistor counts continue to grow
- Clock speed and single-thread performance have stagnated

- **No longer expect traditional CPUs** to keep up with demands from particle physics
- **Coprocessors**: specialized hardware attached to CPU, dedicated to specific tasks

**LPC Topic of the Week**

Kevin Pedro
Development for Coprocessors

- Large speed improvement from hardware accelerated coprocessors
  - Architectures and tools are geared toward **machine learning**

**Option 1**
- **re-write physics algorithms for new hardware**
- Language: OpenCL, OpenMP, HLS, CUDA, …?
- Hardware: FPGA, GPU

**Option 2**
- **re-cast physics problem as machine learning problem**
- Language: C++, Python (TensorFlow, PyTorch,…)
- Hardware: FPGA, GPU, ASIC

**Why (Deep) Machine Learning?**
- Common *language* for solving problems: simulation, reconstruction, analysis!
- Can be universally expressed on optimized computing hardware (follow industry trends)
• ResNet-50: 25M parameters, 7B operations
• Largest network currently used by CMS:
  o DeepAK8, 500K parameters, 15M operations
• Newer approaches w/ larger networks in development…
Top Tagging

- High $p_T$ top quarks are boosted: form a single large-radius jet with substructure
- Top tagging started as simple cuts on high-level variables (right)
- Now advanced to particle-level deep neural networks (next slide)
- (Can also do Higgs tagging, W/Z tagging, etc.)
Top Tagging with Deep Learning

- DNNs outperform simpler ML algorithms (such as BDTs)
- Many approaches now developed
  - Some use much larger networks than DeepAK8: Particle cloud, ResNet-like
- Metrics: AUC, accuracy, $1/\varepsilon_B @ \varepsilon_S=30\%$

Table:

<table>
<thead>
<tr>
<th>Approach</th>
<th>AUC</th>
<th>Acc.</th>
<th>$1/\varepsilon_B (\varepsilon_S=0.3)$</th>
<th>Contact</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoLa</td>
<td>0.980</td>
<td>0.928</td>
<td>680</td>
<td>GK / Simon Leiss</td>
<td>Preliminary number, based on LoLa</td>
</tr>
<tr>
<td>LBN</td>
<td>0.981</td>
<td>0.931</td>
<td>863</td>
<td>Marcel Rieger</td>
<td>Preliminary number</td>
</tr>
<tr>
<td>CNN</td>
<td>0.981</td>
<td>0.93</td>
<td>780</td>
<td>David Shih</td>
<td>Model from Pulling Out All the Tops with Computer Vision and Deep Learning (1803.00107)</td>
</tr>
<tr>
<td>P-CNN (1D CNN)</td>
<td>0.980</td>
<td>0.930</td>
<td>782</td>
<td>Huilin Qu, Loukas Gouskos</td>
<td>Preliminary, use kinematic info only (<a href="https://indico.physics.lbl.gov/indico/event/546/contributions/1270">https://indico.physics.lbl.gov/indico/event/546/contributions/1270</a>)</td>
</tr>
<tr>
<td>6-body N-subjettiness (+mass and pT) NN</td>
<td>0.979</td>
<td>0.922</td>
<td>856</td>
<td>Karl Nordstrom</td>
<td>Based on 1807.04769 (Reports of My Demise Are Greatly Exaggerated: N-subjettiness Taggers Take On Jet Images)</td>
</tr>
<tr>
<td>8-body N-subjettiness (+mass and pT) NN</td>
<td>0.980</td>
<td>0.928</td>
<td>795</td>
<td>Karl Nordstrom</td>
<td>Based on 1807.04769 (Reports of My Demise Are Greatly Exaggerated: N-subjettiness Taggers Take On Jet Images)</td>
</tr>
<tr>
<td>Linear EFPs</td>
<td>0.980</td>
<td>0.932</td>
<td>380</td>
<td>Patrick Komiske, Eric Metodiev</td>
<td>d&lt;= 7, chi &lt;= 3 EFPs with FLD, Based on 1712.07124: Energy Flow Polynomials: A complete linear basis for jet substructure.</td>
</tr>
<tr>
<td>Particle Flow Network (PFN)</td>
<td>0.982</td>
<td>0.932</td>
<td>888</td>
<td>Patrick Komiske, Eric Metodiev</td>
<td>Median over ten trainings. Based on Table 5 in 1810.05165: Energy Flow Networks: Deep Sets for Particle Jets.</td>
</tr>
<tr>
<td>Energy Flow Network (EFN)</td>
<td>0.979</td>
<td>0.927</td>
<td>619</td>
<td>Patrick Komiske, Eric Metodiev</td>
<td>Median over ten trainings. Based on Table 5 in 1810.05165: Energy Flow Networks: Deep Sets for Particle Jets.</td>
</tr>
<tr>
<td>2D CNN [ResNeXt50]</td>
<td>0.984</td>
<td>0.936</td>
<td>1086</td>
<td>Huilin Qu, Loukas Gouskos</td>
<td>Preliminary from indico.cern.ch/event/745718/contributions/3202526</td>
</tr>
<tr>
<td>DGCNN</td>
<td>0.984</td>
<td>0.937</td>
<td>1160</td>
<td>Huilin Qu, Loukas Gouskos</td>
<td>Preliminary from indico.cern.ch/event/745718/contributions/3202526</td>
</tr>
</tbody>
</table>

G. Kasieczka
Deep learning in industry focuses on image recognition.

Jets are not images, but can pretend in order to test industry networks.

Convert jets into images using constituent $p_T$, $\eta$, $\varphi$: 224x224 pixels.

Standardized top quark tagging dataset is publicly available: https://goo.gl/XGYju3
Top Tagging with ResNet-50

- Retrain ResNet-50 on publicly available top quark tagging dataset
  → New set of weights, optimized for physics
    - Add custom classifier layers to interpret features from ResNet-50

- ResNet-50 model that runs on FPGAs is “quantized”
  - Tune weights to achieve similar performance
    ➢ State-of-the-art results vs. other leading algorithms
Beyond Tagging

- R&D to use novel network architectures for tracking and clustering
- Optimal use of complex new detectors
- May be crucial for Phase 2

HEP.TrkX
Message Passing Neural Network
Graph-based ML for tracking

MPNN for HGCal
Exploit hexagonal cells & 3D structure

J. Kieseler

(GravNet)

Fermilab
LDRD
L2019.017
NOvA was the first particle physics experiment to publish a result obtained using a CNN

- Achieved similar efficiency for $\nu_\mu$ (58% vs. 57%), improvement for $\nu_e$ (49% vs. 35%) vs. existing algorithms (arXiv:1604.01444)

- Used to constrain oscillation parameters: (arXiv:1703.03328)
  - Inverted mass hierarchy w/ $\theta_{23}$ in lower octant excluded at 93% CL for all $\delta_{CP}$ values
Neutrino Recognition with ResNet-50

- ResNet-50 can also classify neutrino events to reject cosmic ray backgrounds
- Use *transfer learning*: keep default featurizer weights, retrain classifier layers
- Events above selected w/ probability > 0.9 in different categories
- CNN inference already a large fraction of neutrino reconstruction time
  ➢ Prime candidate for acceleration with coprocessors
Why Accelerate Inference?

• Training is viewed as “hard” part of machine learning

• But…

• DNN training happens ~once/year/algorithm
  
  o Cloud GPUs or new HPCs are good options

• Once DNN is in common use, inference will happen billions of times
  
  o MC production, analysis, prompt reconstruction, high level trigger…

➢ Training is done by developers, inference is done by everyone
Inference as a Service

• Inference as a service:
  o Minimize disruption to existing computing model
  o Minimize dependence on specific hardware
  o Avoid need for large input batches
    ▪ Large batches are necessary to use GPUs efficiently
    ▪ Not a good fit for most particle physics applications: prefer to load one (complex) event into memory, then run all algorithms on it

• Performance metrics:
  o Latency (time for a single request to complete)
  o Throughput (number of requests per unit time)
What are FPGAs?

- Field Programmable Gate Arrays
- “Programmable hardware”: arrange gates to perform desired task
- Execute many instructions in parallel on input data: spatial computing
  - vs. CPUs, which execute instructions in series: temporal computing
- Can be reconfigured in 100ms–1s
- Ideal for deep networks: many operations, fixed model parameters
- ASICs (Application Specific Integrated Circuits) not configurable – require stable problem to solve
Coprocessors: An Industry Trend

Specialized coprocessor hardware for machine learning inference

LPC Topic of the Week

Kevin Pedro
Microsoft Brainwave

- Provides a full service at scale (more than just a single co-processor)
- Multi-FPGA/CPU fabric accelerates both computing and network
- Weight retuning available: retrain supported networks to optimize for a different problem

Brainwave supports:
- ResNet50
- ResNet152
- DenseNet121
- VGGNet16
**Event-based processing**

- Events are very complex with hundreds of products
- Load one event into memory, then execute all algorithms on it

- Most applications not a good fit for large batches, which are required for best GPU performance
Accessing Heterogeneous Resources

- New **CMSSW** feature called **ExternalWork**:
  - Asynchronous task-based processing
  - Non-blocking: schedule other tasks while waiting for external processing
- Can be used with GPUs, FPGAs, cloud, …
  - Even other software running on CPU that wants to schedule its own tasks
  - Now demonstrated to work with Microsoft Brainwave!
SONIC in CMSSW

• Services for Optimized Network Inference on Coprocessors
  o Convert experimental data into neural network input
  o Send neural network input to coprocessor using communication protocol
  o Use ExternalWork mechanism for asynchronous requests
• Currently supports:
  o gRPC communication protocol
    ▪ Callback interface for C++ API in development
      → wait for return in lightweight std::thread
  o TensorFlow w/ inputs sent as TensorProto (protobuf)
• Tested w/ Microsoft Brainwave service (cloud FPGAs)
• gRPC SonicCMS repository on GitHub
Cloud vs. Edge

- Cloud service has latency
- Run CMSSW on Azure cloud machine → simulate local installation of FPGAs ("on-prem" or "edge")
- Provides test of ultimate performance
- Use gRPC protocol either way

LPC Topic of the Week
Kevin Pedro
SONIC Latency

- Remote: cmslpc @ FNAL to Azure (VA), \( \langle \text{time} \rangle = 60 \text{ ms} \)
  - Highly dependent on network conditions
- On-prem: run CMSSW on Azure VM, \( \langle \text{time} \rangle = 10 \text{ ms} \)
  - FPGA: 1.8 ms for inference
  - Remaining time used for classifying and I/O
• Run N simultaneous processes, all sending requests to 1 BrainWave service
• Processes only run JetImageProducer from SONIC → “worst case” scenario
  o Standard reconstruction process would have many non-SONIC modules
• FPGA performs inference serially (1 image at a time)
SONIC Latency: Scaling

mean ± std. dev.

• Tests: $N = 1, 10, 50, 100, 500$

• Only moderate increases in mean, standard deviation, and long tail for latency
  
  ○ Fairly stable up to $N = 50$
• Each process evaluates 5000 jet images in series
• Remarkably consistent total time for each process to complete
  o Brainwave load balancer works well
• Compute inferences per second as \(\frac{5000 \cdot N}{\text{total time}}\)
• \(N = 50\) ~fully occupies FPGA:
  o Throughput up to 600 inferences per second (max ~650)
CPU Performance

- Above plots use i7 3.6 GHz, TensorFlow v1.10
- Local test with CMSSW on cluster @ FNAL:
  - Xeon 2.6 GHz, TensorFlow v1.06
  - 5 min to import Brainwave version of ResNet-50
  - 1.75 sec/inference subsequently
• Above plots use NVidia GTX 1080, TensorFlow v1.10
• GPU directly connected to CPU via PCIe
• TF built-in version of ResNet-50 performs better on GPU than quantized version used in Brainwave
Performance Comparisons

<table>
<thead>
<tr>
<th>Type</th>
<th>Note</th>
<th>Latency [ms]</th>
<th>Throughput [img/s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU*</td>
<td>Xeon 2.6 GHz</td>
<td>1750</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>i7 3.6 GHz</td>
<td>500</td>
<td>2</td>
</tr>
<tr>
<td>GPU**</td>
<td>batch = 1</td>
<td>7</td>
<td>143</td>
</tr>
<tr>
<td></td>
<td>batch = 32</td>
<td>1.5</td>
<td>667</td>
</tr>
<tr>
<td>Brainwave</td>
<td>remote</td>
<td>60</td>
<td>660</td>
</tr>
<tr>
<td></td>
<td>on-prem</td>
<td>10 (1.8 on FPGA)</td>
<td>660</td>
</tr>
</tbody>
</table>

*CPU performance depends on:
- clock speed, TensorFlow version, # threads (=1 here)

**GPU caveats:
- Directly connected to CPU via PCIe – not a service
- Performance depends on batch size & optimization of ResNet-50 network

SONIC achieves:
- 175× (30×) on-prem (remote) improvement in latency vs. CMSSW CPU!
- Competitive throughput vs. GPU, w/ single-image batch as a service!
Summary

• Particle physics experiments face extreme computing challenges
  o More data, more complex detectors, more pileup

• Growing interest in machine learning for reconstruction and analysis
  o As networks get larger, inference takes longer

• FPGAs are a promising option to accelerate neural network inference
  o Can achieve order of magnitude improvement in latency over CPU
  o Comparable throughput to GPU, without batching
    ➢ Better fit for event-based computing model

• SONIC infrastructure developed and tested
  o Compatible with any service that uses gRPC and TensorFlow

➢ Paper with these results in preparation

• Thanks to Microsoft for lots of help and advice!
  o Azure Machine Learning, Bing, Project Brainwave teams
  o Doug Burger, Eric Chung, Jeremy Fowers, Kalin Ovtcharov, Andrew Putnam
Continuing Work

• Continue to translate particle physics algorithms into machine learning
  o Easier to accelerate inference w/ commercial coprocessors

• Develop tools for generic model translation
  o E.g. graph NNs used for HEP.TrkX and other projects

• Explore broad offering of potential hardware
  o Google TPUs, Xilinx ML suite on AWS, Intel OpenVINO, …

• Continue to build infrastructure and study scalability/cost
  o Adapt SONIC to handle other protocols, other network architectures and ML libraries, other experiments (e.g. neutrinos)
A single FPGA can support many CPUs → cost-effective

- SONIC throughput results indicate 1 FPGA for 100–1000 CPUs running realistic processes (many algorithms, only some ML inferences)

- Install small “edge” instances at T1s and T2s
  - Can also install a dedicated instance for CMS HLT farm at CERN
Backup
Jet Substructure

- $u,d$ or $s$ jet
- $c$ or $b$ jet
- Gluon jet
- Pileup jet
- $W$ or $Z$ jet
- Higgs jet
- Top jet

LPC Topic of the Week
Kevin Pedro
External Work in CMSSW (1)

Setup:

- TBB controls running modules
- Concurrent processing of multiple events
- Separate helper thread to control external
- Can wait until enough work is buffered before running external process
External Work in CMSSW (2)

Acquire:

- Module *acquire()* method called
- Pulls data from event
- Copies data to buffer
- Buffer includes callback to start next phase of module running

![Diagram showing the process of acquiring data in CMSSW](image)
External Work in CMSSW (3)

Work starts:

- External process runs
- Data pulled from buffer
- Next waiting modules can run (concurrently)
Work finishes:

• Results copied to buffer
• Callback puts module back into queue
Produce:

- Module *produce()* method is called
- Pulls results from buffer
- Data used to create objects to put into event