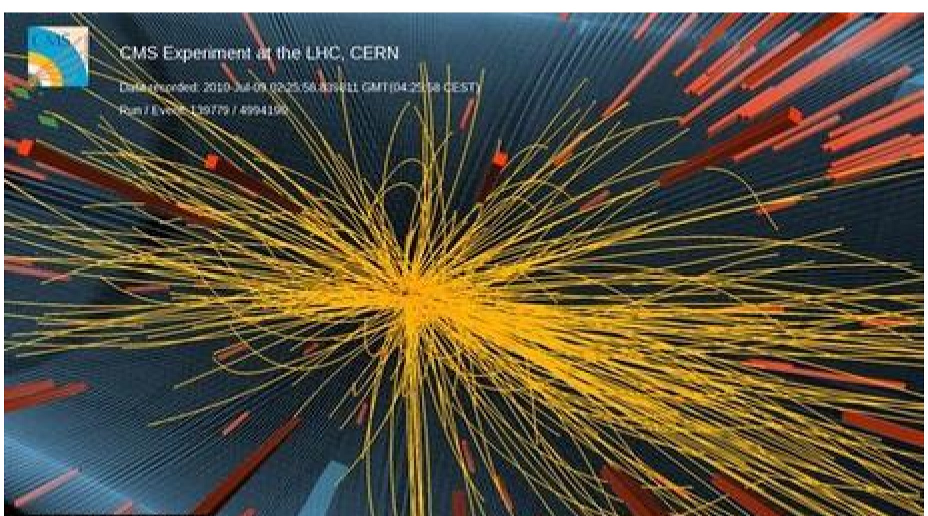
BDTs in the Level 1 Muon Endcap Trigger at CMS

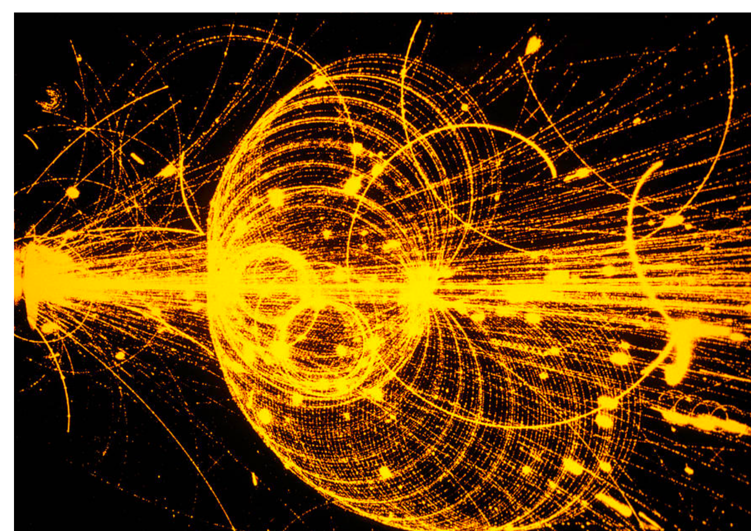
By Andrew Carnes
Darin Acosta, Andrew Brinkerhoff, Elena Busch, Ivan Furic, Sergei Gleyzer, Khristian Kotov, Jia Fu Low, Alexander Madorsky, , Jamal Rorie , Bobby Scurlock, Wei Shi

Intro

- At the Large Hadron Collider
  - **We want to save as much data as possible**
  - **But… there's way too much**
    - So throw out uninteresting events (proton collisions)
    - Keep interesting events
  - The Trigger decides which to throw out and which to keep
  - Needs to operate quickly!

- **Implemented machine learning to classify interesting vs uninteresting Muons** at one of the detectors called CMS
  - Implemented it in hardware: Field Programmable Gate Arrays (FPGAs)
  - First implementation of Machine Learning in a Level 1 Trigger at the LHC

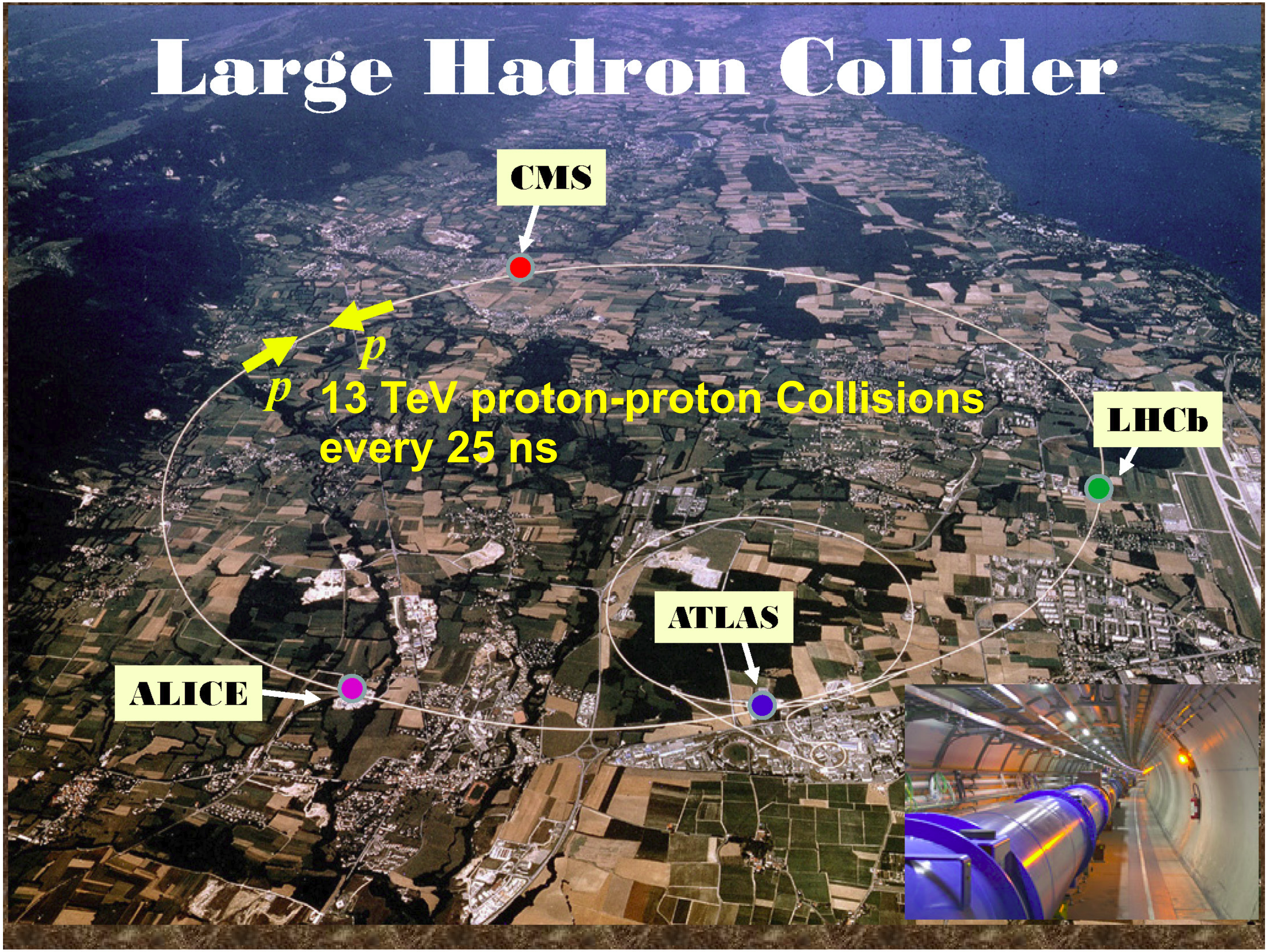




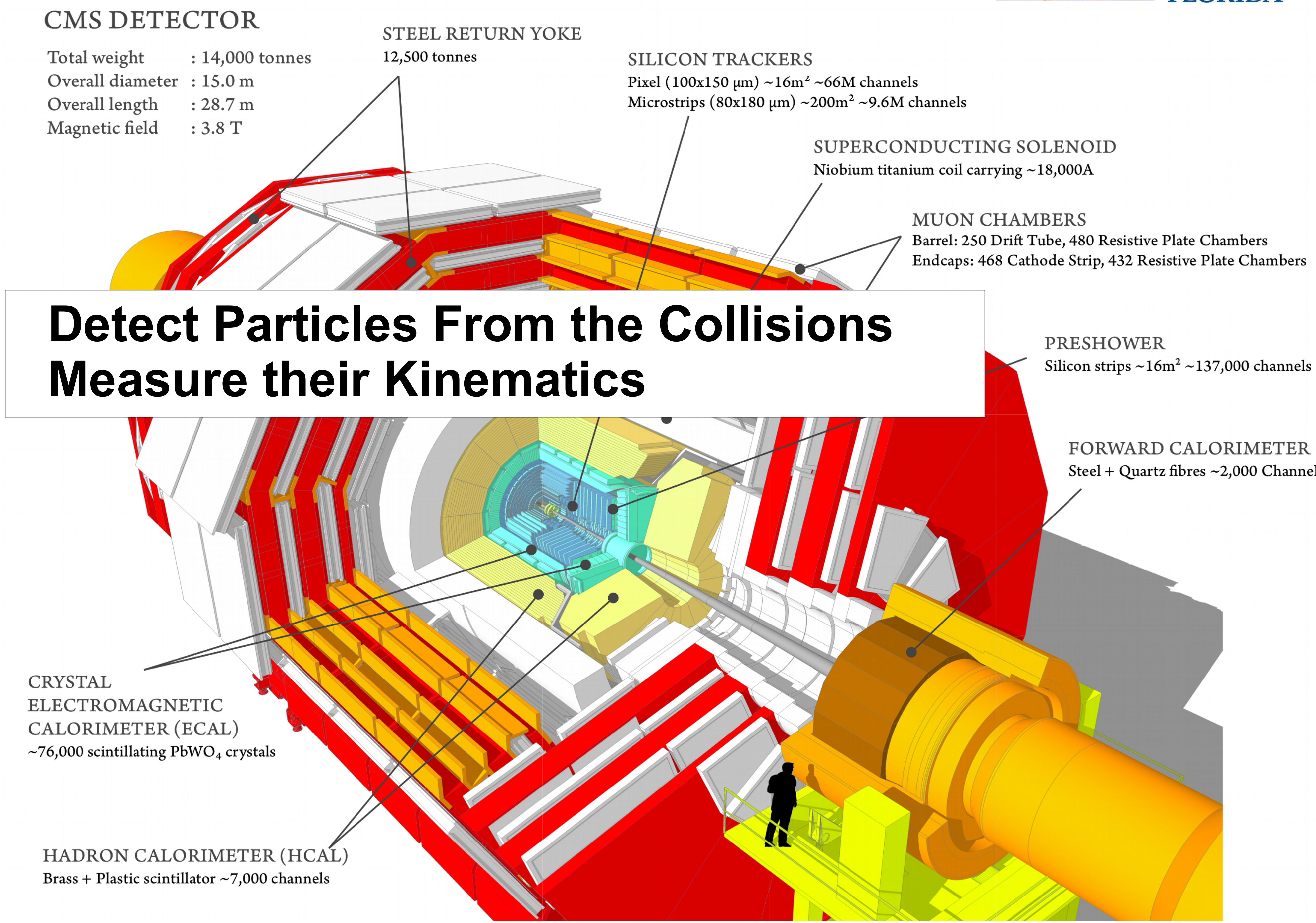
Outline

- Very Brief Context of the Project
  - The Large Hadron Collider
  - The Compact Muon Solenoid (CMS) Detector
  - The Trigger System at CMS
- Implementation of BDTs in the Endcap Muon Trackfinder (EMTF)
  - Machine Learning implemented in Hardware (FPGAs)
  - Runs online in real time
- Results
  - Substantial Improvements!

The Large Hadron Collider and The Compact Muon Solenoid Detector


Large Hadron Collider
CMS
p
p
13 TeV proton-proton Collisions
every 25 ns
LHCb
ATLAS
ALICE

Compact Muon Solenoid

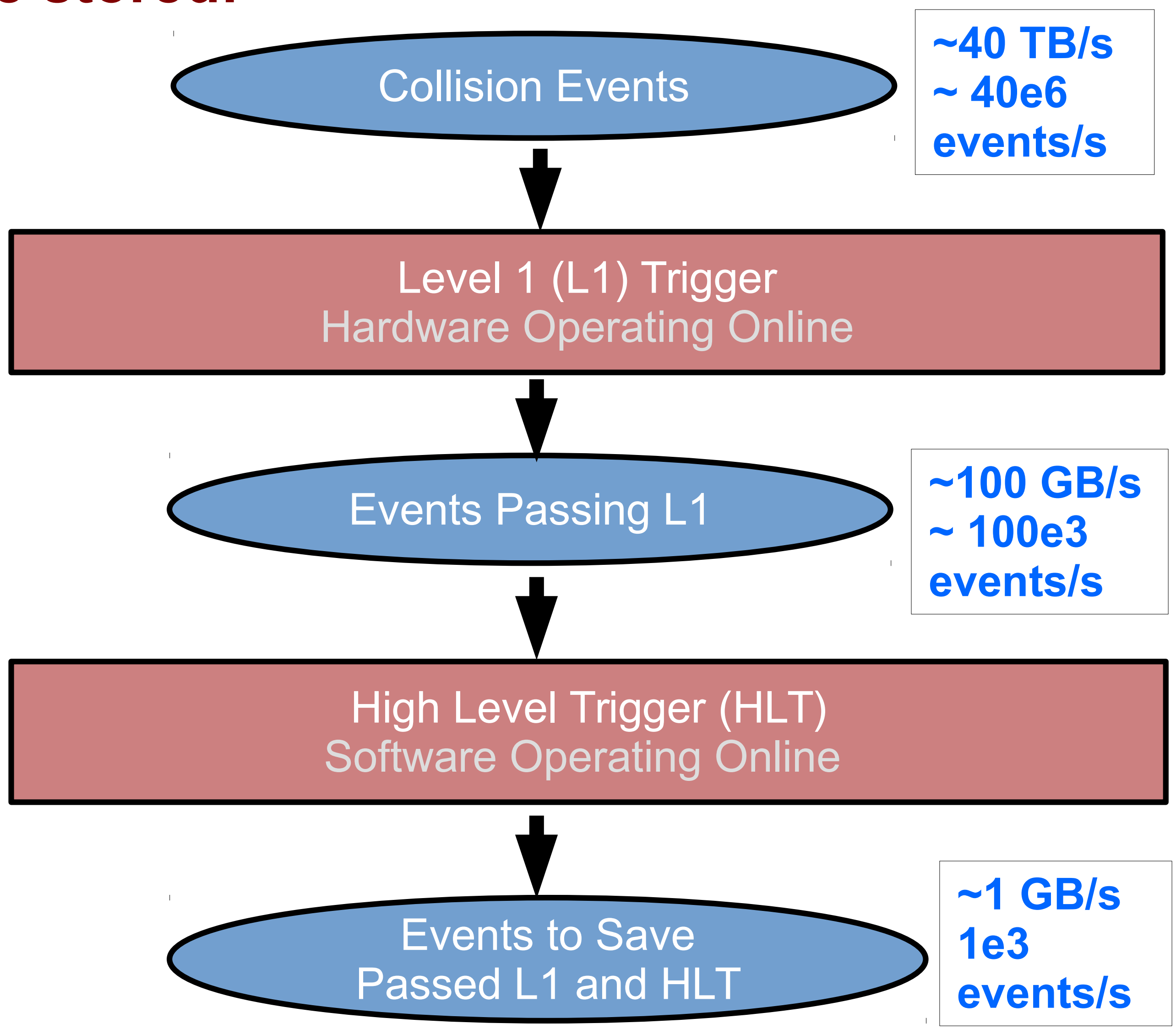


Detect Particles From the Collisions
Measure their Kinematics

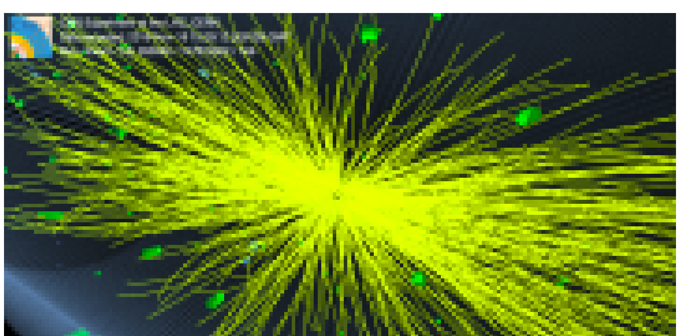
The Level 1 Trigger and the EMTF

CMS Trigger Overview

- **Too much data to save!**
- **The triggers filter events until a manageable amount of data can be stored!**
  - **40 Million/sec IN**
  - **1000/sec OUT**



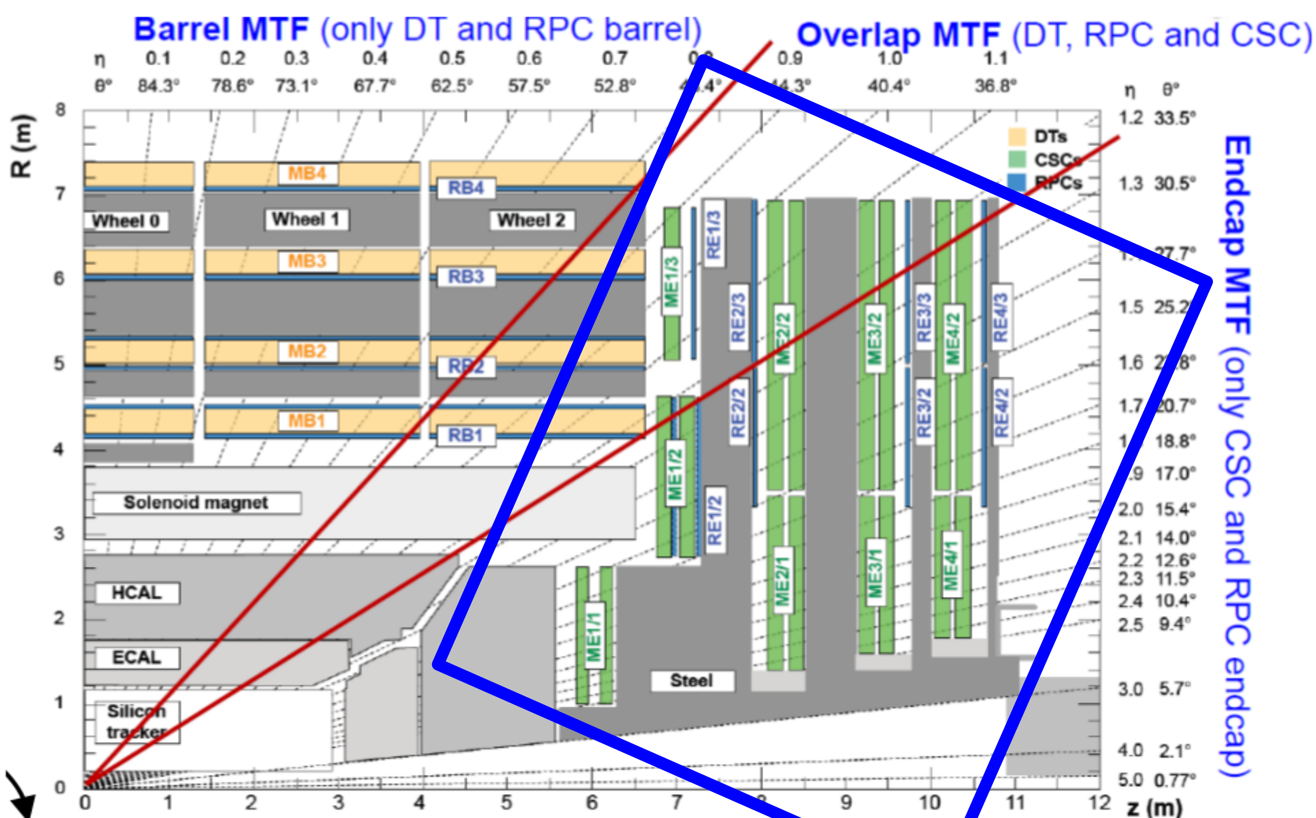
Event: bunches of protons collide

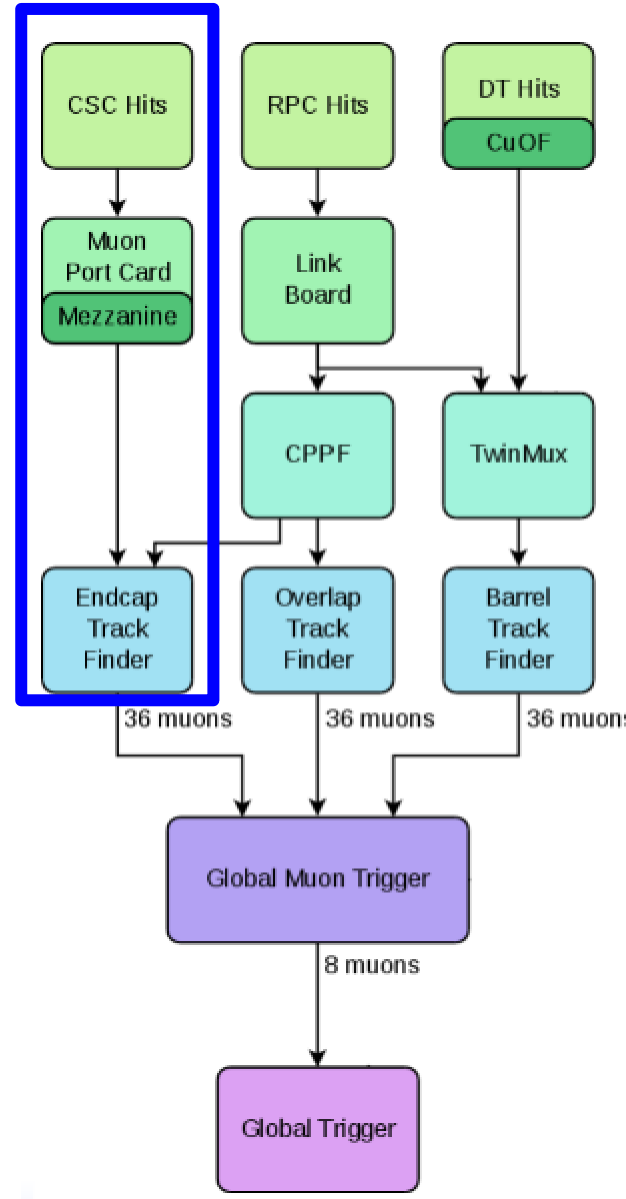


An Event at CMS

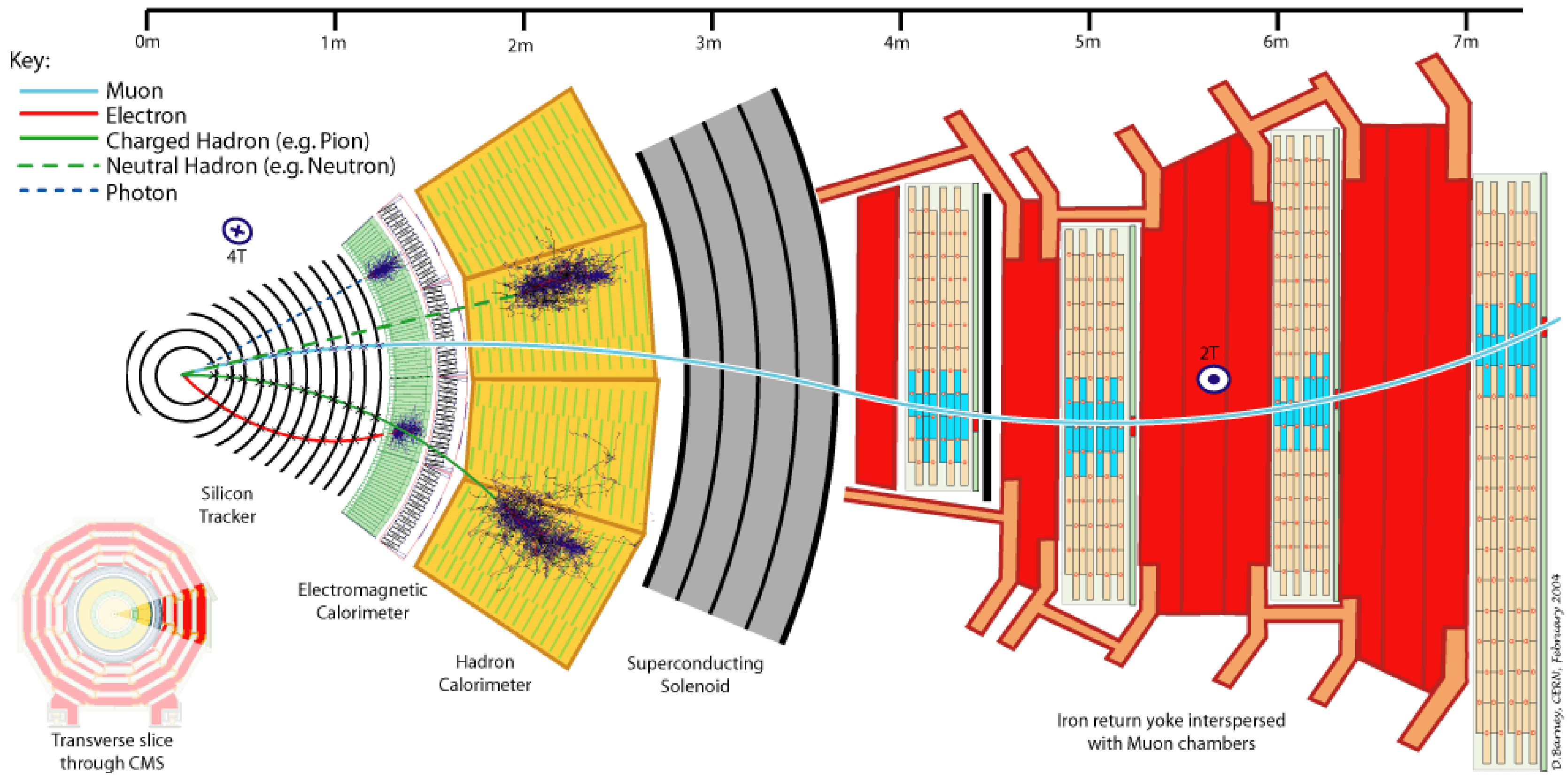
L1 Trigger and the EMTF

- **Level 1 (L1) Trigger** is responsible for selecting 100k interesting events out of 40 Million events every second at the LHC
- **Only have 3.0 μs for the entire process**
- **Endcap Muon Track Finder (EMTF)**
  - Part of the L1 Trigger System dedicated to Muons
  - Needs to operate FAST (~ 500 ns)
  - No tracker info available, only muon chambers





Muons Leave Tracks

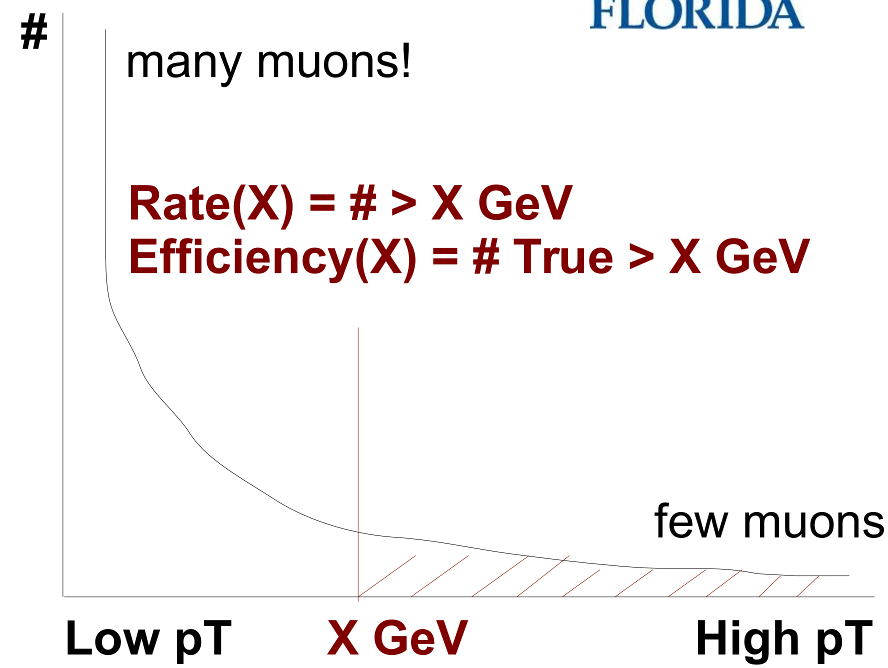


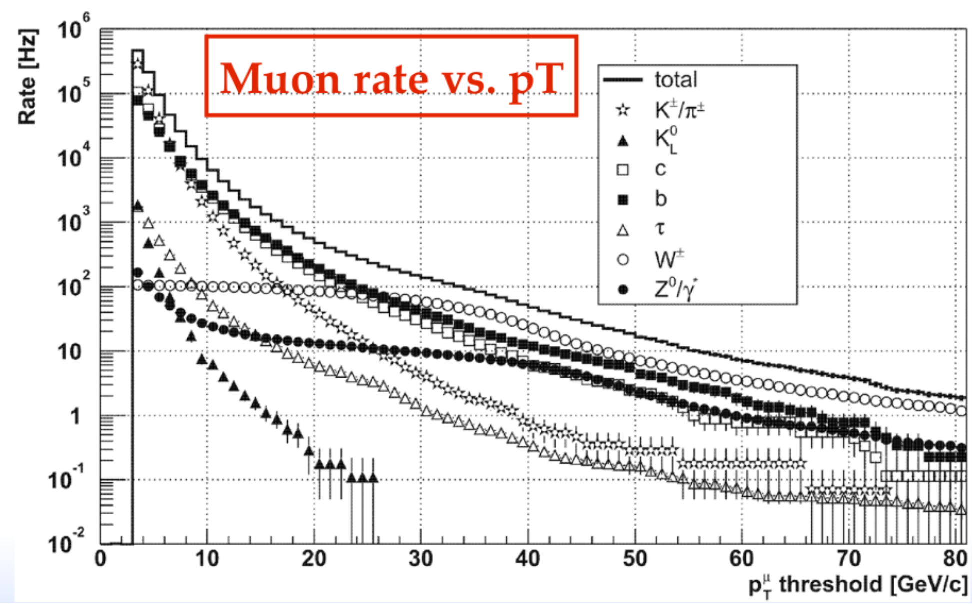
- Interesting muons have a large Transverse Momentum (pT)
- pT is assigned based upon the curvature in the magnetic field
  - Low momentum particles bend more in Φ
  - High momentum particles bend less in Φ
- EMTF needs to process hits and assign a momentum in ~500 ns

*the picture shows the barrel not the endcap, but gets the point across

EMTF Objectives

- **Metrics of Success**
  - **Rate(X)** – The number of muons predicted to be greater than X GeV
    - True AND False Positives
  - **Efficiency(X)** – The number of muons predicted to be greater than X GeV that should have been
    - AKA True Positives
- **EMTF Objective**
  - Minimize Rate while Maximizing Efficiency
  - In simpler terms
    - pass as little data > X GeV
    - but keep those actually > X GeV
- **Typical "Interesting" Event has pT > 25 GeV**
  - 1000 5 GeV muons for every 25 GeV Muon
  - Critical to reject as many low-pT as possible
  - Predicting low pT above threshold increases rate substantially



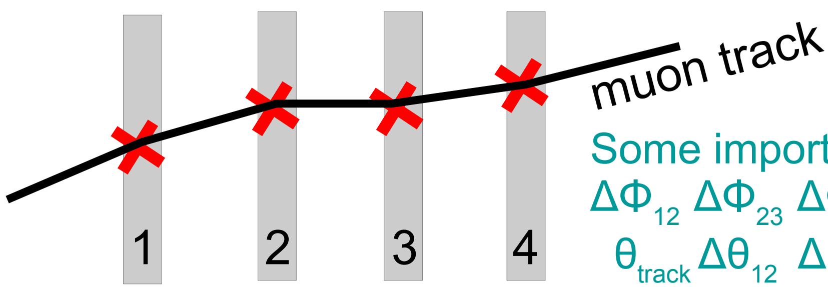


EMTF pT Assignment

Predict the pT well and the trigger will operate well

We have a **regression problem** with **many features***

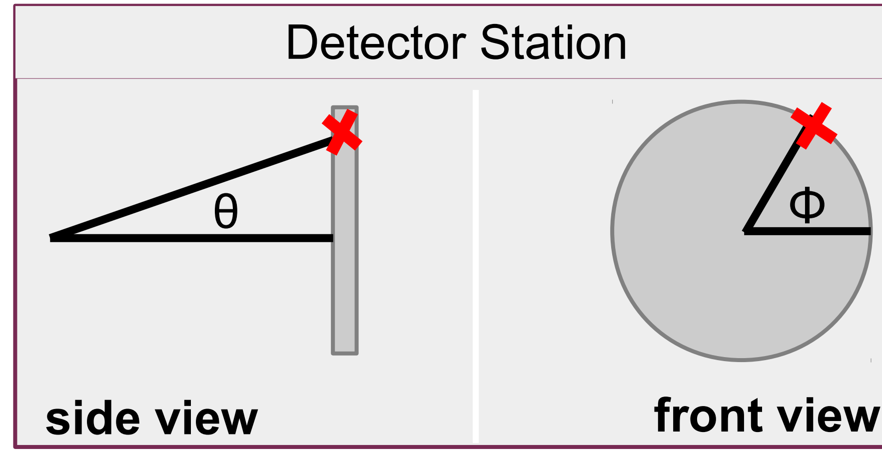
- 4 detection stations with Φ, θ info for each



Some important features

$\Delta\Phi_{12}$ $\Delta\Phi_{23}$ $\Delta\Phi_{34}$

θ_{track} $\Delta\theta_{12}$ $\Delta\theta_{12}$ $\Delta\theta_{12}$



Complicated Dependencies

- Non-uniform magnetic field in the endcap
- The muons may scatter between stations
- Muons shower charged particles from the material at high pT
- low pT muons may spiral completely before getting to the next station
    - looks like a straight line
    - actually went in a full circle

Many variables with complicated dependencies

- **Machine Learning should perform well**
- But evaluation is slow
- And the logic to implement the algorithm would take up lots of logic space from the FPGA...

*many, but not as many as say a picture

Getting Machine Learning into Hardware

How to Have your cake and Eat it Too

- **Want machine learning (ML) for accurate pT Assignment!**
- **Want it to operate in hardware quickly!**

- **Take a standard ML algorithm and estimate if it is fast enough**
  - Boosted Decision Tree with standard settings* would take about 2500 ns
  - only have 500 ns total for **ALL** EMTF calculations
    - Need most of the 500 ns to process measurements from wires and strips, build tracks, and then evaluate θ, Φ values
    - Standard evaluation of ML algorithm is not feasible on these time scales!
    - Moreover we would need to store all of the ~15,000 logical (<,>,+) operations for the BDT onto the FPGA… takes up too much logic
      - and that's in addition to the logic already present

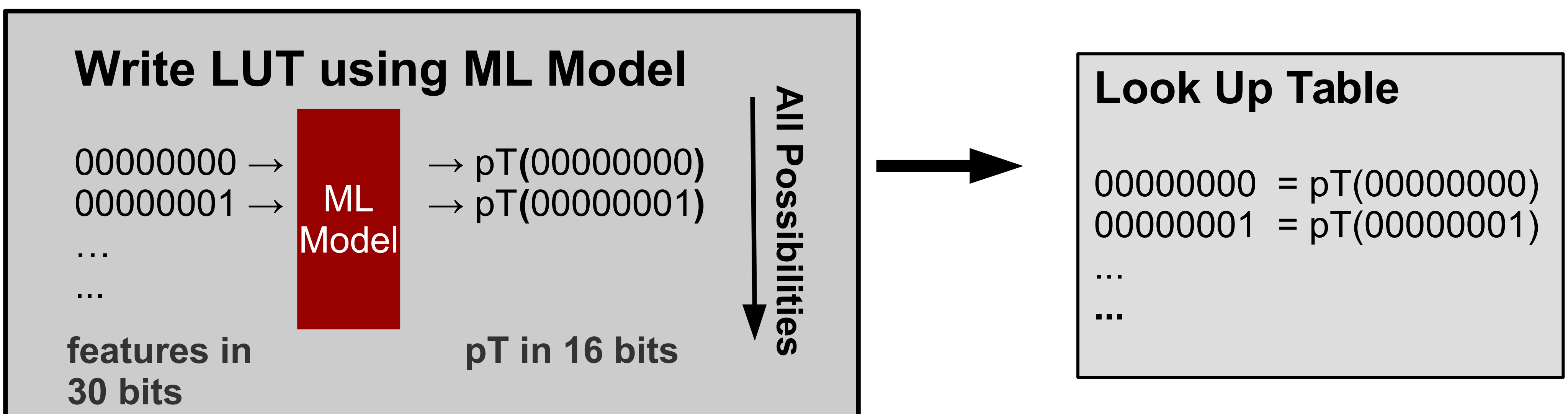
- **2500 operations to assign the pT for a single track! No thanks!**
  - Reduce the 2500 operations into 1 operation

* 500 Trees, depTh of 4, estimated for ~ 1 GHz processor

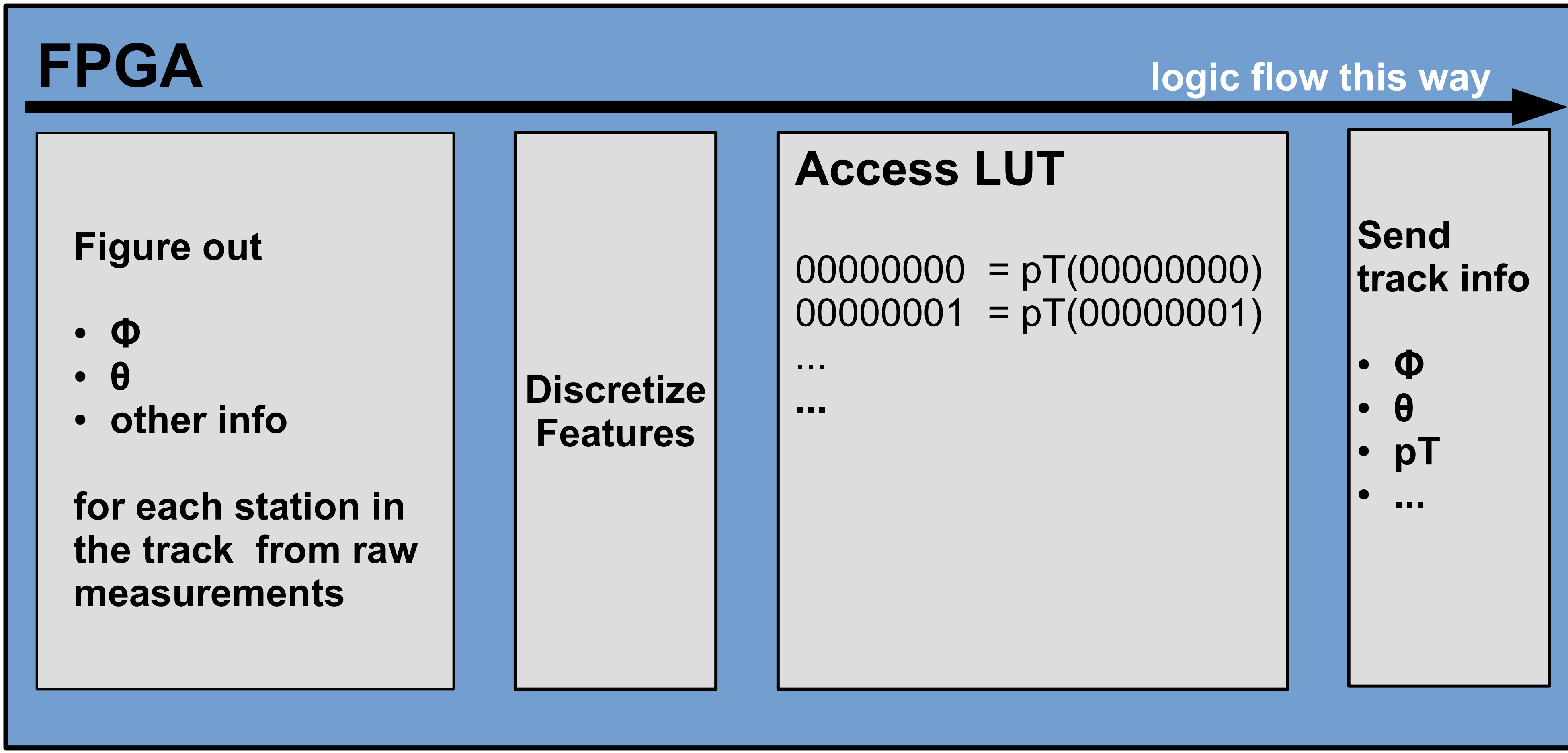
Create a Look Up Table

- **Turn evaluation from a Machine Learning (ML) Model into a single operation**
- **Trade time for memory**
  - **Create a Look Up Table (LUT)!**
  - Create offline, use online
  - Discretize features and fit into 30 bits
    - e.g. var1 = 10 bits, var2 = 5 bits, var3 = 5 bits, var4 = 5 bits, var5 = 5 bits
    - input = [var1 | var2 | var3 | var4 | var5] = 30 bits
  - Map each input to the ML model output and save the map
  - 2^30 possibilities w/ 9 bit outputs = 1.2 GB LUT

- **Versatile method that works for any fit algorithm**
  - However… Lose resolution on the features
  - Hard to fit lots of features into 30 bits



Summarizing the Logic

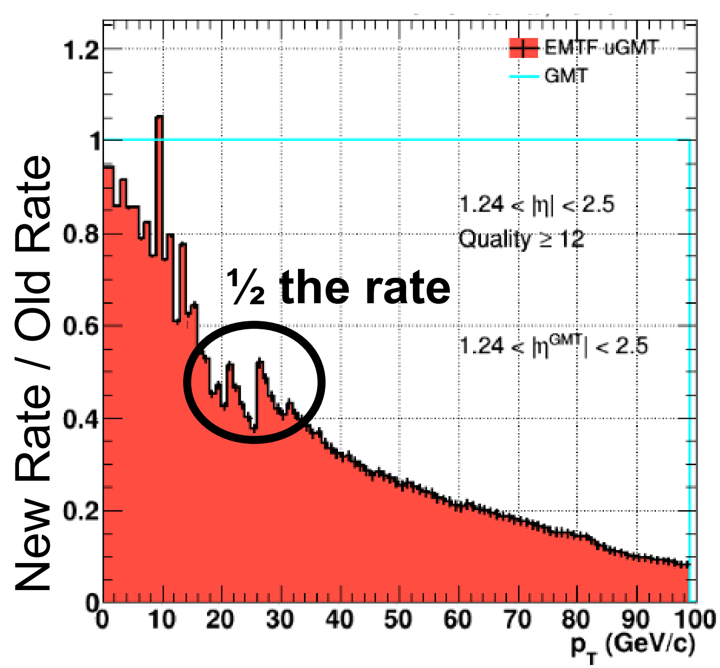


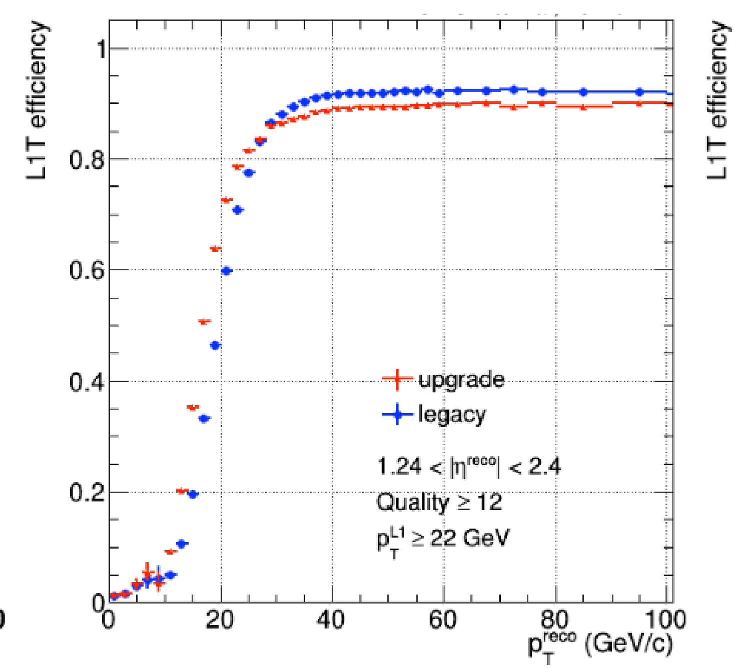
Results and Conclusions

Results in Practice

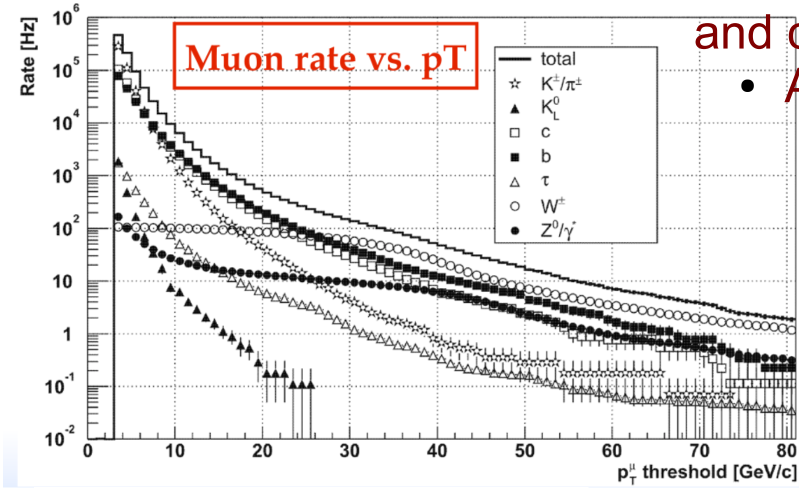
- **At the EMTF**
  - We trained a forest of Boosted Decision Trees (BDTs)
  - Then discretized features fitting them into 30 bits
  - Converted 2^30 possible features into a 2 GB LUT
  - Put the LUT into the FPGA
  - **Implemented this design in 2016/2017 data taking**

- **Improved the EMTF trigger by a factor of 2!**
  - 2x rate reduction (for pT > 22 GeV) with small loss of efficiency
  - Comparing 2016/2017 BDT based EMTF to the 2015 EMTF trigger

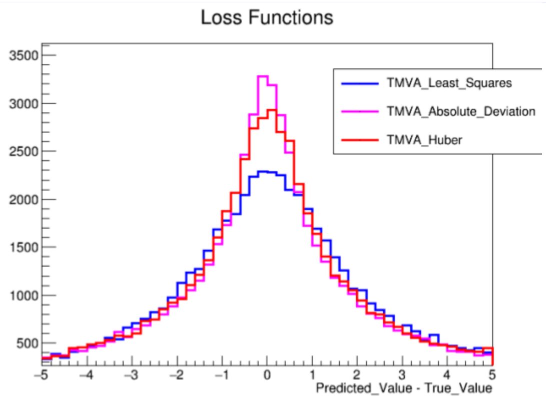




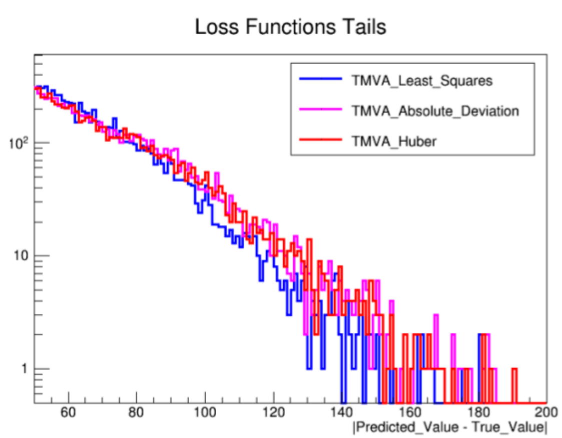
Lessons Learned



- Interesting problem since it is somewhere between regression and classification
  - Above or below pT threshold? For many thresholds
- **Main Problem**
  - **1000 5 GeV muons for every 25 GeV muons**
  - **Really need to focus on low momentum events**
    - If low pT events are predicted greater than their actual pT the rate increases substantially!



- **Use a Transformation + Loss Function to focus on low pT events**
  - Targeting 1/pT makes differences in low pT large, count more in loss
  - Loss = |1/pT – 1/pT_true|$^{\mathbf{2}}$ ← Change exponent to penalize differences more/less
  - Focus on low pT more → lower rate (good) , lower efficiency (bad)
  - Focus on low pT less → higher rate (bad), higher efficiency (good)



Create variables to identify outliers

- Problem: strange dΦ bend between two chambers due to scattering or showering throws off pT assignment
- Add Feature: average dΦ, |dΦ| calculated without the outlier
- Add Feature: variable identifying the outlier station

Conclusions

- **Implemented Boosted Decision Trees in a Field Programmable Gate Array**
  - Created a Look Up Table (LUT)
    - Make offline, use online
    - Map from 2^30 possible discretized feature values → 9 bit pT
  - LUT turns pT assignment into an O(1) operation running in << 500 ns
  - Accurate pT assignment improved our trigger by a factor of 2
- **LUT method is versatile and possible for any Machine Learning method**
  - Great for implementing a ML method where fast decisions are important (like a trigger)
  - Might be difficult to fit all important features into ~30 bits total
- **Some Future Ideas**
  - Train on Data rather than MC using HLT tracker pT as the “truth”
    - Very high statistics for training and testing very quickly
    - The pT distribution and hence the rate of muons differs between Data and MC
  - Craft a loss function that directly models our metrics: Rate and Efficiency

The End