Usage of Machine Learning in CMS Level-1 Endcap Muon Trigger

Efe Yiğitbaşı[2], Darin Acosta[2], Osvaldo Miguel Colin[2], Aleksei Greshilov[2], Sergo Jindariani[1], Patrick Kelling[2], Jacobo Konigsberg[3], Jia Fu Low, Alexander Madorsky[3], Gauri Pradhan[1], Ryan Rivera[1], Suzanne Rosenzweig[3], John Rotter[2]

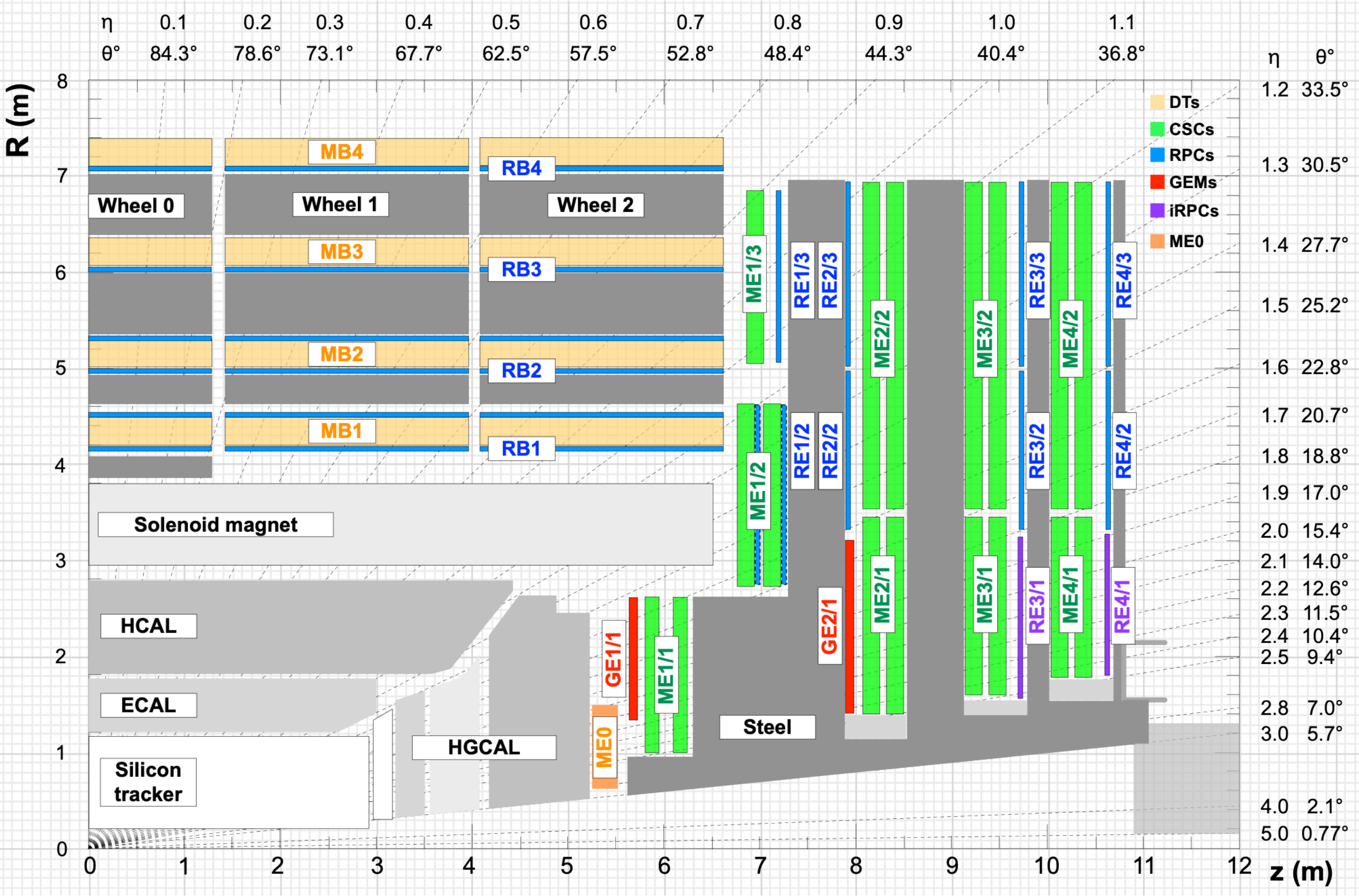
30th November 2022
CPAD2022

[1] Fermilab

[2] RICE

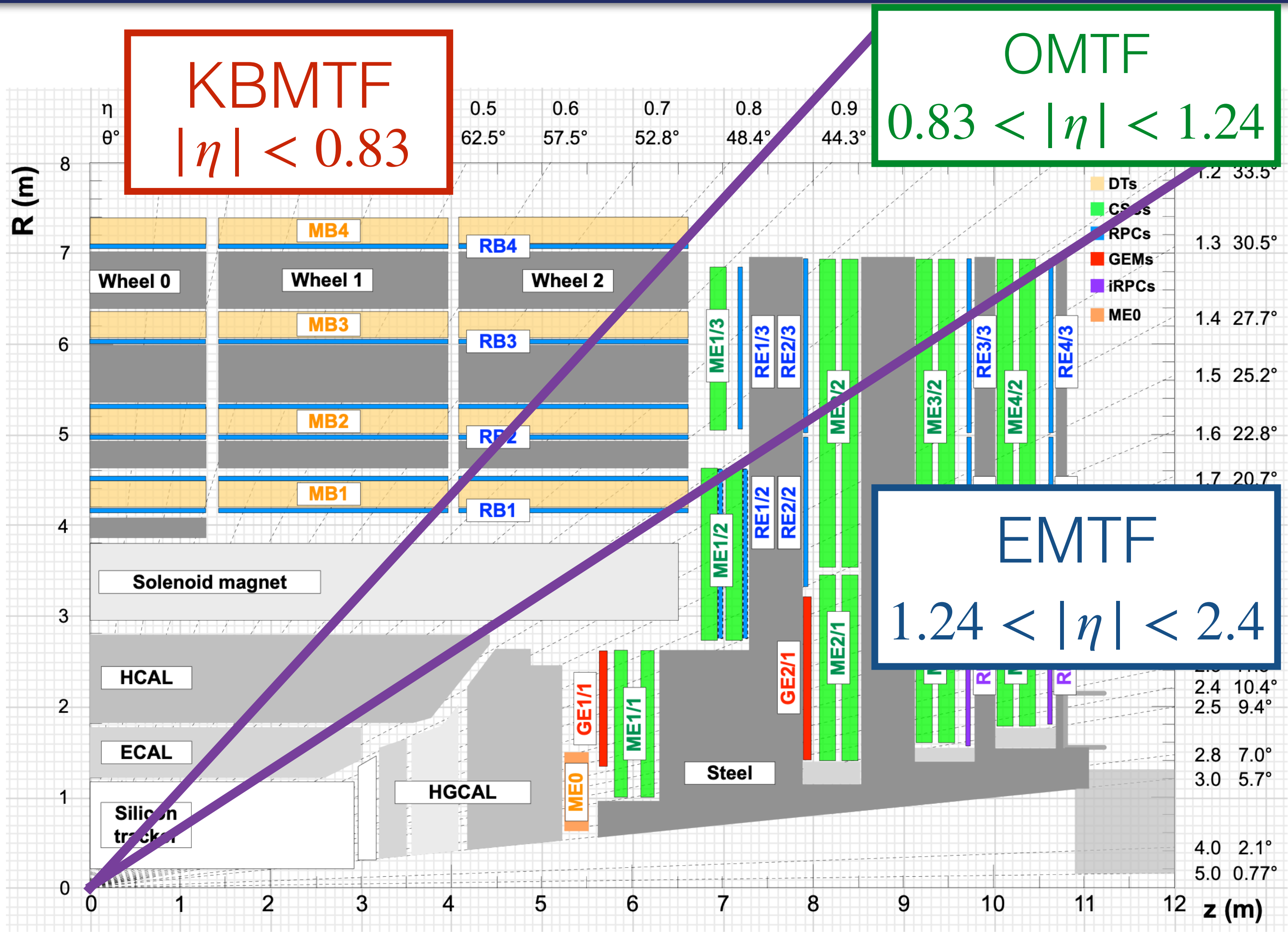
[3] UF UNIVERSITY of FLORIDA

CMS Level-1 Muon Trigger

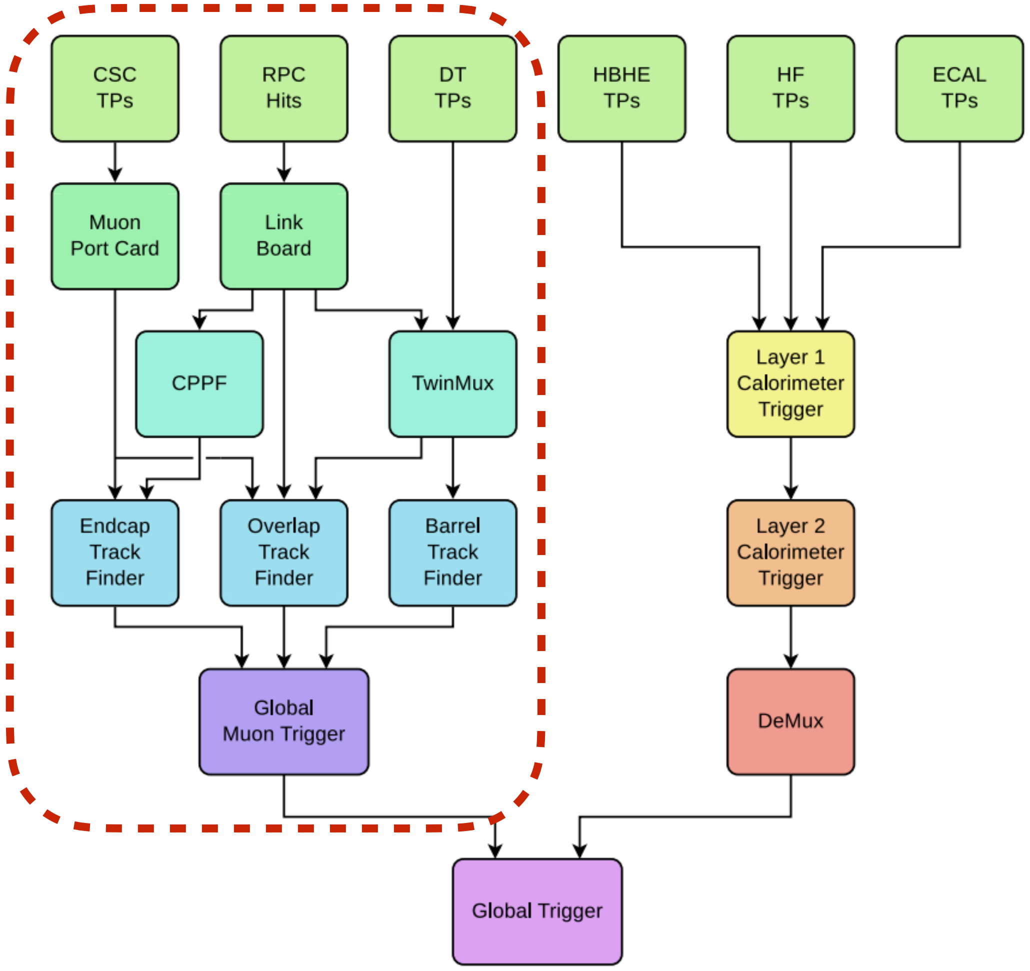


- CMS uses a two-level trigger system:
  - Level-1 (L1): Uses custom electronics to reduce event rate from 40 MHz to 100 kHz within 4 µs latency
  - High Level Trigger (HLT): Uses large CPU/GPU farms to further reduce the event rate to 1 kHz
- L1 muon trigger system uses “trigger primitives” (TPs) or “stubs” created by CMS muon detectors:
  - CMS muon system currently consists of Drift Tubes (DT) and Resistive Plate Chambers (RPC) in the barrel, and Cathode Strip Chambers (CSC), RPCs, and Gas Electron Multipliers (GEM) in the endcaps.
  - For the HL-LHC upgrade, the forward muon system will be enhanced with new GEM type detectors (ME0 and GE2/1) as well as improved RPCs (iRPC).

CMS Level-1 Muon Trigger



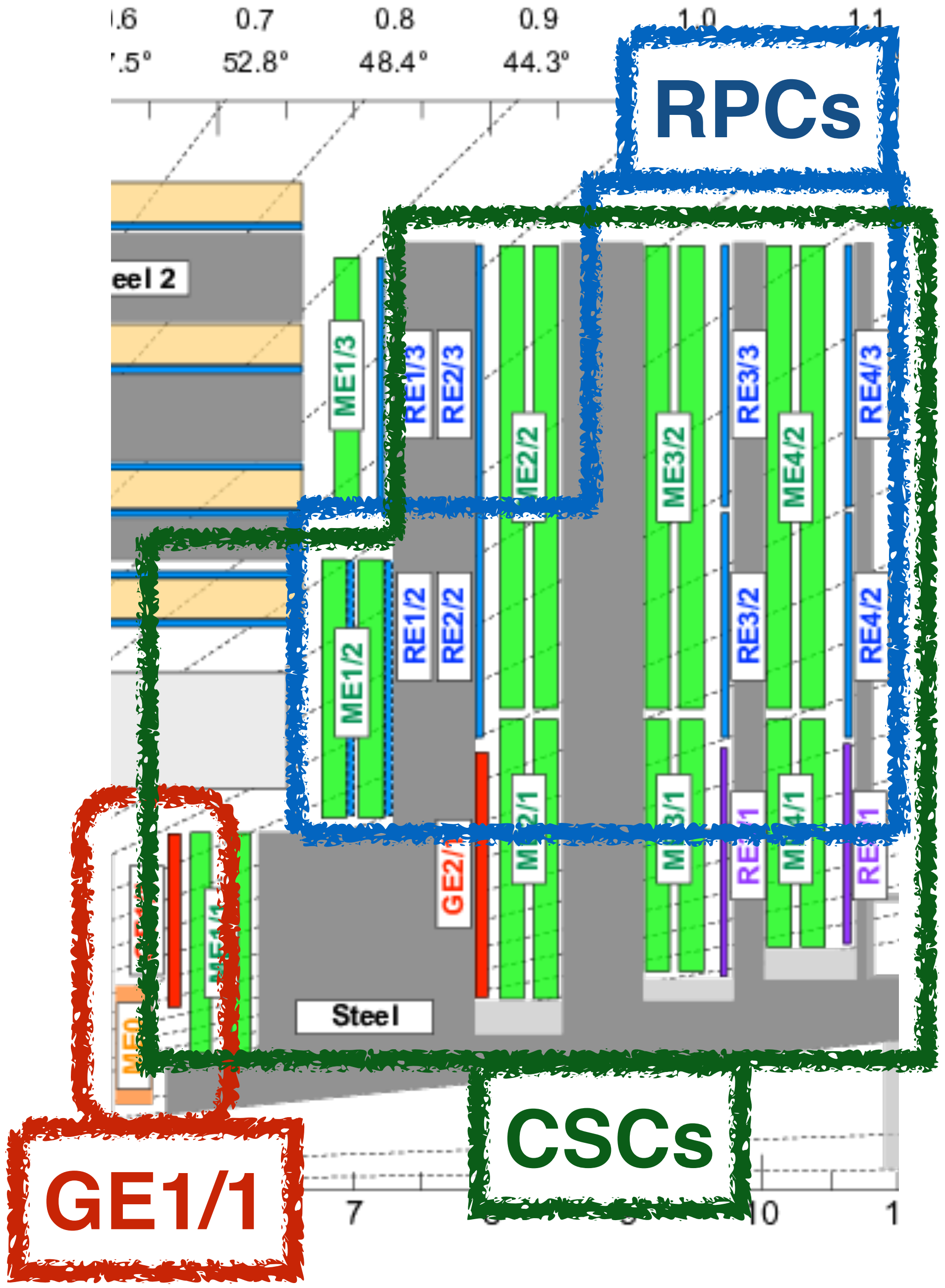
The Level-1 Trigger



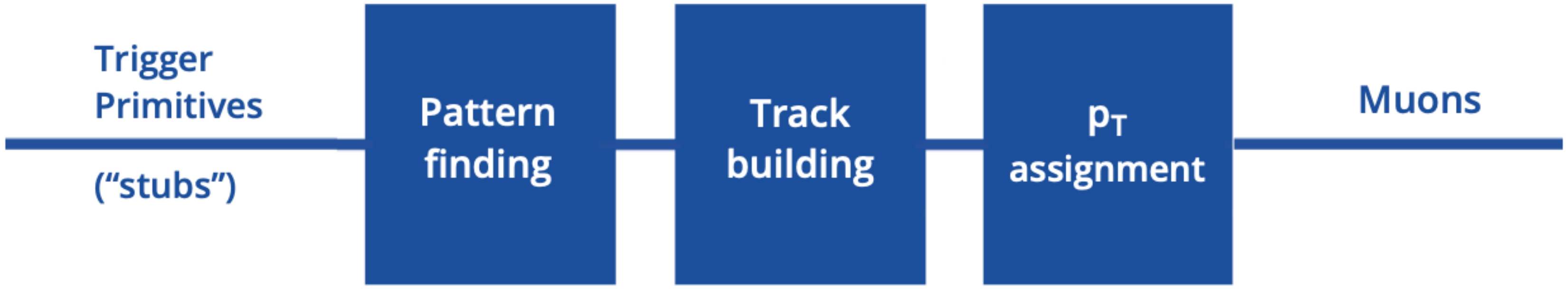
- Level-1 muon trigger receives inputs from different muon detectors.
- 3 track finders (TF) build muon tracks from muon detector stubs and measure track parameters such as η, ϕ, p_T, q.
  - Best tracks are sent to Global Muon Trigger (μGMT) which are then merged, cross cleaned, sorted, and sent to Global Trigger (μGT) where the L1 trigger decision happens.

The EMTF Algorithm

- Goal:
  - Build tracks by associating stubs from different stations and measure track p_T, η, ϕ etc. in a very short timescale
- Challenges:
  - Non-uniform magnetic field in the endcaps whose effect gets weaker in the forward direction
  - Different detector technologies with different spatial and timing resolutions
  - Large collision backgrounds which increase with increasing η which can lead to non-linear pileup dependence



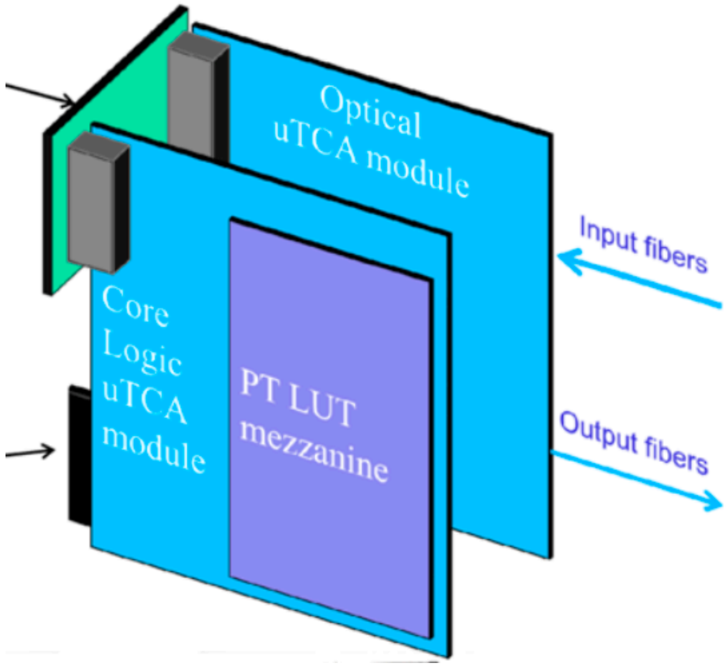
The EMTF Algorithm



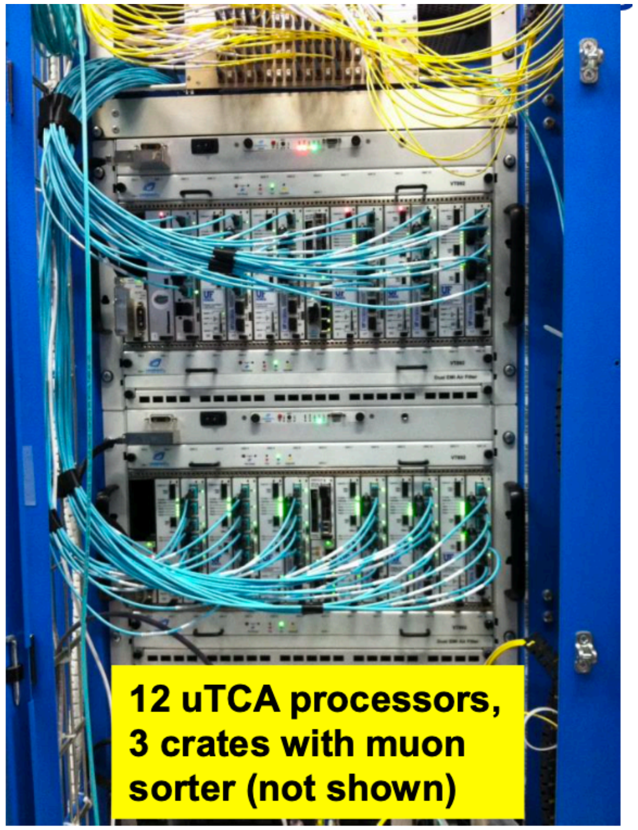
- The EMTF algorithm mainly consists of 3 blocks:
  - Pattern finding:
    - Use predetermined patterns to find stubs in different stations that are consistent with a muon pattern
  - Track building:
    - Find best matching stub to the pattern in each station. Calculate station to station track parameters ($\Delta\phi$, $\Delta\theta$ etc)
  - p_T assignment:
    - Determine p_T of the track based on the track curvature using the best possible method.
    - Challenges mentioned before become a real problem at this step. Not trivial to do this analytically. Ideal problem for ML based solutions.
    - How do we use the available EMTF resources for ML algorithms?

EMTF Hardware for Run 3

- EMTF algorithm is currently running at CMS in MTF-7 cards using Virtex-7 690T-2 FPGAs
  - Optimized for maximum input from muon detectors (84 input links, 28 output links)
  - Dual card with large capacity for RAM (~1 GB) to be used for p_T assignment via LUTs
- Total of 12 sector processors (6 per endcap) each working with inputs from a 60° endcap sector (+10° or 20° from neighbor sector).



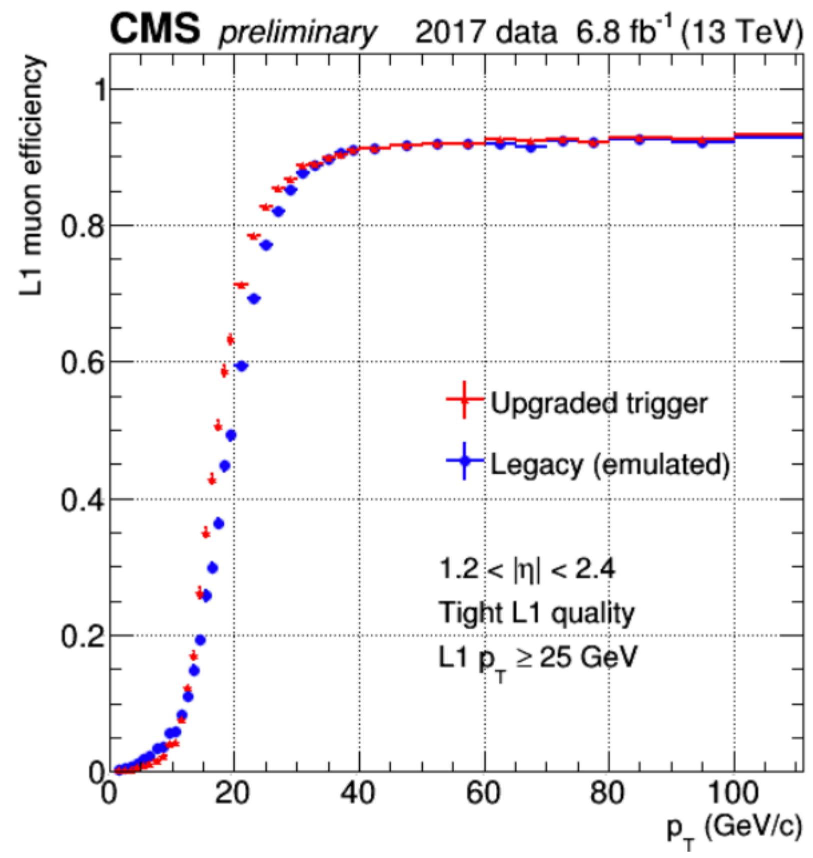
In USC

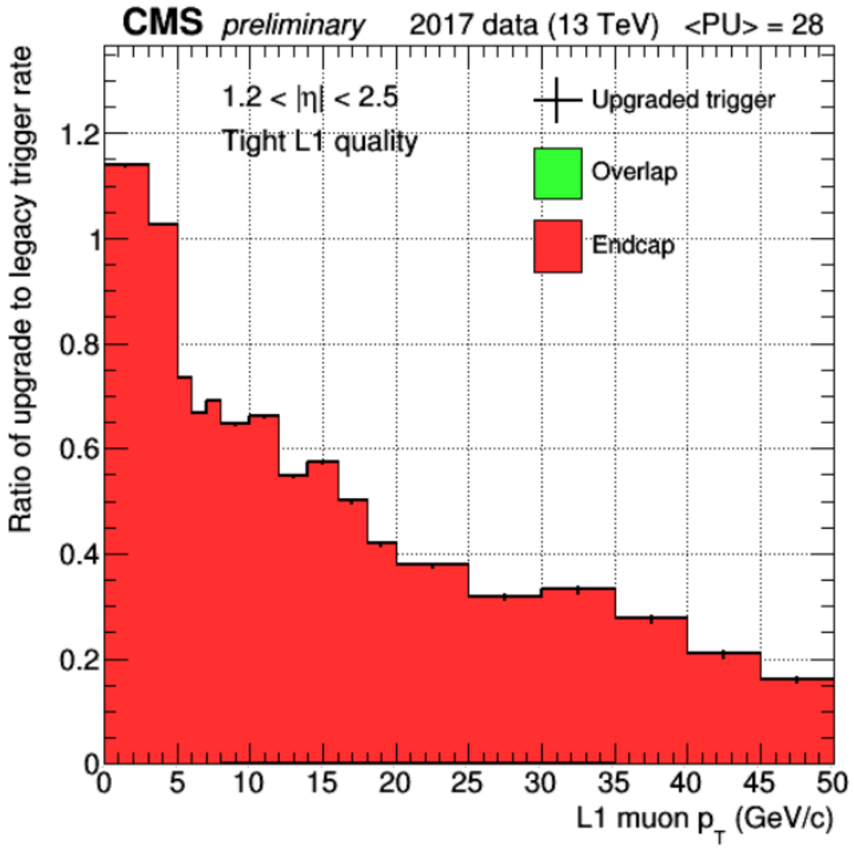


12 uTCA processors, 3 crates with muon sorter (not shown)

EMTF Prompt BDT

- Plan for p_T assignment is to use the large (~1 GB) fast LUT.
  - Each 30 bit address is used for a p_T value
  - Address is calculated from a combination of track parameters (Δϕ, Δθ etc) depending on which station hits are used in building the track.
- For Run 2, EMTF team decided to use a Boosted Decision Tree (BDT) regression to create the p_T LUTs.
  - BDT is trained offline using TMVA with MC simulation samples.
  - Output of the BDT is converted to a LUT where a 9-bit p_T value is assigned for each of the 2^{30} addresses.
  - In 2016, the BDT method achieved a factor of 3 rate reduction compared to the legacy trigger while maintaining the same plateau efficiency.
- For Run 3 EMTF BDT was retrained mostly due to changes in CSC stubs and to anticipate the inclusion of GE1/1 in L1T:
  - Current training was done using Run 2 CSC+RPC stubs. GE1/1 will be added later.
  - We adjusted BDT training parameters compared to Run 2 to improve the high p_T performance which showed inefficiencies due to new CSC stubs.
- Run 3 BDT achieves slightly better turn-on behaviour, improved efficiencies for very high p_T muons (> 100 GeV) while keeping the single muon trigger rate similar to 2018 values (within 10%)



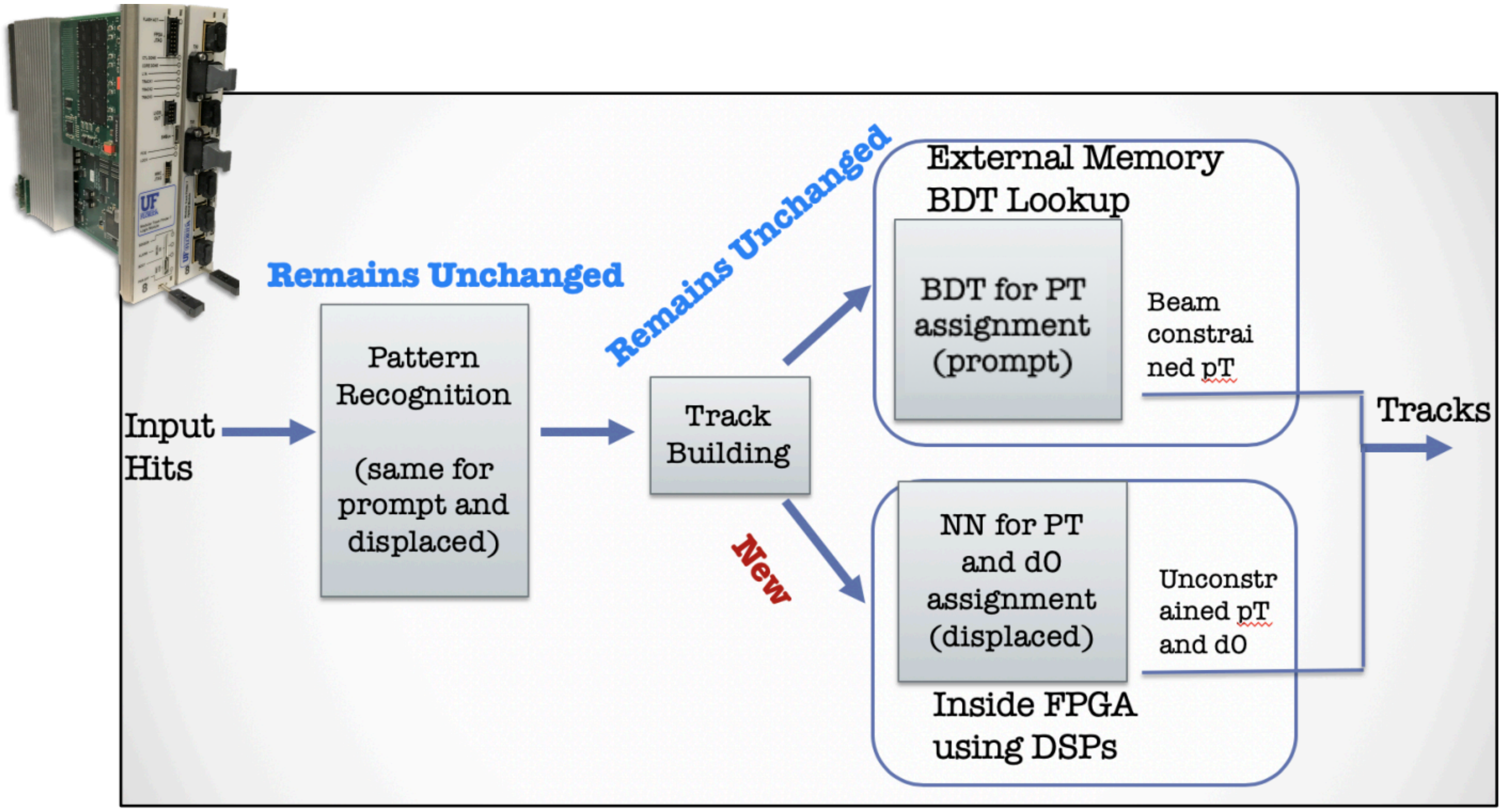


https://cds.cern.ch/record/2289251

EMTF Displaced NN

- EMTF includes a separate path for p_T assignment to displaced muons in Run 3 using an Neural Network (NN):
  - Displaced muons can appear in decays of long-lived particles (LLPs). LLP searches are one of the focuses of CMS in Run 3.
  - The displaced neural network, which will handle vertex unconstrained p_T and d_0 assignment, was developed for HL-LHC upgrade of EMTF (EMTF++) and is described in CMS L1T Phase 2 TDR.
  - The idea is to run the NN directly on FPGA, which removes the bottlenecks of using LUTs by instead using logic and DSP resources of FPGA.
  - Without the NN, current largest EMTF resource usage is LUT (54%) and BRAM (60%). Only ~1% of DSPs are used.
- For Run 3, a smaller version of the Phase 2 NN was implemented to EMTF during 2020-2022 to drastically improve our capabilities of triggering on displaced muons.
  - The Run 3 NN uses only Run 3 stubs and is slightly smaller due to FPGA constraints.
  - Run 3 EMTF will use prompt patterns for both prompt and displaced muons while EMTF++ will have dedicated displaced patterns.
- These changes limits the performance of the NN for Run 3. However, the improvements for displaced triggering are still substantial.

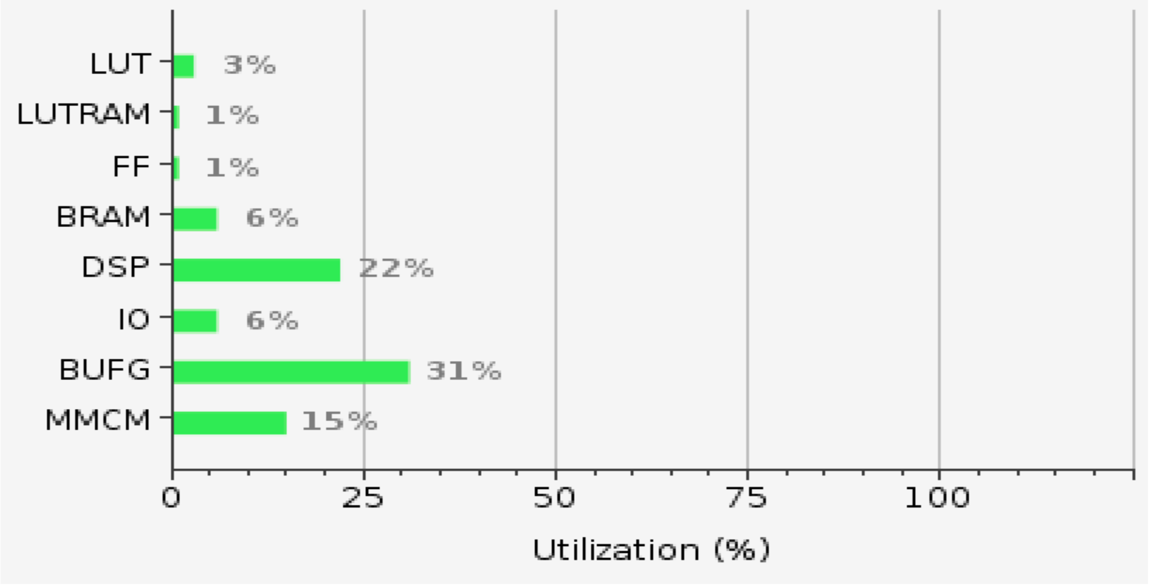
Run 3 EMTF:


Input Hits
Remains Unchanged
Pattern Recognition (same for prompt and displaced)
Remains Unchanged
Track Building
New
External Memory BDT Lookup
BDT for PT assignment (prompt)
Beam constrai ned pT
NN for PT and d0 assignment (displaced)
Unconstr ained pT and d0
Inside FPGA using DSPs
Tracks

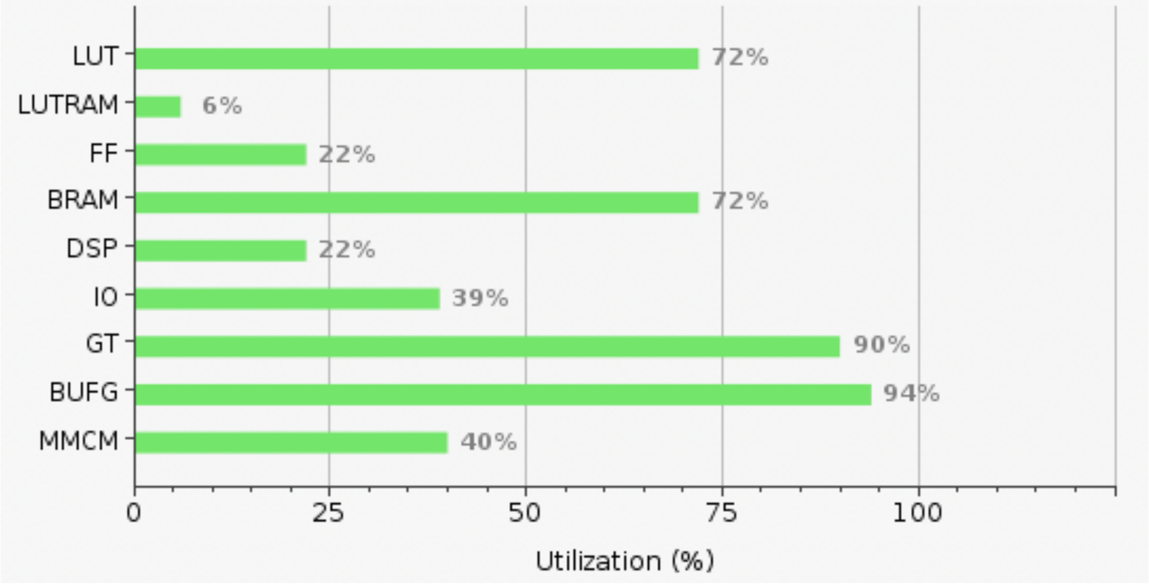
EMTF Displaced NN Implementation

- The NN model is trained in software and implemented in firmware using hls4ml workflow [1].
  - hls4ml is a toolkit to implement fast neural network inferences in FPGAs using Vivado High-Level Synthesis (HLS).
  - NN inputs are current EMTF p_T assignment inputs that are also used for creating the LUT address.
  - NN outputs are converted to 8b p_T and 3b (1 sign + 2 value) d_0 via 2 LUTs to be sent to μGMT.
- Software implementation uses tensorflow and floating point precision.
- The current version (NNv10) is slightly smaller than initially foreseen back in 2021 due to resource usage in FPGA.
  - NNv10 was designed to use 16 clocks @ 120 MHz, 1 clock is saved for NN output conversion LUTs. In order to achieve this the NN was reduced to 2 dense layers with 20/15 nodes from 3 dense layers of 30/25/20.
  - In Summer 2022, we realized that total NNv10 latency in firmware is 3 BX too large. Currently a new training is underway to further reduce the latency to fit into our latency budget. New NN version will use 13 clocks @ 120 MHz and it will not have input/output conversion LUTs.

NNv10
resource utilization



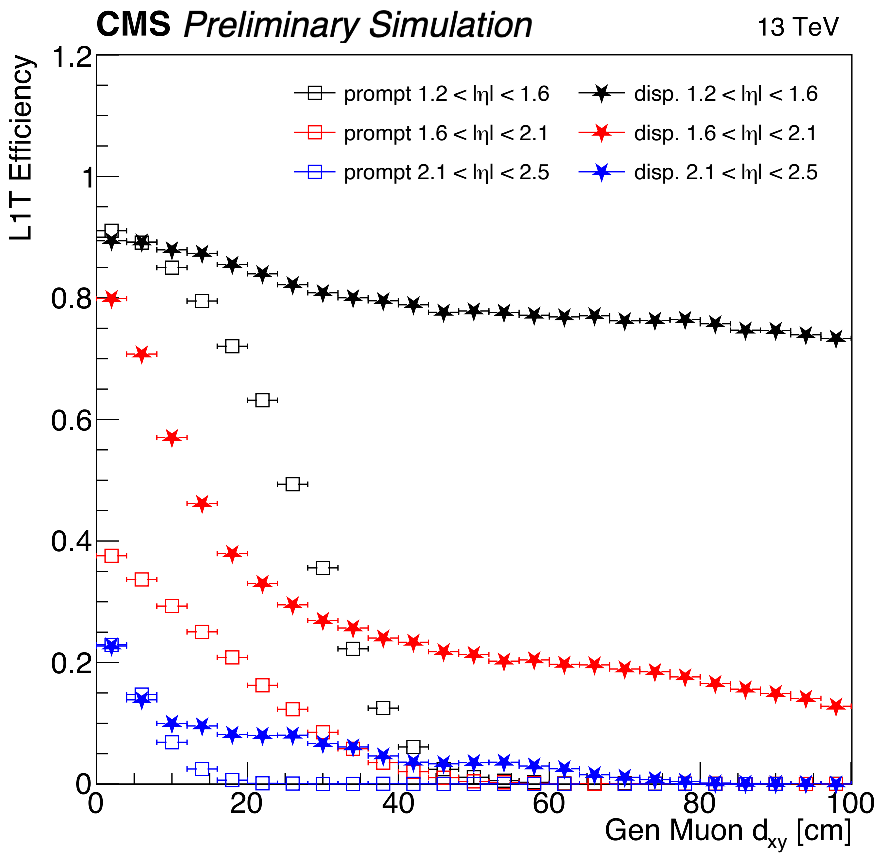
EMTF resource utilization
with NN



[1] https://arxiv.org/abs/1804.06913

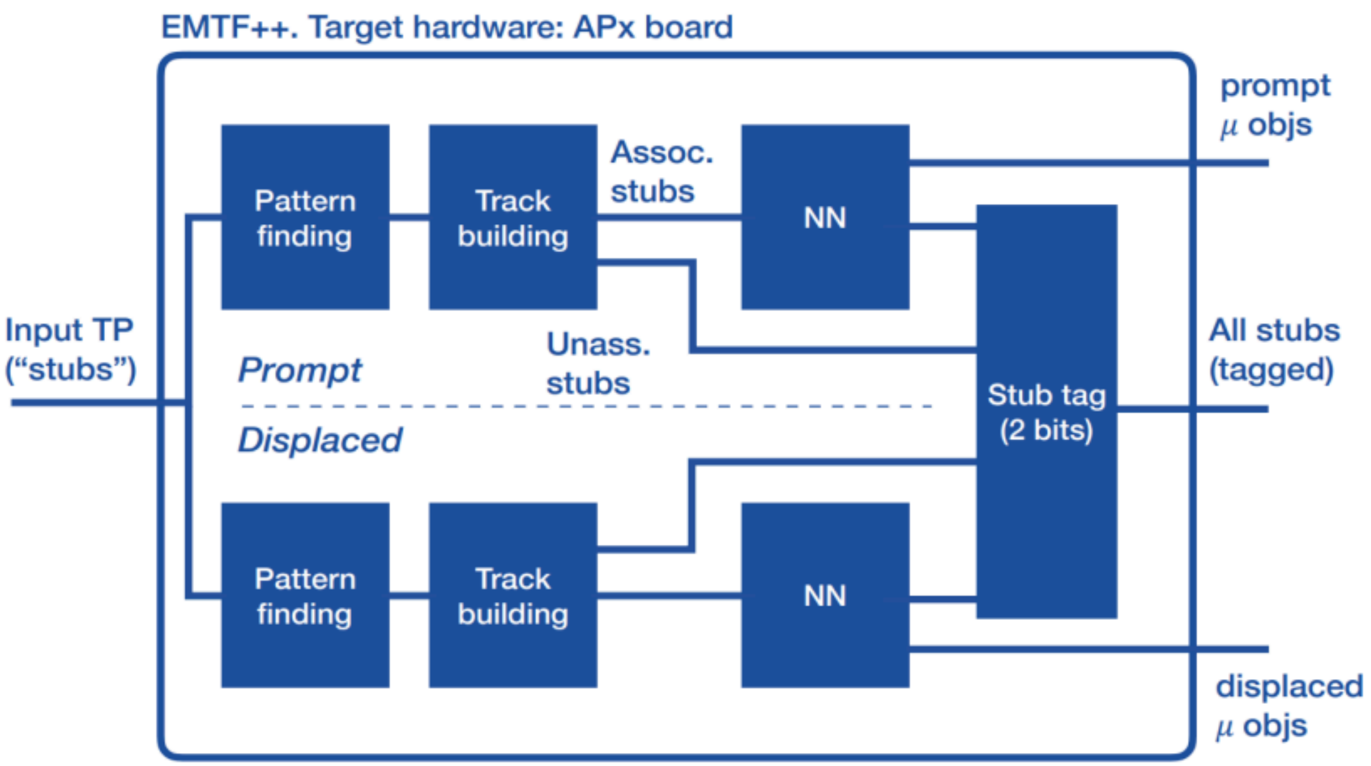
EMTF Displaced NN Performance

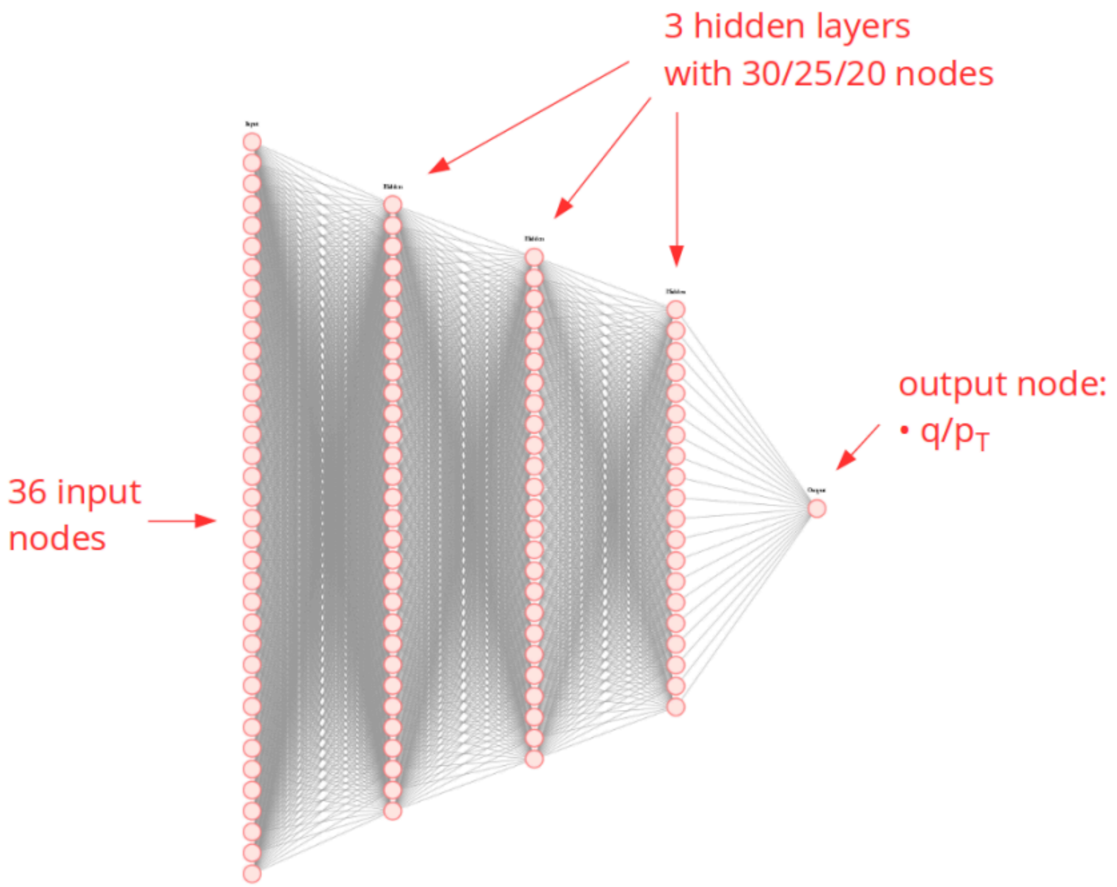
- NNv10 was implemented in software and firmware in 2022.
  - Substantial efficiency gain for displaced muons, mostly coming from the outer ring region while keeping impact on rate under control (< 1 kHz).
    - Displaced triggering efficiency is ~80% and stable with increasing d_{xy} for $1.2 < |\eta| < 1.6$, compared to prompt efficiency which falls rapidly after 10 cm.
    - Reduced size of NNv10 is one of the main reasons for widely varying performance between η regions.
- NN performance in Run 3 could be improved by including Run 3 CSC and GE1/1 stubs, and by further optimizations.



EMTF NN For Phase 2

- For Phase 2 upgrade of EMTF algorithm (EMTF++), we decided to replace the prompt BDT with two NNs: one for prompt muons and one for displaced muons.
  - The idea is to run NNs directly on modern FPGAs implemented via hls4ml workflow.
  - EMTF++ will have separate prompt/displaced patterns and prompt/displaced NNs
  - More muon detector inputs (ME0, GE2/1, iRPC) and more FPGA resources will lead to better performing NNs
- Switching purely to NNs in FPGA rather than LUTs allows us to use all the low level information available to us directly as inputs with more precision, without pre-processing them for the LUTs.



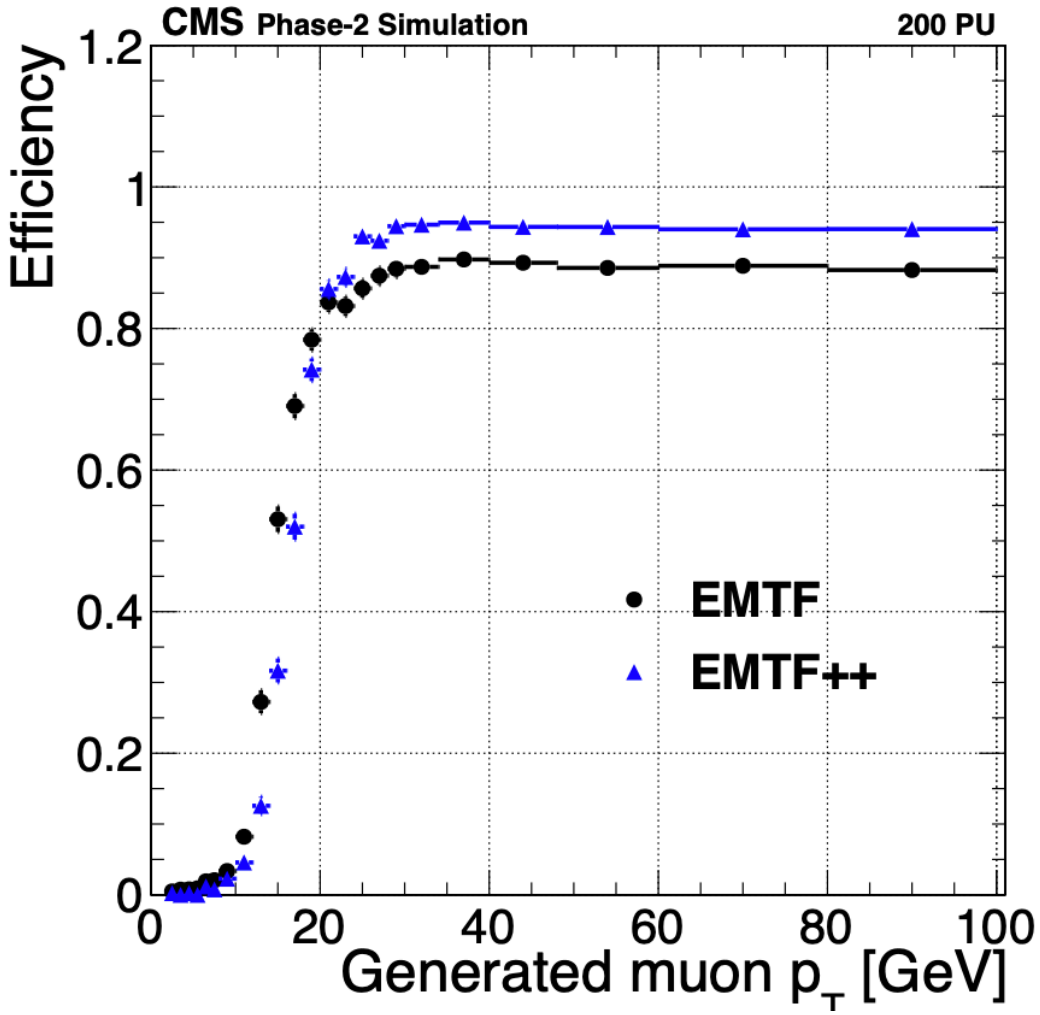

3 hidden layers with 30/25/20 nodes

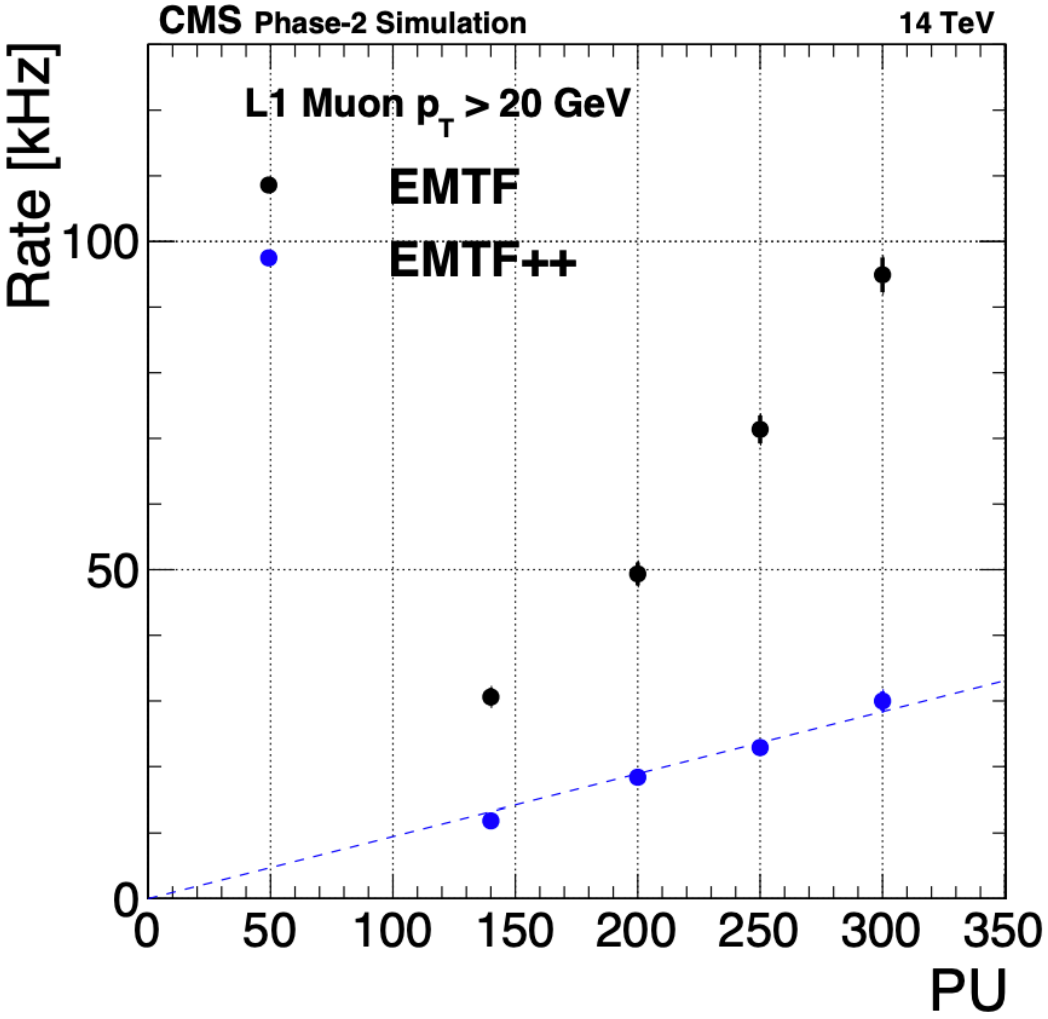
36 input nodes

output node:
- q/p_T

Phase 2 Prompt NN Performance

- The goal for prompt EMTF++ algorithm is to retain the same level of efficiency as Run 2/3 also in Phase 2 while improving the rate vs PU behaviour to a sustainable level.
  - Run 3 EMTF algorithm has a non-linear PU dependence and results in very large rates at 200 PU.
- EMTF++ NN design uses 36 inputs and has 3 dense layers of 30/25/20 nodes. The output is q/p_T
- Performance studies for Phase 2 TDR shows EMTF++ has improved turn-on and higher plateau efficiency while reducing rate by a factor of 2.5 and keeping rate vs PU relationship quite linear.

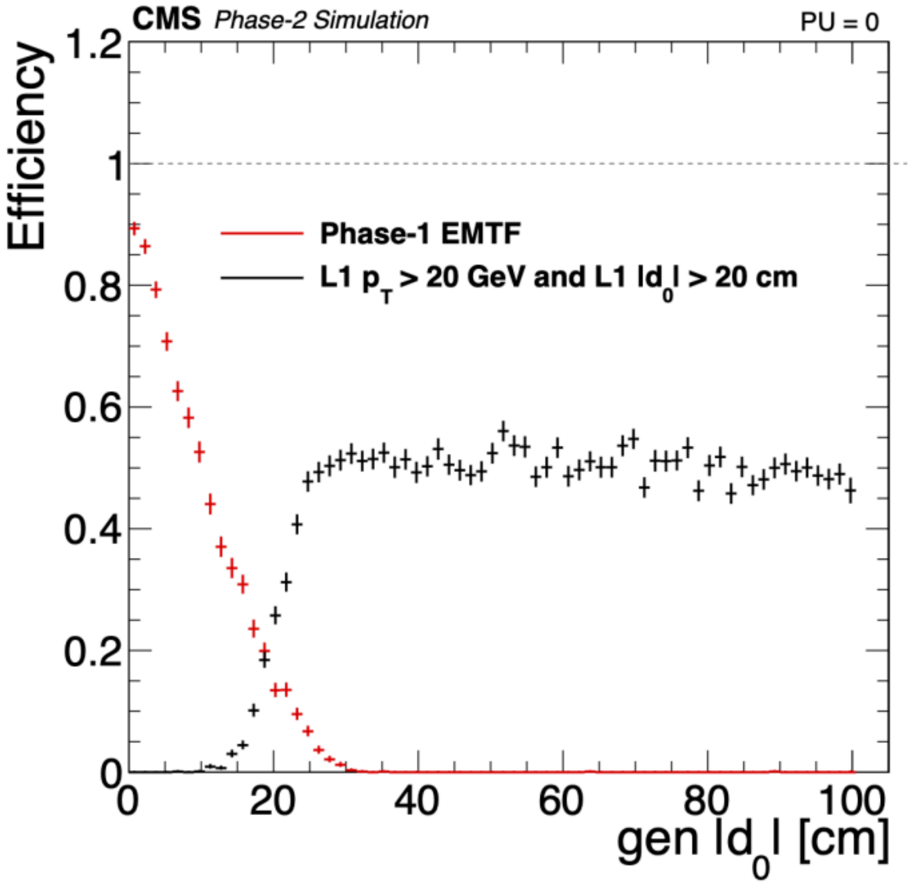


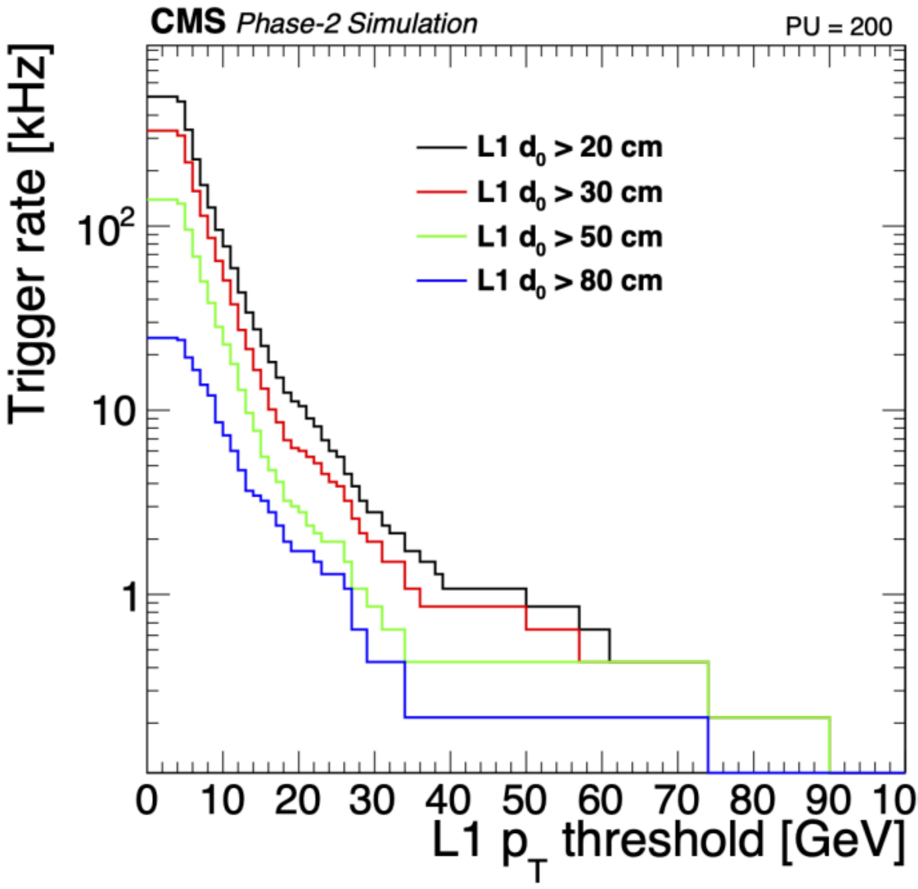


https://cds.cern.ch/record/2714892/

Phase 2 Displaced NN Performance

- The displaced NN was originally developed for Phase 2 and designed the same way as prompt NN.
- Performance of the displaced NN was evaluated for L1T Phase 2 TDR. Plots were made using Phase 2 EMTF patterns but Run 2 EMTF inputs (23 inputs instead of 36) due to unavailable Phase 2 inputs at the time.
  - Efficiency plots were made with 0 PU and L1 $p_T > 20$ GeV.
    - Using L1 $p_T > 20$ GeV and $d_0 > 30$ cm, efficiency ~50% up to 100 cm.
  - Rate plots were made with 200 PU.
    - Estimated rate for $d_0 > 20$ cm and L1 $p_T > 20$ GeV is around 10 kHz.





https://cds.cern.ch/record/2714892/

Conclusions

- EMTF is a good example of how ML algorithms can help trigger systems since Run 2.
  - Challenging conditions and/or hard to reconstruct objects can benefit greatly from ML solutions
- EMTF prompt BDT has been very successful in Run 2 and we will continue to support and use it for Run 3
- EMTF displaced NN is a great example of running ML algorithms directly in FPGAs and the hls4ml workflow.
  - Many lessons were learned during the integration process which will be very useful for Phase 2 NN implementations.
- Phase 2 EMTF (EMTF++) will use 2 NNs running directly in FPGAs and the current estimates show substantial improvements to muon triggering in the endcaps even in challenging conditions of HL-LHC.

Acknowledgement

- This material is based upon work supported by the U.S. Department of Energy, DOE Grant: DE-SC0023351