Imperial College London

Deployment of ML in Changing Environments

CHEP 2023

M. Barbone, **C. Brown**, B. Radburn-Smith, A. Tapper
On Behalf of the CMS Collaboration

Outline

Changing Environments

- Typical timescales for CMS

Possible Solutions

- Retraining, Uncertainty Quantification, **Continual Learning**

Continual Learning in the Phase-2 CMS Trigger

- Example algorithm, trained on degraded CMS tracker

Summary

Phase-2 CMS Level-1 Trigger for High-Lumi LHC

- Hardware trigger
- 12.5 μs to make a decision
- ML used throughout
- All-FPGA architecture
- Unless triggered, data is lost
- Use fast track finding for vertex finding and pileup rejection

- We will use a Phase-2 CMS Level-1 Trigger [1] as an example throughout
- Identifying incorrectly reconstructed vertices in the L1 trigger
- Evaluate the impact of tracker degradation on this task

[1] CMS-TDR-021

Changing Environments

There are many situations in which a Machine Learning (ML) system is deployed in an evolving environment

- This can be with changing conditions which leads to a target dataset varying significantly from the training dataset

In particle physics experiments we tend to train ML systems on large Monte Carlo simulation datasets and then deploy the systems on real data which can have variable conditions

- The detector itself might not respond in the desired manner, alignment and calibrations may need to be performed on the data

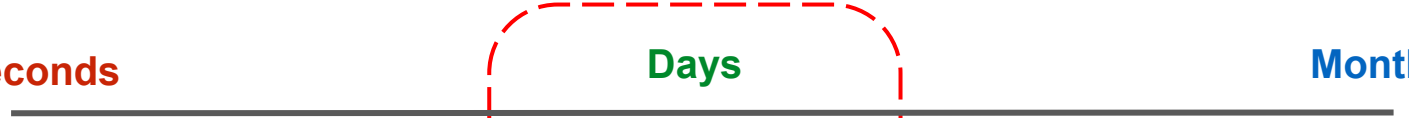
The Phase-2 Level 1 Trigger (L1T) uses ML to make decisions: small, low-latency models, might not be robust to the changing detector environment

Timescales for CMS

Seconds	Days	Months
Beam fluctuations	Beam conditions or small degradation	Significant detector changes
No time to create training data or retrain a model L1 can monitor ML robustness Online CL could be explored	Time to collect data directly from detector e.g. scouting or full reconstruction Between fill calibration [1][2]	Time to accurately emulate detector performance in large MC campaigns

[1] CMS-DP-2022-042 [2] CMS-DP-2022-068

Timescales for CMS



Beam fluctuations

Beam conditions or small degradation

Significant detector changes

No time to create training data or retrain a model
L1 can monitor ML robustness
Online CL could be explored

Time to collect data directly from detector e.g. scouting or full reconstruction
Between fill calibration [1][2]

Time to accurately emulate detector performance in large MC campaigns

Marco's CHEP talk

This talk

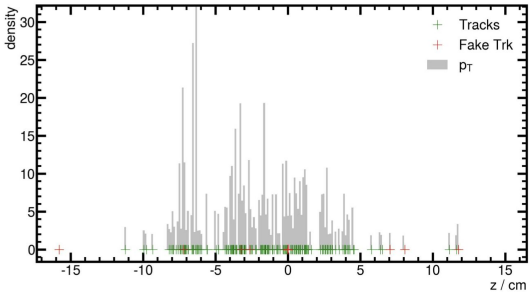
[1] CMS-DP-2022-042 [2] CMS-DP-2022-068

Possible Solutions

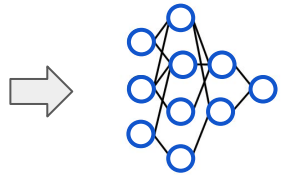
There are a various different options in order to cope with the changing environments, for example:

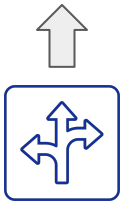
- Offline retraining and redeployment ➔ good for month timescales
- Uncertainty quantification ➔ good for seconds timescales
- Continual learning ➔ good for days timescales?

Each option has its own advantages and disadvantages

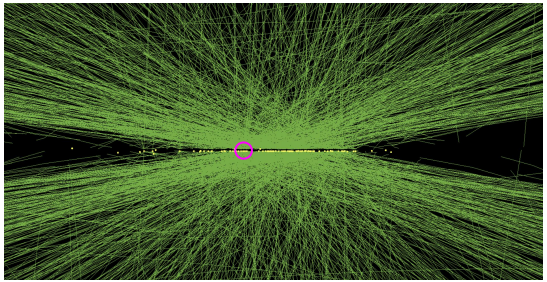


Sketch of Vertex Finding process, histogramming all tracks in z_0 weighted by p_T, a Phase-2 CMS L1 algorithm





New data



Finding a vertex in a high multiplicity HL-LHC event

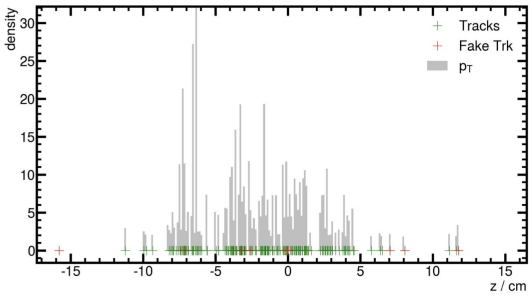
Possible Solutions

There are a various different options in order to cope with the changing environments, for example:

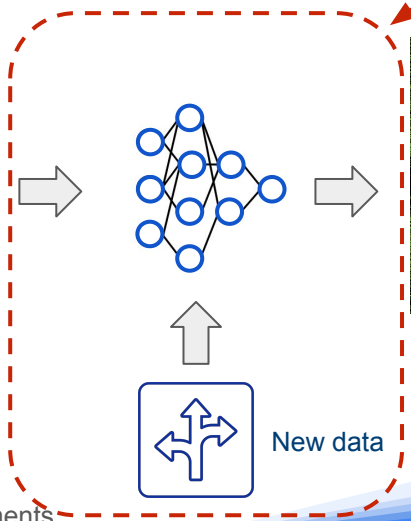
- Offline retraining and redeployment ➔ good for month time
- Uncertainty quantification ➔ good for seconds timescales
- Continual learning ➔ good for days timescales?

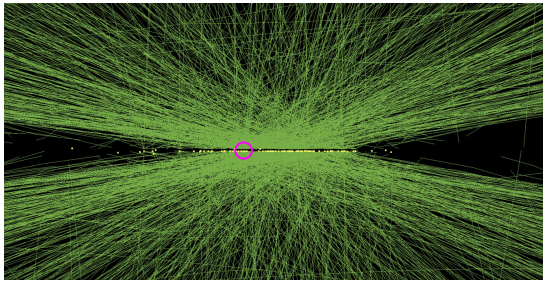
How do we update ML depending on the timescale?

Each option has its own advantages and disadvantages



Sketch of Vertex Finding process, histogramming all tracks in z_0 weighted by p_T, a Phase-2 CMS L1 algorithm





Finding a vertex in a high multiplicity HL-LHC event

Continual Learning (CL)

Concept: Train a model with a continuous stream of data

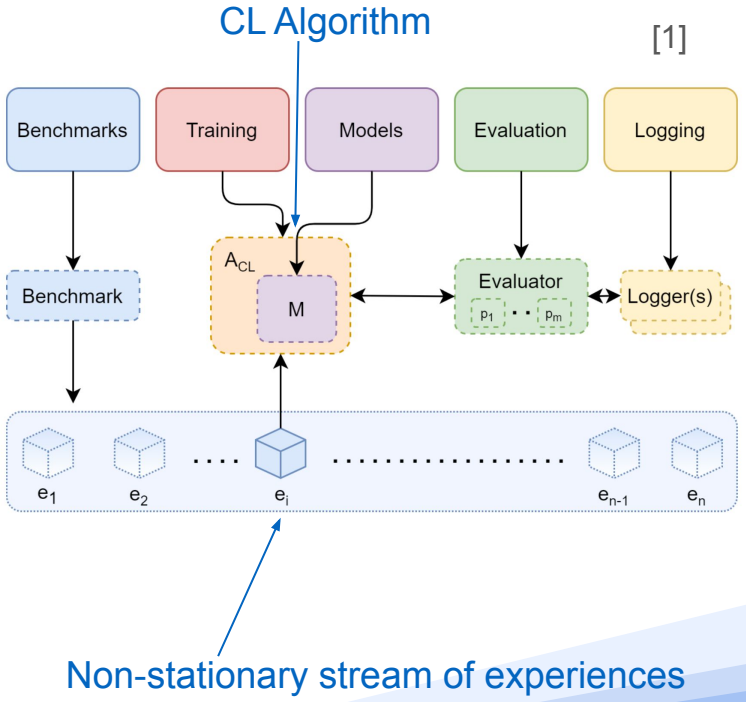
Learns from a sequence of partial experiences rather than all the data at once

Advantages:

- Avoids catastrophic forgetting → initial training is not disregarded
- Adapts to a changing data stream → don't need to quantify how the environment changes
- Don't need to store previous training data

Disadvantages:

- Some methods see the network change dynamically → in a fast ML setting this makes deployment on FPGA difficult
- For supervised learning need a ~continual stream of labelled data which might not be accessible



[1] avalanche

Continual Learning in the CMS Phase-2 Trigger

CMS Phase-2 trigger uses tracker primitives reconstructed using the outer tracker

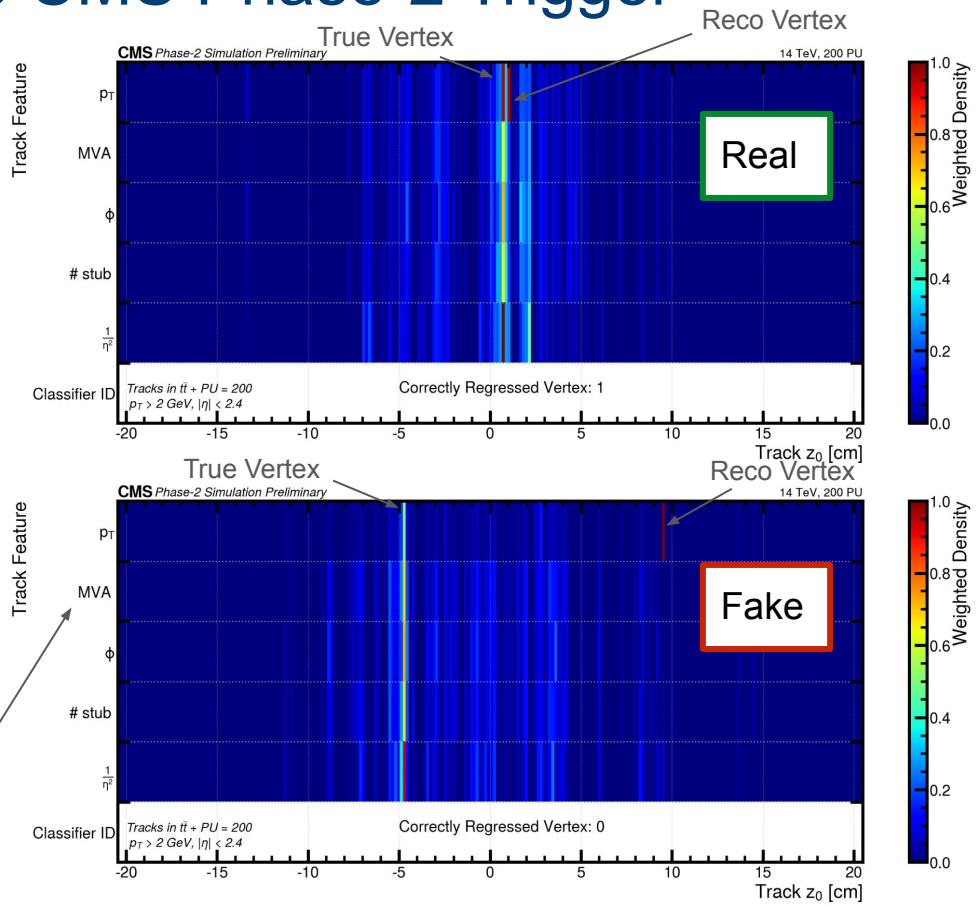
Vertex finding is performed by binning these tracks in the z-direction upon which a sliding window passes over to find the highest track p_T position

Use datasets with degraded outer tracker modules

Use a CNN model that classifies real and fake vertices

1. Does the ML model performance drop with new samples?
2. Does retraining the model with the new samples improve performance?
3. Does using CL to retrain the model improve performance further?

Use histogrammed track features (MVA is from a track quality BDT) [CMS-CR-2022-236]

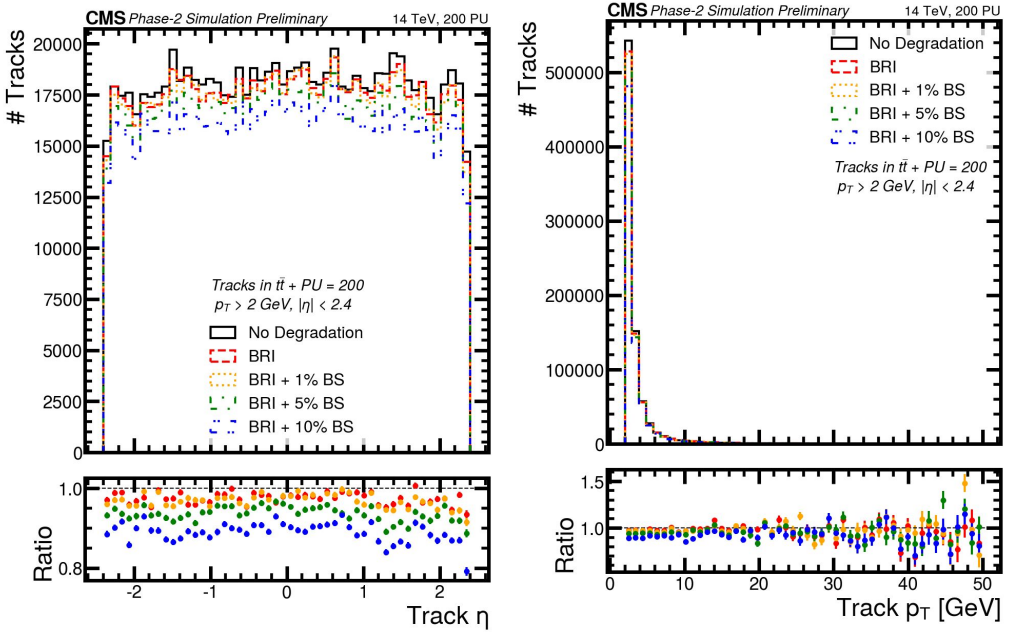


Characterising Detector Degradation

CMS Outer tracker has PS modules made up of PS-p (pixel) and PS-s (strip) submodules and 2S modules (strips only)

Have specific datasets that emulate the CMS outer tracker degrading [1] all generated from a top quark pair production sample with an additional 200 pileup:

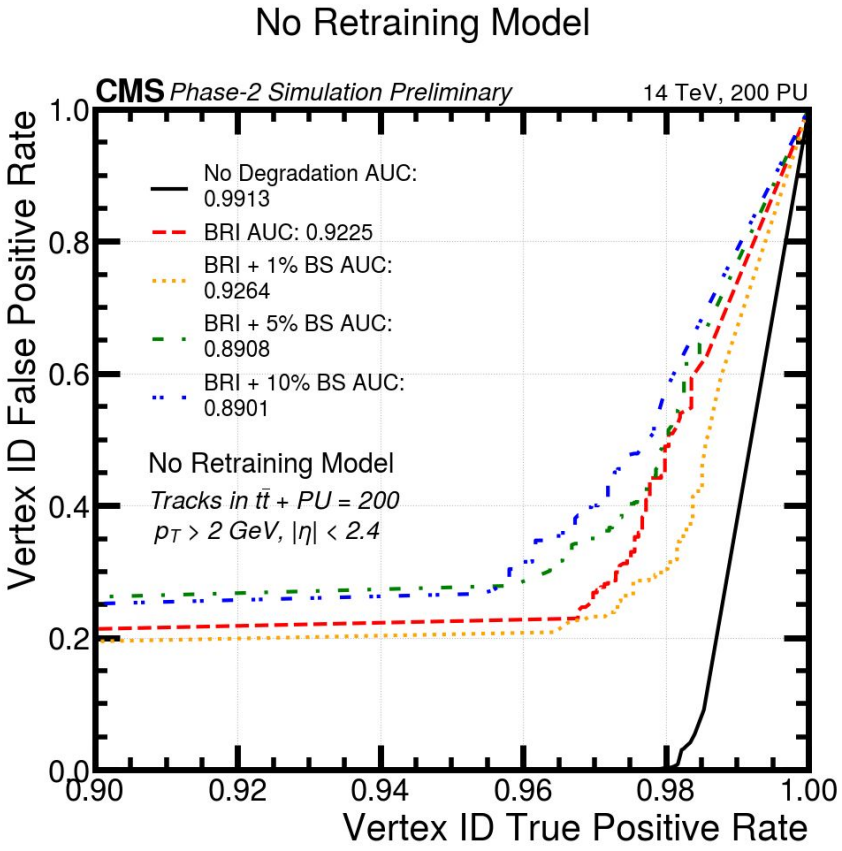
- No Degradation: no inefficiency (reference)
- BRI: PS-p bias rails inefficiency only
- BRI + 1% BS: PS-p bias rails inefficiency + 1% bad strips in PS-s and 2S sensors
- BRI + 5% BS: PS-p bias rails inefficiency + 5% bad strips in PS-s and 2S sensors
- BRI + 10% BS: PS-p bias rails inefficiency + 10% bad strips in PS-s and 2S sensors



Fewer tracks as the detector degrades especially at low p_T, quality of the tracks broadly unchanged

[1] CMS-TDR-014

Result - No Retraining Model



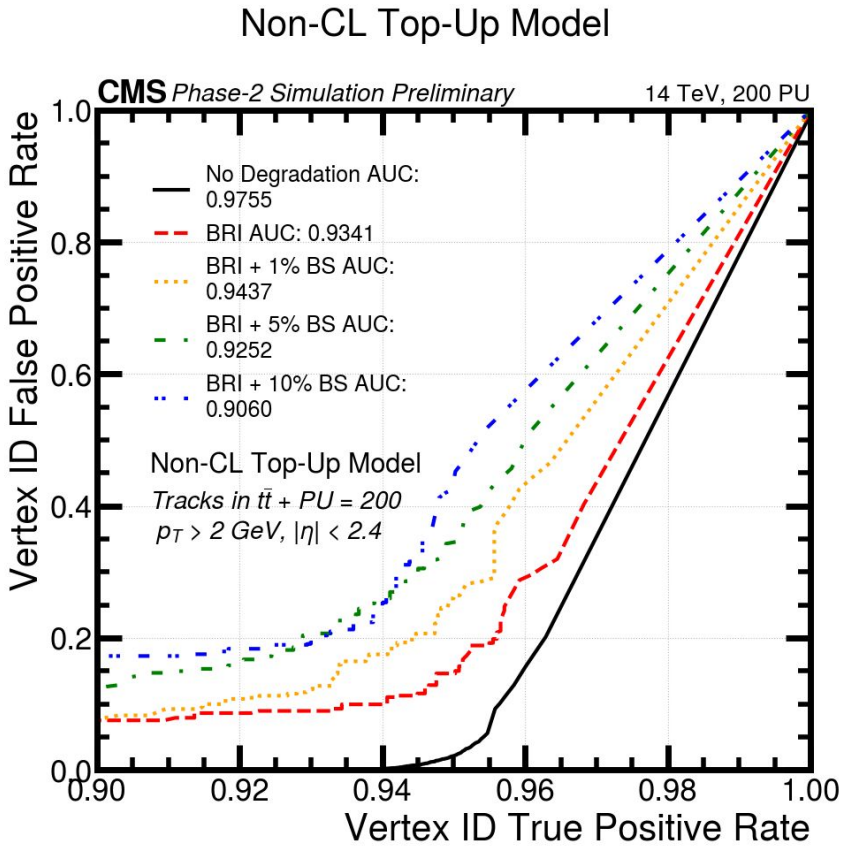
No Retraining model is as if we left our ML in the trigger without changing anything

Does the ML model performance drop with new samples?

Performs well on what it was trained on

Performance drops of as detector degrades, not robust to the changing environment

Result - Top-Up Train Model No CL



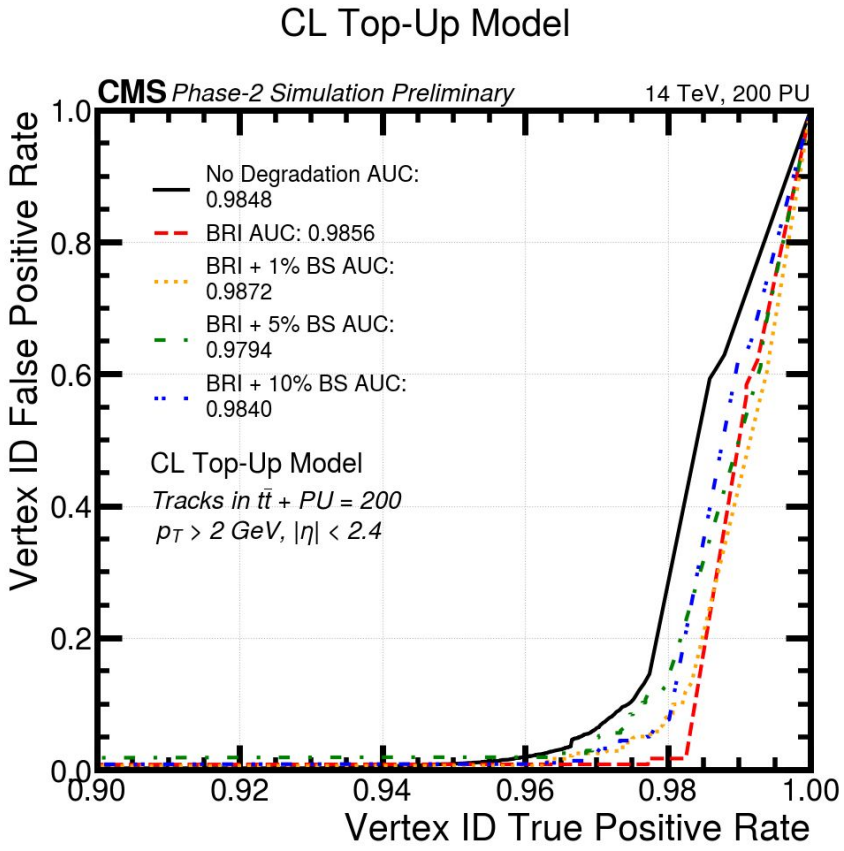
Start with no retraining model, top-up training with new datasets, 7000 training events, no fancy algorithm just same model new data, lower learning rate

Does top-up training the model with the new samples improve performance?

Performance across the degraded samples is better, generally more robust to the degraded detector

Performance in the no degradation dataset reduced as our retraining has disrupted the original model. Causing forgetting

Result - Top-Up Train Model Using CL Replay



Start with no retraining model, top-up train the model with new datasets in a CL manner, 7000 training events, replay algorithm, every batch has a subset of previous training datasets mixed in, always repeating old data as it trains on new

Does using CL to retrain the model improve performance further?

Far better performance across all datasets, robustness to the detector change

Less forgetting of original training

Summary

One of the key issues for real-time ML in **any area of science** is in the differences between the training data and the “real-world” data which can lead to performance degradation

We have described a few of the timescales at which this can occur and shown possible solutions for the problem

- In particular we focus on Continual Learning

- We face this issue in many areas of HEP including at the Phase-2 L1T
- The detector state can change over time which leads to different responses at the Phase-2 L1T
- We have demonstrated how ML in the Phase-2 L1T can be affected by a degraded detector and while top-up training does improve performance, the use of CL has a larger impact on the robustness of the ML model

Backup

Retraining Old Models

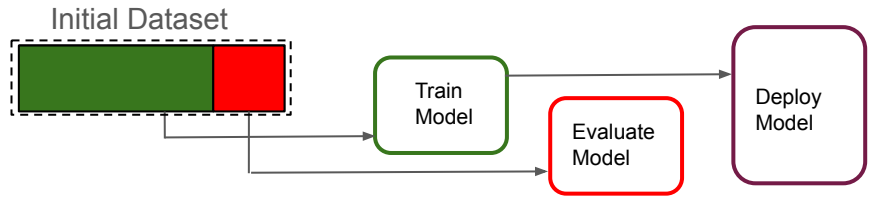
Regularly retrain and redeploy models on large datasets

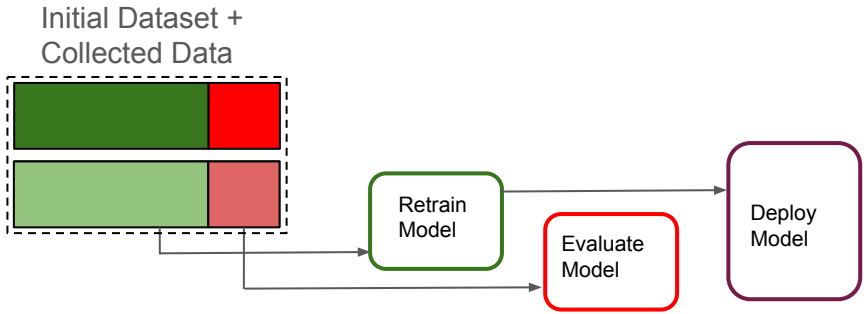
Advantages:

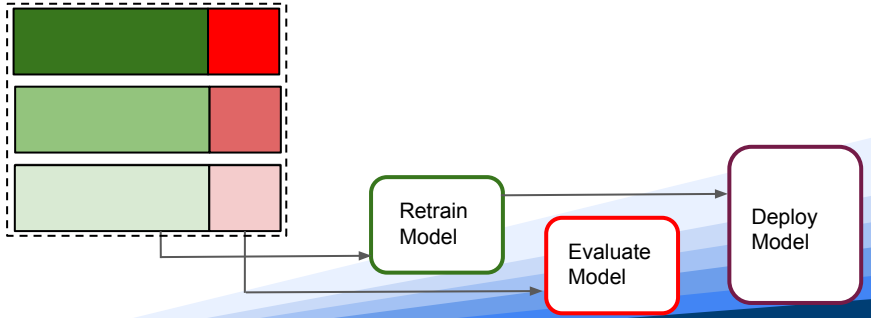
- Can selectively create datasets based on understanding the changing environment
- Can store previously trained models

Disadvantages:

- Retrained model might have different footprint to original model (pruning or quantisation might change)
- Catastrophic forgetting can occur where model performance on previously learned task reduces when new data is given







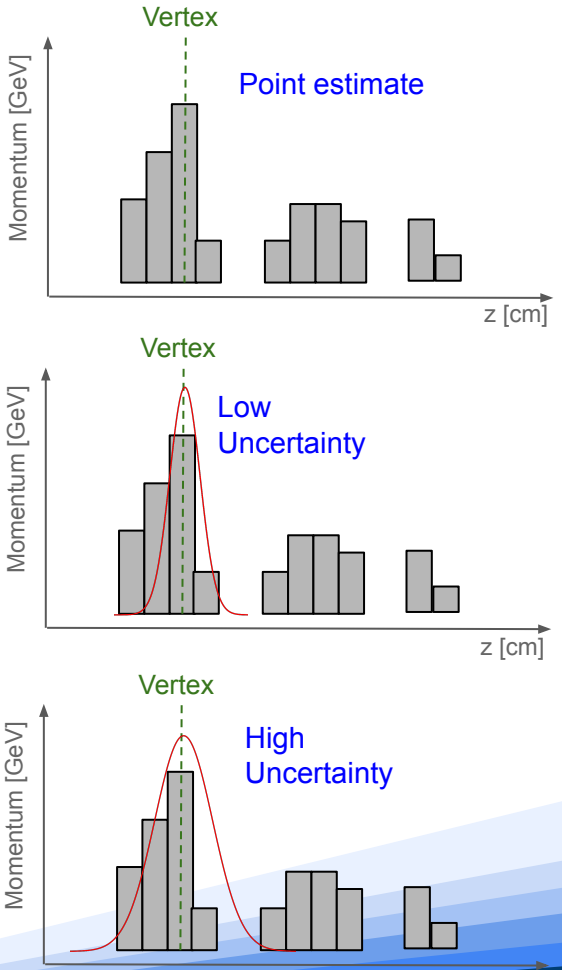
Uncertainty Quantification

Learn (or quantify) the uncertainty of a model and use this as a handle for our confidence in the model

This uncertainty can be either used by downstream users or as a flag to retrain the model

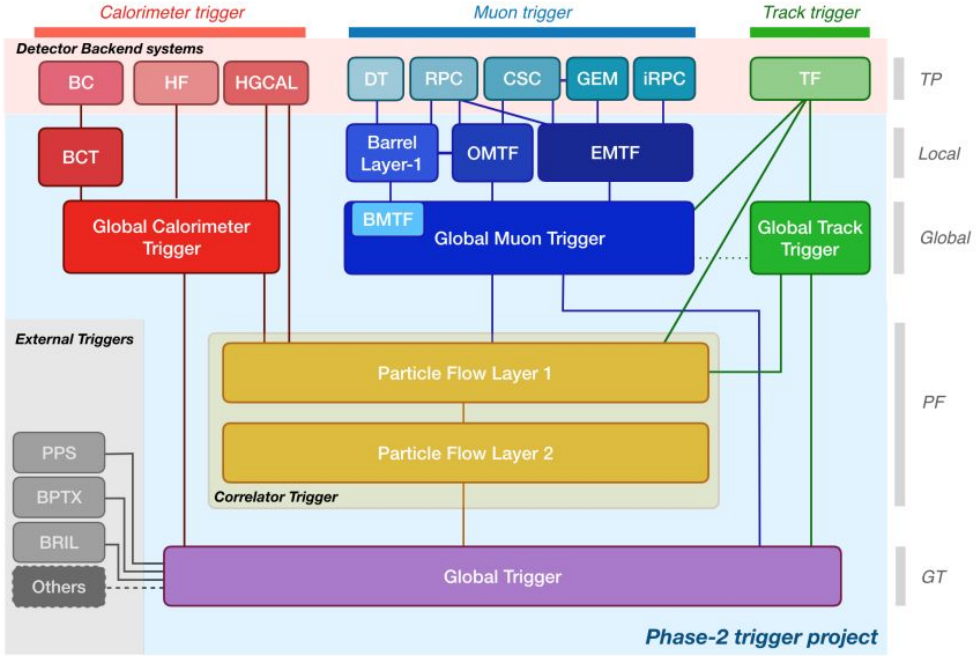
Can only inform us on how confident we should be in our model, doesn't give insight into why

Same issues as retraining models, core model still needs to be retrained but have an understanding of when



CMS Phase-2 Level-1 Trigger

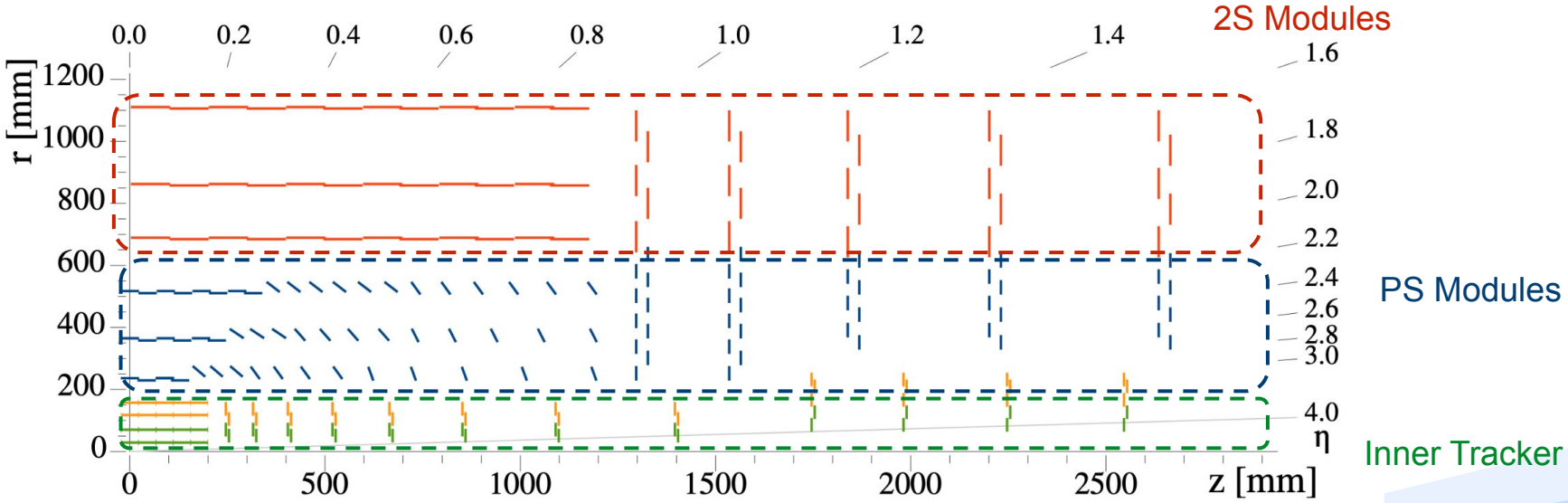
- All new trigger, with increased latency (12.5 μs) and 750 kHz read-out rate [1]
- All-FPGA based with custom designed ATCA boards
- Tracks from outer tracker, a first for CMS L1 Trigger
- Opportunities for complex algorithms, Vertex Finding, PUPPI, all to maintain physics performance in HL-LHC
- ML used in multiple places, e.g, for vertex finding [2] or VBF triggers in the Global Trigger



[1] CMS-TDR-021 [2] JoP

CMS Phase-2 Tracker

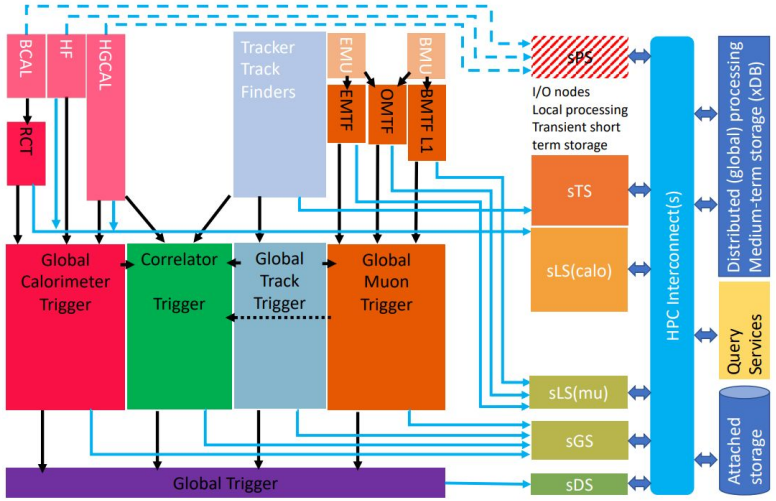
- CMS Phase-2 L1 Trigger will have track finder primitives [1]
- Reconstructed tracks p_T > 2 GeV, |η| < 2.4, only using CMS outer tracker, 4 μs latency
- Outer tracker is made up of 2S Modules, strips only and PS modules which have two sub modules, the PS-p (pixels) and PS-s (strips) [2]



[1] CMS-TDR-021 [2] CMS-TDR-014

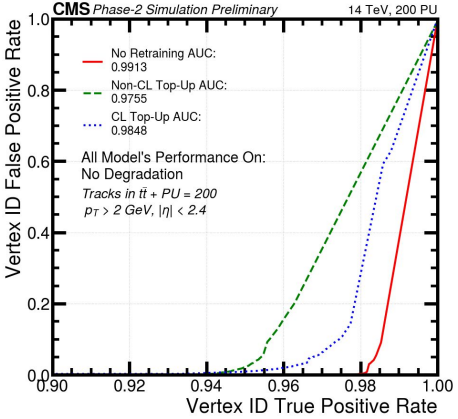
L1 Scouting

- The Phase-2 L1 Trigger will have 40 MHz scouting, input from L1 trigger multiple sources
- Envisaged to be used for monitoring, diagnostics, luminosity measurements
- Can be used as an unbiased continual stream of data for L1 ML algorithms after offline reconstruction truth labelling
- Collects data independently to L1 accept

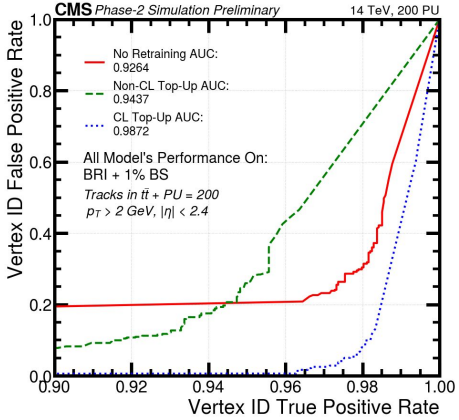


Architecture of Phase-2 L1 Trigger with scouting datapath Phase-2 L1 Trigger TDR

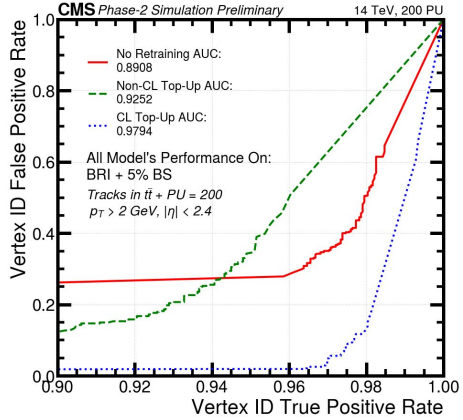
All Model's Performance On: No Degradation



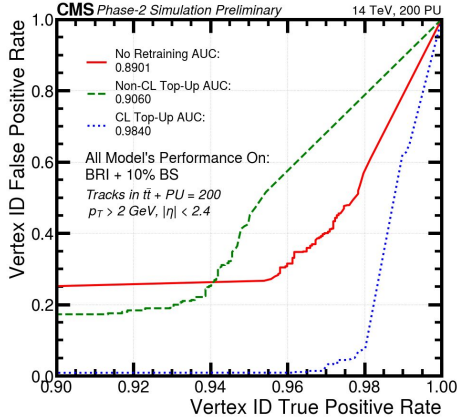
All Model's Performance On: BRI + 1% BS



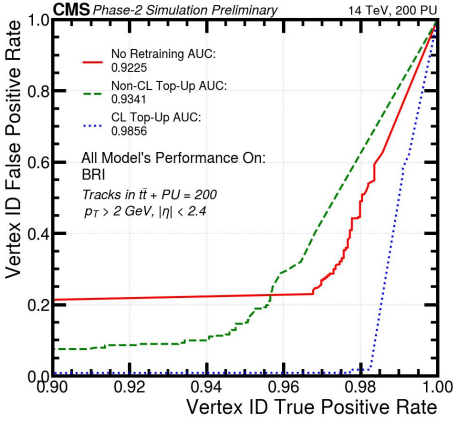
All Model's Performance On: BRI + 5% BS



All Model's Performance On: BRI + 10% BS



All Model's Performance On: BRI



Receiver Operating Characteristic curves for each of the different failure scenarios. In red the No Retraining model performs poorly as it has never seen the degraded samples. In green the Non-CL model performs better with a lower false positive rate. In blue the CL model outperforms the Non-CL and has stable performance across all failure scenarios as the detector degrades demonstrating the effectiveness of a CL learning strategy in these top-up trainings.