Machine Learning predictor for measurement-to-track association for the ATLAS Inner Detector trigger

Nisha Lad

Supervised by: Dmitry Emeliyanov & Prof. Nikos Konstantinidis

Department of Physics & Astronomy
University College London

3-6 April 2022

Outline

1. Introduction
2. Tracking & Motivation
3. Measurement-to-track Association
4. Performance Evaluation
5. Summary

The ATLAS Detector & Inner Detector (ID)

- General purpose detector at the LHC, aims to make Standard Model precision measurements & test BSM theories
- The ID is dedicated to track & vertex reconstruction, & consists of 3 sub-systems
- **Pixel detector & Insertable B-Layer (IBL)** are closest to the beamline; **have the highest hit occupancy**
- Semiconductor Tracker (SCT)
- Transition Radiation Tracker (TRT)
- The trigger is part of the High Level Trigger (HLT) & performs **fast online track & vertex finding**. The ID **Fast Tracking algorithm** uses seeded track finding using combinatorial track following

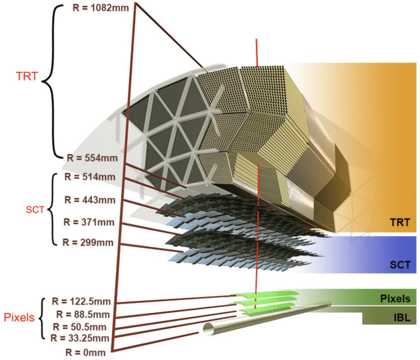


Figure 1: The ATLAS ID layers

[1] https://doi.org/10.48550/arXiv.2007.07624

Tracking Finding as a Pattern Recognition Problem

- **Associate individual measurements into sequences** representing tracks
- Typical scale: $O(10^5)$ hits per event & several 1000s of tracks
- Track finding algorithm for the LHC Run-2 data taking period is based on **combinatorial track following** using track seeds
- Typically the number of seeds **scales non-linearly** with the number of hits (∼cubical)
- Motivation for novel Machine Learning (ML) approaches in track finding that could lead to large savings in CPU as pileup increases
- **Aims:** create a ML algorithm to **predict if a pair of hits belong to the same track** given input hit features & optimize the HLT ID track seeding by **reducing the proportion of fake seeds**

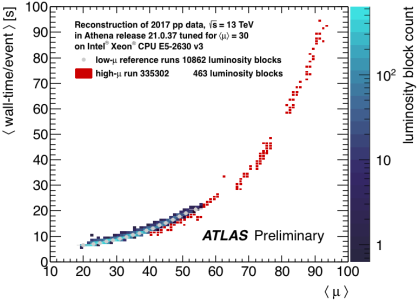


Figure 2: Wall time per event for ID reconstruction in 2017

[2] https://twiki.cern.ch/twiki/bin/view/AtlasPublic/ComputingandSoftwarePublicResults

Data Exploration & Feature Extraction

Training Data

- Monte Carlo (MC) $t\bar{t}$ events at 13 TeV and mean pileup multiplicity $< \mu > = 80$, using run-2 geometry
- **Seeds:** groups of three spacepoints located in different detector layers; fake seeds are groups of spacepoints that do not originate from the same simulated track
- Seeds constructed at the combinatorial stage of ATLAS track seeding were extracted
- **Doublets (hit pairs) are extracted from pixel-only seeds**, the inner doublet is denoted by spacepoints: (1,2) & outer doublet: (2,3)

Ground truth labels obtained from MC truth

- **Correct hit association (1):** hit pairs belong to the same track & correspond to a truth particle
- **Incorrect hit association (0):** hit pairs do not belong to the same track

Input Features

- Consider the $r - z$ plane
- **Absolute inverse track inclination** $|cot(\theta)|$ where θ is the angle of inclination to the doublet hits from the z axis
- w_η **pixel cluster width measured in the** η **direction**
- $\eta \equiv -ln[tan(\theta/2)]$

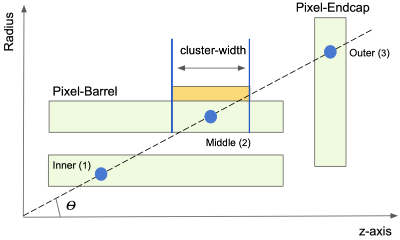
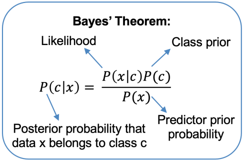


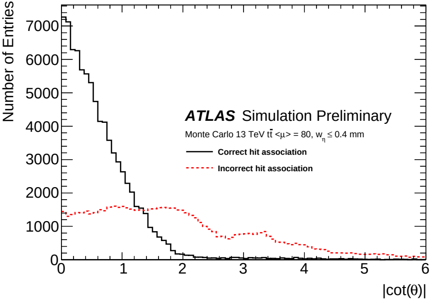
Figure 3: Illustration of a triplet seed in the $r - z$ plane pixel layers of the ID

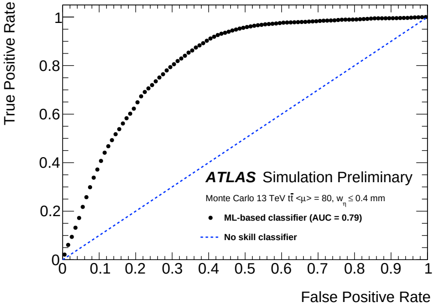
Classifier Development & Training

The Model

- Binary classification using a **'Not-so-Naïve' Bayes' classifier** with a generative model
- **Likelihood is computed via a Kernel Density Estimate** fitted to each doublet class
- $|cot(\theta)|$ forms the input feature to be learned for varying **pixel cluster width** w_η
- ROC curve is used to adjust probability threshold cut
- Each classifier was tuned to yield **True Positive Rate (TPR) = 0.95** to maintain high purity of doublets





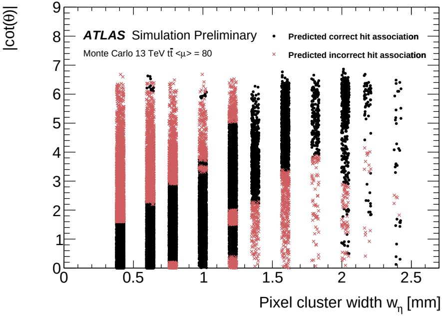


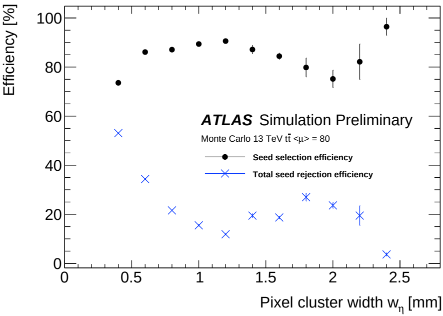
[3] https://twiki.cern.ch/twiki/bin/view/AtlasPublic/HLTTrackingPublicResults #ATL_COM_DAQ_2021_003_Machine_Lea

Classifier Development & Training

The Model

- Predictions were made for each 1D distribution & plotted as an **'acceptance-rejection'** region
- Acceptance region (black) converted to Look-Up Table (LUT), ensures fast look-up in HLT ID track seeding
- **Reduces computational overhead** & training does not need to be done on the fly
- Classifiers for pixel-barrel doublets & pixel-endcap doublets are trained separately
- Seed selection efficiency 74.8 ± 0.1% & total rejection rate 41.5 ± 0.1%, gives an indication of relative speed-up



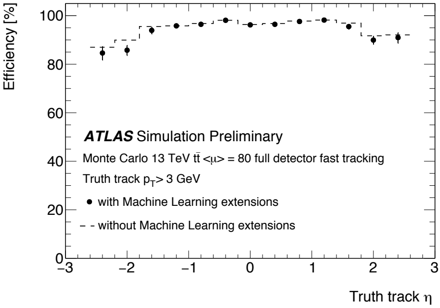


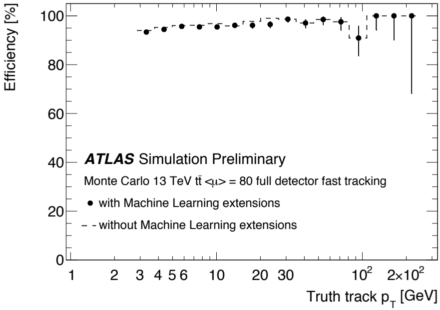
[3] https://twiki.cern.ch/twiki/bin/view/AtlasPublic/HLTTrackingPublicResults #ATL_COM_DAQ_2021_003_Machine_Lea

Performance: Efficiency vs. Track Parameters

Key Results

- Pixel barrel & endcap LUT fed directly into the HLT Fast Tracking algorithm
- $<\mu>= 80$: **93.9% average tracking efficiency** (nominal 95%)
- An overall **2.3× speed-up factor**
- Efficiency loss is mainly observed at large $|\eta|$





[3] https://twiki.cern.ch/twiki/bin/view/AtlasPublic/HLTTrackingPublicResults #ATL_COM_DAQ_2021_003_Machine_Lea

Performance: CPU Time Comparison

- Greatest saving in CPU time is achieved during the **Seed Processing** stage
- ML filtering for pixel seeds yields $\sim$ **78% fewer seeds**

Table 1: Breakdown of speed-up factors observed for different stages of the HLT ID Fast Tracking algorithm at $<\mu>= 80$

Total Speed-up Factor	Seed Generation	Seed Processing	Track Fitting
2.3	1.3	3.3	1.5

Table 2: Performance of the full detector tracking at various average pile-up multiplicities with application of ML filtering on pixel seeds in the ID

$<\mu>$	Efficiency Loss (%)	Total Speed-up Factor
40	0.7	1.6
60	0.7	2.1
80	1.1	2.3

[3] https://twiki.cern.ch/twiki/bin/view/AtlasPublic/HLTTrackingPublicResults
#ATL_COM_DAQ_2021_003_Machine_Lea

Summary

- The application of a ML-based classifier for seed selection in the ATLAS Inner Detector has provided **significant CPU savings** on trained MC data at various pileup levels
- The trained predictor in the form of a LUT yields **2.3$\times$ speed-up with minimal loss in efficiency (1.1%) at $< \mu > = 80$** compared with the standard trigger tracking
- The developed ML pipeline provides a way to generate custom LUTs by training the predictor to yield a required TPR, dependent on the degree of efficiency required
- Reducing the proportion of fakes at an earlier stage in the ATLAS HLT track seeding, ensures the reduction in CPU usage overall
- **Efficient use of computing power** will become an increasingly important factor in the selection of physics objects as the luminosity and pileup increase during future upgrades of the LHC program

Backup

ATLAS Trigger System & the ID Trigger System

ATLAS trigger system is separated in 2 main stages

- **L1: Level 1 hardware stage**
  - Identifies Regions of Interest (RoI)
  - Reduces data rate to 100kHz, $< 2.5\mu s$
- **HLT: High Level Trigger software stage**
  - Processes RoIs identified by L1
  - Reduces data rate to 1.5kHz
  - Between 200ms & 500ms

ID trigger is part of HLT system & performs fast online track and vertex finding

- Data preparation: Detector elements are reconstructed for given spatial Region of Interest (RoI)
- Fast tracking algorithm optimised for track finding efficiency provides initial track fit & parameters
- Seeded track finding using combinatorial track following
- Precision tracking:
  - Applies offline track fit using tracks from the Fast Tracking algorithm
  - Runs the ambiguity solver algorithm to remove duplicate tracks

Precision & Recall

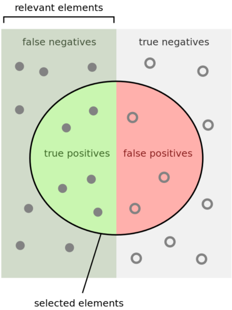
Definitions

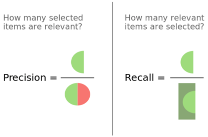
- **Precision = TP/TP+FP** Ratio of correctly predicted positive observations to the total predicted positive observations
- **Recall (Sensitivity) = TP/TP+FN** Ratio of correctly predicted positive observations to the all observations in actual class
- **F1 Score = 2*(Recall * Precision) / (Recall + Precision)** Weighted average of Precision & Recall, taking both FPs and FNs into account

Predicted Class

Actual Class		**Positive**	**Negative**	
	Positive	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

(Recall)





[4] https://en.wikipedia.org/wiki/Precision_and_recall