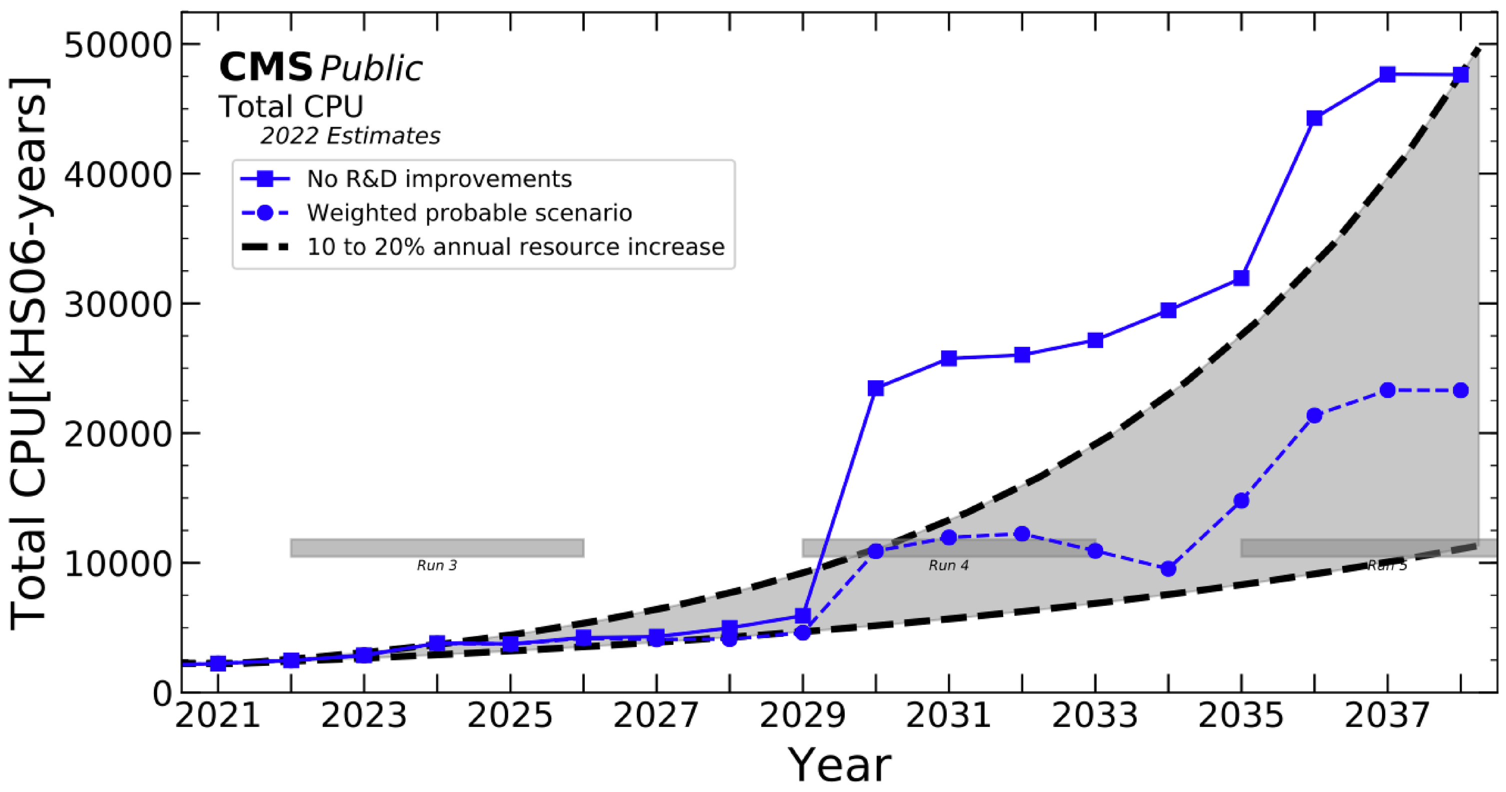
Deployment of inference as a service at the US CMS Tier-2 data centers

Burt Holzman, **Kevin Pedro**, Nhan Tran (FNAL); Philip Coleman Harris, Noah Paladino (MIT); Ethan Colbert, Dmitry Kondratyev, Miaoyuan Liu, Garyfallia Paspalaki, Stefan Piperov, Jan-Frederik Schulte, Yao Yao (Purdue); Javier Duarte (UCSD); Philip Chang, Kelci Ann Mohrman (UF); Yongbin Feng (TTU)
on behalf of the CMS Collaboration

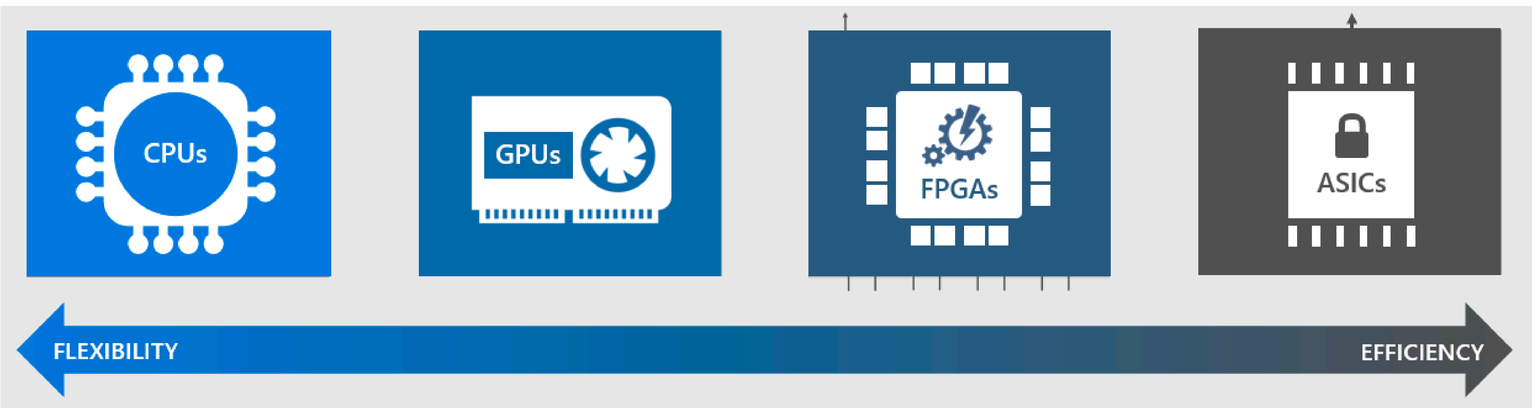
Computing Demands

- Large computing demands for HL-LHC, but CPU performance increases expected to be limited [1]



Coprocessors

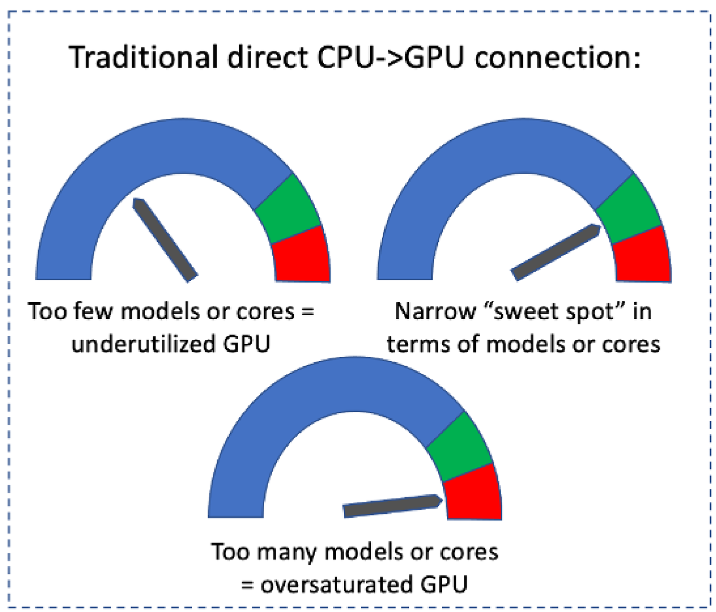
- Recent performance improvements in coprocessors rather than CPUs
- Tradeoffs between flexibility and efficiency

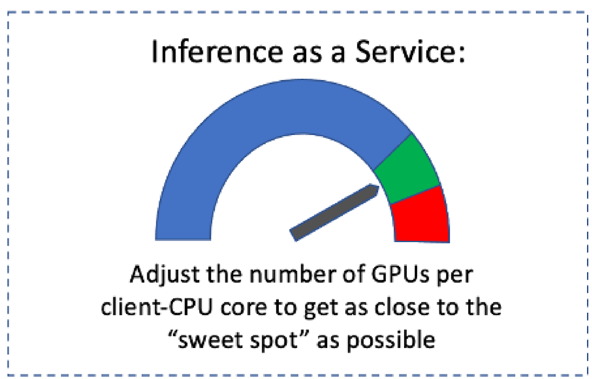


- *Heterogeneous computing*: make best use of each processor type

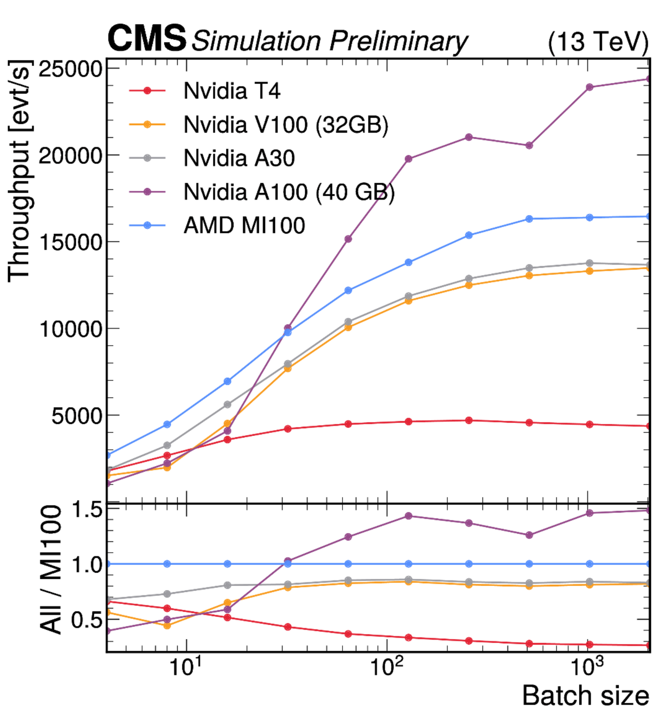
Inference as a Service

- **SONIC**: Services for Optimized Network Inference on Coprocessors [2]
  - *Design pattern* for inference as a service in experiment software
- Build on industry technologies: gRPC, Nvidia Triton inference server
- Advantages:
  - *Isolation*: factorize ML frameworks out of experiment software
  - *Simplicity*: client code only handles input/output conversions
  - *Flexibility*: CPU-GPU ratios can be adjusted dynamically
  - *Efficiency*: optimize CPU-GPU ratios to ensure full usage (minimizes cost)
  - *Portability*: use CPU, GPU, FPGA, etc. with no client-side code changes
  - *Accessibility*: use remote coprocessors if none available locally





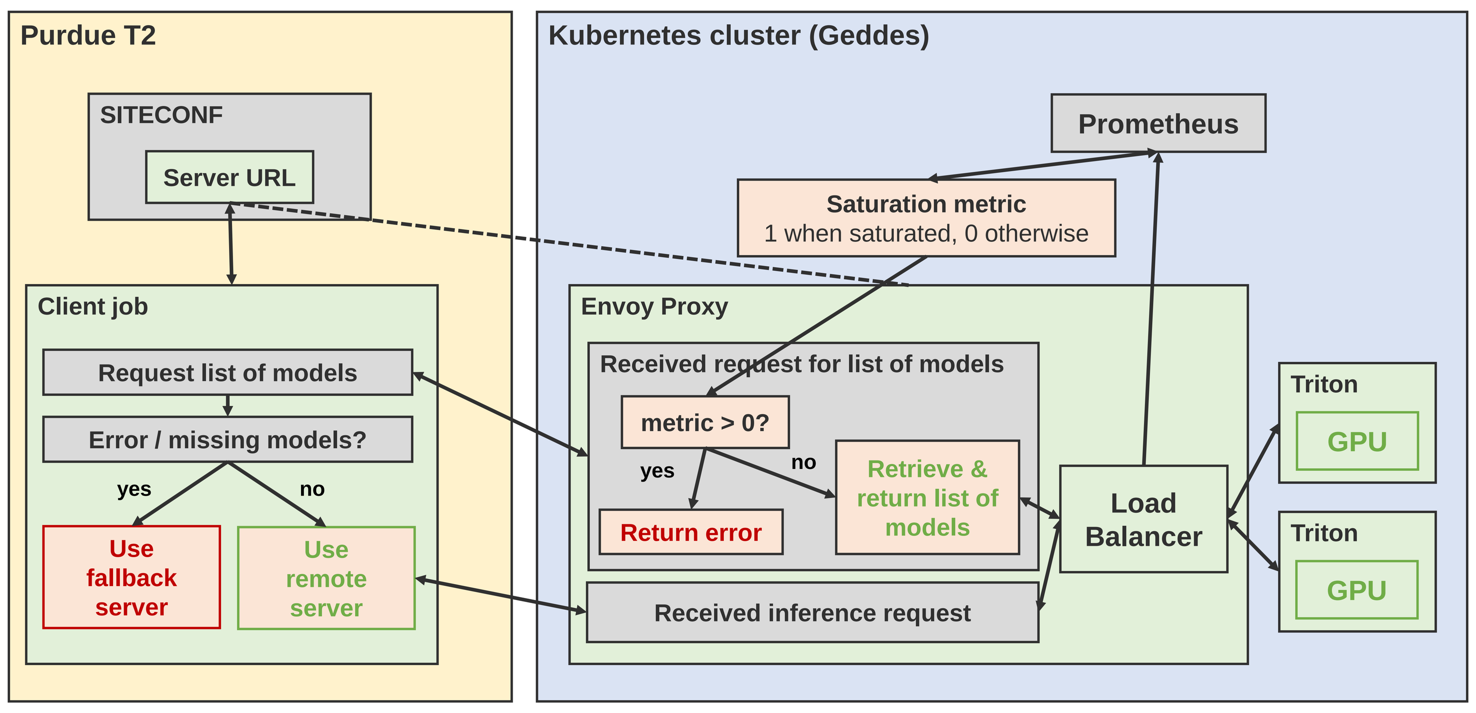
AMD GPUs



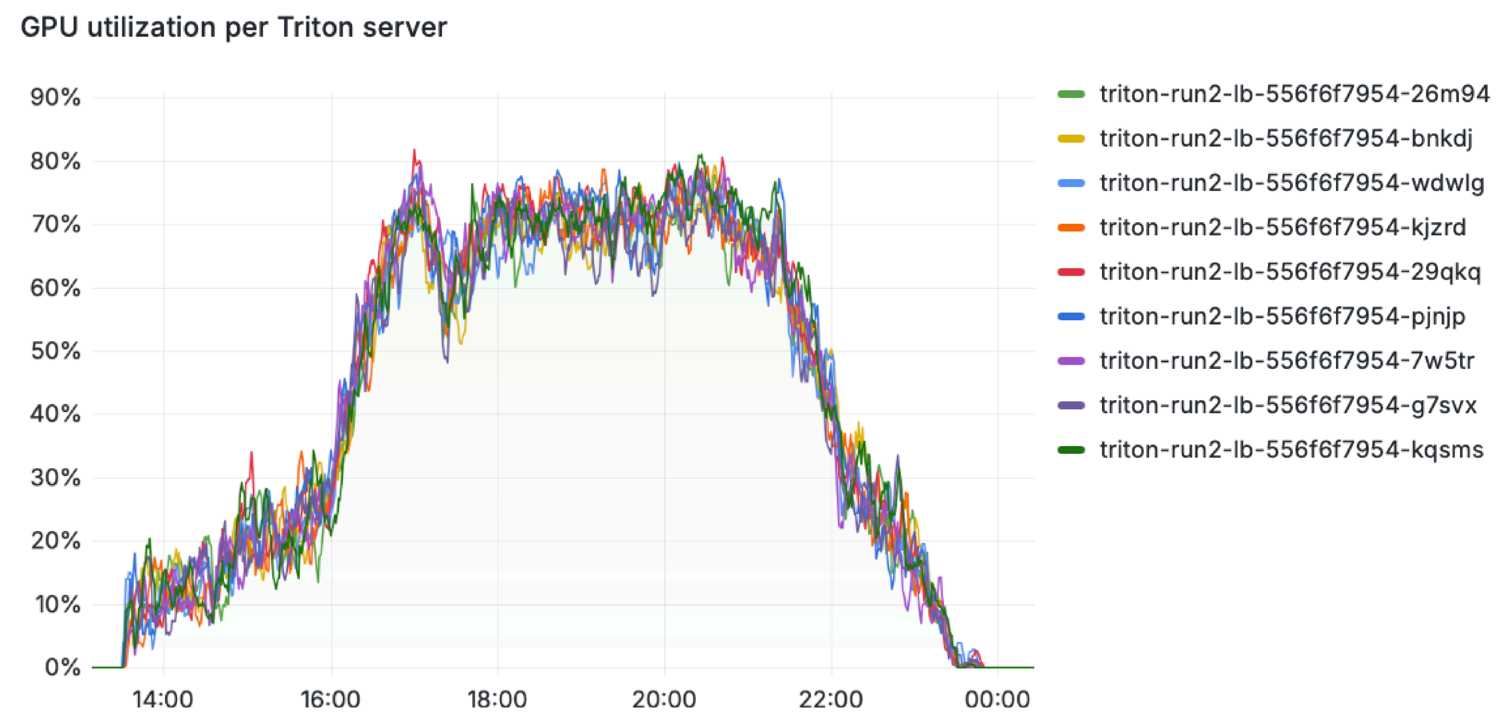
- First demonstration of non-Nvidia GPU usage, using important CMS ParticleNet [3] algorithm
- AMD MI100 has superior throughput to several existing GPU types (even A100s at smaller batch sizes)
- AMD GPUs can be accessed through the Triton server using a custom backend: dedicated instructions loaded by server via Python (or compiled into shared library)

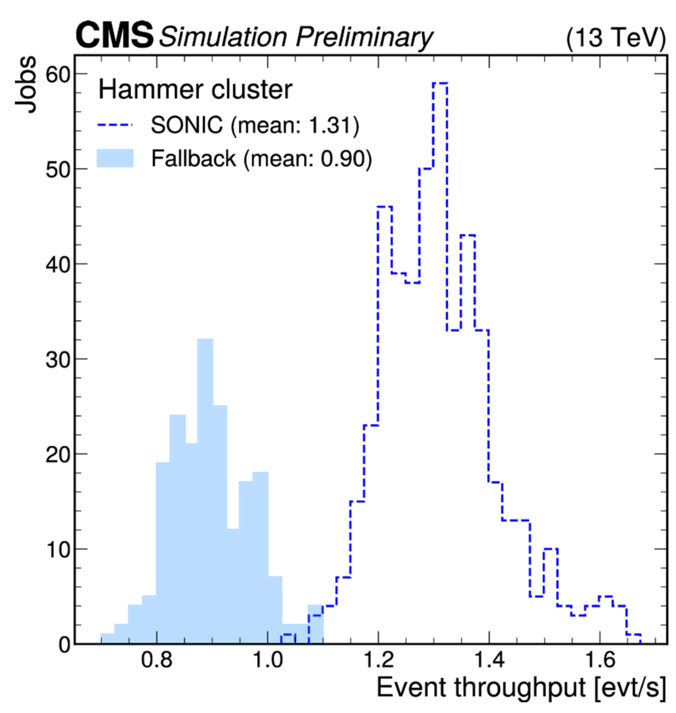
Architecture at Purdue T2

- Server discovery through official site configuration
- Prevent connections to saturated servers based on queue latency
- Load balancing via Envoy Proxy
- Autoscaling via KEDA (Kubernetes Event-Driven Autoscaling)
- Configuration bundled into Helm chart to deploy at other T2 sites



Scaling and Load Balancing

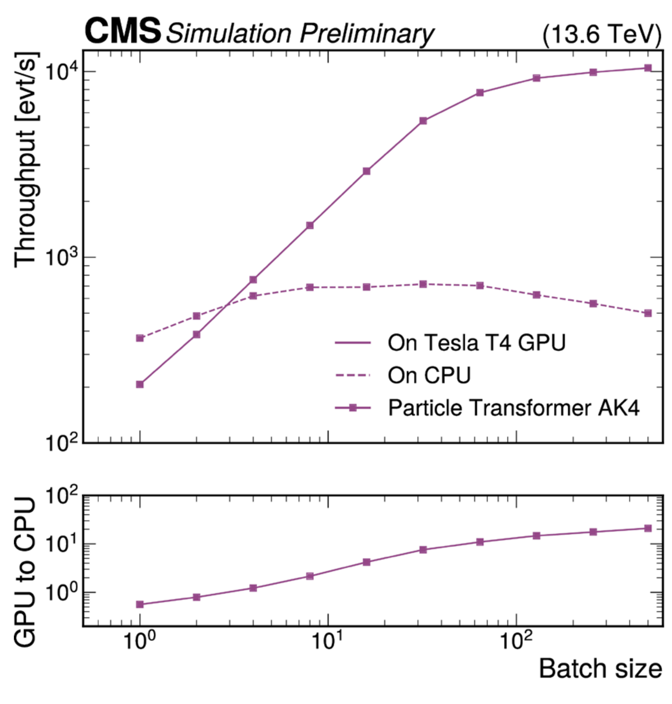




- Production-like "continuous flow" of jobs (via CRAB): 1000 jobs in batches of 50, every 10 min
- New load balancer distributes load *per request*: consistent and uniform load across GPUs for hours
- **45%** speedup in Run 2 miniAOD workflow when offloading ML inference to GPUs vs. falling back to CPU-only processing
  - Depends on CPU properties

Run 3: Transformers

- CMS Run 3 miniAOD processing now includes Particle Transformer (ParT) [4], successor to ParticleNet
- Factor 10 speedup demonstrates advantages of batching
  - Dynamic batching (combining requests from different threads/jobs) only possible via SONIC/Triton!
- Overall miniAOD workflow speedup: **33%** w/ ParT on GPU through SONIC



References

[1] CMS-NOTE-2022-008
[2] CSBS 8 (2024) 17
[3] PRD 101 (2020) 056019
[4] PMLR 162 (2022) 18281

FERMILAB-POSTER-24-0301-CMS-CSAID-PPD

This document was prepared by the CMS Collaboration using the resources of the Fermi National Accelerator Laboratory (Fermilab), a U.S. Department of Energy, Office of Science, Office of High Energy Physics HEP User Facility. Fermilab is managed by Fermi Research Alliance, LLC (FRA), acting under Contract No. DE-AC02-07CH11359.