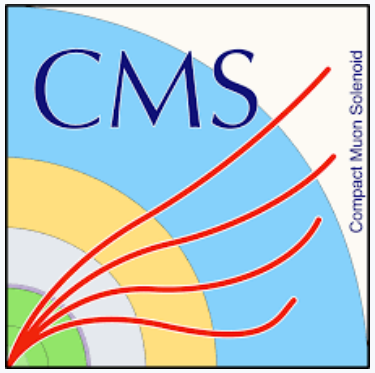
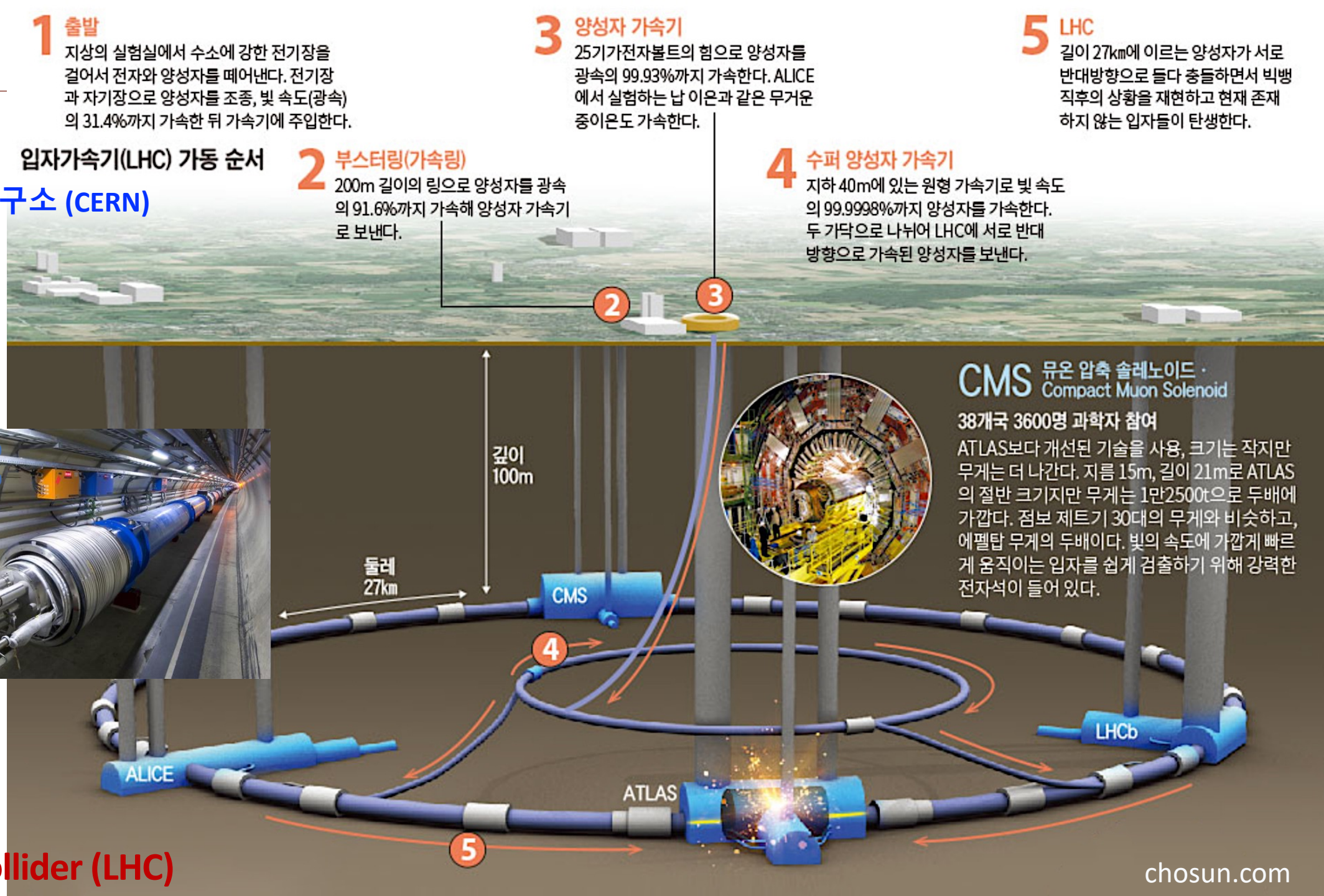


Artificial Intelligence for High-Energy Physics

SEUNGJIN YANG (KHU), CHANG-SEONG MOON (KNU)

ON BEHALF OF KOREA CMS (KCMS) ML GROUP

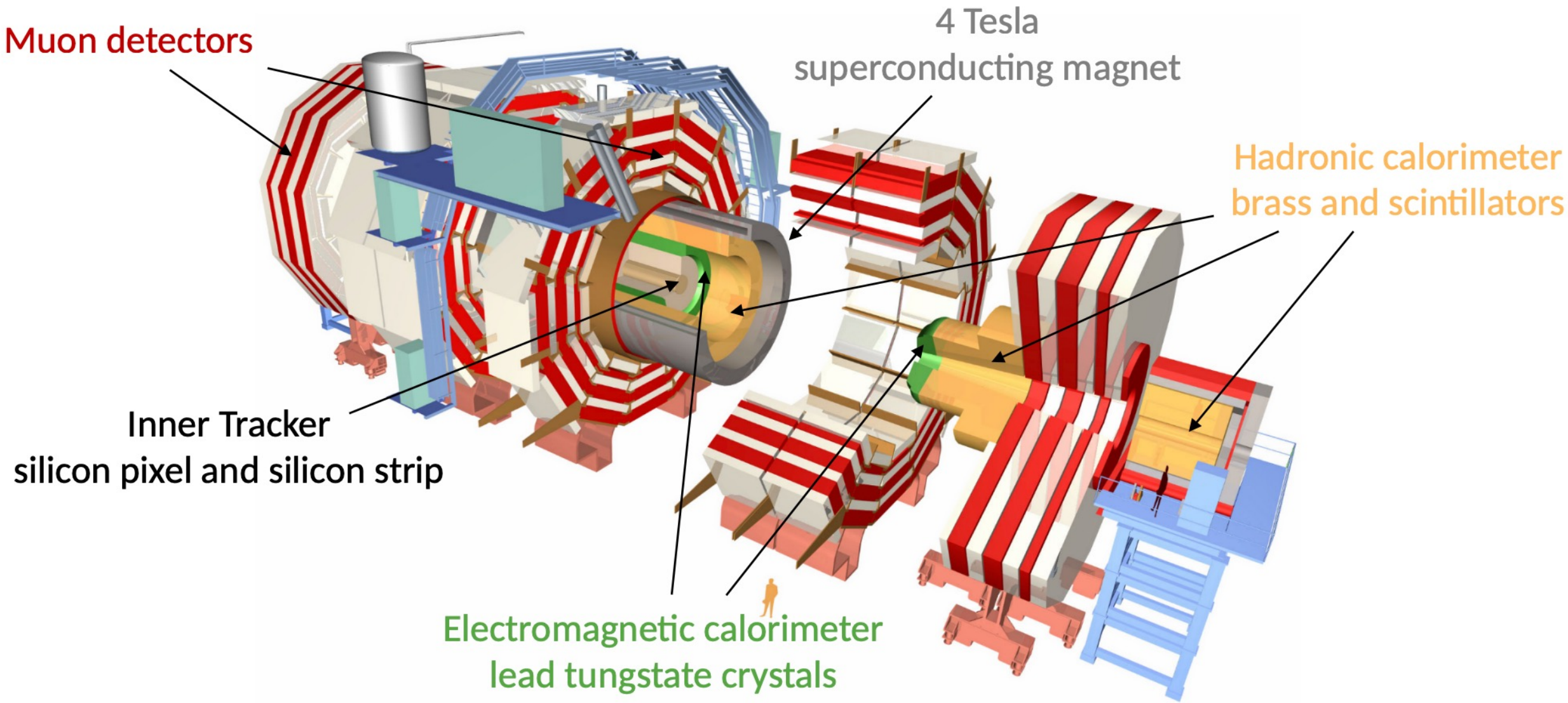
유럽입자물리 연구소 (CERN)



The tool:
Large Hadron Collider (LHC)

CMS detector

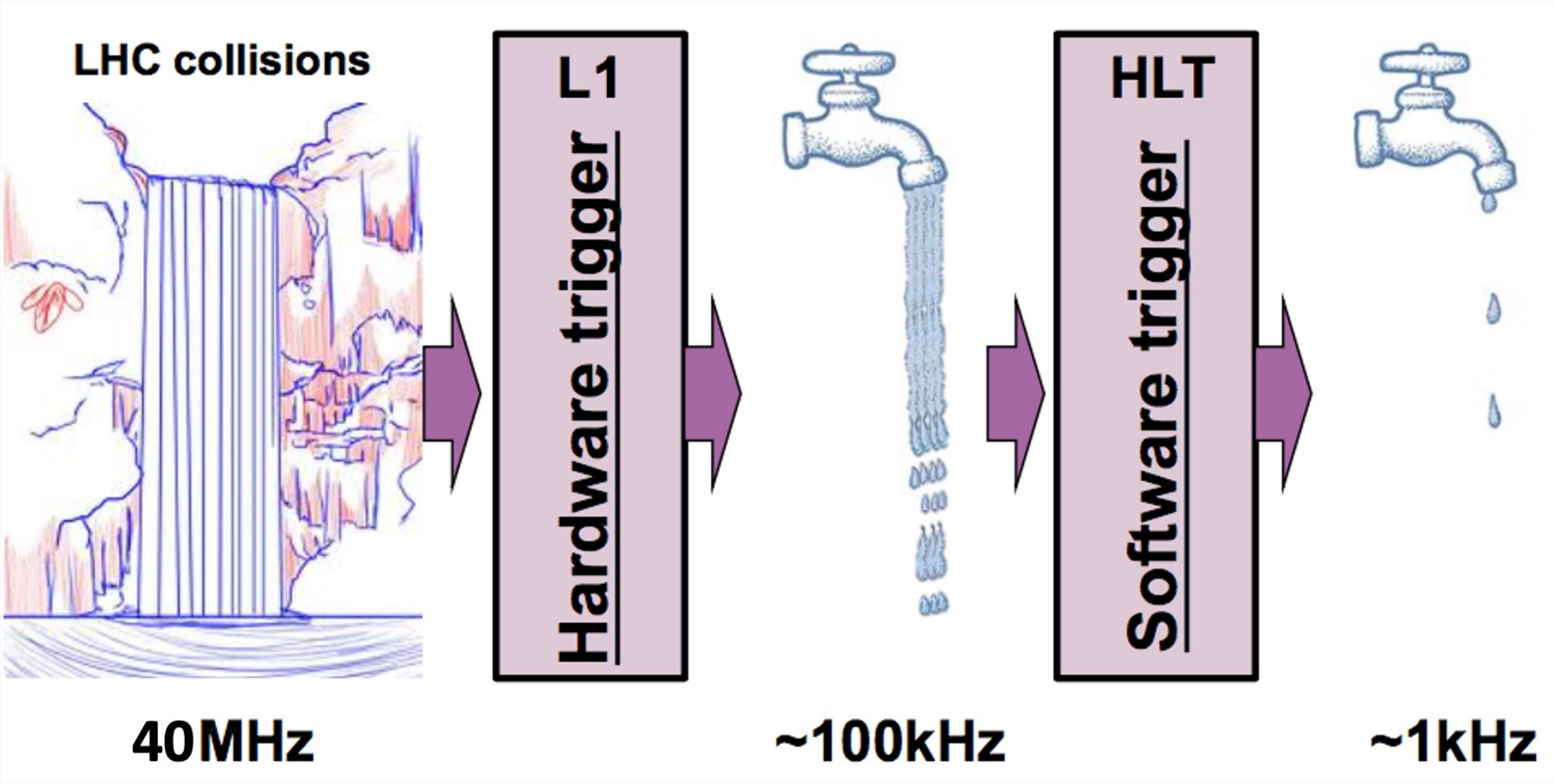
High-granularity detectors
Order of 100 Million channels



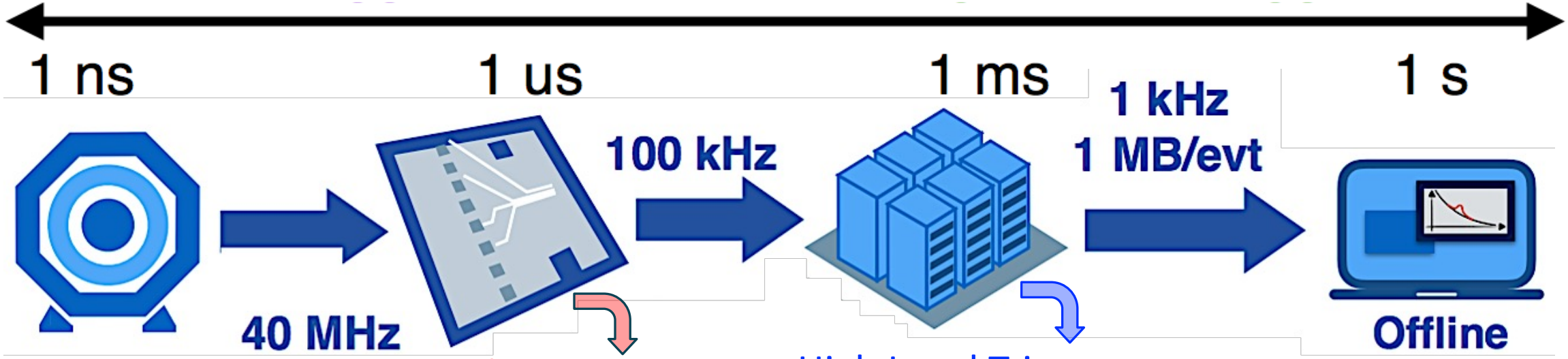
Real-Time Graph Neural Networks for Missing E_T at CMS

Bongho Tae, Chang-Seong Moon (Kyungpook National University)

CMS 데이터 트리거 원리 : 실시간 이벤트 필터링



Current CMS Data Processing

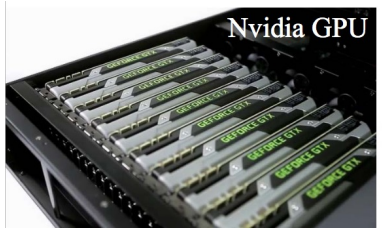


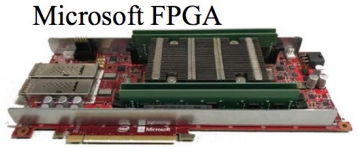
- 40 MHz LHC clock
- ASICS and FPGAs
- Decision made in ~4 μs
- ML inference with FPGAs
- 99.75% (399/400) events are rejected

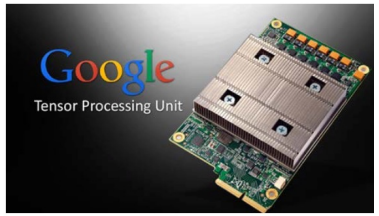
- 100 kHz Input rate
- CPU Farm : 30,000 CPU cores
- Decision made in 300 ms
- 99% (99/100) events are rejected

After triggering, 99.9975% (39999/40000) of events are gone forever!

Heterogeneous computing





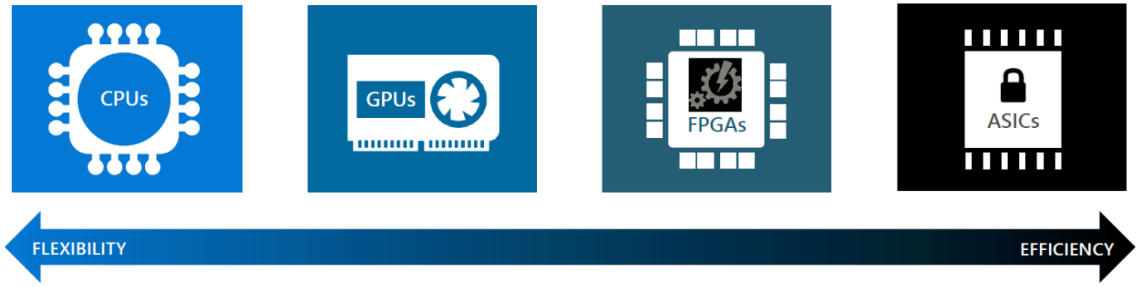


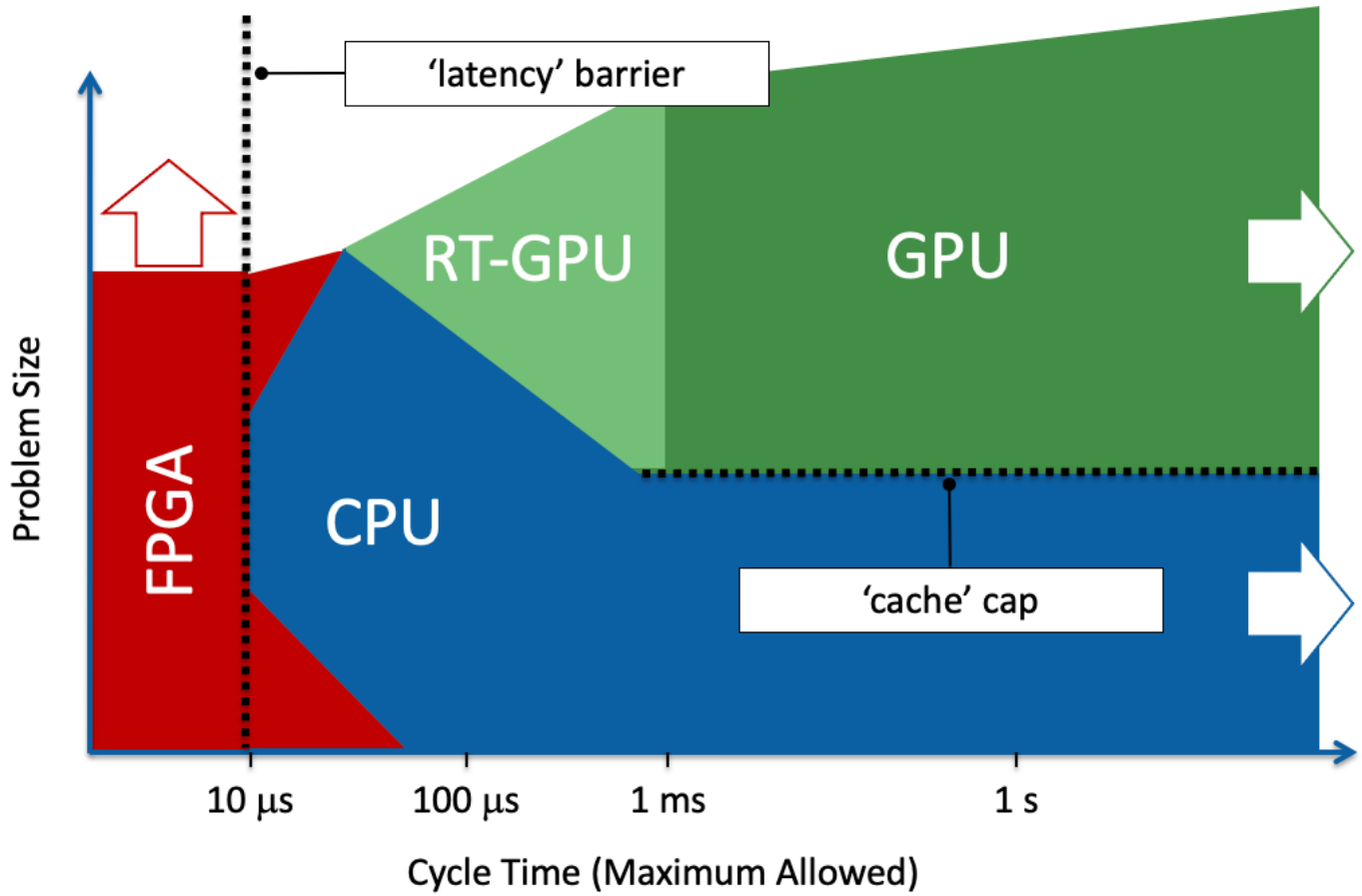
CPU
- Small models
- Small datasets
- Useful for design space exploration

GPU
- Medium-to-large models, datasets
- Image, video processing
- Application on CUDA or OpenCL

TPU
- Matrix computations
- Dense vector processing
- No custom TensorFlow operations

FPGA
- Large datasets, models
- Compute intensive applications
- High performance, high perf./cost ratio





XILINX

https://www.xilinx.com/publications/powered-by-xilinx/cerncasestudy-final.pdf

Artificial Intelligence Accelerates Dark Matter Search

Integrating Inference Acceleration with Sensor Pre-processing in Xilinx FPGAs Delivers Performance Unachievable by GPUs and CPUs

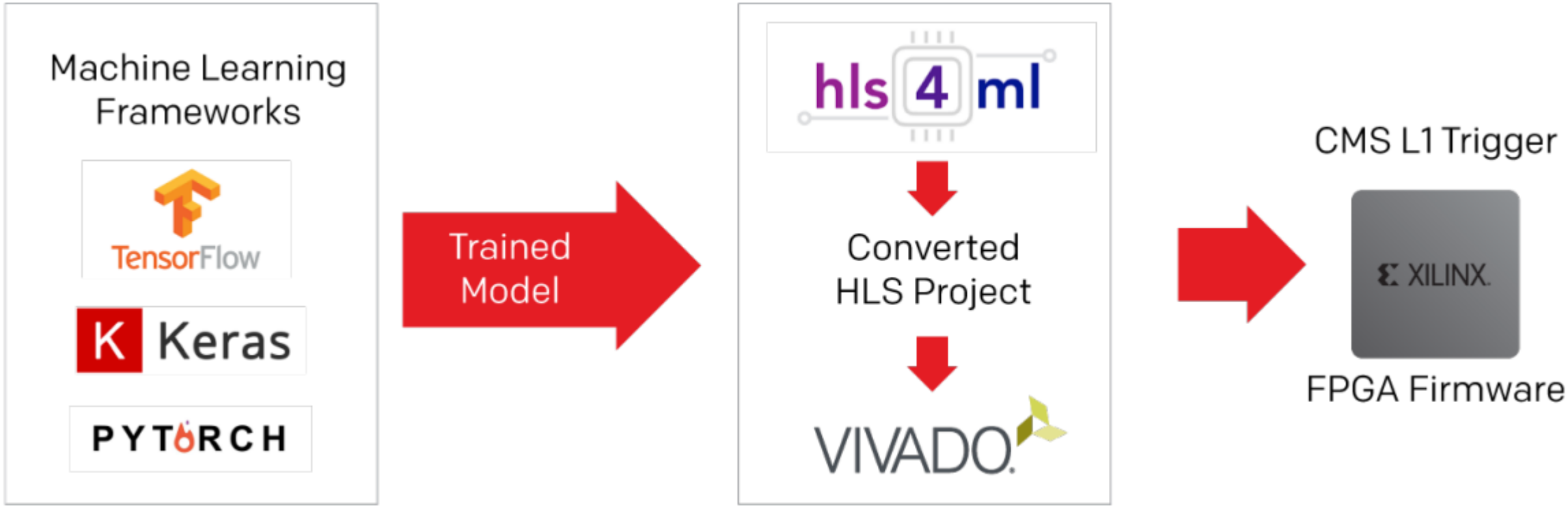
AT A GLANCE:

Customer: High energy physics researchers from an association of leading international institutions (CMS Institute) conducting experiments at the European particle physics laboratory, CERN.

Industry: Scientific Research

Employees: CMS Institute has more than 4,000 global scientific collaborators representing 200 institutes and universities from more than 40 countries.

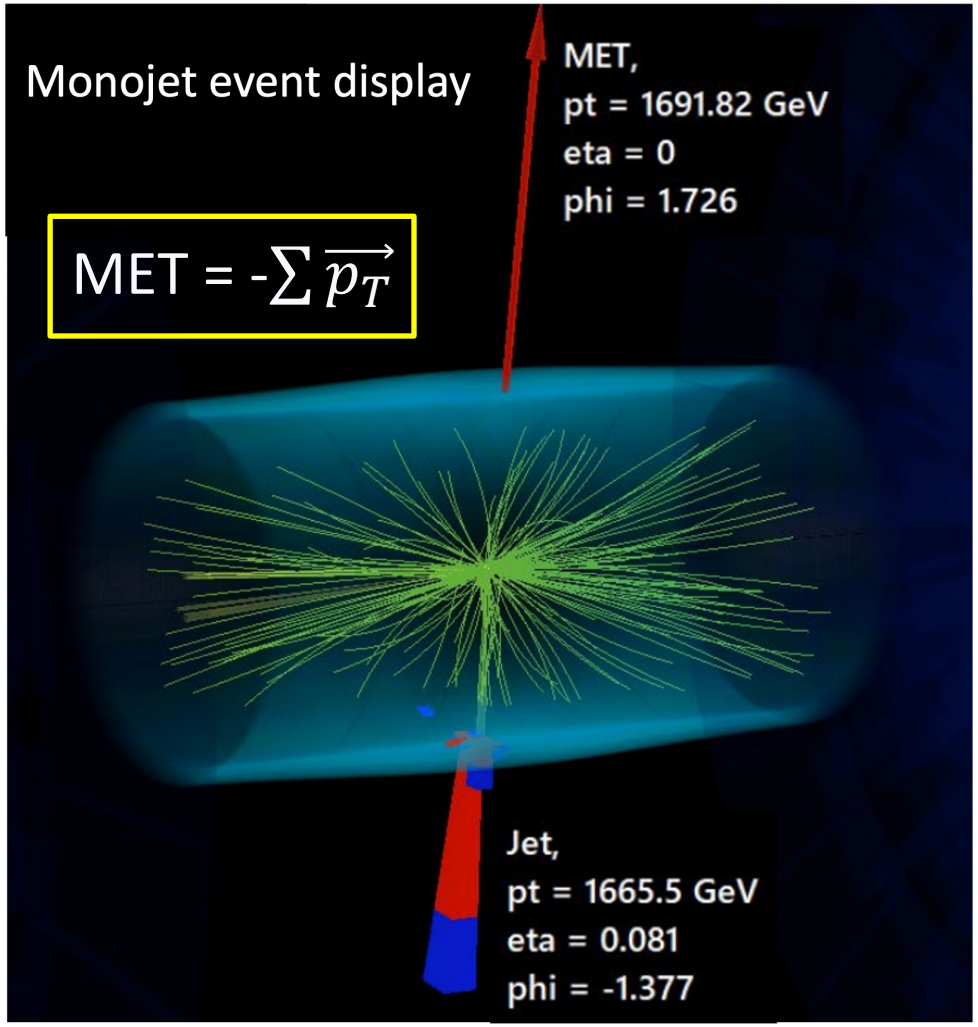
Customized hls4ml Flow Leveraging Vivado HLS

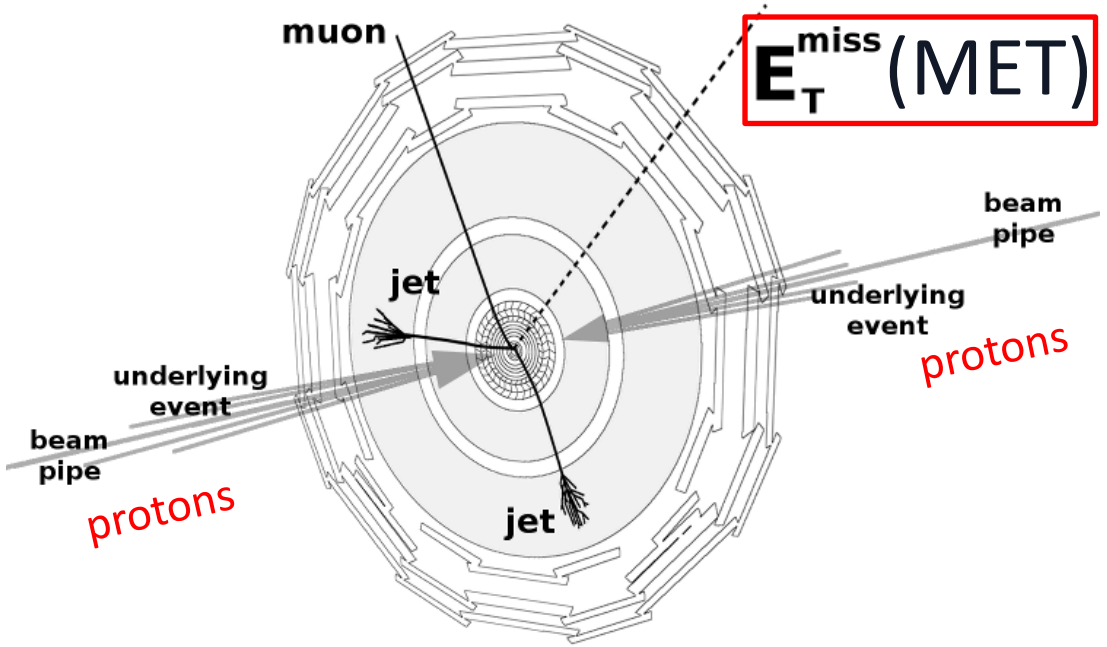


- Xilinx Vivado HLS
  - Creating machine learning algorithms for the CMS level-1 trigger.
- The hls4ml tool has a number of configurable parameters that enable users
  - Customize the space of latency, initiation interval, and resource usage tradeoffs for their application.
  - Perform the optimization through automated neural network translation and FPGA design iteration.

Missing Transverse Energy (MET)

- Energy that is not detected in a detector
  - By conservation of momentum, the sum of transverse momentum should be zero





- MET calculated by PUPPI (Pileup Per Particle Identification) algorithm using CMS detector

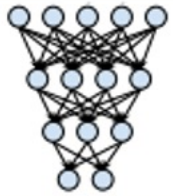
Real-time online event selection using Machine Learning

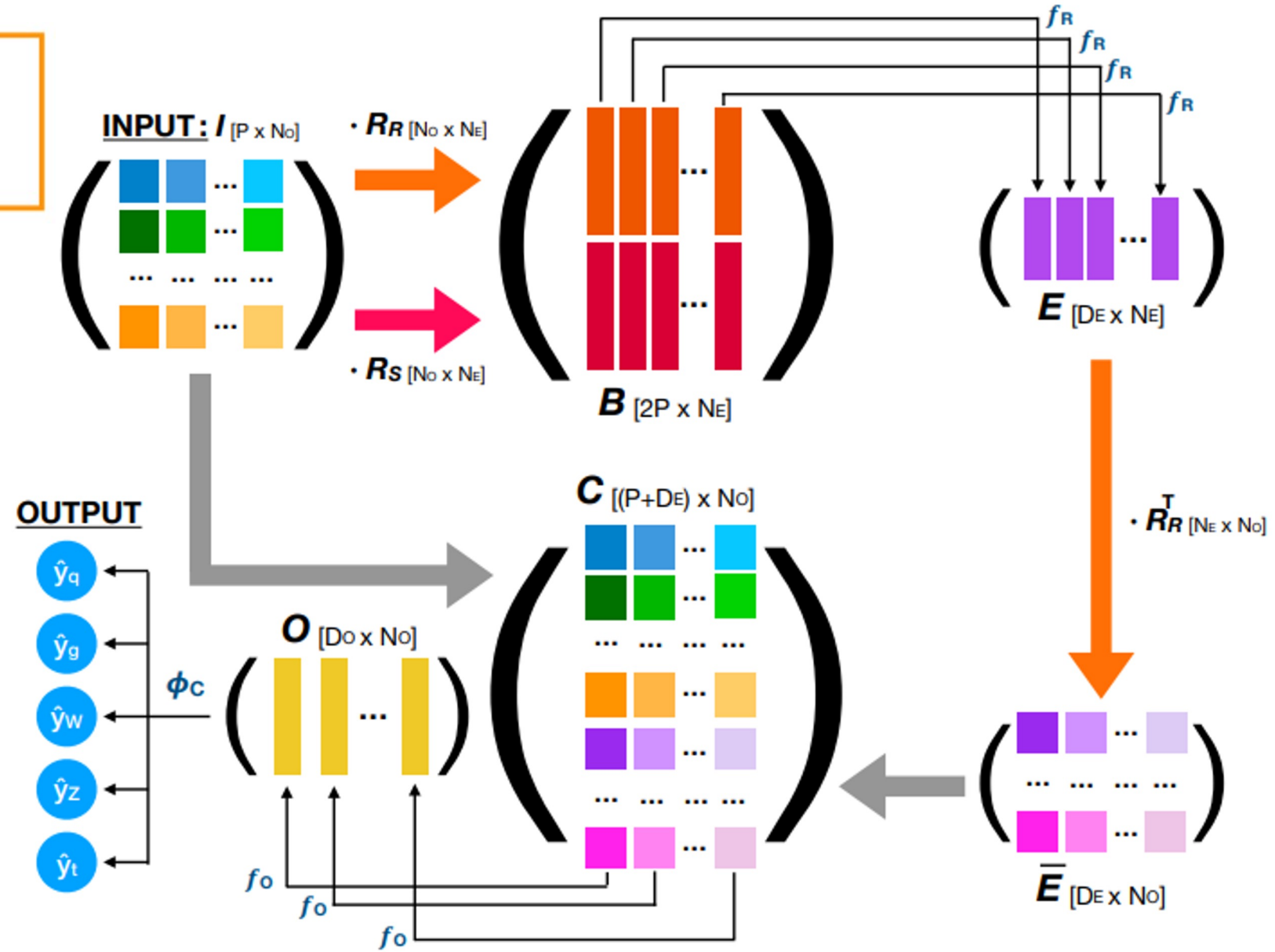
- In the CMS experiment, the missing transverse energy (MET) is reconstructed by the '**PUPPI**' algorithm (Pileup Per Particle Identification).
- To estimate missing transverse energy more precisely, we use a neural network, a model based on '**JEDI-net**'.
  - The '**JEDI-net**' is a jet identification algorithm based on **interaction network**.
- The **interaction network** is model that can perform an analogous form of reasoning about objects and relations in complex systems.
  - This model is based on the **graph neural network (GNN)**.
- The ultimate goal is to apply a trained learning matrix using models to the L1 trigger for Phase-2 CMS Detector Upgrade.
  - Improve measurement of MET at Level-1 Trigger.

JEDI-net Model Architecture (1)

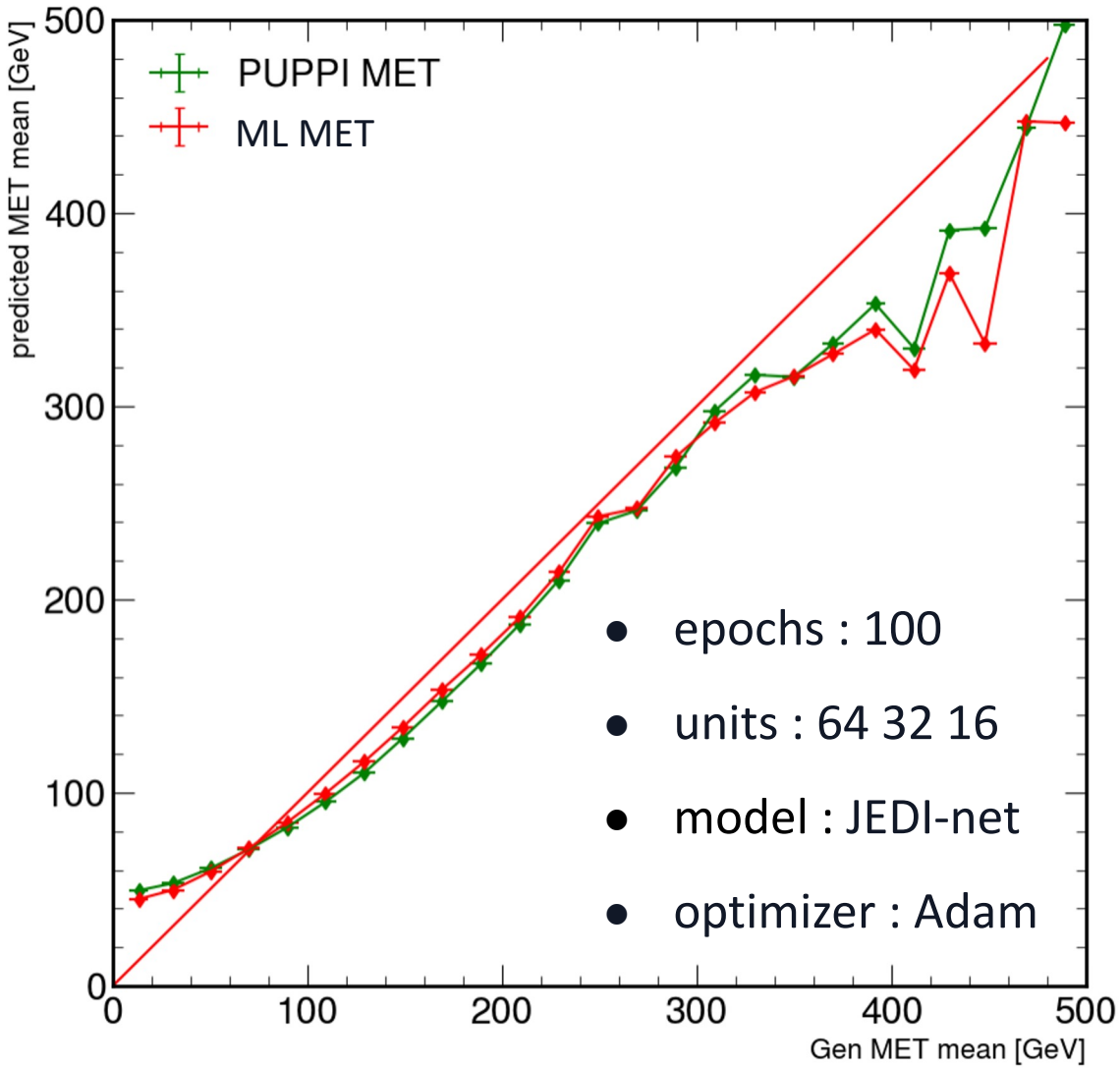
N_O: # of constituents
P: # of features
$N_E = N_O(N_O-1)$: # of edges
D_E: size of internal representations
D_O: size of post-interaction internal representation

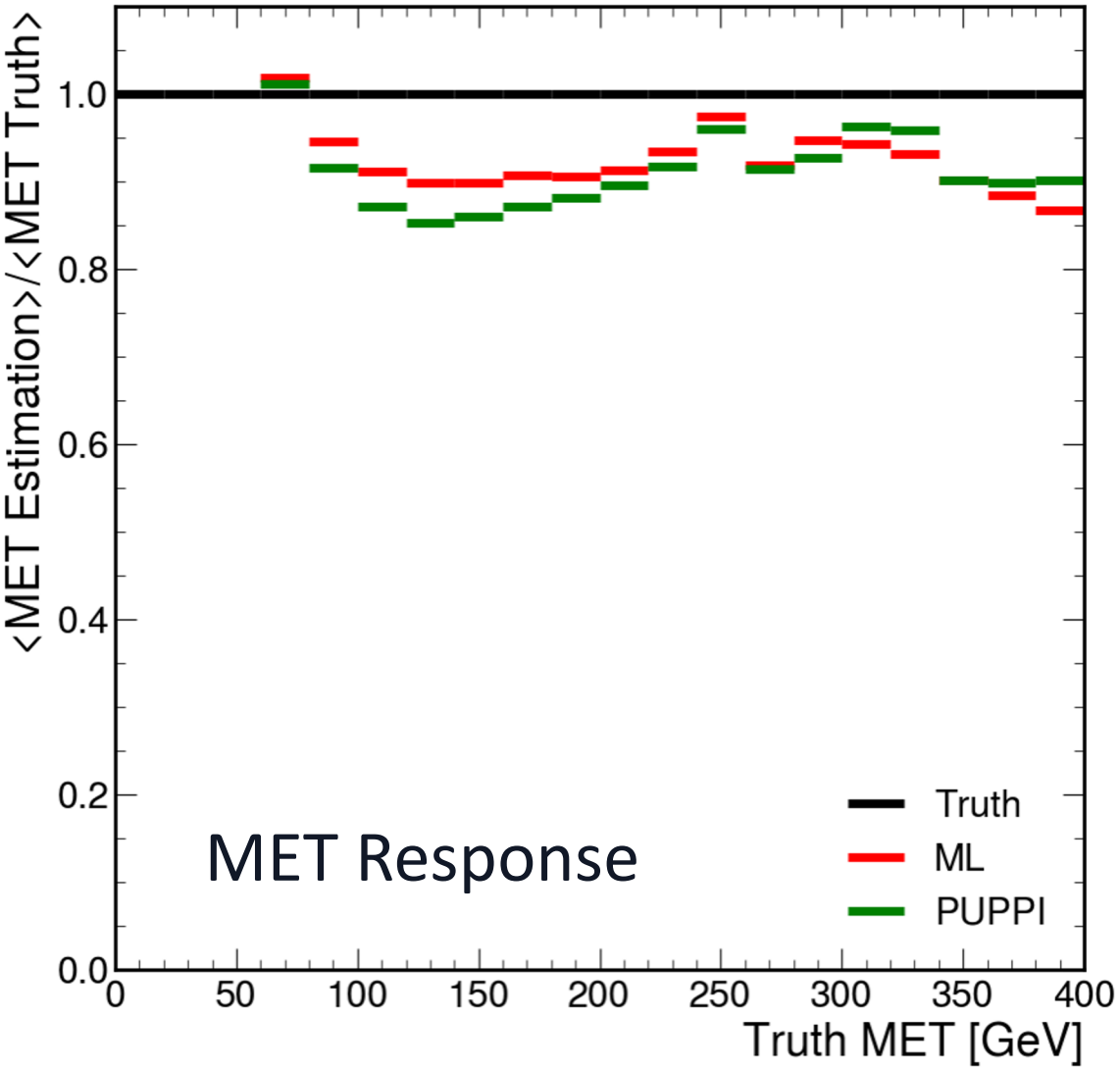
ϕ_C, f_O, f_R expressed as dense neural networks





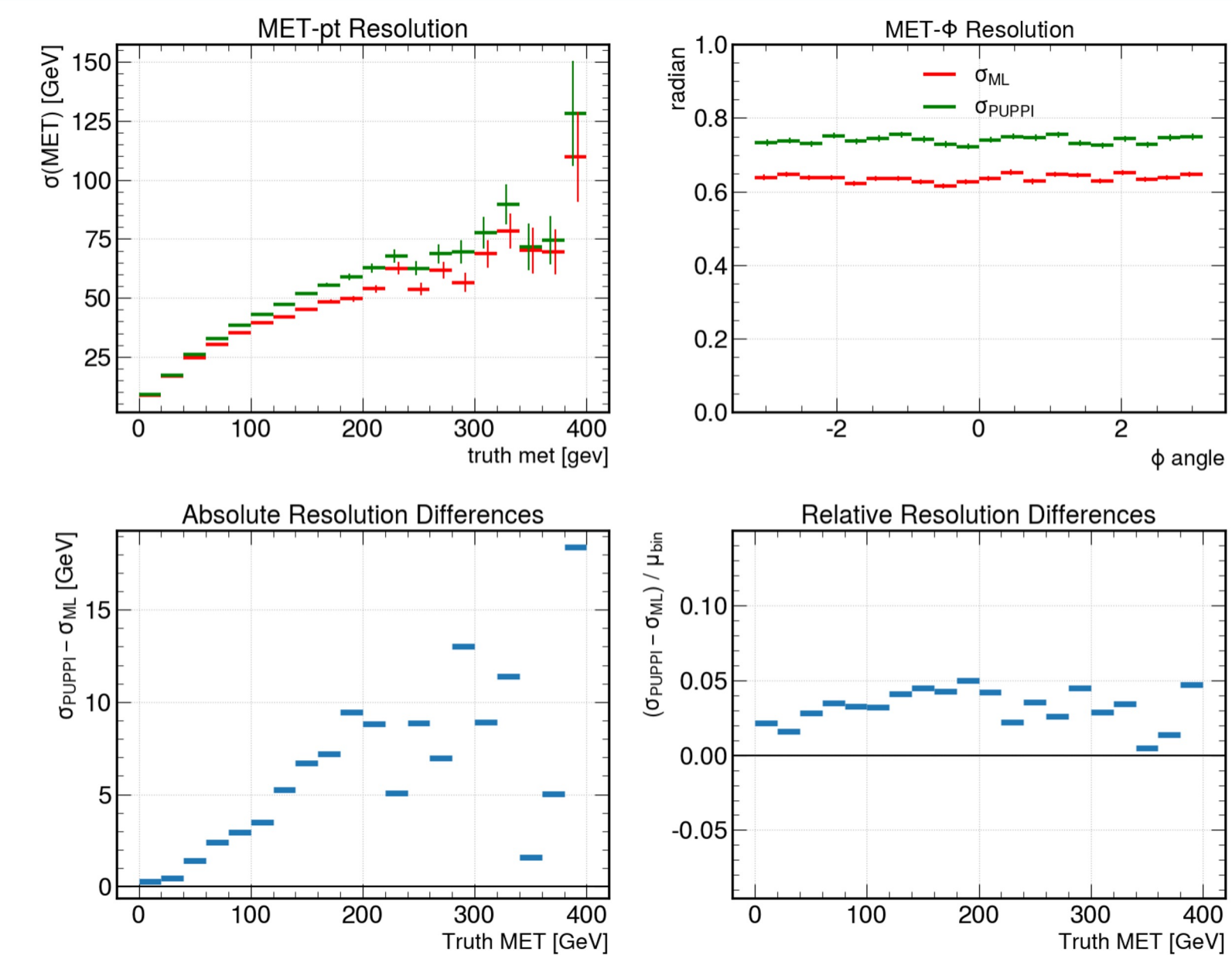
Gen MET vs ML MET





We have verified that in areas with sufficient statistics, ML MET exhibits a closer approximation to the truth value compared to PUPPI MET.

Resolution for MET pT & phi

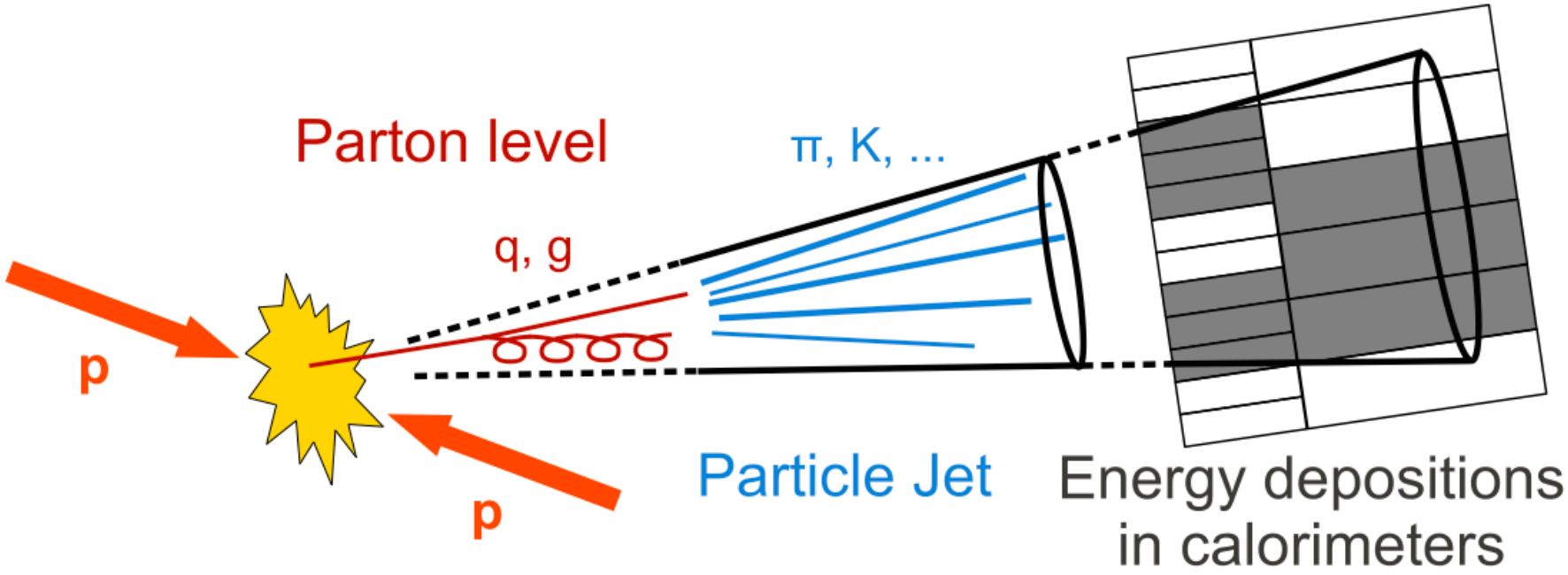


15% improved for MET resolution with respect to PUPPI MET

Transformers for Jet tagging at CMS

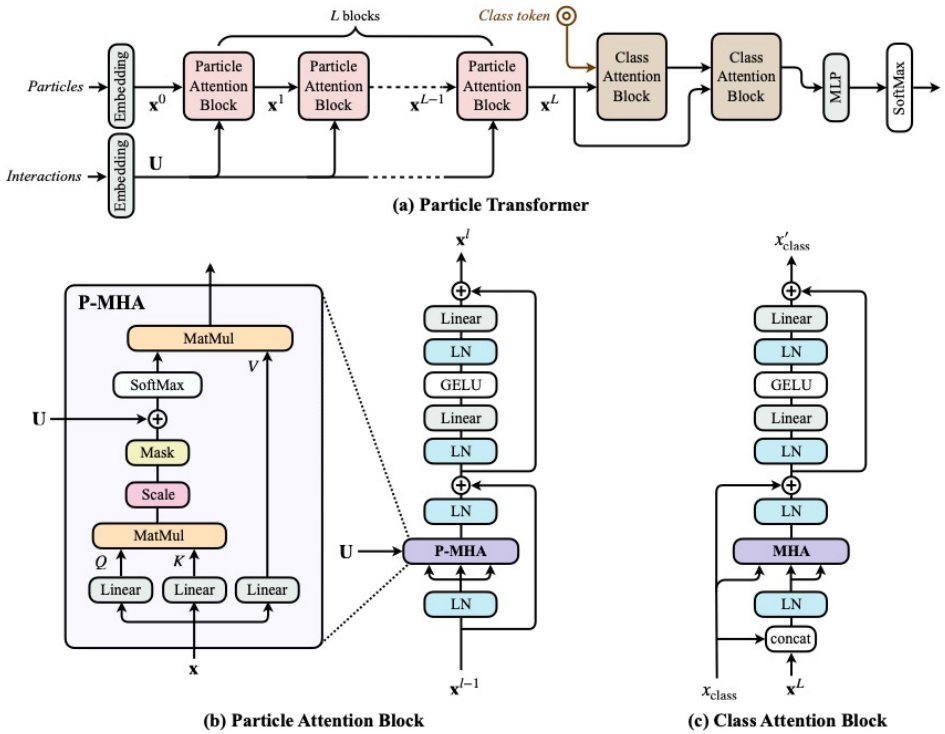
Jets and Transformers in CMS

- A Transformer is a neural network architecture that compares many inputs at once using self-attention.
- A Transformer is the backbone of most large language models, including ChatGPT and Claude
  - For jets, particles act like words
- A jet is a narrow spray of particles made when a high-energy quark or gluon leaves the collision.
- Jet tagging asks what made the jet, b quark, c quark, light quark, or gluon, so it supports Higgs and top-quark studies.
- CMS uses ParticleTransformer, a Transformer-based jet tagger for Run 3



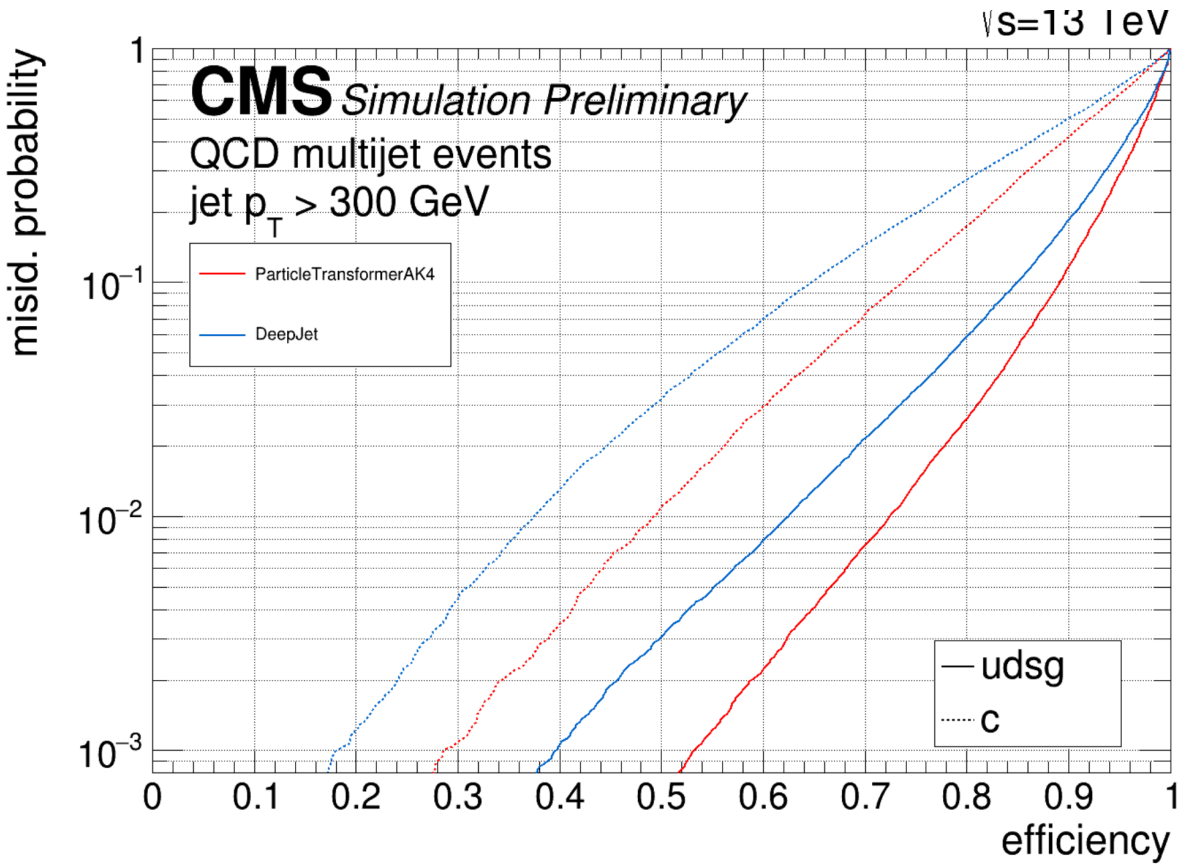
Particle Transformers

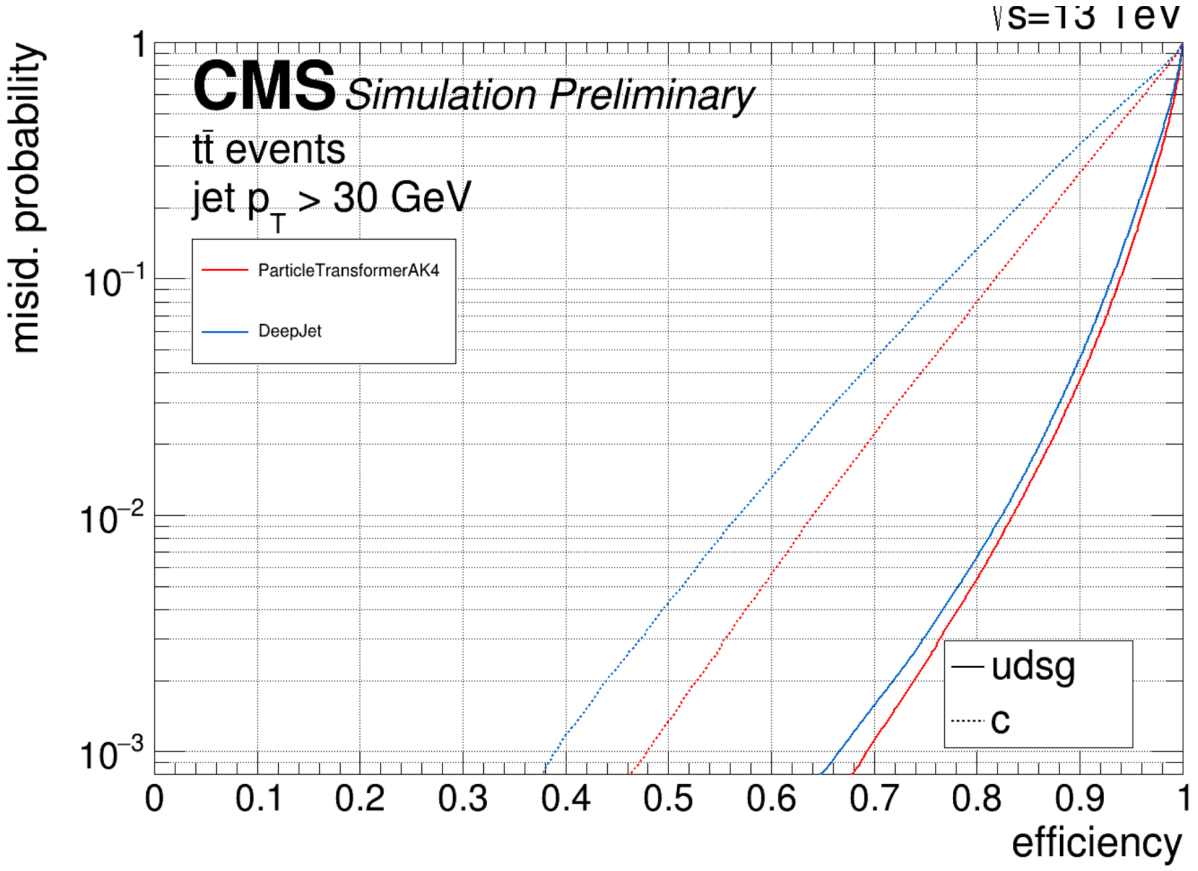
- A Transformer can learn relationships (or physics) between all pairs of elements simultaneously via attention mechanism
- ParticleTransformer features P-MHA (Particle Multi-Head Attention), adding a physics-inspired pairwise interaction matrix U to standard attention weights
  - $\mathbf{P-MHA}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \mathbf{softmax}\left(\frac{\mathbf{1}}{\sqrt{\boldsymbol{d_k}}}\boldsymbol{QK^T} + \boldsymbol{U}\right)\boldsymbol{V}$
- U encodes QCD-motivated features for every particle pair:
  - Angular separation, $\Delta = \sqrt{(y_a - y_b)^2 + (\phi_a - \phi_b)^2}$
  - Relative transverse momentum, $k_T = \min\left(p_{T,a}, p_{T,b}\right)\Delta$
  - Momentum fraction, $\mathrm{z} = \frac{\min(p_{T,a}, p_{T,b})}{p_{T,a}+p_{T,b}}$
  - Invariant mass, m
- In CMS, secondary vertices are also included as inputs alongside jet constituents



Results for b-tagging Performance

ParticleTransformerAK4 (Red) vs DeepJet (Blue)





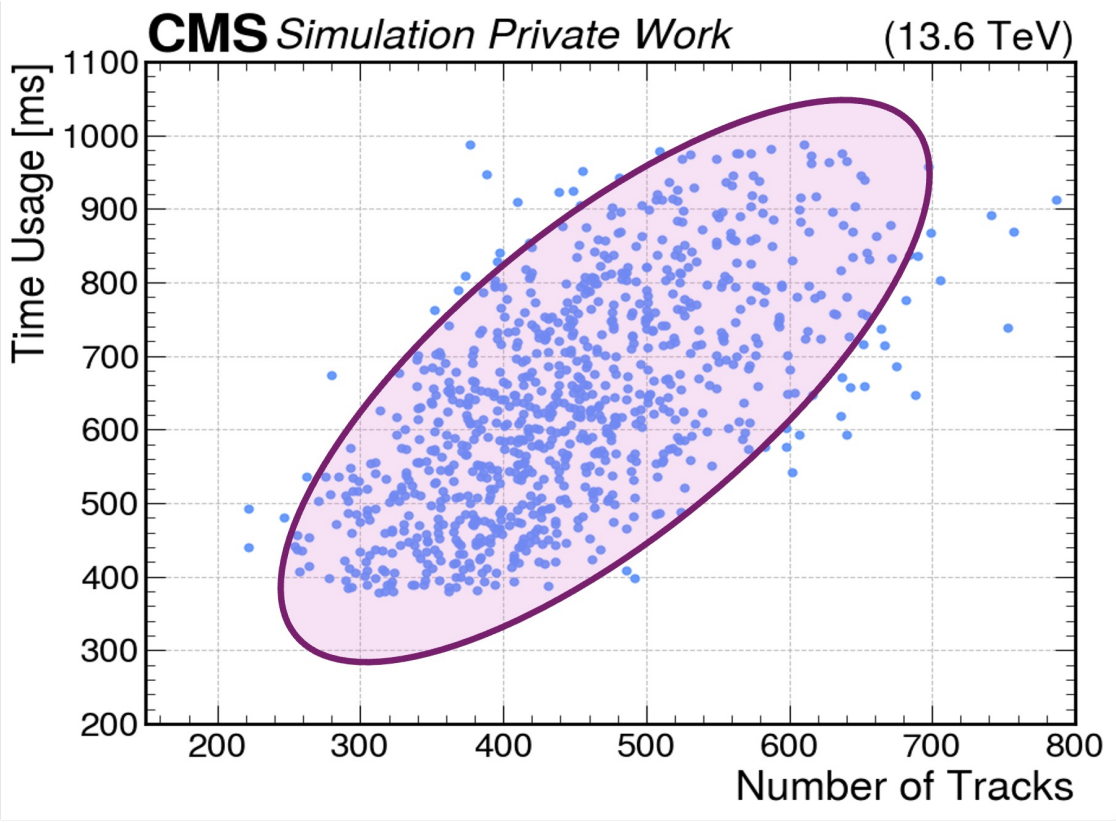
- Lower misidentification rate for the same b-tagging efficiency
- The improvement is particularly significant in the high p_T region.

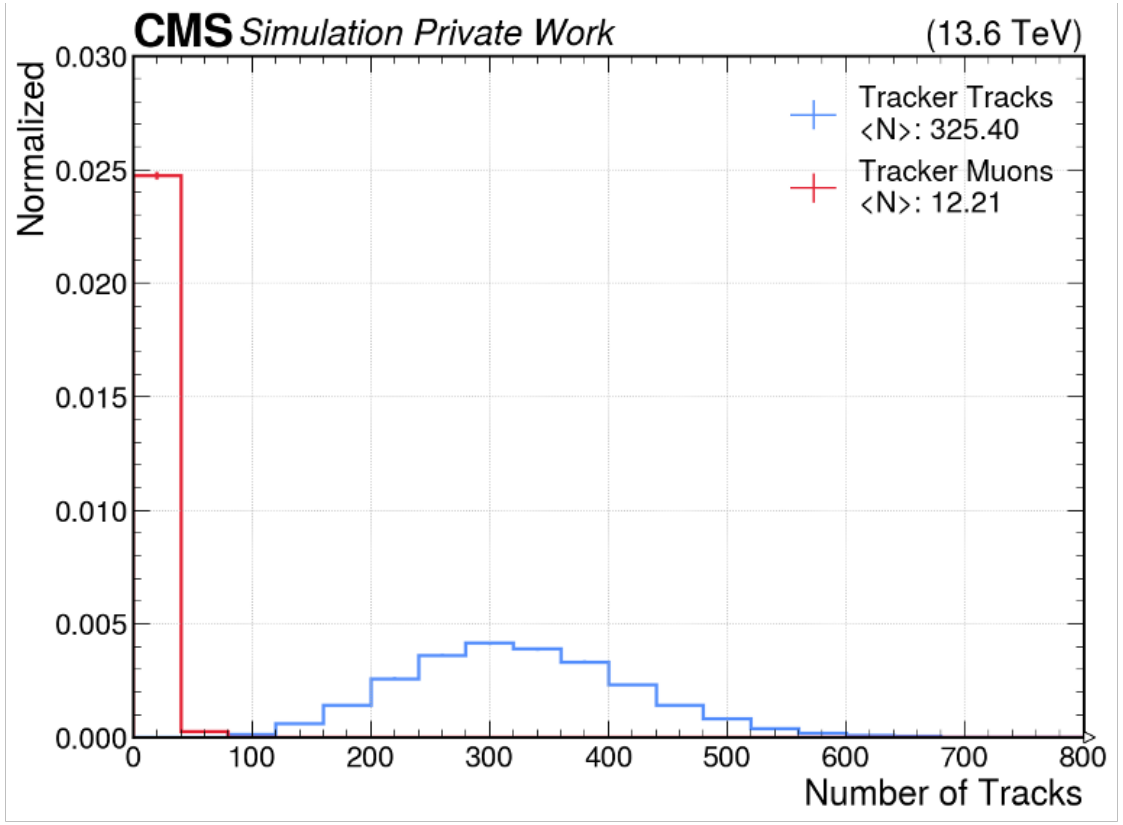
Transformers for Tracker Muon Reconstruction at CMS

Team. Seungjin Yang, Jongwon Shin, and Junghwan Goh (Kyung Hee University)

Motivation: Tracker Muon Reconstruction

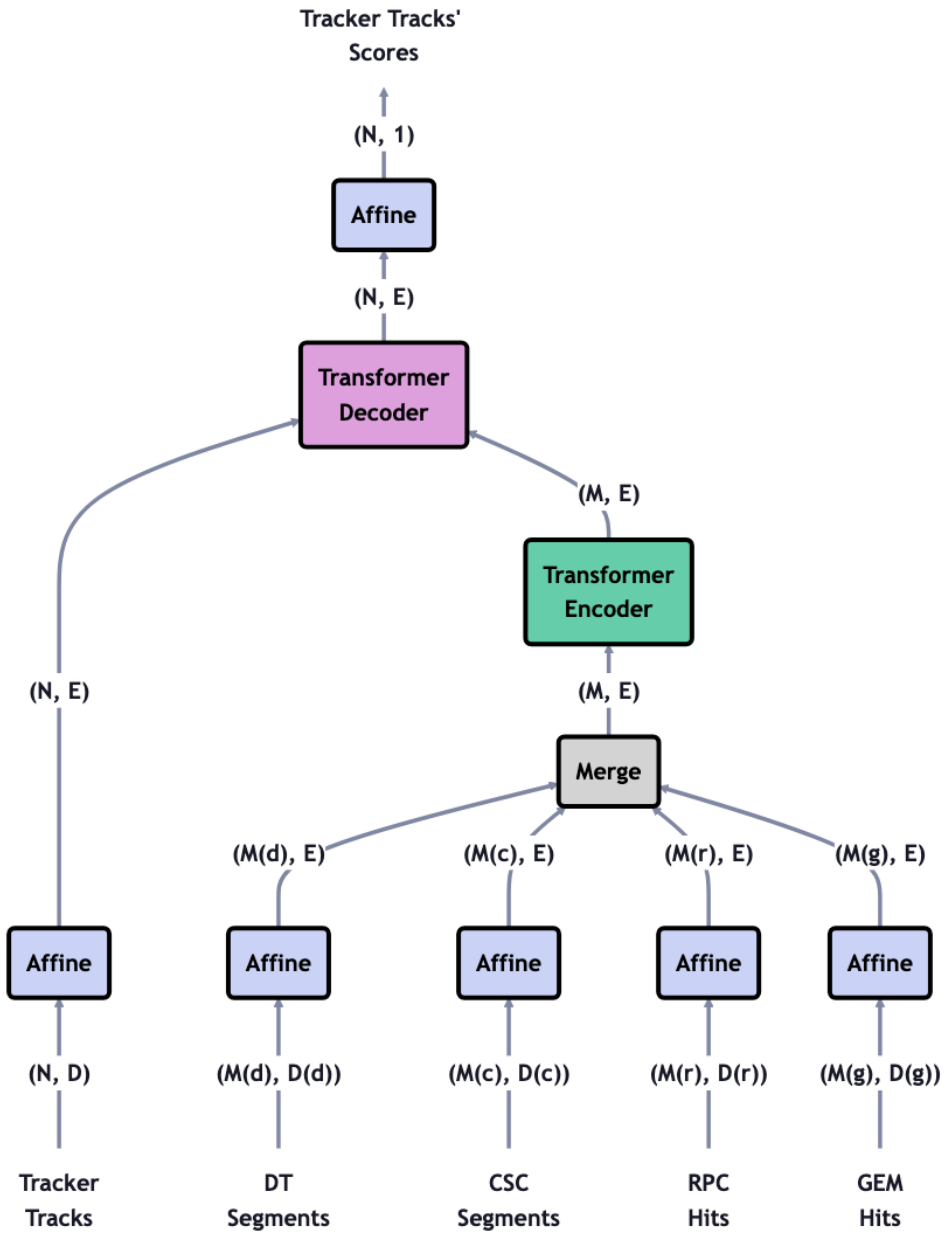
- To reconstruct muons, CMS extrapolates charged particle tracks in tracker system to muon detector system, and attempts to match each track to measurements in muon detector system
  - Extrapolation is computationally expensive
  - Reconstruction time scales with track multiplicity
  - Only a small fraction of tracker tracks are promoted to tracker muons
- Idea: Use a DL model to pre-select likely muon candidates before the expensive matching step





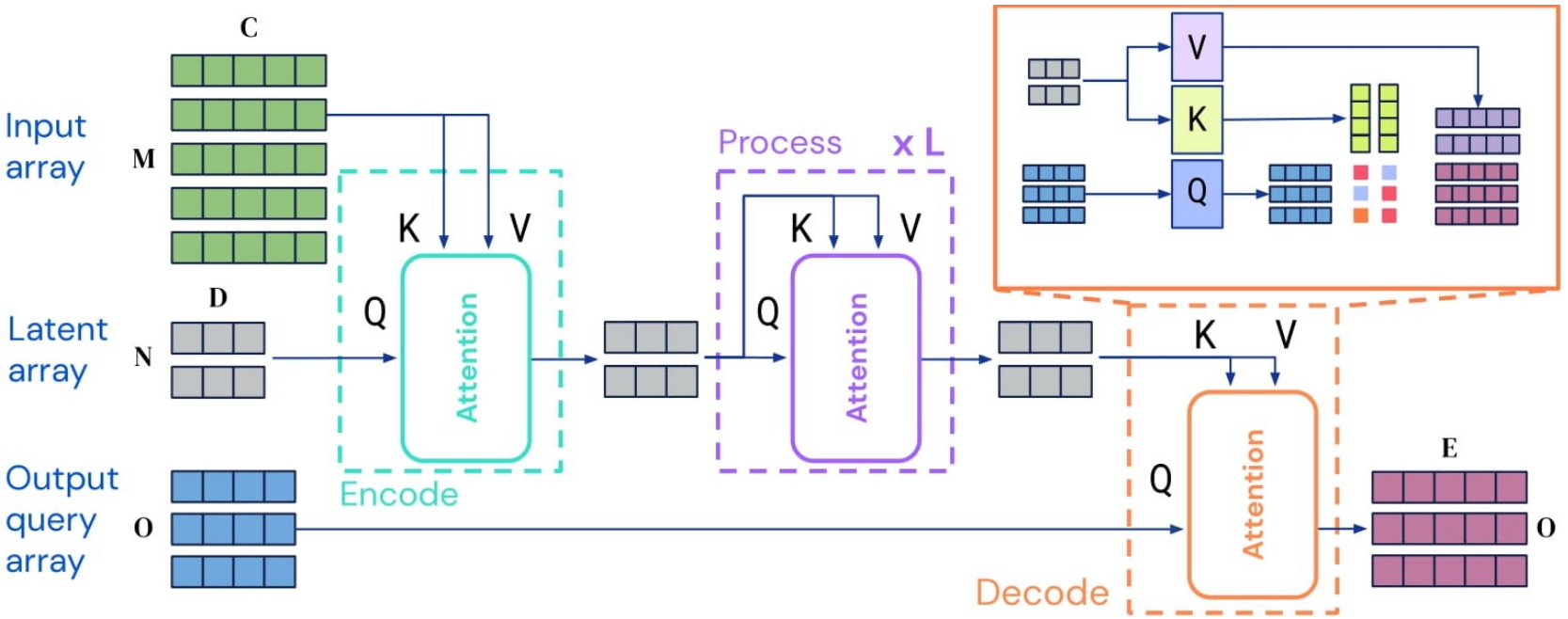
Transformer for Track Pre-selection

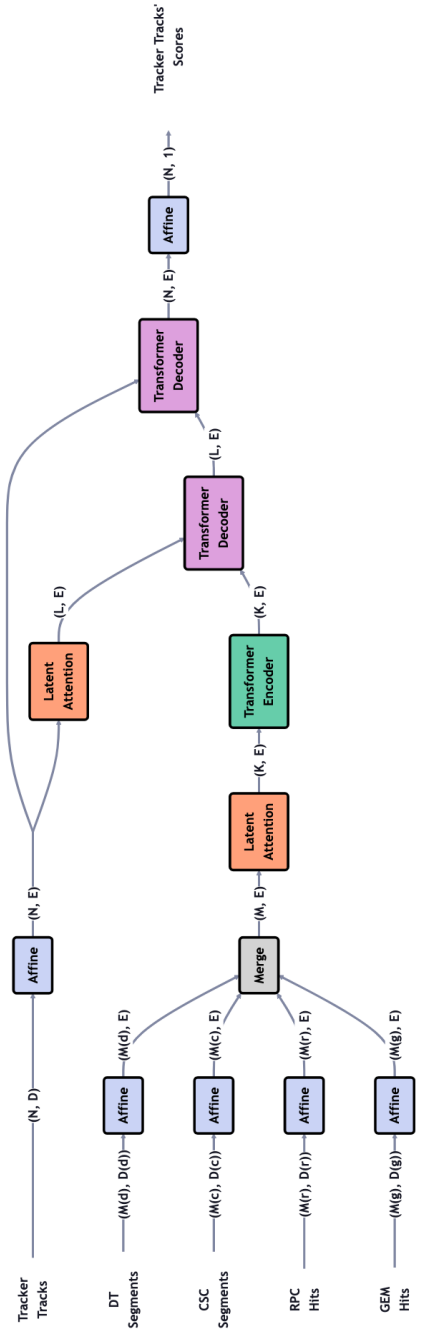
- Input features
  - Tracker track kinematics
  - muon detector segment/hit positions
- Transformer
  - Encoder captures dependencies among muon detector measurements
  - Decoder attends to encoded detector features for each tracker track
  - Final linear layer
- Binary classification
  - muon track vs non-muon track
- Dataset
  - muon gun simulation (Run 3 conditions, 640k events)



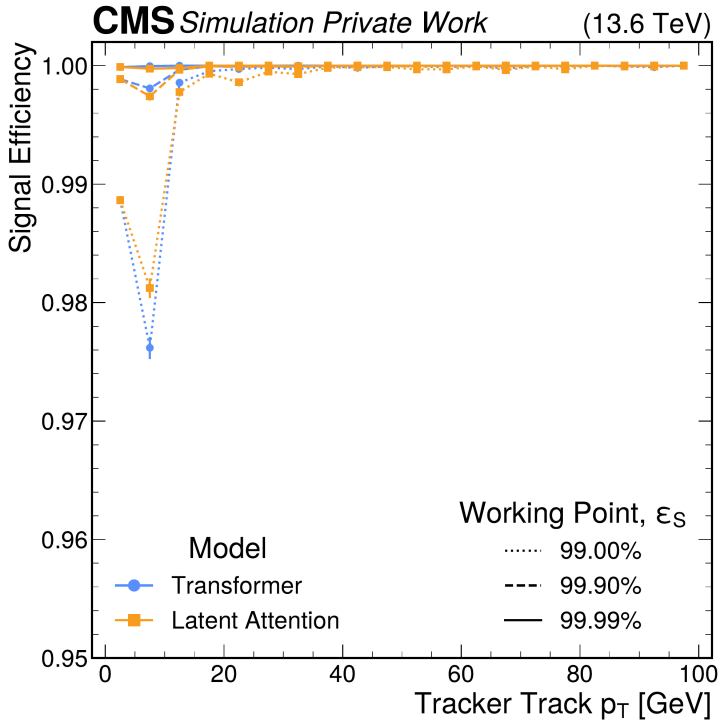
Latent Attention: Linear-complexity Alternative

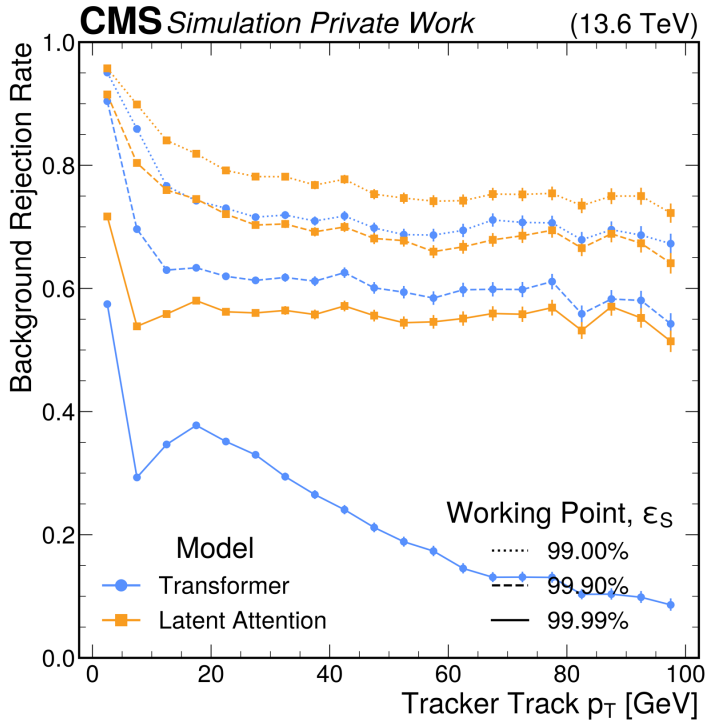
- Standard self-attention scales as $O(N^2)$ → Expensive for large track multiplicities
- Latent attention uses a small fixed set of learnable Z vectors to compress the input into a compact representation, reducing complexity to $O(NZ)$ and $O(Z^2)$
- In Run3,
  - $\langle N_{\text{tracker track}} \rangle \approx 325 \rightarrow \mathrm{Z} = 64$:
  - $\langle N_{\text{muon segments}} \rangle \approx 36 \rightarrow \mathrm{Z} = 16$:

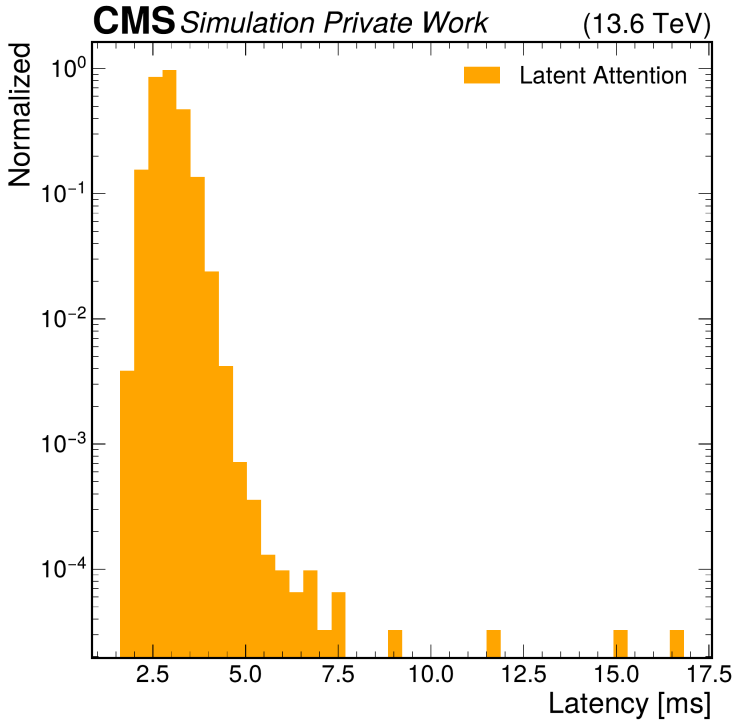




Results for Tracker Muon Reconstruction





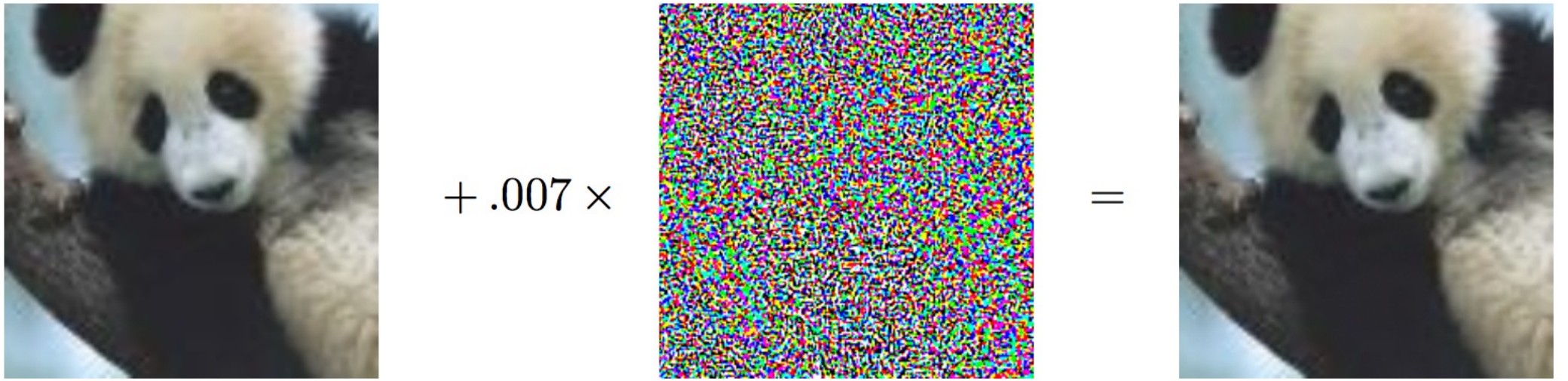


- Both models maintain high non-muon track rejection across the full p_T range, with minimal efficiency loss
- In Run 3, tracker-muon reconstruction takes 400–1000 ms per event.
- The latent-attention model adds only ~3 ms of average CPU inference overhead, even without batch processing.

AI Safety at CMS

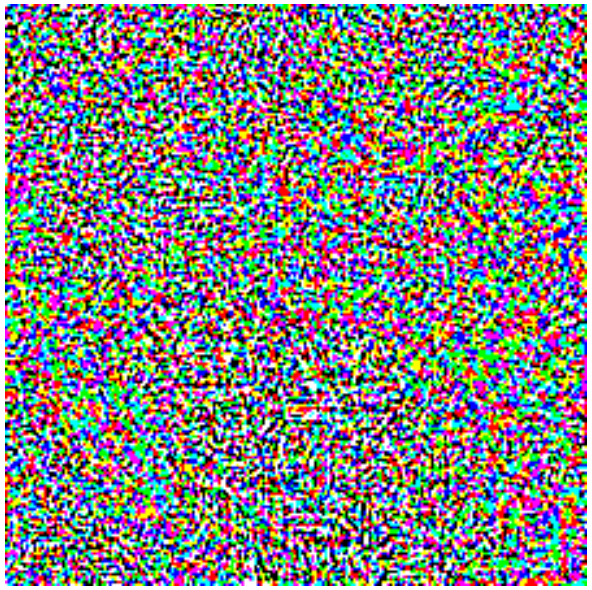
Adversarial Training

- Jet flavor tagging algorithms are trained on simulation but evaluated on both data and simulation.
  - Data/simulation differences must be calibrated.
  - **Adversarial training** reduces these discrepancies before calibration by exposing the model to perturbed inputs during training.
- CMS studied adversarial training on the DeepJet *b*-tagging algorithm using Run 2 and Run 3.



Small perturbations can change AI classifications

(a) “panda”
57.7% conf.



(b) “adversarial perturbation”
$\epsilon \cdot \mathrm{sgn}(\nabla_x J)$, where $\epsilon = 0.007$

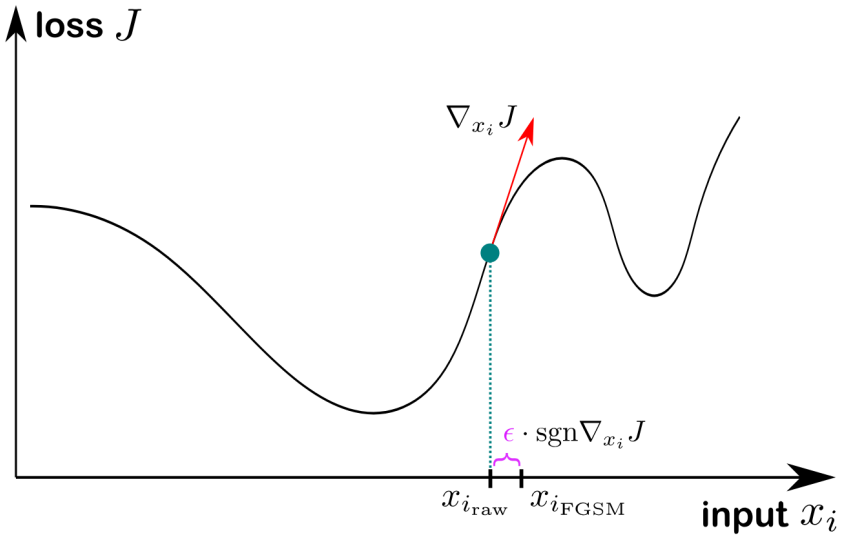
(c) “nematode”
(for comparison)

(d) “gibbon”
(for comparison)

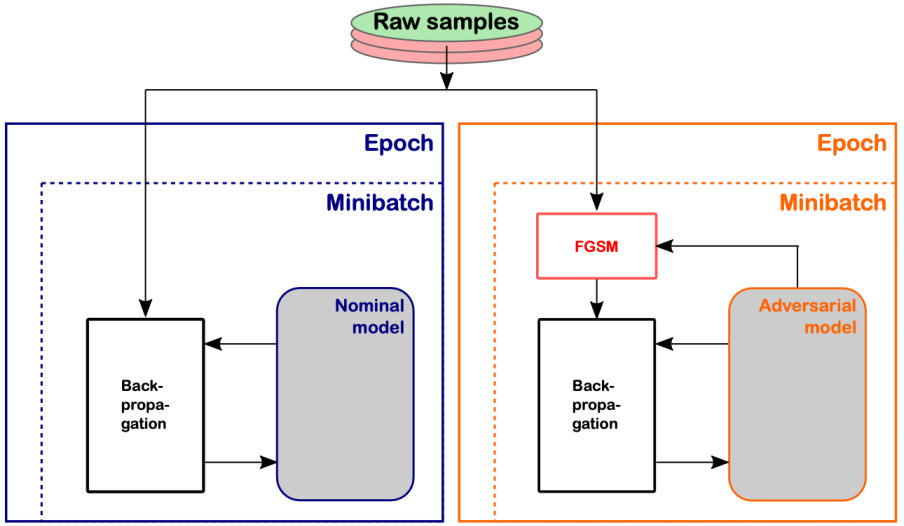
- Adding an imperceptible perturbation ($\epsilon = 0.007$, the least-significant bit of an 8-bit image) causes an incorrect classification in GoogLeNet.
  - The model labels the panda as a gibbon with 99.3% confidence.
  - The perturbation pattern is visually similar to a nematode.
- Small, structured perturbations can move a neural network toward a confident wrong label.

Adversarial training - CMS DeepJet

- DL models can be fooled by tiny, intentional changes to their inputs.
- ***Fast gradient sign method (FSGM)*** generate the worst-case perturbation that nudge each input slightly in the direction that most increases the model's error.
- In the top sketch, "loss" means error: the red arrow shows the small input change that makes the error grow.
- ***Adversarial training*** adds perturbated examples during training, so the model learns not to overreact.
- For CMS DeepJet, the adversarial training is applied to continuous inputs such as:
  - Particle candidate
  - Secondary vertex
  - event-level variables
- Keep normal performance while making the model more robust to realistic shifts in data.

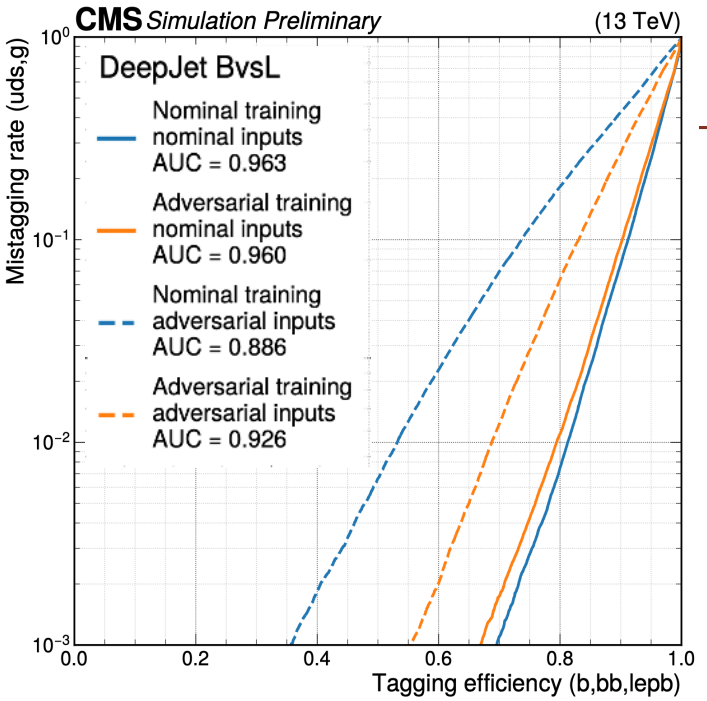


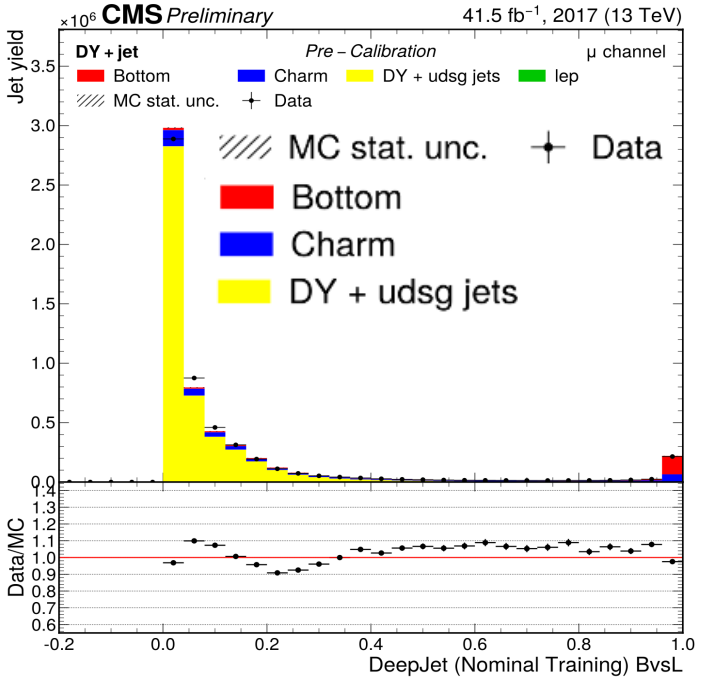
Defense — Adversarial training
(instead of nominal training)



Results: Robustness in Simulation

JS divergence (a.u.) DeepJet 2017 (13 TeV)	udsg jets			c jets			b jets		
	BvsL	CvsB	CvsL	BvsL	CvsB	CvsL	BvsL	CvsB	CvsL
Nominal training	0.000358	0.000353	0.000947	0.002632	**0.002350**	0.002263	0.003506	**0.002528**	0.004820
Adversarial training	**0.000063**	**0.000058**	**0.000466**	**0.001887**	0.003074	**0.001766**	**0.003329**	0.003005	**0.002924**



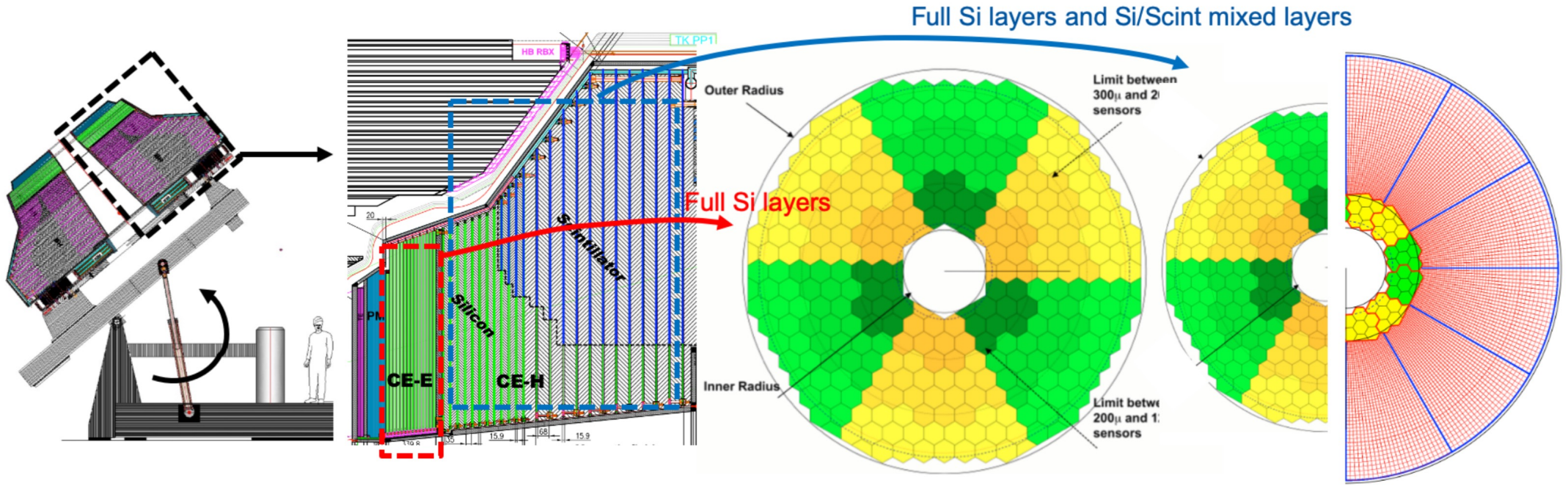


- Adversarial training (orange) preserves tagging AUC on nominal inputs
- The Jensen–Shannon divergence quantifies the difference between two distributions
  - JS divergence = 0 means perfect data/MC agreement
- Adversarial training improves agreement in 7 out of 9 cases
  - Light jets show near-perfect agreement
- **Summary: adversarial training helps to mitigate discrepancies between data and MC**

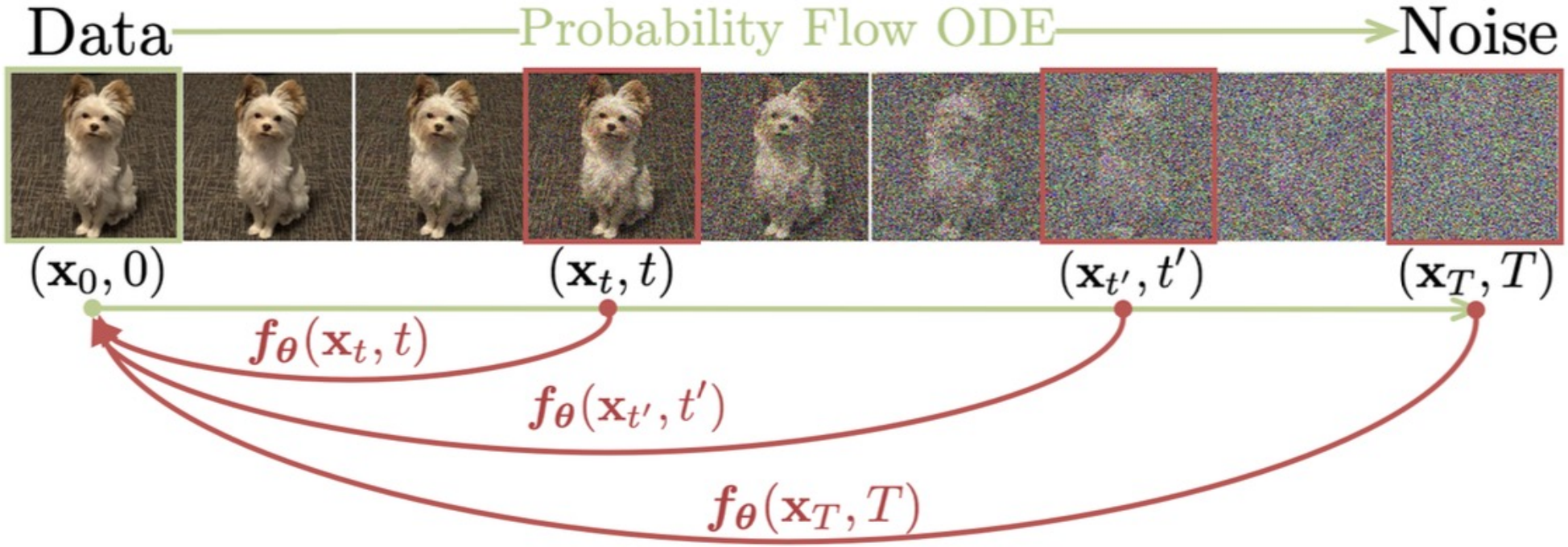
Fast Simulation using Generative AI at CMS

AI speeds HGCAL simulation at CMS

- CMS needs faster detector simulation for the HL-LHC era, where High-Granularity Calorimeter (HGCAL) adds about 3 million hexagonal silicon sensors.
- Geant4 remains accurate, but is too slow to produce the event volumes needed for reconstruction and analysis studies.
- CMS adopts CaloClouds II, which is a diffusion model that generates particle showers as geometry-independent point clouds for the CMS Phase-2 HGCAL.
- The study also models hit timing, a key input for pileup mitigation and HGCAL reconstruction performance.
- Goal: preserve physics fidelity while reducing simulation time enough to support large-scale CMS workflows.

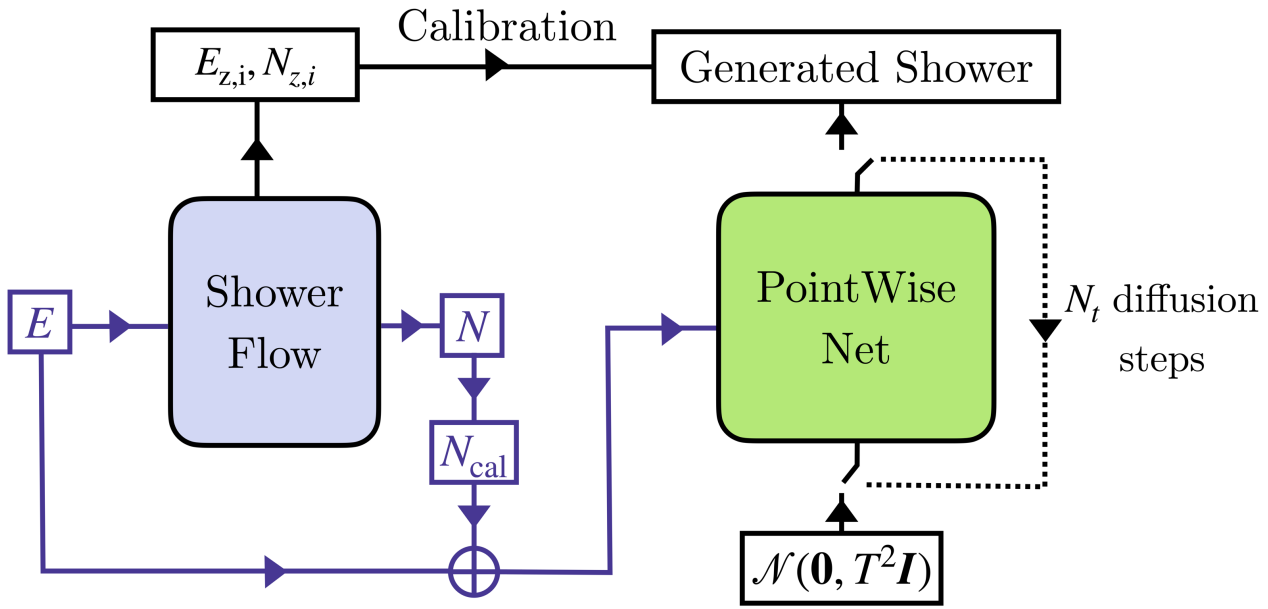


Diffusion Models

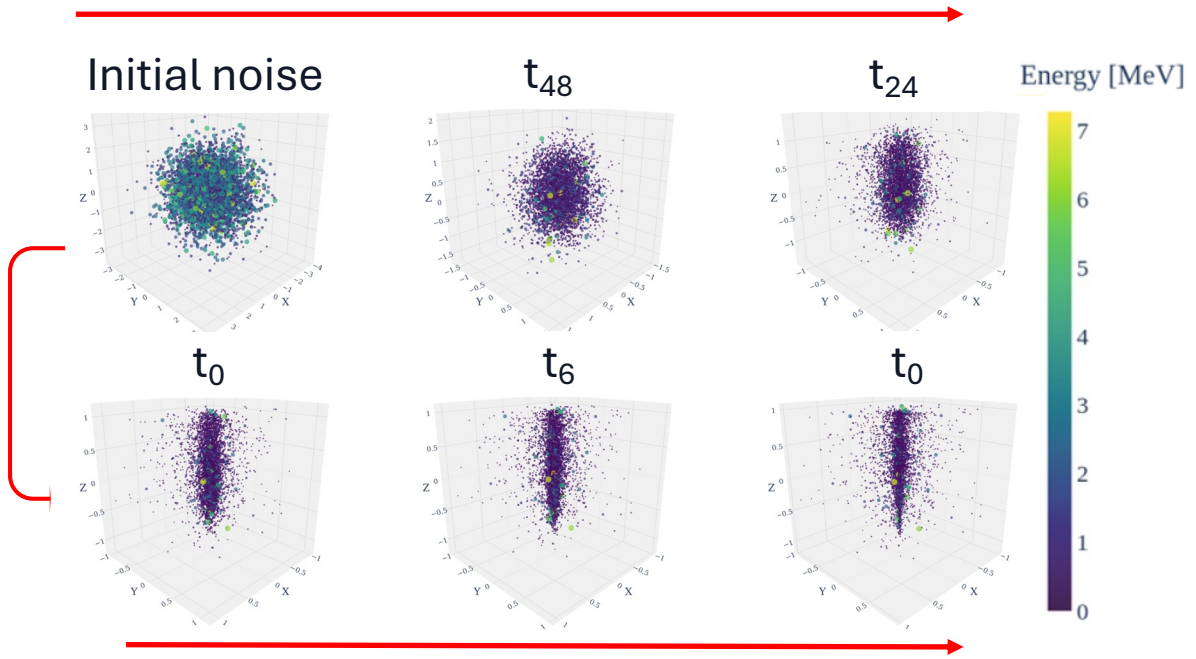


- Diffusion models are a kind of generative Ais.
- During training, data is gradually corrupted by adding small amounts of noise.
- The diffusion model learns to reverse this process by removing the noise step by step.
- The model learns how to reverse this process by predicting and removing noise step by step.
- After training, the model can generate new data by starting from random noise and progressively denoising it.

CaloClouds



Sampling procedure of the CaloClouds II.

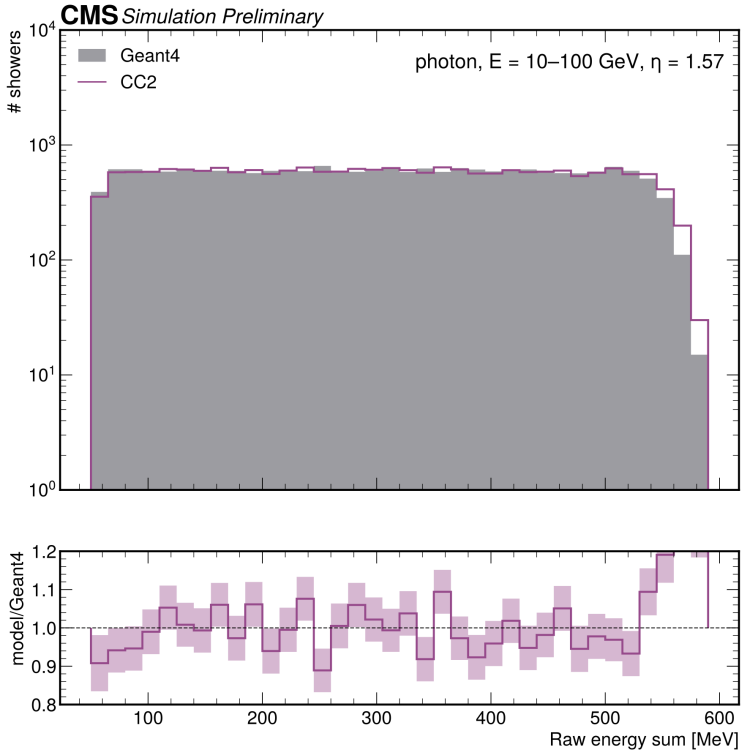


Reverse diffusion process for a photon shower point cloud

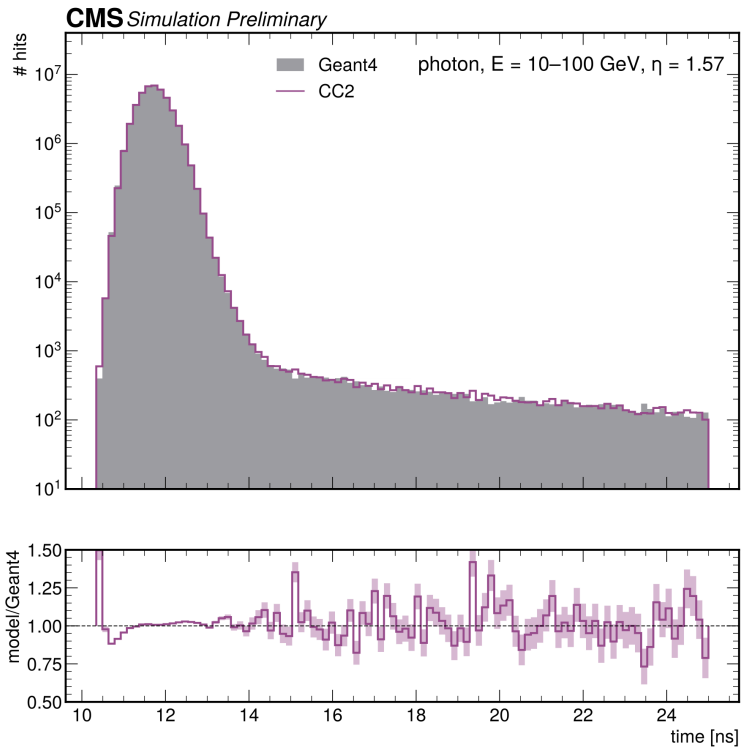
- CaloClouds II is a continuous-time diffusion model and achieves fast sampling speed using one-step generation through consistency distillation.
- CMS adopts the CaloClouds Ⅱ (CC2) architecture for HGCAL photon showers including timing information.

Results: HGCAL photon showers

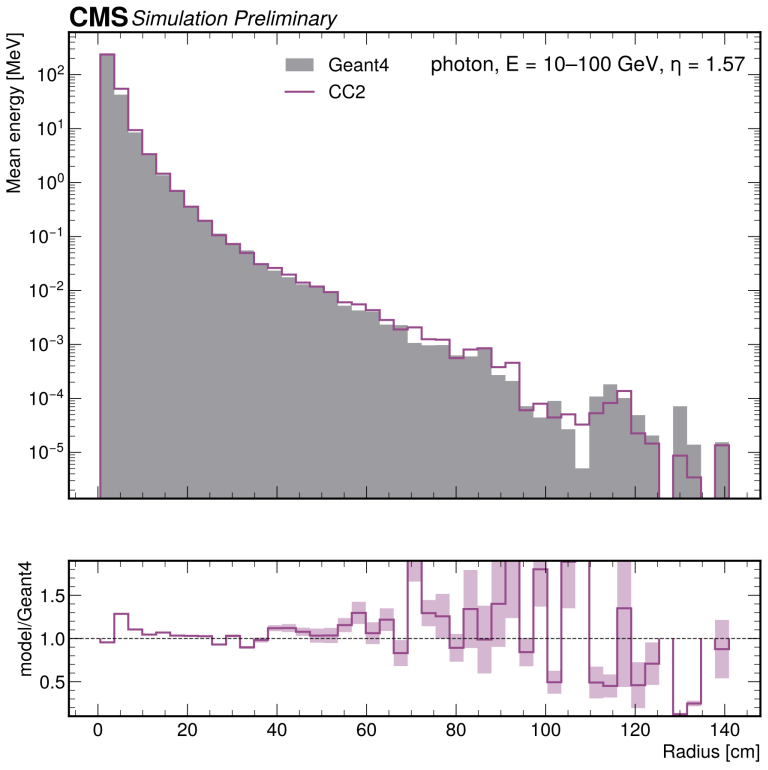
- CMS compares 20k Geant4 and CC2 HGCAL photon showers after projecting generated point clouds to nearest HGCAL cells.
- Very good agreement is reported for raw shower energy and radial shower profiles.
- Timing distributions are also well reproduced, especially in the core timing region.



raw shower energy



core timing region



radial shower profile

Results (Speed)

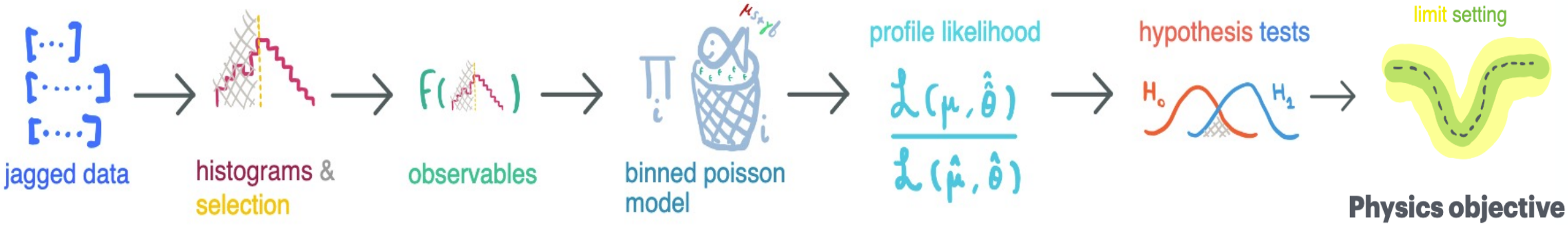
Hardware	Simulator	Batch Size	Time / Shower [ms]	Speed-up
CPU	Geant4	–	3153.64 ± 1490.68	× 1
CPU	CaloClouds II (CC2)	1	714.33 ± 288.22	× 4
GPU	CaloClouds II (CC2)	64	31.04 ± 0.67	× 102

- CPU: a single core of an AMD EPYC 7543 32-core
- GPU: NVIDIA A100 with about 40 GB VRAM
- Benchmark
  - incident photon energy uniformly distributed between 10 and 100 GeV
  - 2,000 showers, 10 repeats,
- CC2 diffusion-based point-cloud simulation reproduces key HGCAL photon-shower observables while delivering substantial speed-ups over Geant4 (up to ~ ×100 on GPU)

Differentiable Analysis

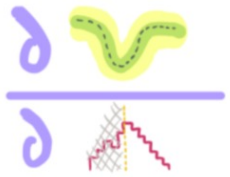
Mohamed Aly and Lino Gerlach (PRINCETON UNIVERSITY) , Andrzej Novák (MIT) from CHEP 2026

Differentiable analysis



Taken from Lino Gerlach's talk

- A typical HEP analysis maps events through selection, an observable or neural network score, histograms, and likelihood-based inference.
- Analysis optimization usually tunes cuts and compare observables by hand or with proxy metric, then validates the statistical result afterward.
- **Differentiable programming** is a paradigm in which numerical programs are written so that their outputs are differentiable with respect to their inputs, enabling end-to-end gradient-based optimization via automatic differentiation through frameworks such as JAX, PyTorch, and TensorFlow.
- **Differentiable analysis = Differential programming for HEP anaylsis**
  - The field has moved from toy demonstrations toward reusable JAX-based components for realistic HEP workflows.
  - It needs gradients from the final objective back through every stage of the workflow.

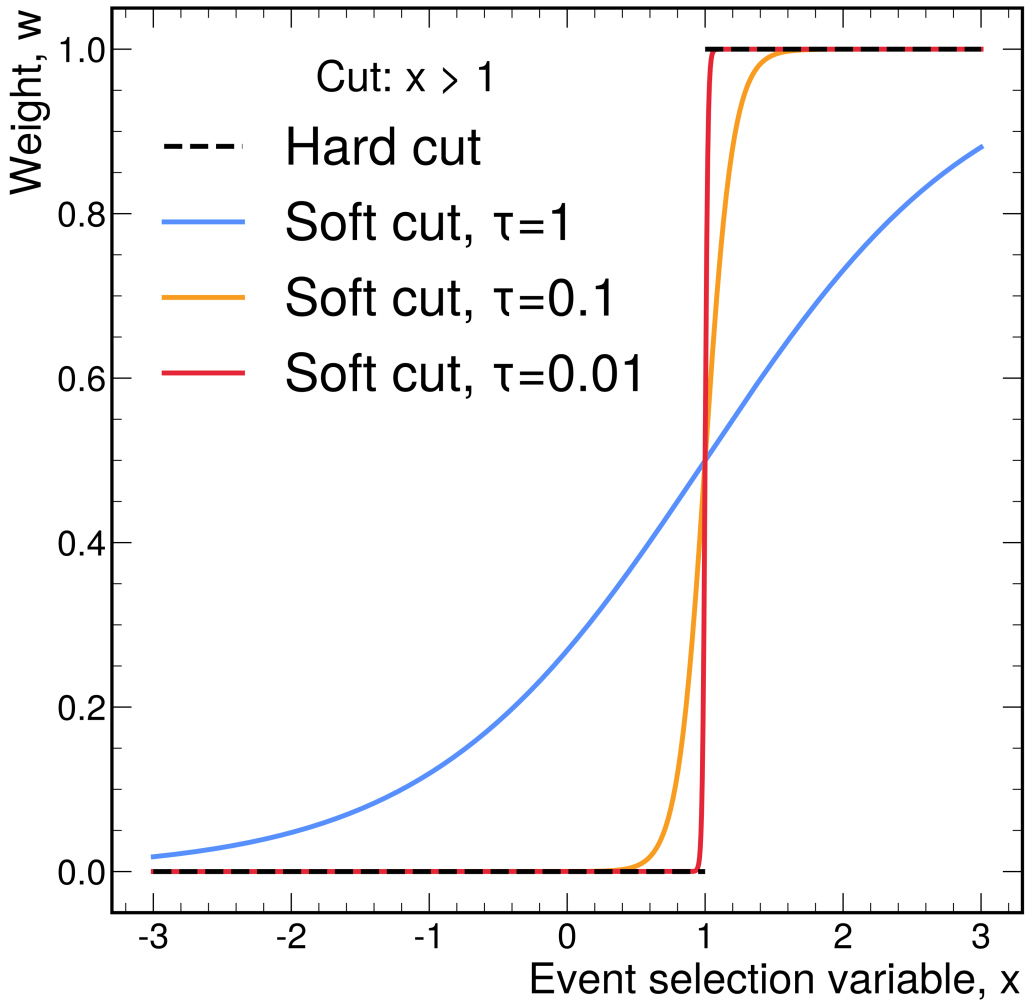


Differentiable Event Selection

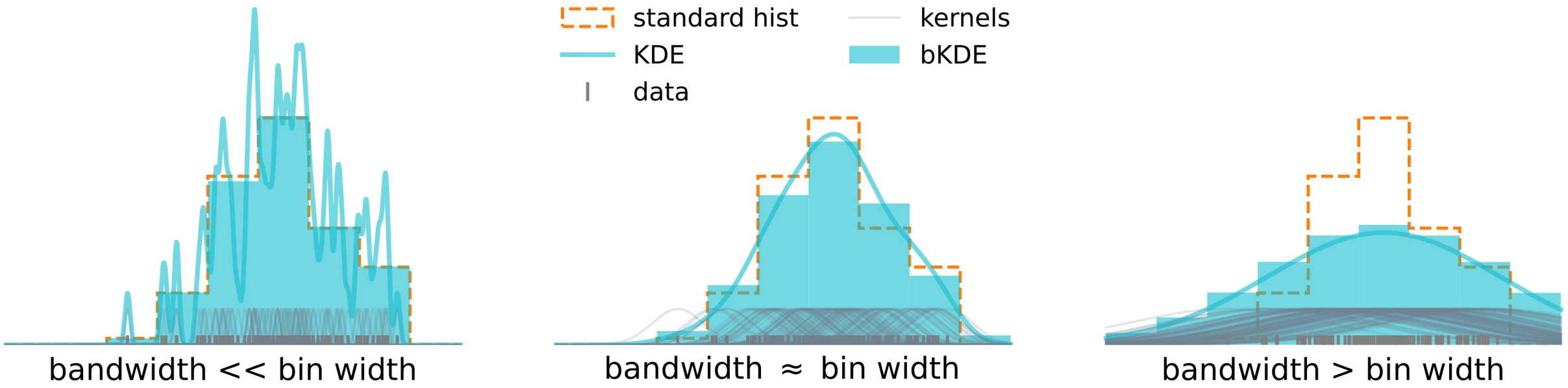
- A hard cut behaves like a step function: events pass or fail, but the threshold has zero or undefined gradient almost everywhere.
- A soft cut replaces the step with a smooth sigmoid, e.g.

$$w(x) = \frac{1}{1 + e^{-\frac{x-c}{\tau}}}$$

- The temperature τ controls the tradeoff: small values resemble a hard cut, while larger values give smoother and more useful gradients



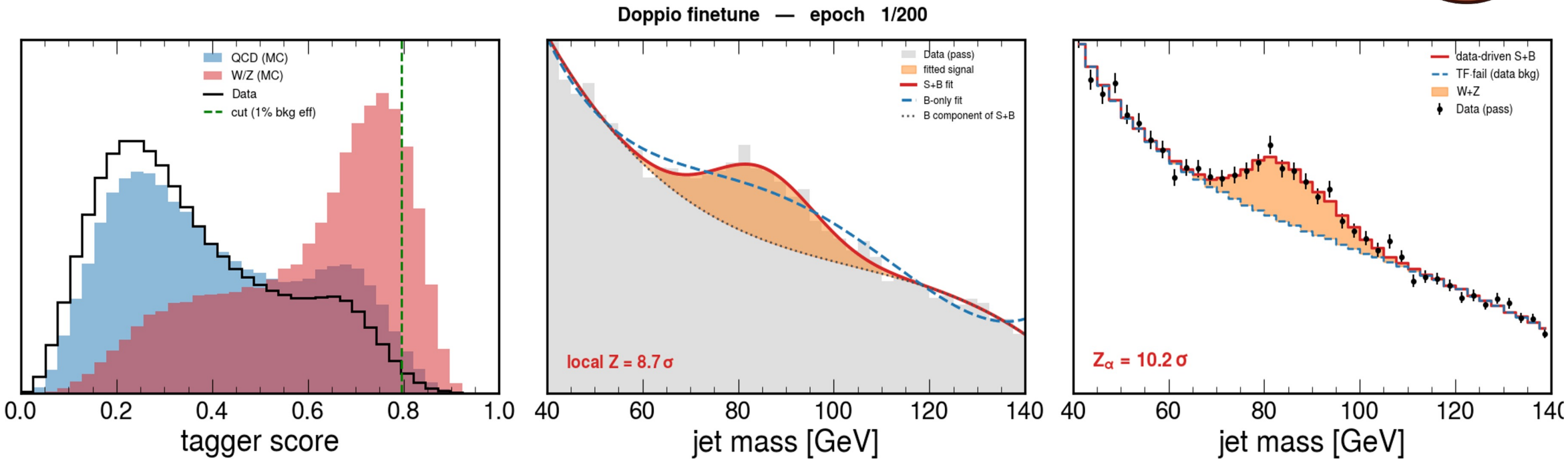
Differentiable Histogram



- Histograms connect a learned summary statistic to template likelihoods, but hard bin counts are discontinuous as events cross bin edges.
- Binned kernel density estimation (bKDE) produces histogram-like templates by integrating each smooth kernel between bin edges.
- Bandwidth controls the bias–stability tradeoff: small bandwidth recovers a hard histogram with noisy gradients, while larger bandwidth trains more smoothly but biases the template.

Early studies show sensitivity gains when the fit defines the loss

- Anomaly detection, W/Z tagging, top-Higgs, and open-data demonstrations differ in details, but share the same training signal.
- evermore: Differentiable likelihoods
  - CMS ttH benchmark: Optimizing a DNN on expected μ(ttH) uncertainty decreases the POI uncertainty.
  - σ_μ: 6.9 → 1.9
- GRAEP: Differentiable analysis Framework
  - CMS Z′ → ttbar study optimized b-tag and kinematic cuts with classifier retraining
- DOPPIO: bump hunt using differentiable analysis
  - LHC Olympics bump hunt: 1.7σ → ~10σ
    - Local significance grows as the anomaly score is trained through the fit
  - CMS Open Data W/Z tagging: 11.1σ → 16.2σ
    - Same 1% background working point; reported as a 46% significance gain.



Thank you

AI For HEP

References

1. CMS Collaboration. “Transformer models for heavy flavor jet identification”. CMS-DP-2022-050
2. Huilin Qu, Congqiao Li, Sitian Qian. “Particle Transformer for Jet Tagging ”.
3. CMS Collaboration. “Adversarial training for b-tagging algorithms in CMS”. CMS-DP-2022-049
4. Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. “Explaining and harnessing adversarial examples”.
5. Erik Buhmann, Frank Gaede, Gregor Kasieczka, Anatolii Korol, William Korcari, Katja Krüger, and Peter McKeown. “CaloClouds II: ultra-fast geometry-independent highly-granular calorimeter simulation”
6. CMS Collaboration, "It's about time: a Point Cloud Generative Model for the CMS High Granularity Calorimeter".
7. DeepMind, “Perceiver IO: A general architecture for structured inputs & outputs.”
8. Lilian Weng. “What are diffusion models?”