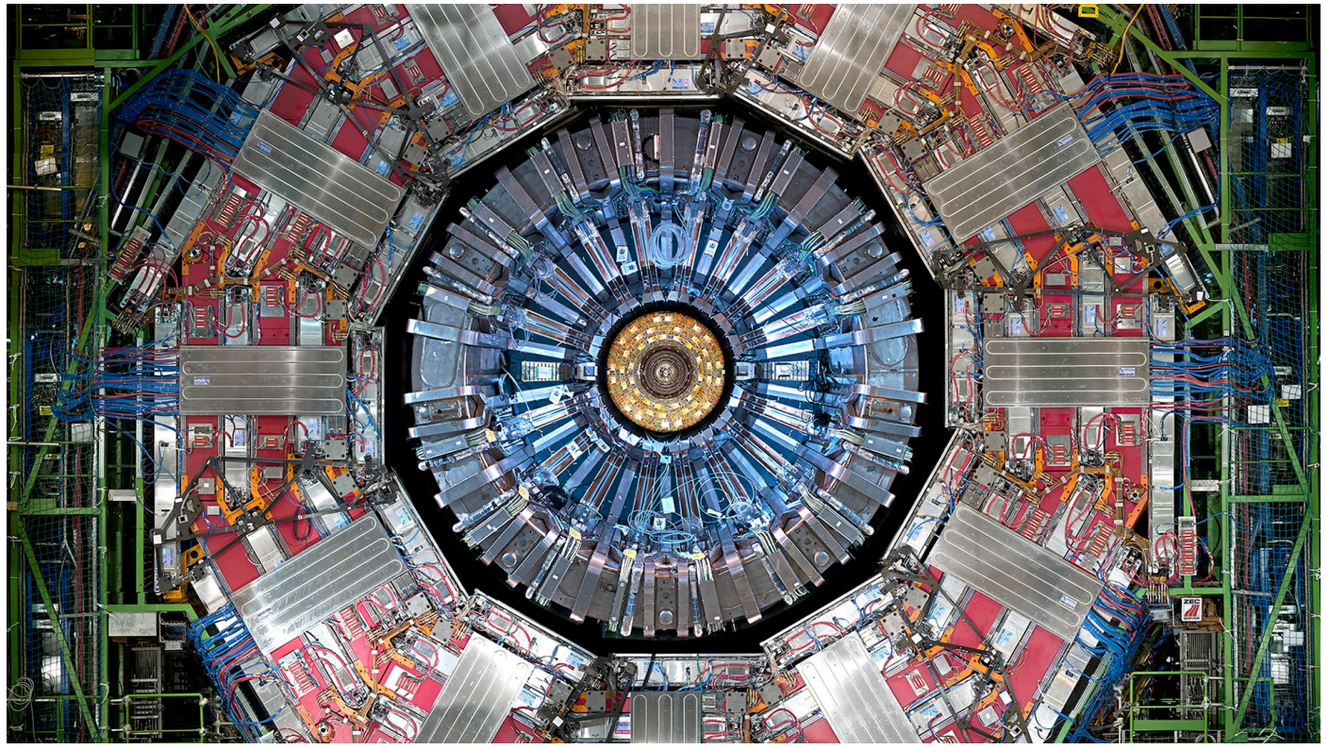
Services for Optimized Network Inference on Co-processors

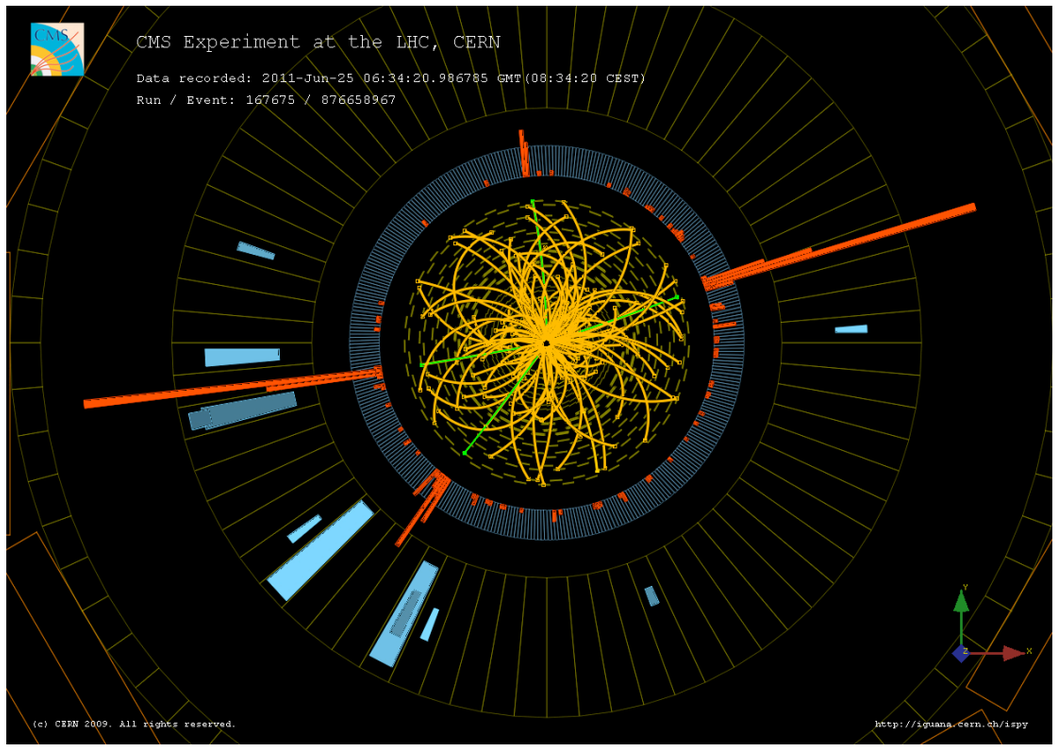
A3D3 kick-off meeting, November 9

Philip Harris, **Jeffrey Krupa**, Zhijian Liu, Patrick McCormack, Dylan Rankin, Simon Rothman (MIT)
Maria Acosta Flechas, Tejin Cai, Yongbin Feng, Ken Herner, Burt Holzman, Kevin Pedro, Steven Timm, Nhan Tran, Michael Wang, Tingjun Yang (FNAL)
Miaoyuan Liu, Stefan Piperov (Purdue)
Javier Duarte (UCSD)
Chun-Yu Lin (NCHC)
Shih-Chieh Hsu, Elham Khoda, Alex Schuy (UW)

CMS detector

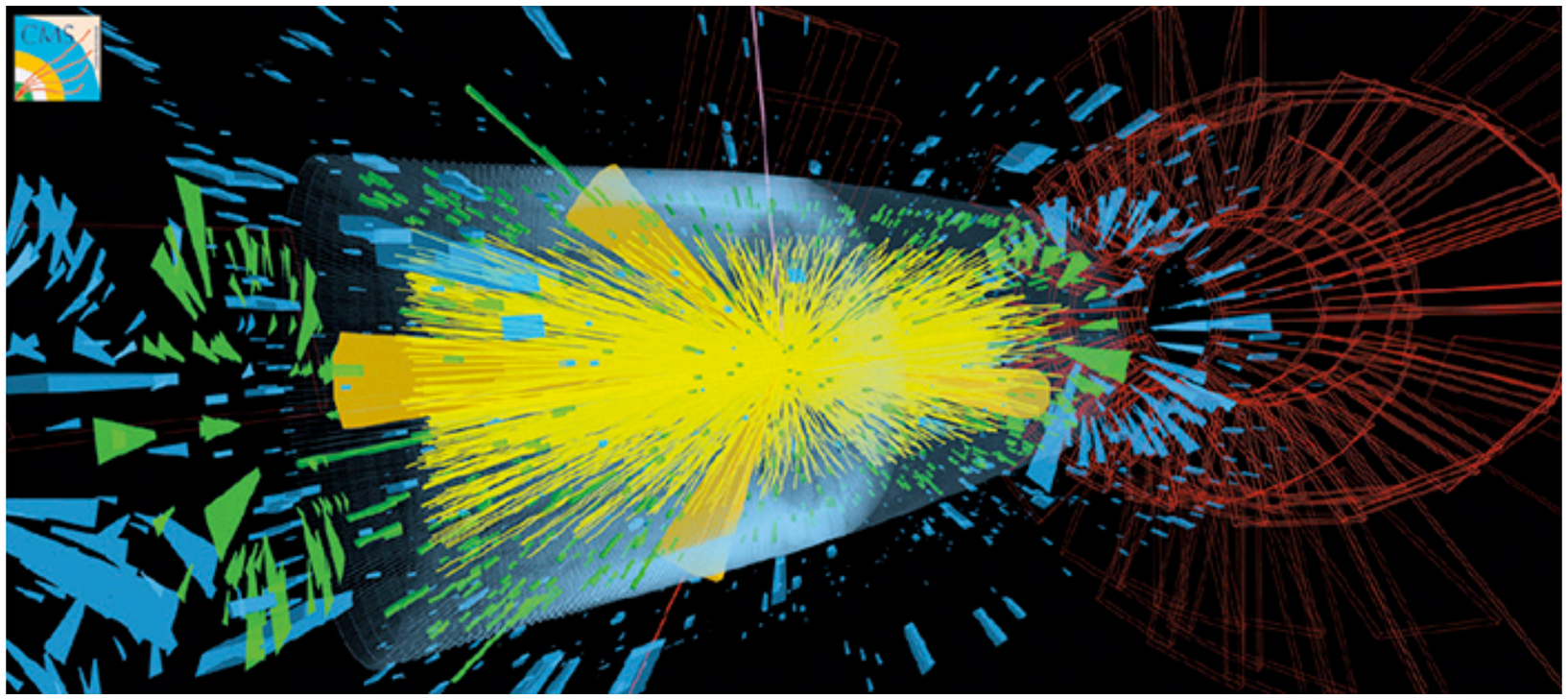
- Tens of millions of channels read @ 40 MHz
- Two tiers of trigger keep only the most interesting collisions (~1 in 40,000)
- These events are fully reconstructed and written to disk
  - LHC experiments have collected ~exabytes of data, processed over ~600k global CPU cores





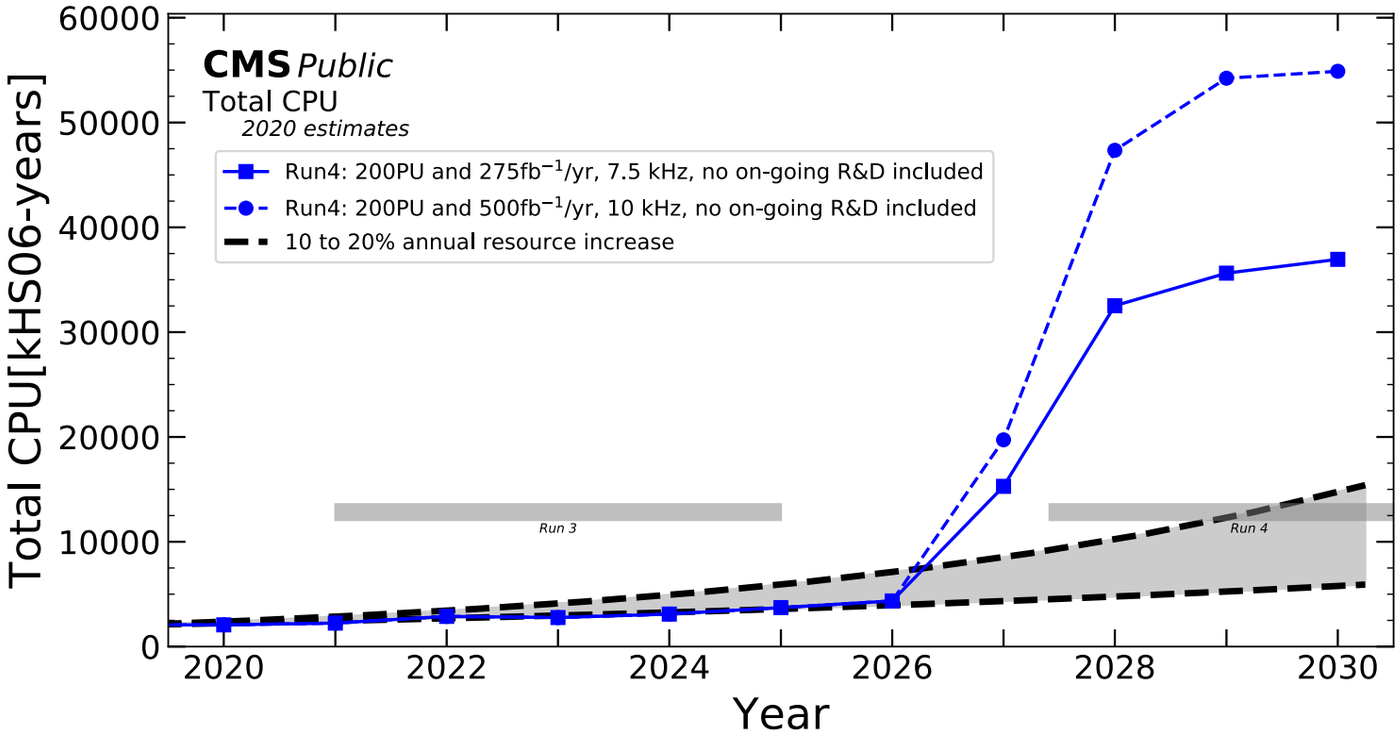
Physics challenge

- The future presents significant challenges and opportunities
- Large effort underway to re-vamp algorithms at all levels
  1. physics gains: what can we learn from more data?
  2. computing: how can we best leverage co-processors?



Computing challenge

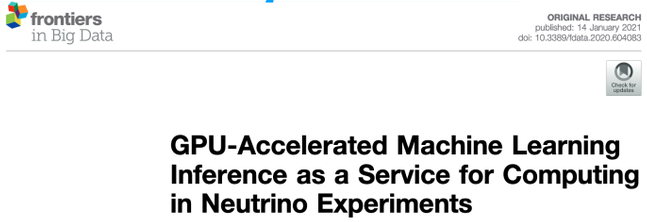
- Computing at LHC experiments will **outpace growth in CPUs**
    1. more segmented detectors with more channels
    2. more collisions, higher trigger rate
    3. high-latency algorithms



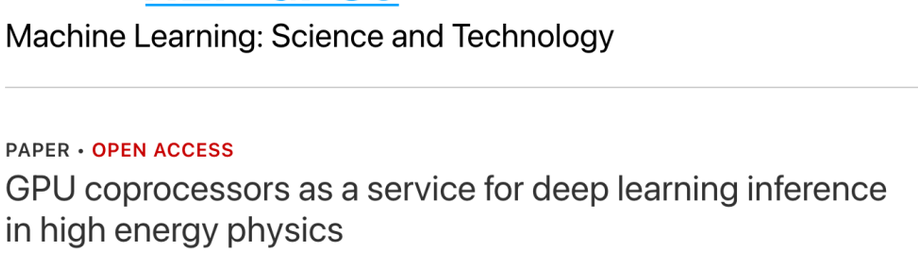
Co-processors

- GPUs and other co-processors will play a roll in solving this
  - We focus on *as-a-service* deployment
- Our work is to develop algorithms for HEP applications and figure out how to scale them to production level
- A number of promising results have already been released

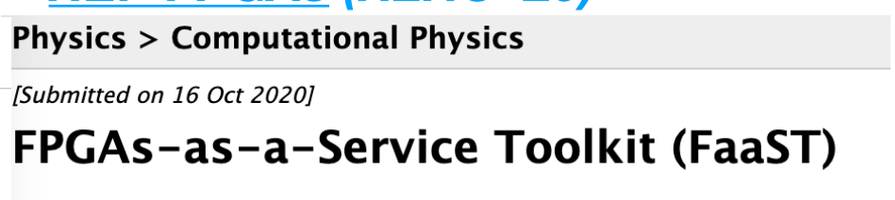
Neutrino experiments



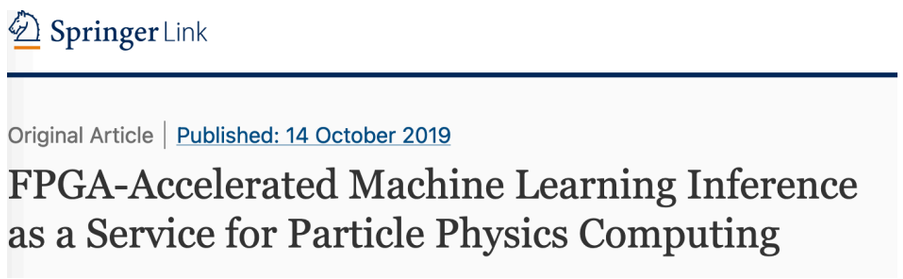
HEP GPUs



HEP FPGAs (H2RC '20)



HEP FPGAs ResNet-50

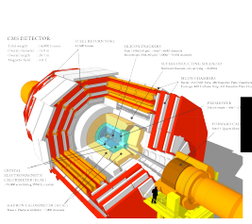
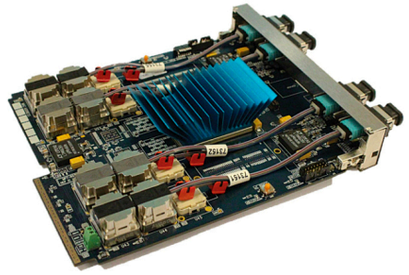
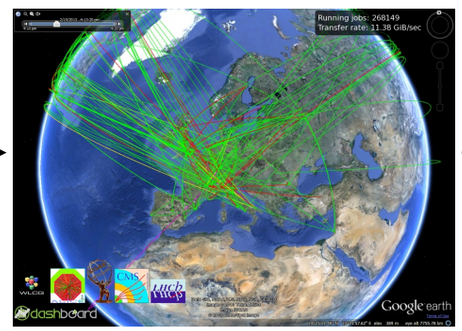
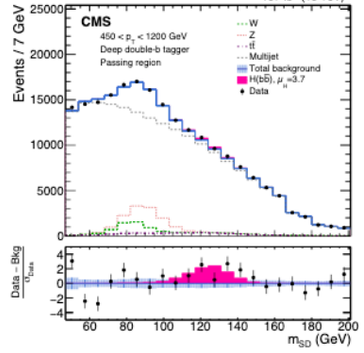


Stay tuned for more!

Data flow

40 MHz → **1 kHz**

100s Tb/s → 10 Gb/s

Radiation Hard ASICs	Level 1 Trigger	High Level Trigger	Offline reconstruction	Analysis
				
10 µs	<500 ms	~10 s		
Local FPGA boards	***30k local CPU cores***	***100k+ world-wide CPU cores***		

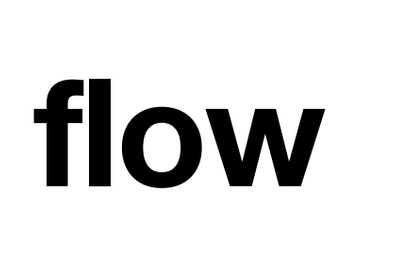
Data flow

40 MHz → **1 kHz**

100s Tb/s → 10 Gb/s

Radiation Hard ASICs

Level 1 Trigger



10 µs

Local FPGA boards

High Level Trigger

<500 ms

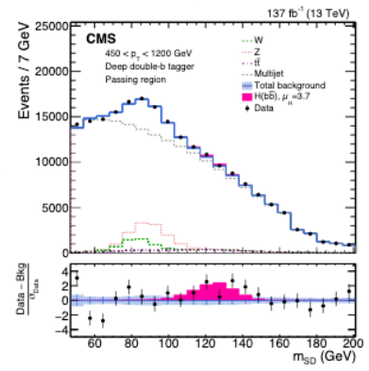
30k local CPU cores

Offline reconstruction

~10 s

100k+ world-wide CPU cores

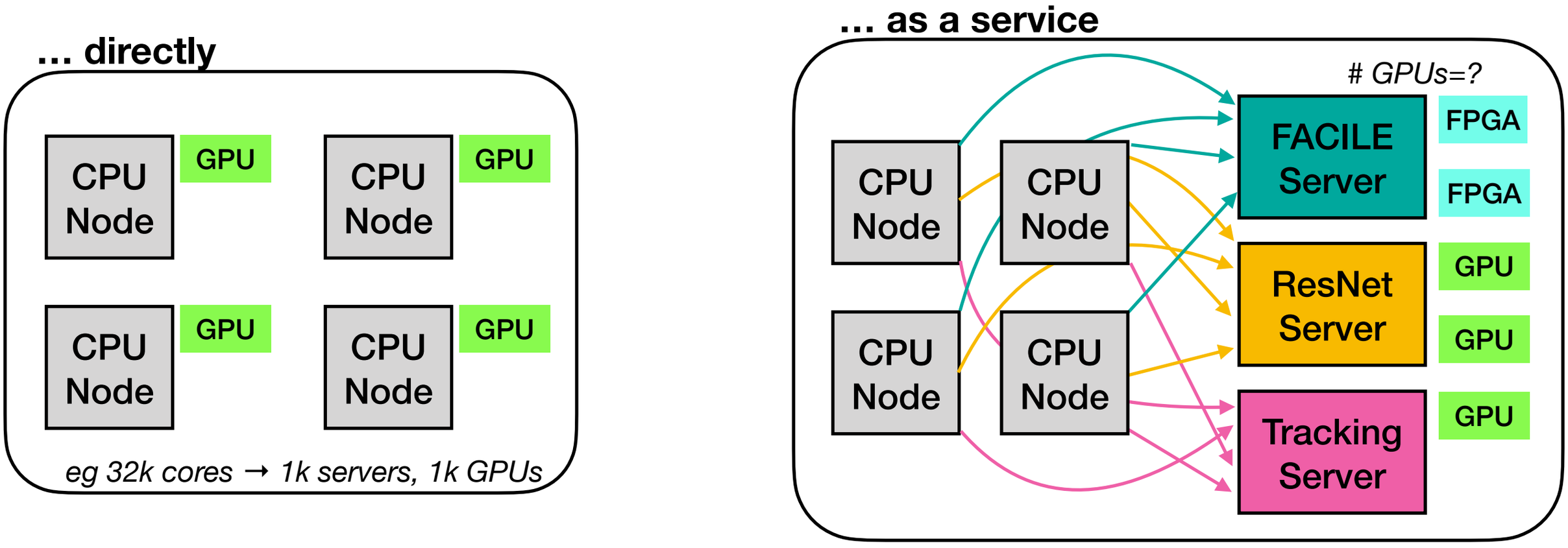
Analysis



DL (+GPUs) is often done on a user-specific basis

This work focuses on introducing algos with heterogeneity in data taking+reco

Talking to GPUs



Communicating with coprocessors as a service:

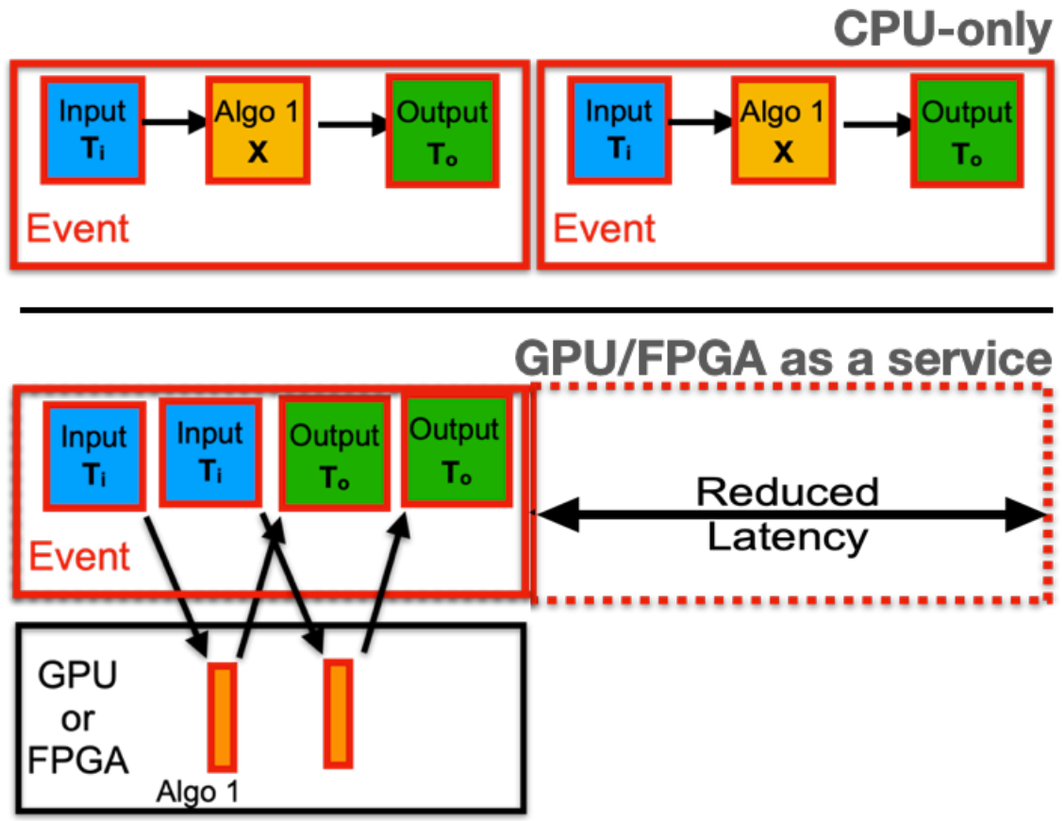
1. Integrates coprocessors within existing experimental software frameworks
2. Removes burden of coprocessor/algorithm-specific code
3. Heterogeneous friendly
   - Can flexibly configure coprocessor type, number of coprocessors per server, …
   - Many coprocessors to choose from
4. Leverages highly optimized inference tools developed by industry

Considerations: added network load, load balancer, sufficient algorithm speedup

SONIC

Services for Optimized Network Inference on Coprocessors

- Integrates as-a-service requests into experimental workflows
- Formats event data for algorithm input
- Makes non-blocking, asynchronous requests
  - Thread is free to do additional processing
- Works with any coprocessor
- Integrated into CMS software
  - Containerizes ML frameworks



NVIDIA TRITON INFERENCE SERVER

hls4ml

XILINX VITIS™

XILINX Xilinx ML Suite

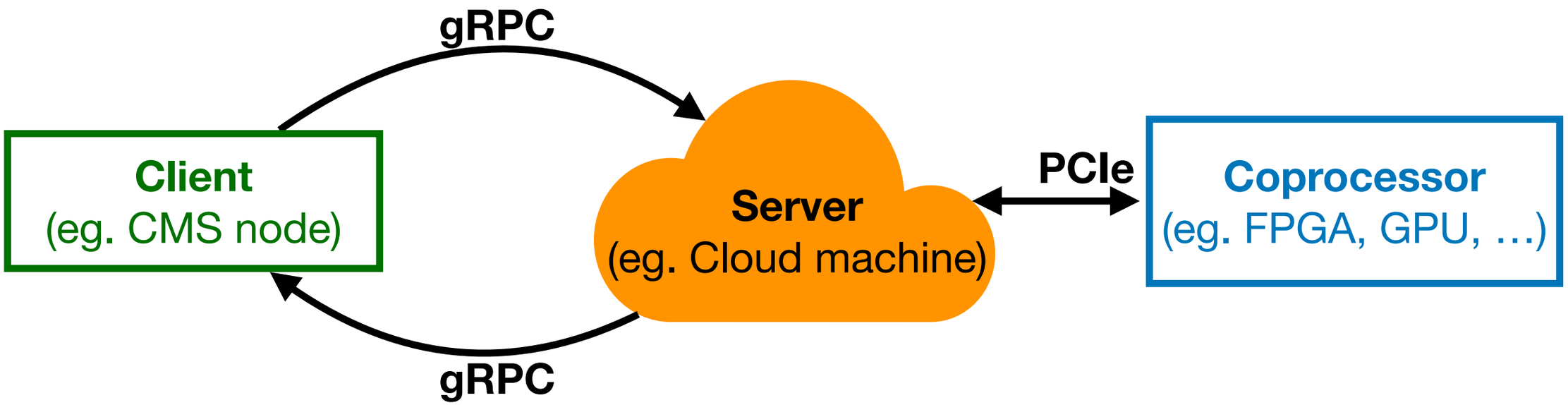
Microsoft Azure

aws

SONIC

Services for Optimized Network Inference on Coprocessors

- For fast inference we focus on remote procedure call (gRPC) protocol
- Triton inference server for inference on NVIDIA GPUs
- Developed custom FPGAs-as-a-Service Toolkit (FaaST) for FPGA



1. Client formats inputs and sends call to server

2. Server receives request and schedules inference on coprocessors

3. Coprocessor runs the inference and returns the result

Tools

NVIDIA

Use NVIDIA triton inference server for GPU + Customized GCP Kubernetes

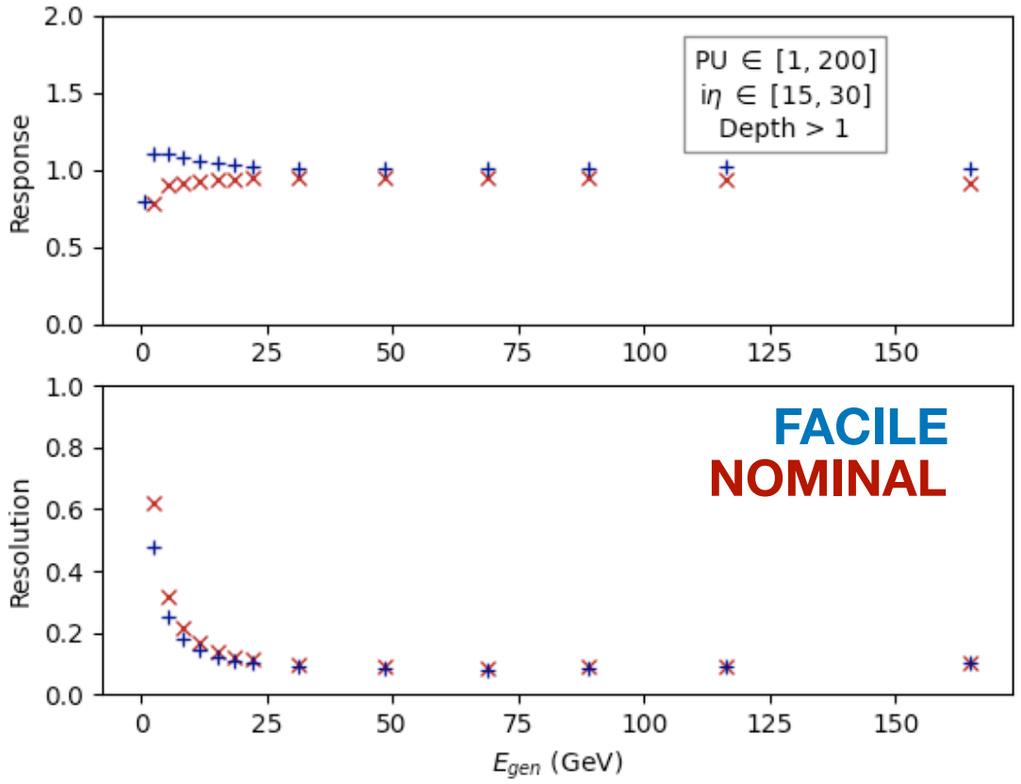
hls4ml

XILINX
ALL PROGRAMMABLE™

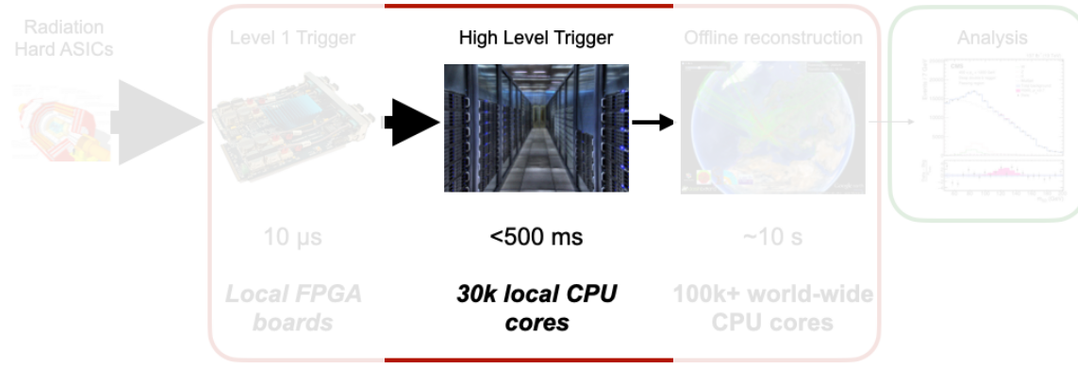
Wrote our own FPGA gRPC inference server

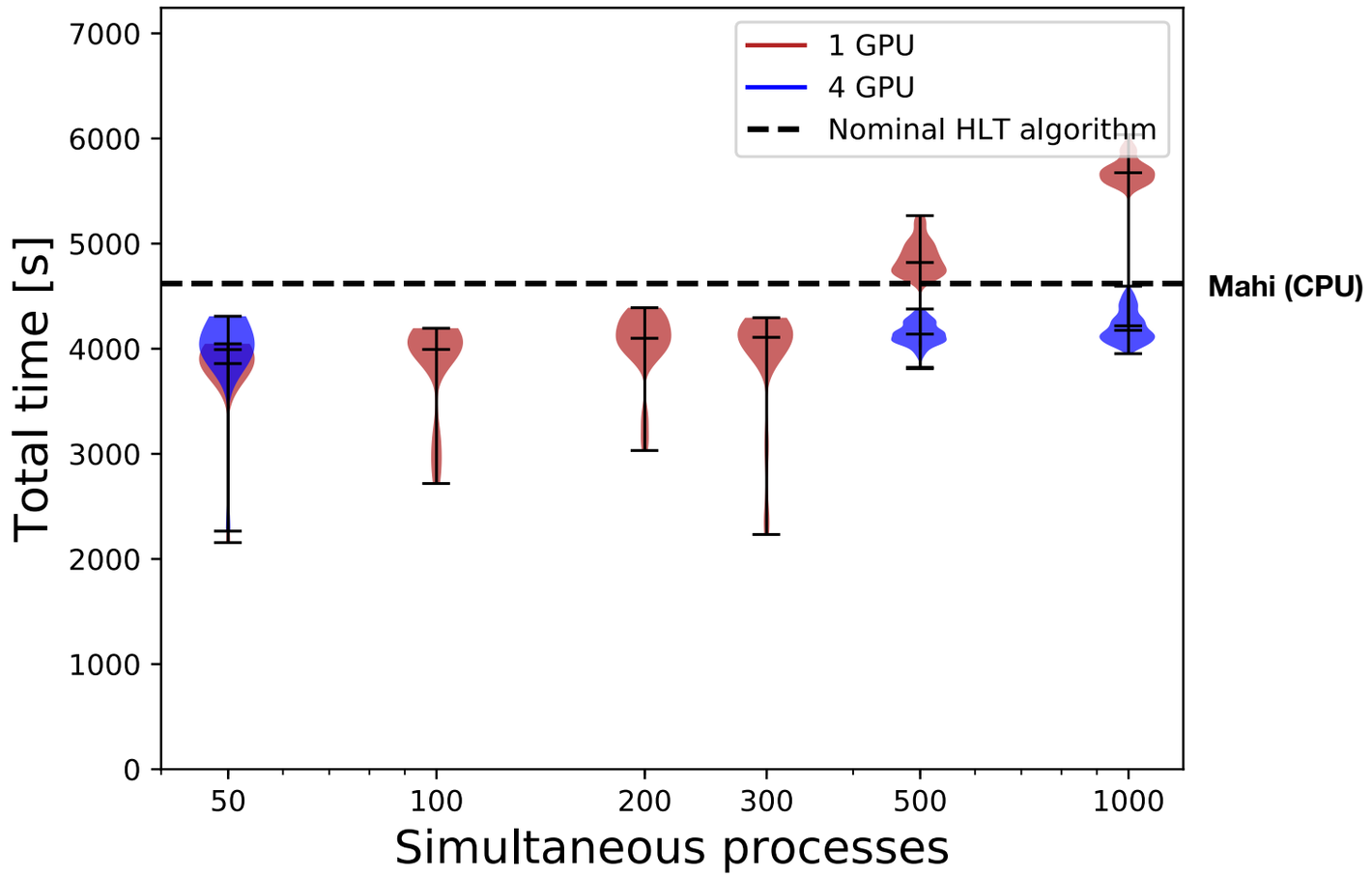
HCAL algorithm

- FACILE: Fast Calorimeter Learning
- Performant (especially at low energy)
- 2 ms GPU latency
  - Nominal algorithm takes 60 ms (10% of online budget)



HLT scaling test



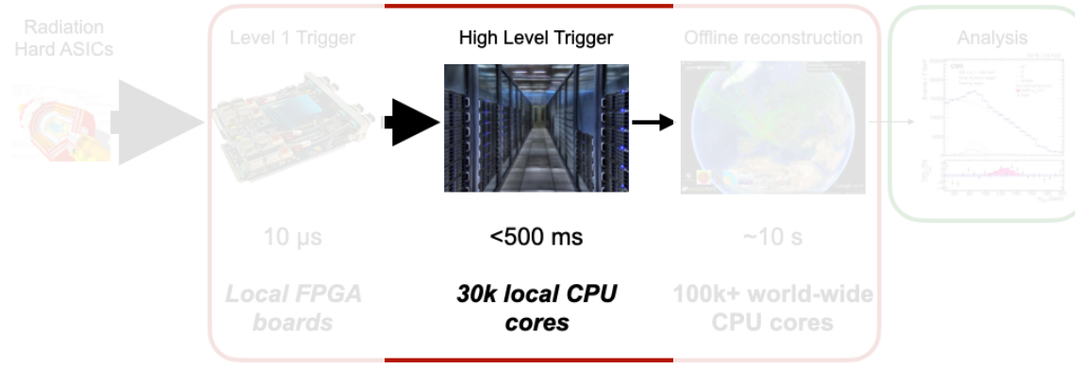


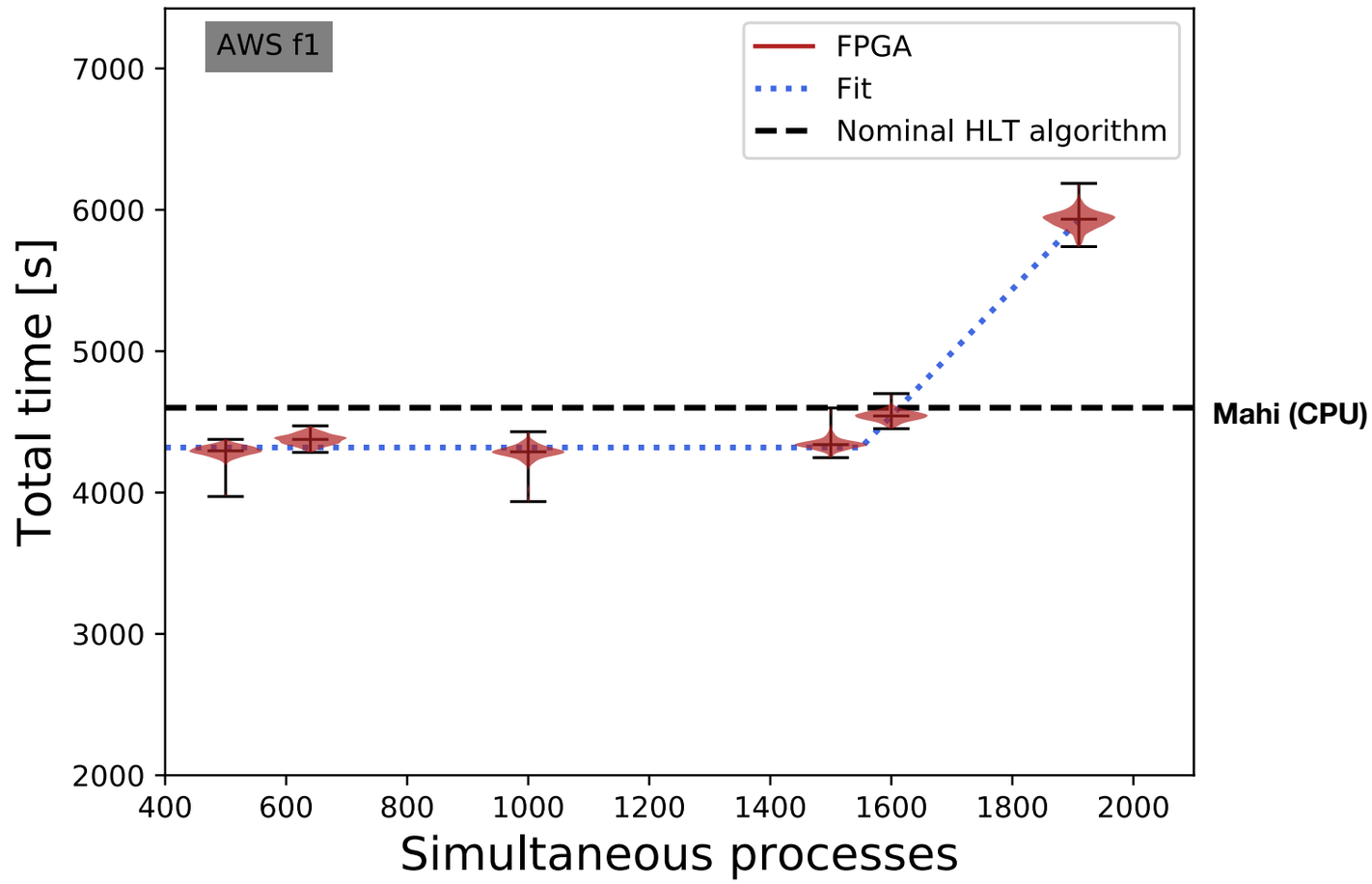
We deployed up to 2k High Level Trigger jobs in Google Cloud

1. 10% reduction in HLT time with HCAL reconstruction latency
2. No increase in latency until ≥300 HLT instances (GPU)

We are currently planning a much larger test

HLT scaling test

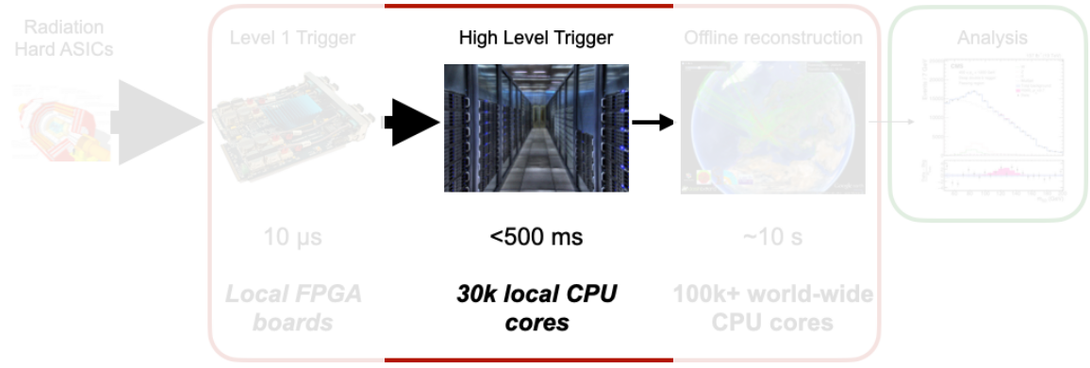




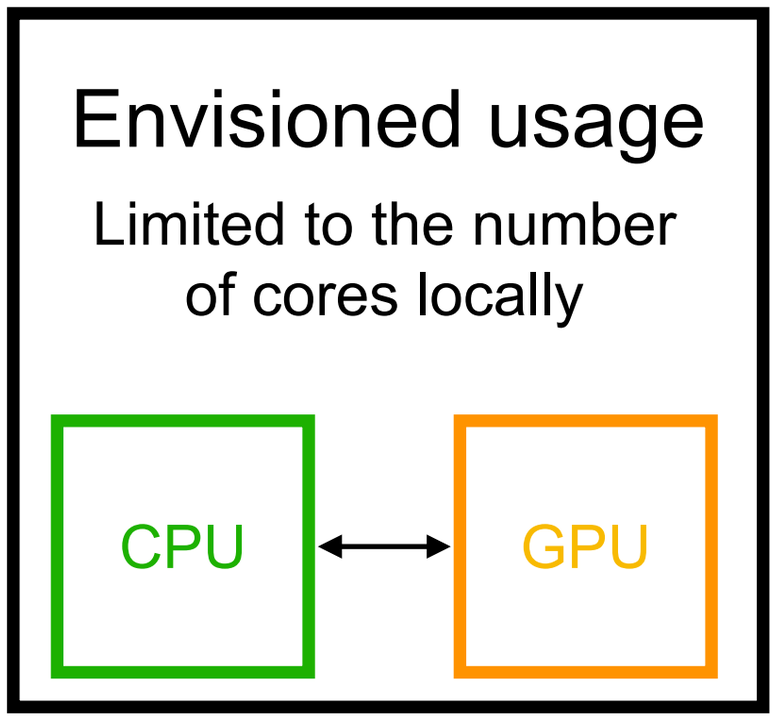
We ran a similar test on FPGAs in the cloud

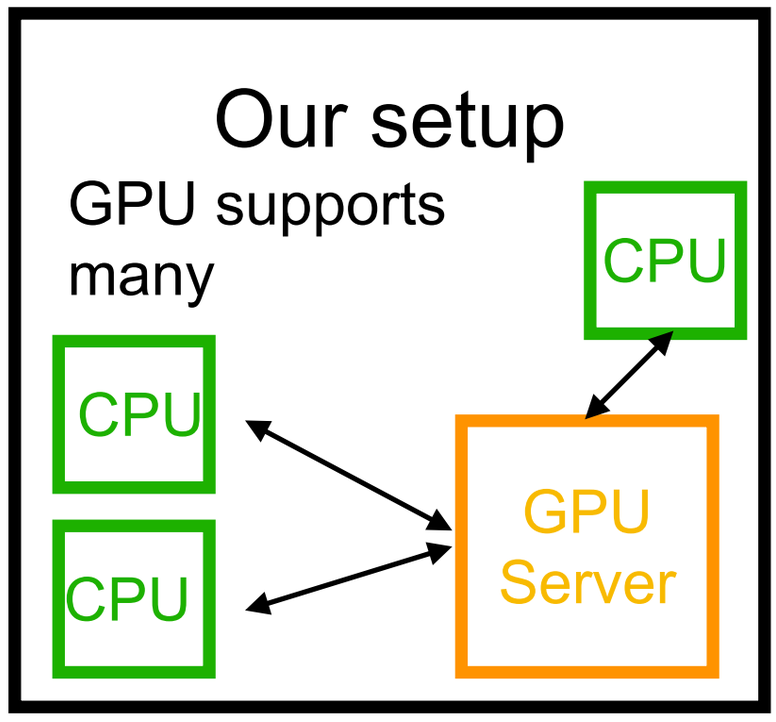
1. Bare latency very low (~70 µs)
2. Bandwidth into FPGA server (not throughput) limited at 25 Gbps
3. Hardware saturation point is ~5000 CPUs!

Patatrack



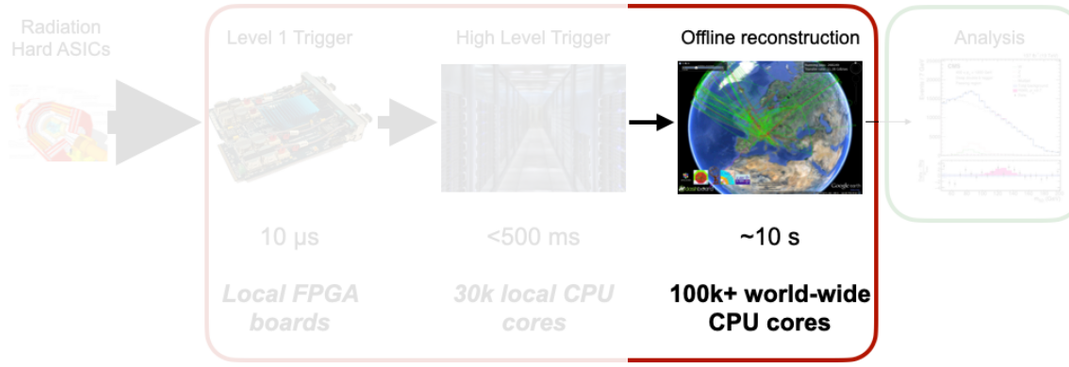
- Currently there is a GPU based tracking for pixels in CMS
  - Plan is to use this non-AI based algo for running in 2022
  - We took this existing code and ran it as-a-service



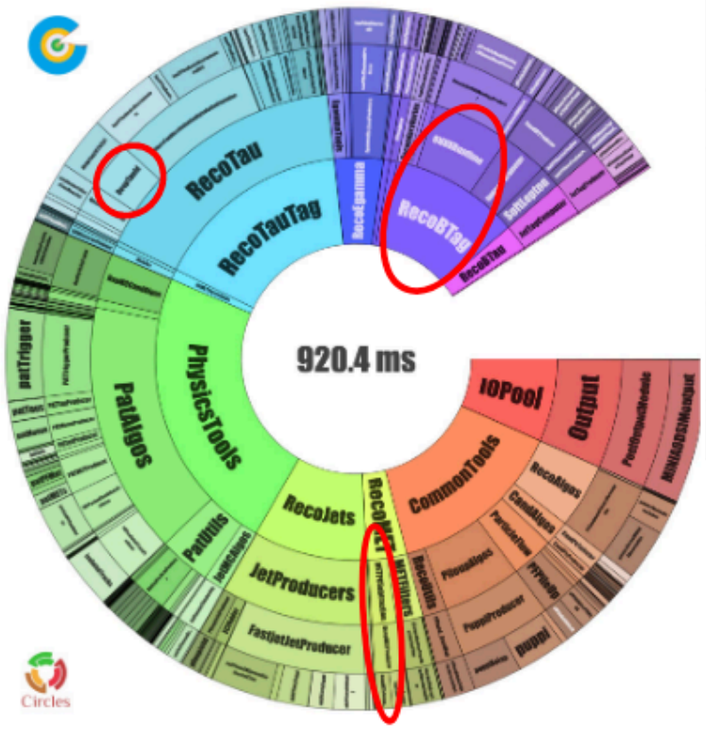


Setup	Max Throughput
Local GPU + 128 cores	100-200 ev/s
Current as-a-service	600 ev/s
Ideal as-a-service	1000 ev/s

Offline scaling test



- Scope of DL offline is increasing
  - Fraction of offline computing time
  - Size and complexity of algorithms
- We ported the three most significant DL algorithms aaS

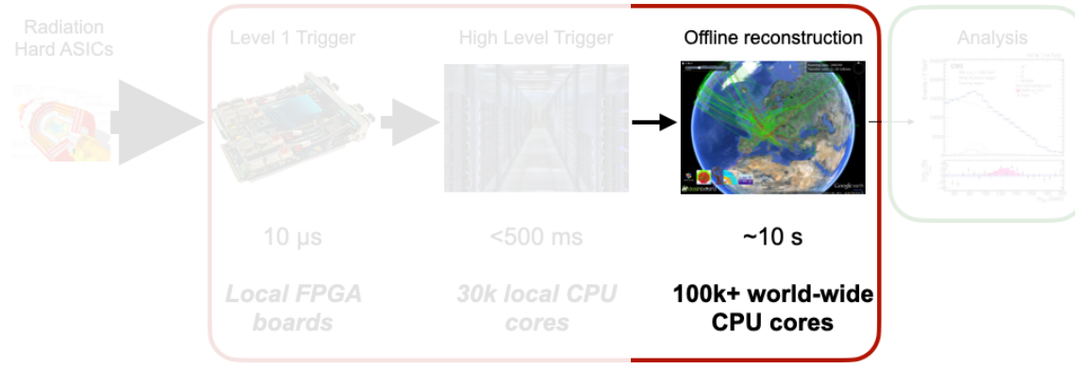


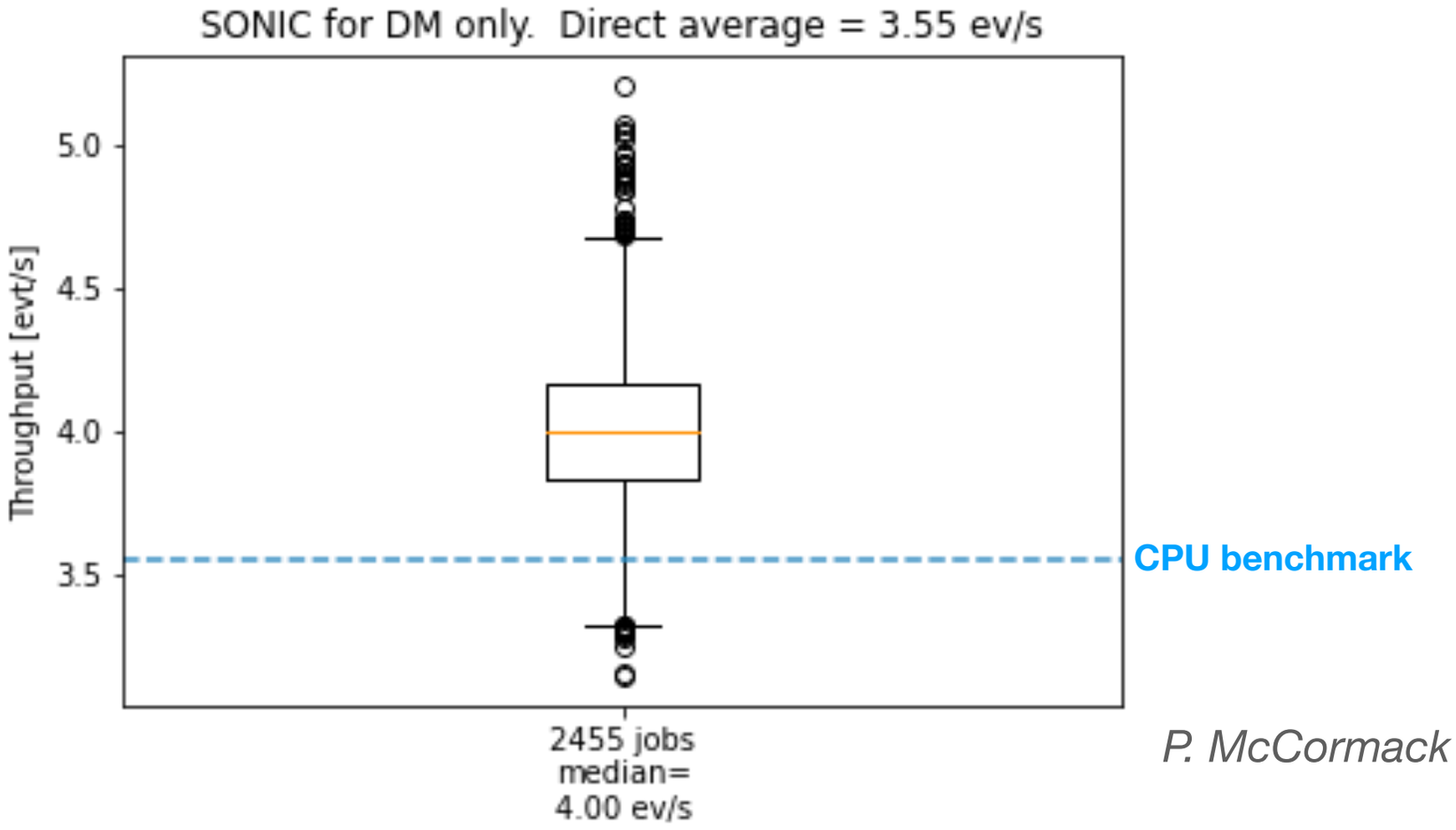
	Time(ms)	Fraction(%)
Total	920.4	100
ParticleNet	43.4	4.7
DeepTau	22.3	2.4
DeepMET	14.0	1.5
Sum	89.7	8.6

RUN 2

P. McCormack

Offline scaling test





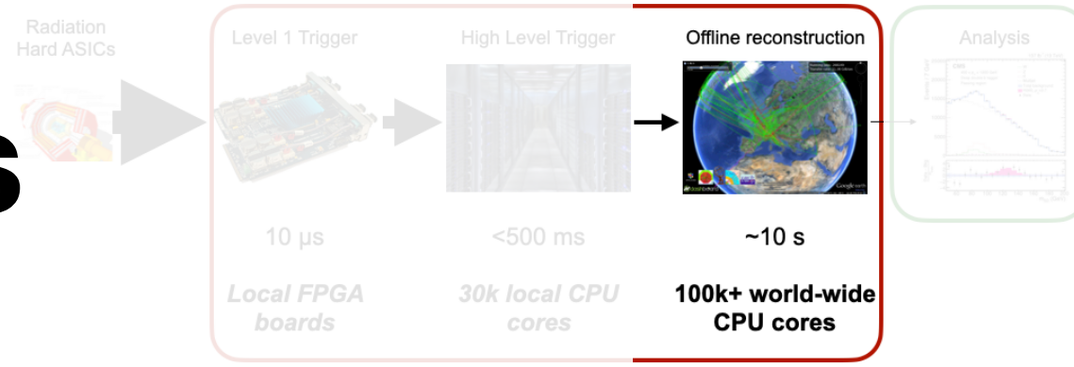
P. McCormack

Deployed **10k CPU cores** with servers hosting ~**100 T4 GPUs**

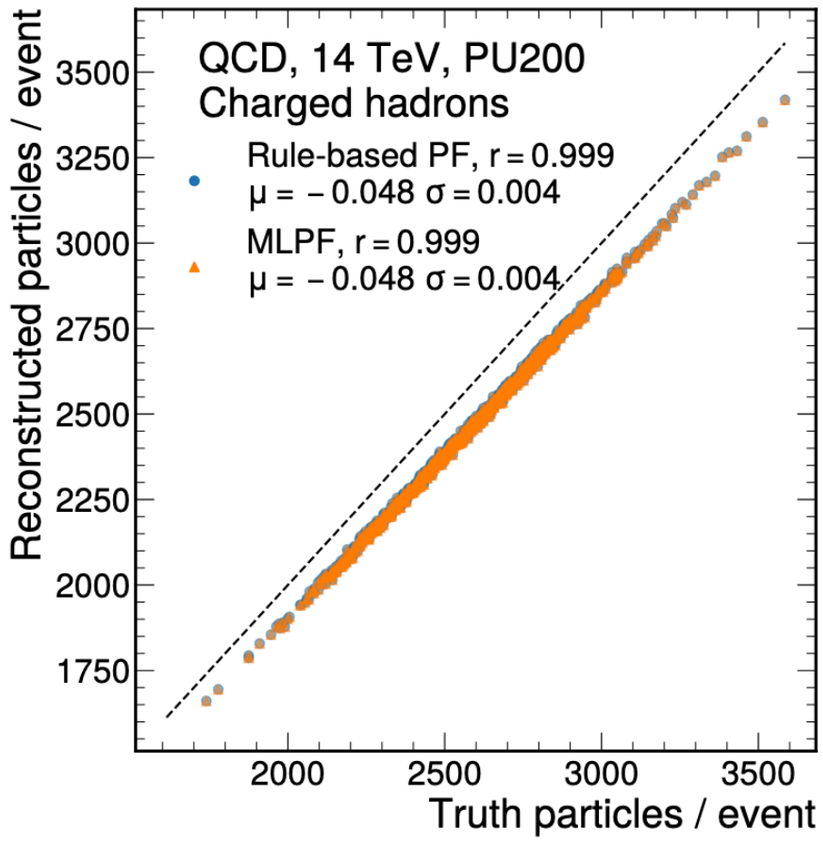
1. Throughput increases by 12% compared to CPU inference
2. Models are spread in optimal ratio across GPUs
3. Bandwidth into servers up to 12 GiB/s

We plan to scale to a 40k CPU test for production-level test

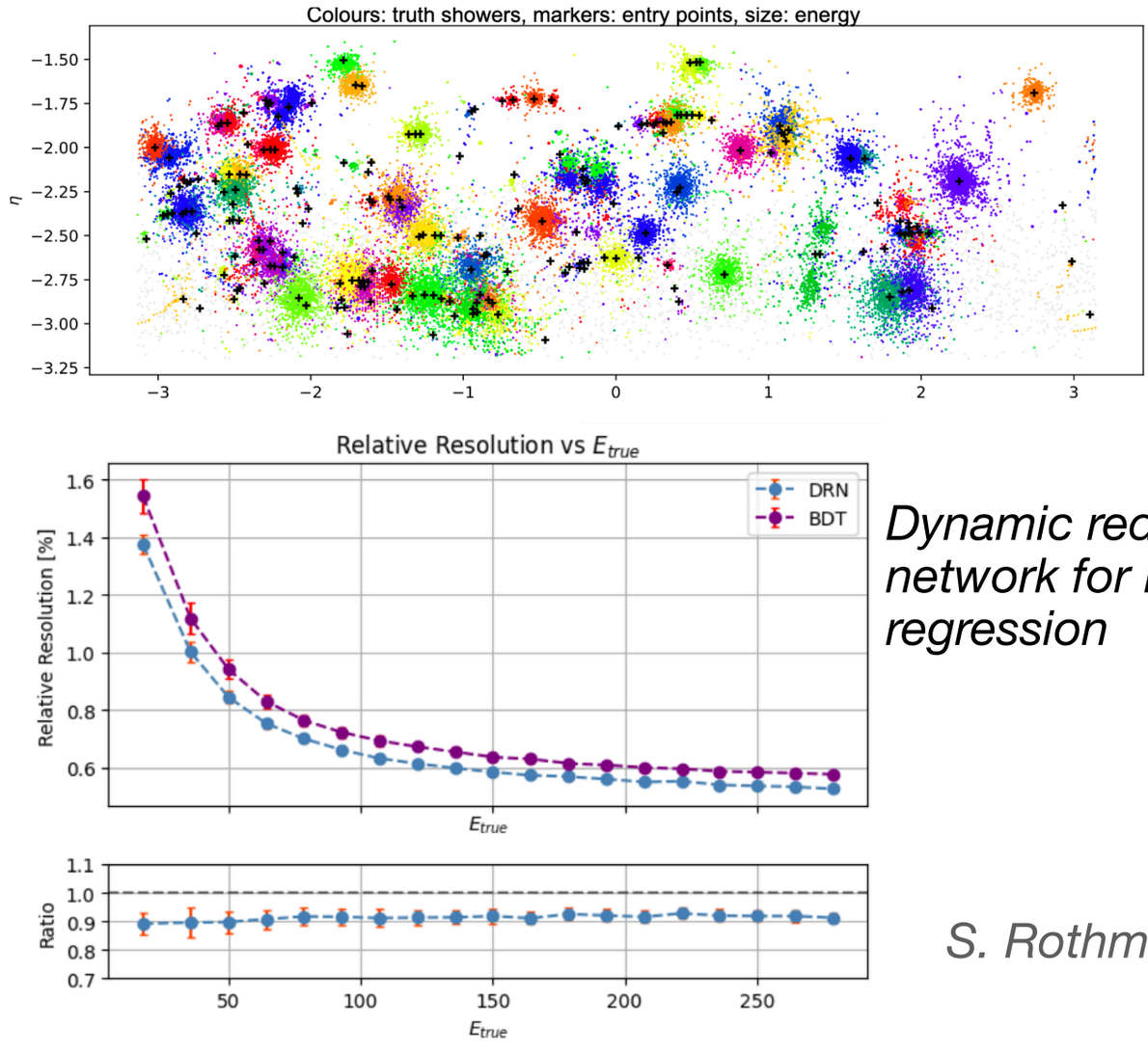
Porting in progress



Particle Flow (MLPF) 2101.08578



Clustering: Graph NNs for HGCAL



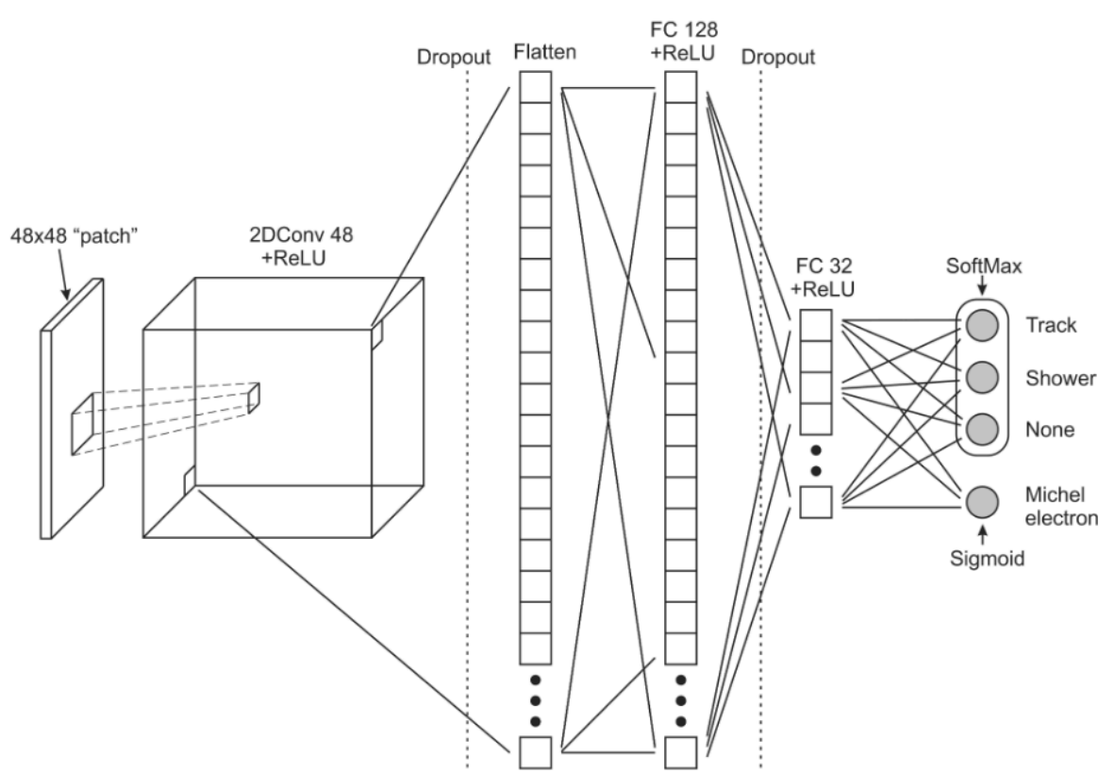
Dynamic reduction network for EGamma regression

S. Rothman

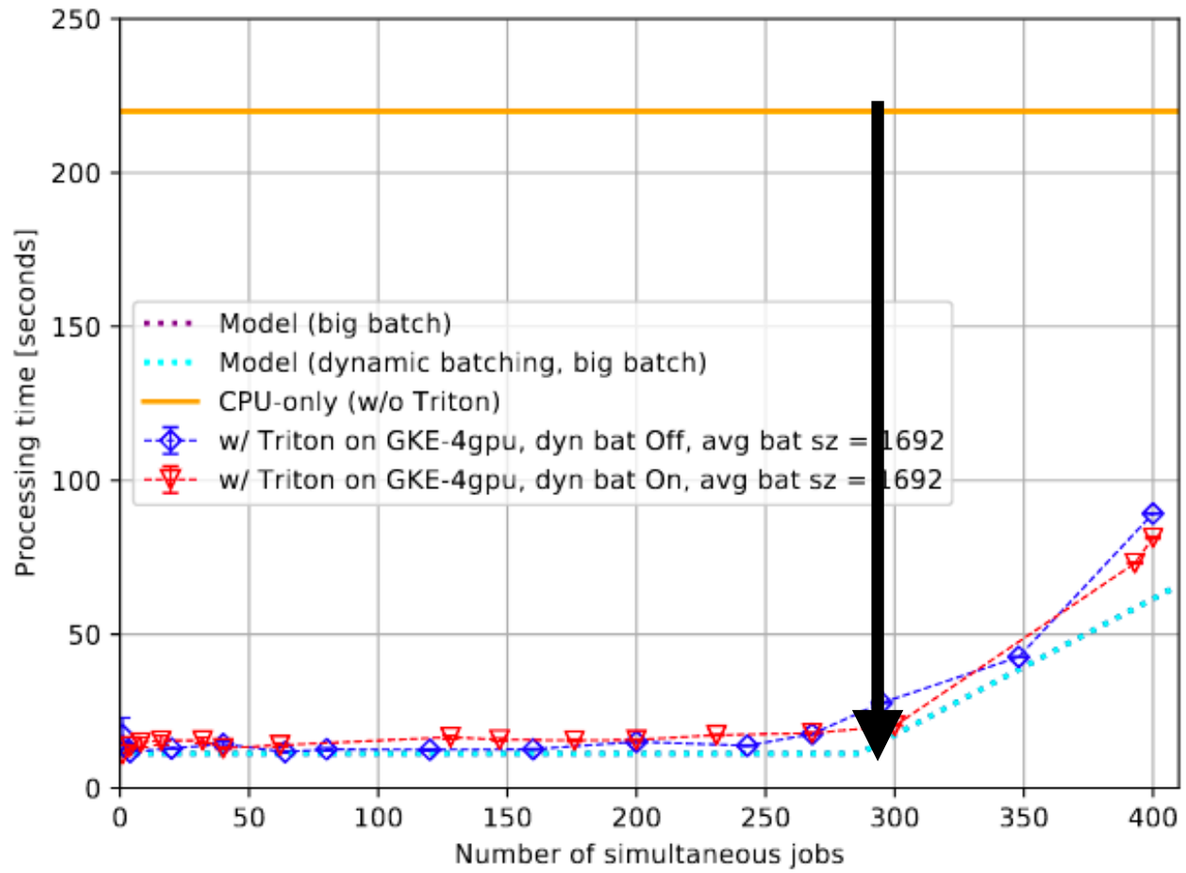
- There are efforts underway to port DL algorithms into SONIC
- Easy to use with low code burden

ProtoDUNE

- ProtoDUNE is a testbed for the Deep Underground Neutrino Experiment
- 2/3 of the reconstruction latency is from *EmMichelTrackId* algorithm
  - 2D CNN classifies electron as a track, shower, or Michel electron



ProtoDUNE



Deploying to GPUs as a service **reduces algorithm latency by 17x**

1. **Hardware efficient** (70 CPU served by single GPU)
2. Related to trigger efforts at DUNE
3. Stay tuned for new result

Summary+outlook

- As we scaled up our studies we've gained experience in a number of areas:
  - This resulted in a large technical toolkit (SONIC, hls4ml, FaaST, …)
  - Operating at very high bandwidths (100+ Gbps), across regions
  - Effectively scaling server resources and configurations to meet demand
  - Optimizing complex workflows, including on GPUs and FPGAs
  - The challenges and opportunities of cloud resources
- Many of these have been lessons shared through collaboration with MMA, neutrino physics, and neuroscience
- We are working our way up to validate our approach by emulating a full-scale LHC data centre test
- We will continue to leverage local resources and HPCs

Thank you and stay tuned for upcoming results!

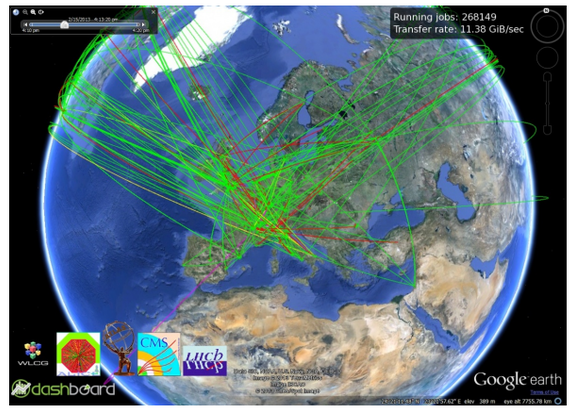
Backup

Integrating into CMSSW

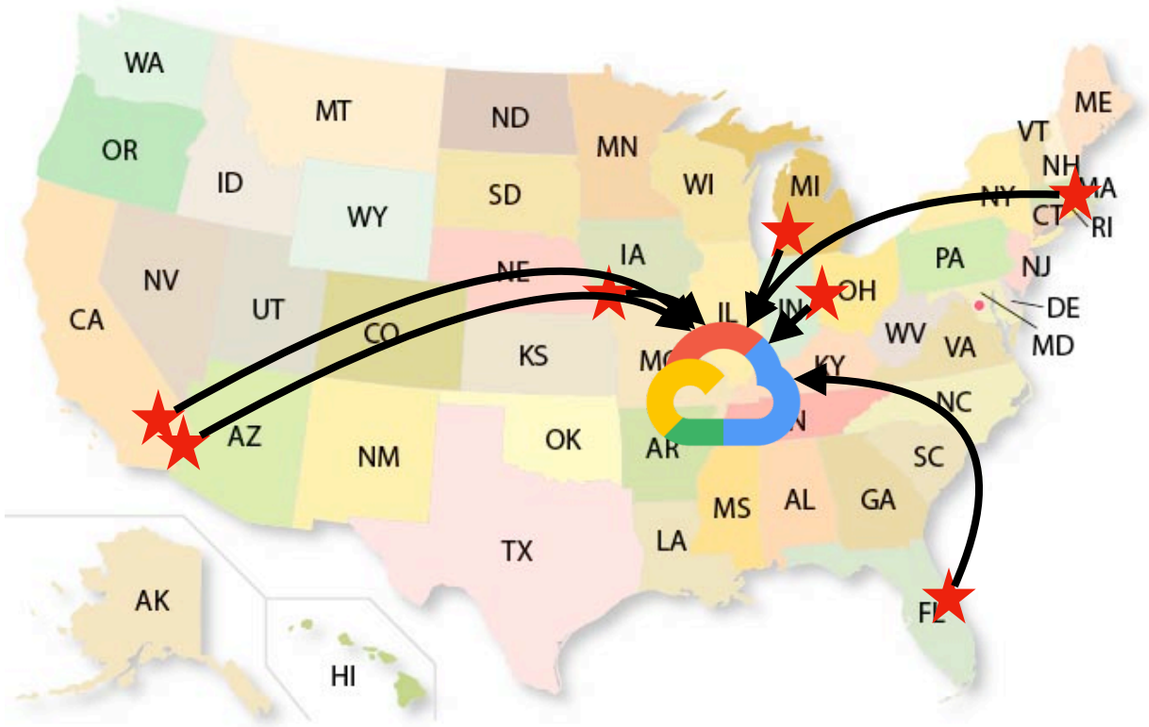
- We integrated FACILE into CMS software
  - HCAL reconstruction performed on GPUs+FPGAs as-a-service
  - User writes a simple producer in c++ (no CUDA)
  - FACILE is simple and has fewer than 100 lines of code
    - Works with either GPUs or FPGAs

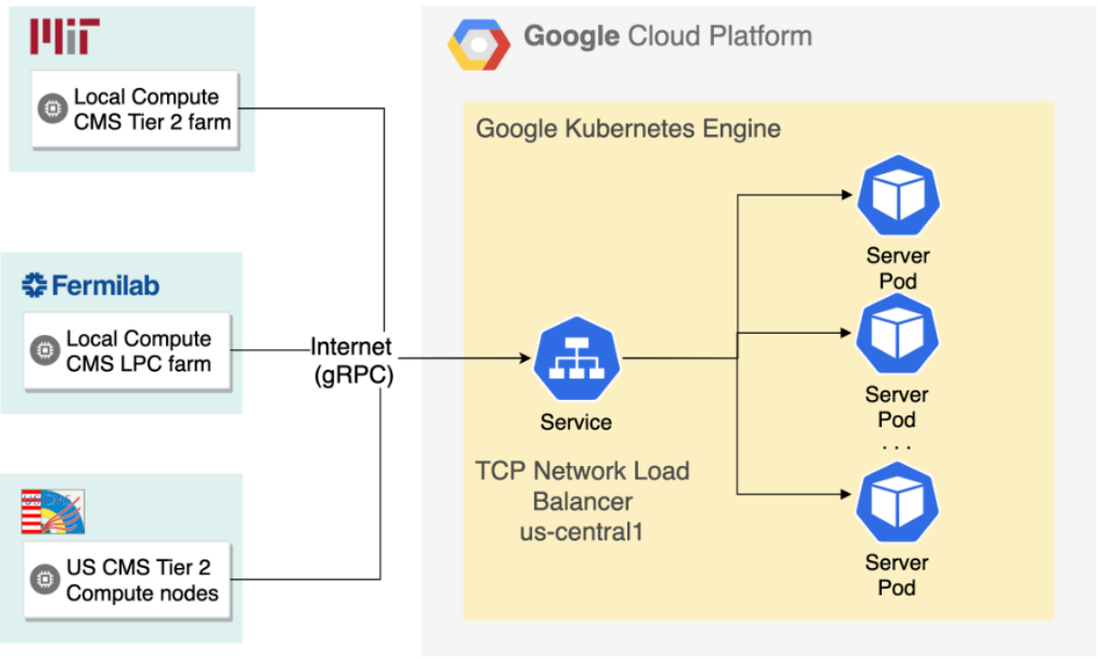
```cpp
#include "HeterogeneousCore/SonicCore/interface/SonicEDProducer.h"
#include "FWCore/Framework/interface/MakerMacros.h"

class MyProducer : public SonicEDProducer<Client>
{
public:
  explicit MyProducer(edm::ParameterSet const& cfg) : SonicEDProducer<Client>(cfg, "MyProducer") {
    //do any necessary operations
  }
  void acquire(edm::Event const& iEvent, edm::EventSetup const& iSetup, Input& iInput) override {
    //convert event data to client input format
  }
  void produce(edm::Event& iEvent, edm::EventSetup const& iSetup, Output const& iOutput) override {
    //convert client output to event data format
  }
  static void fillDescriptions(edm::ConfigurationDescriptions & descriptions) {
    edm::ParameterSetDescription desc;
    Client::fillPSetDescription(desc);
    //add producer-specific parameters
    descriptions.add("MyProducer",desc);
  }
};

DEFINE_FWK_MODULE(MyProducer);
```

Hardware



- For global reconstruction, can we use Cloud resources?
- We explored the outer boundaries by doing a long-distance, high-throughput test with 2k Tier 2 cores (goal=10 Gb/s from MIT to Google Cloud in Iowa)
- Also developed fallback server for local operation of models

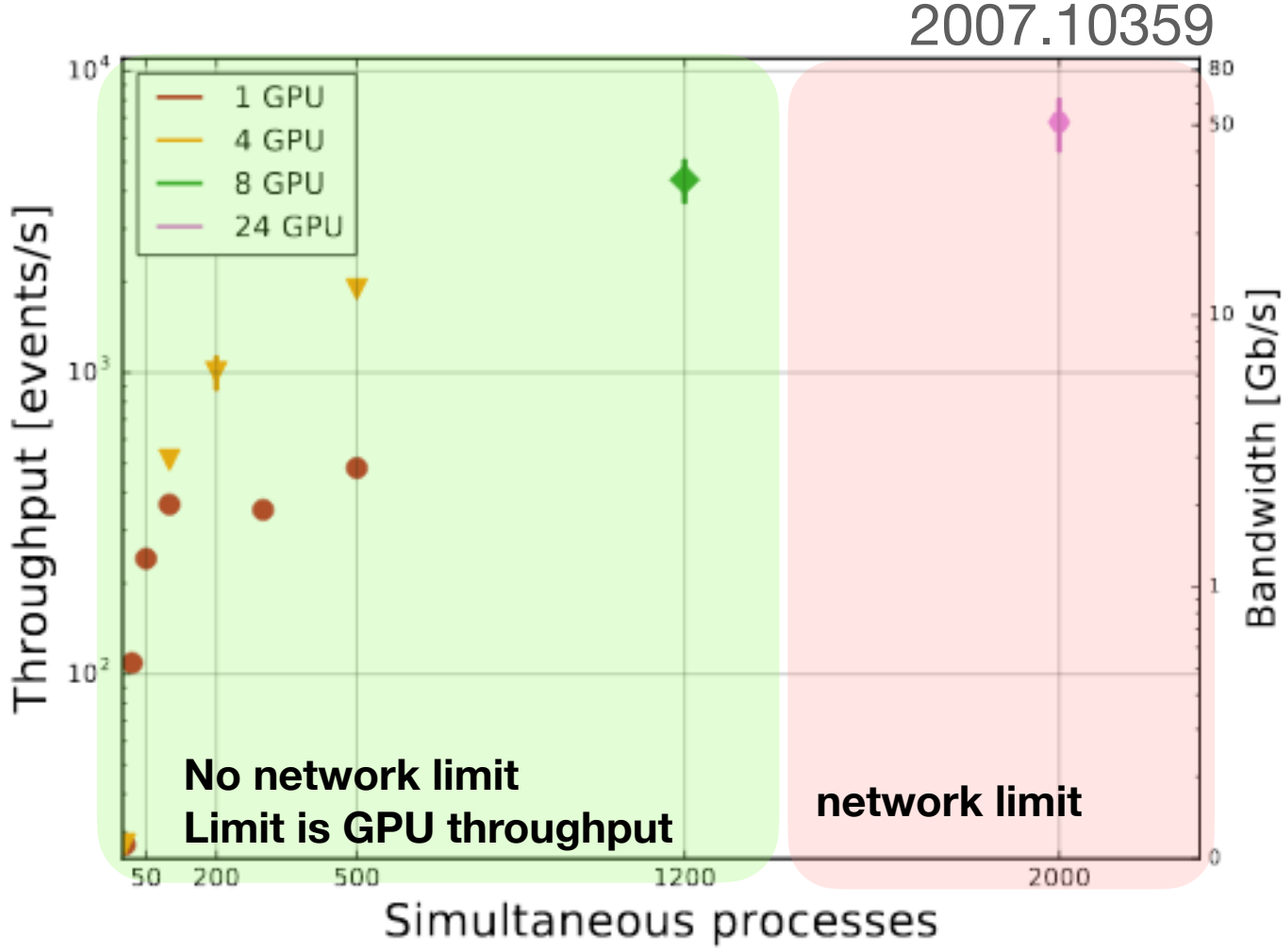




High bandwidth test

- Throughput scales linearly with number of GPUs
- Throughput is stable 60-70 Gb/s (no special links)
  - Far exceeding any realistic use case (offline reco = 10 Gb/s)
  - Custom Kubernetes server: 24 GPUs behind a single IP

2007.10359



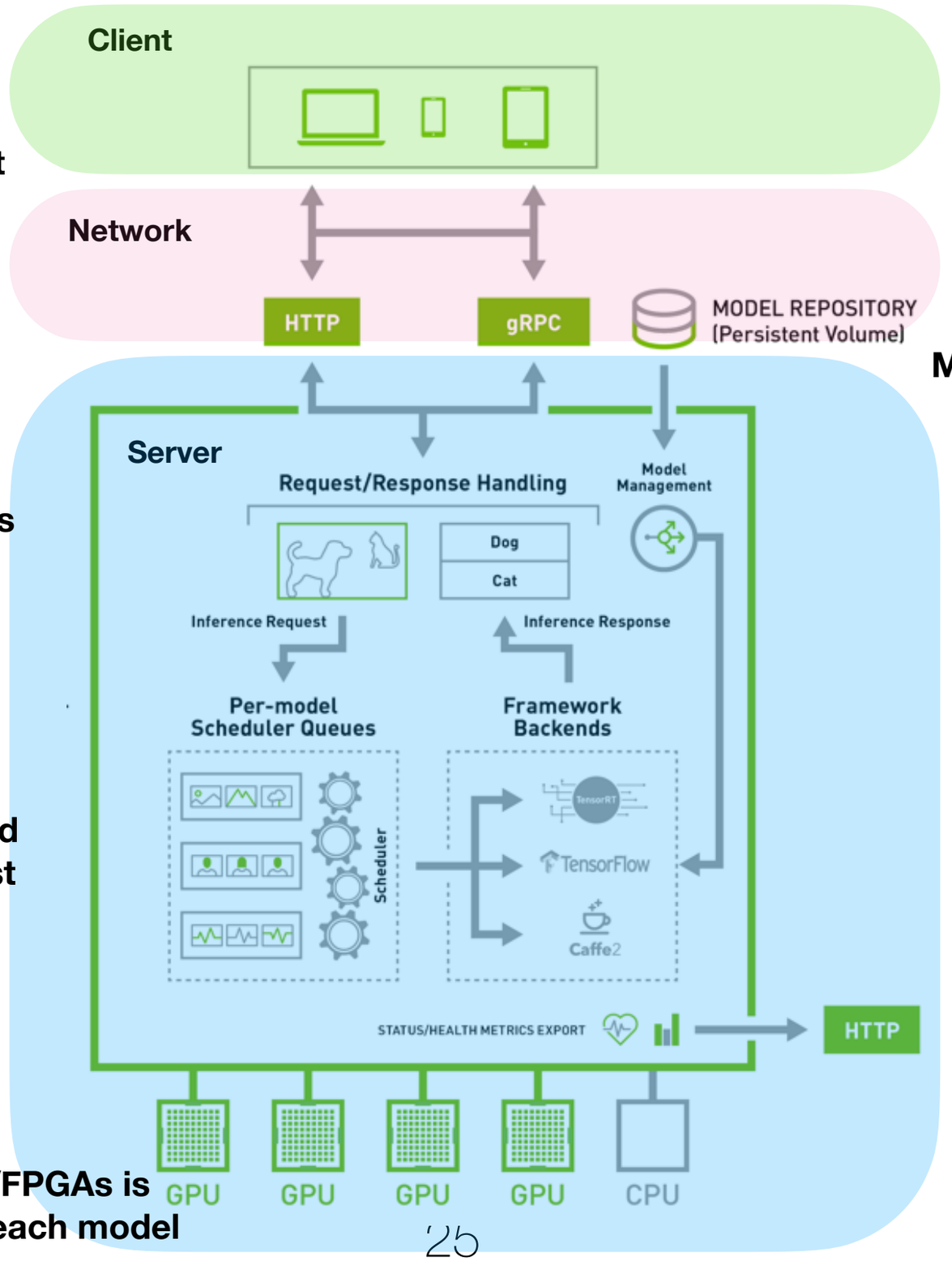
Triton inference server

Client sends request over network

Server receives request

Server queues and schedules request

The number of connected GPUs/FPGAs is scaleable; each has an instead of each model

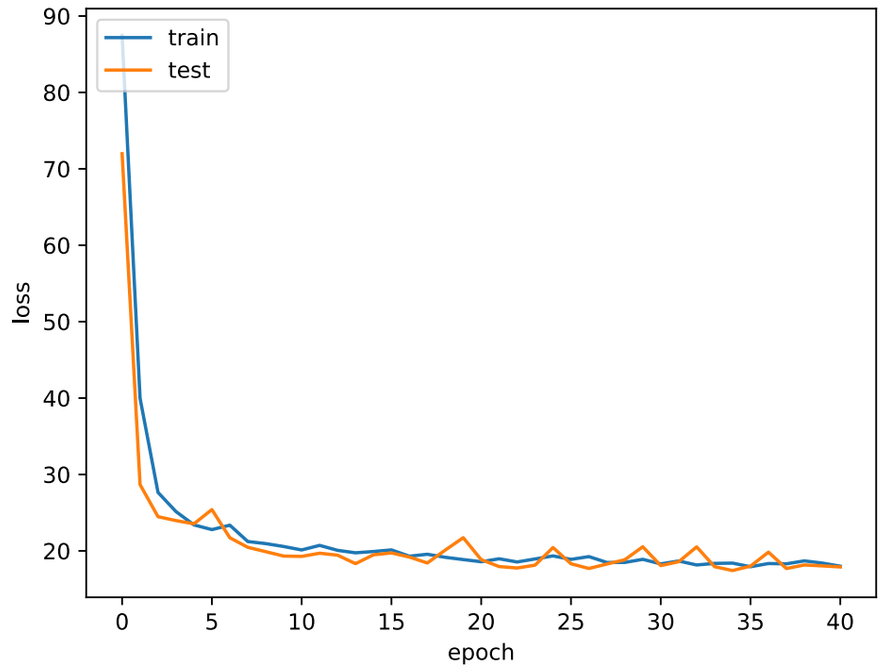


Models are stored in local repository

Many model formats (TensorFlow, Pytorch, TensorRT,…)

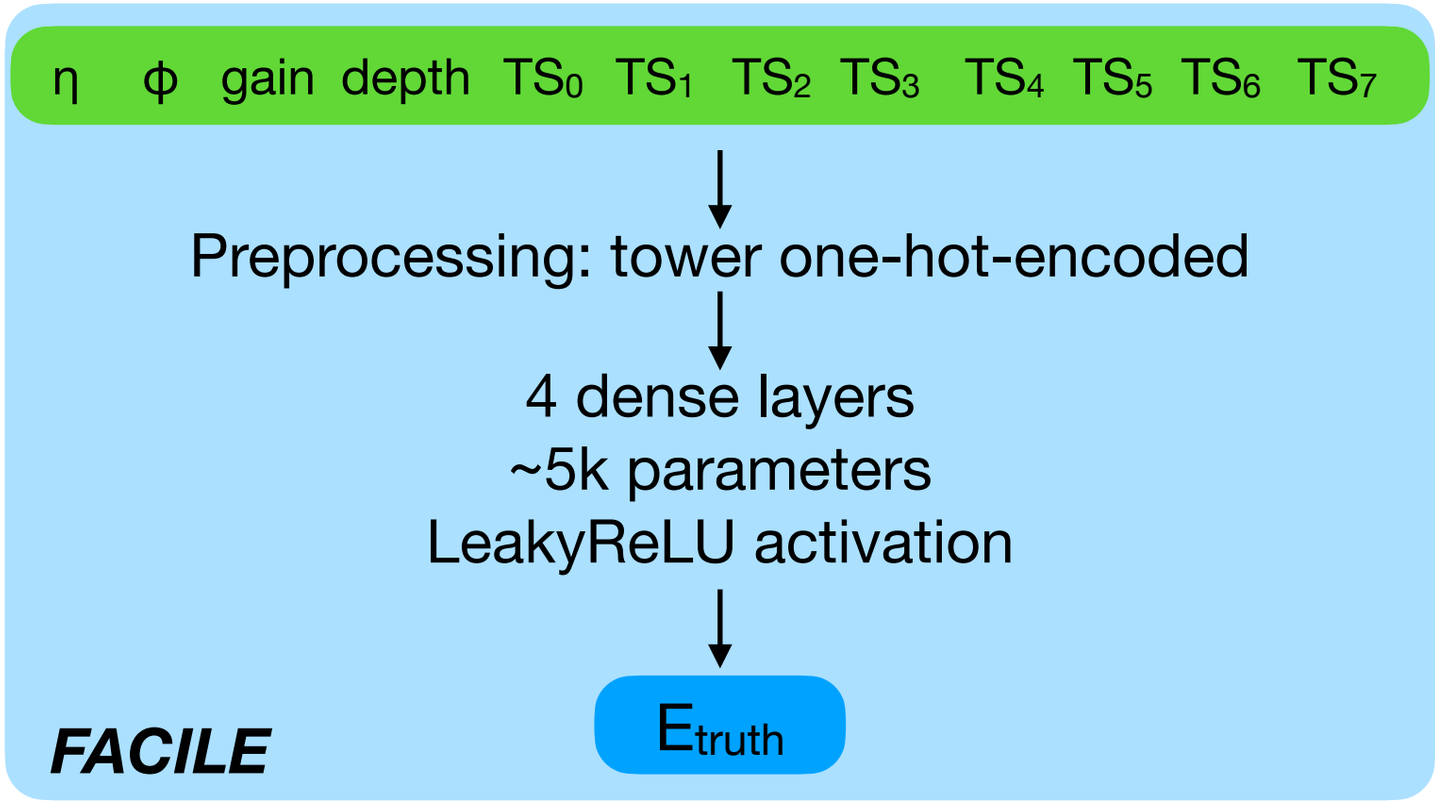
Output monitoring information

FACILE details



Batch size 16000

Architecture	
CPU (interactive FNAL)	8-15 ms
GPU (Nvidia V100)	~2 ms
FPGA (Alveo U250)	~0.2 ms
Gain on GPU/FPGA	**~25x (GPU)** **~50x (FPGA)**



Cloud bandwidth

- Offline reconstruction workflow reaches 12 Gbps