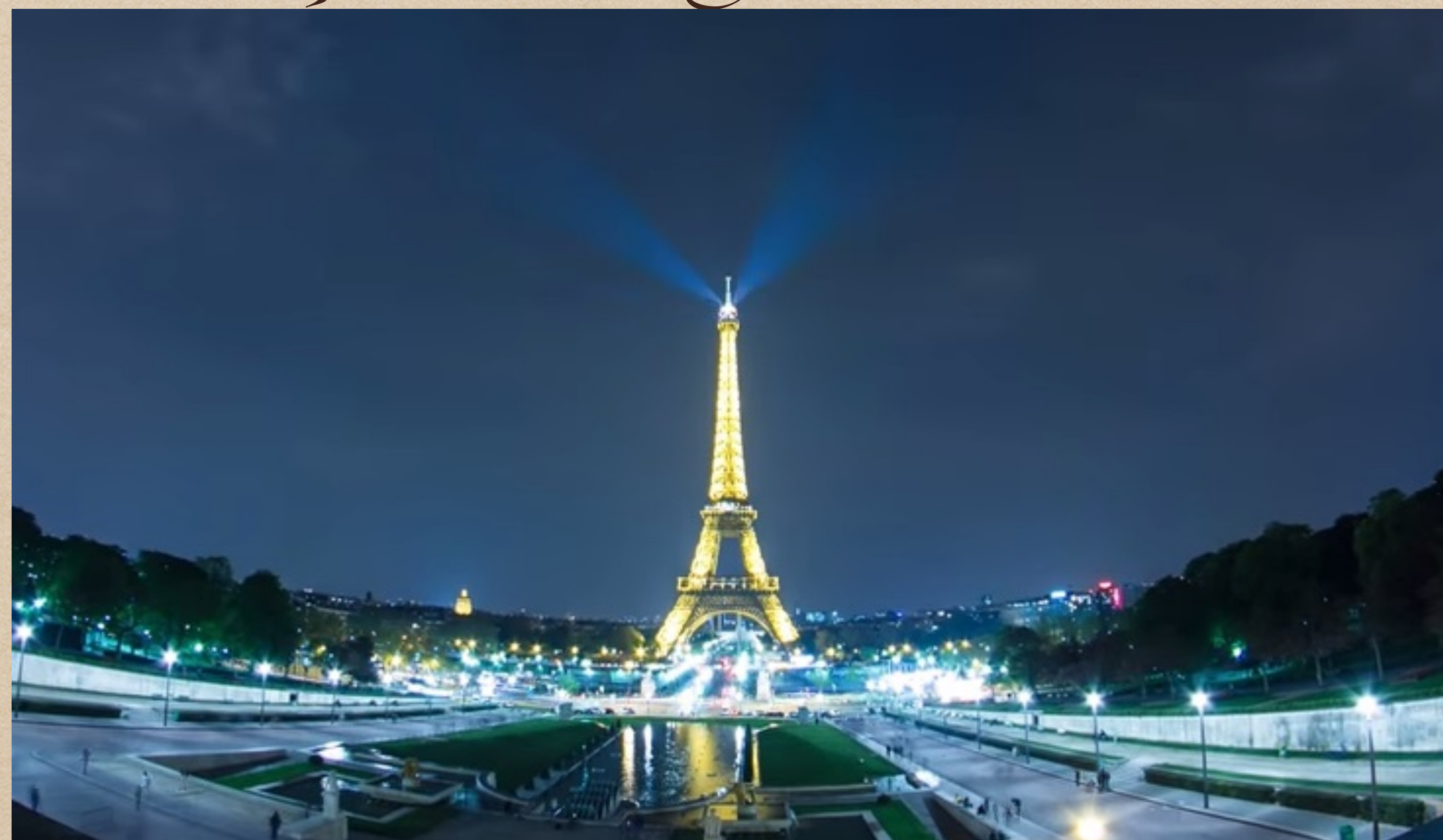
Deep Neural Nets and Bonsai BDTs in the LHCb pattern recognition



Adam Dendek*, on behalf of the LHCb collaboration
*AGH University of Science and Technology, Krakow Poland

Connecting The Dots / Intelligent Trackers 2017

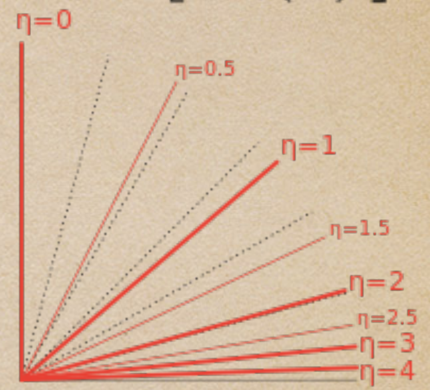
Paris 09.03.17

Part one

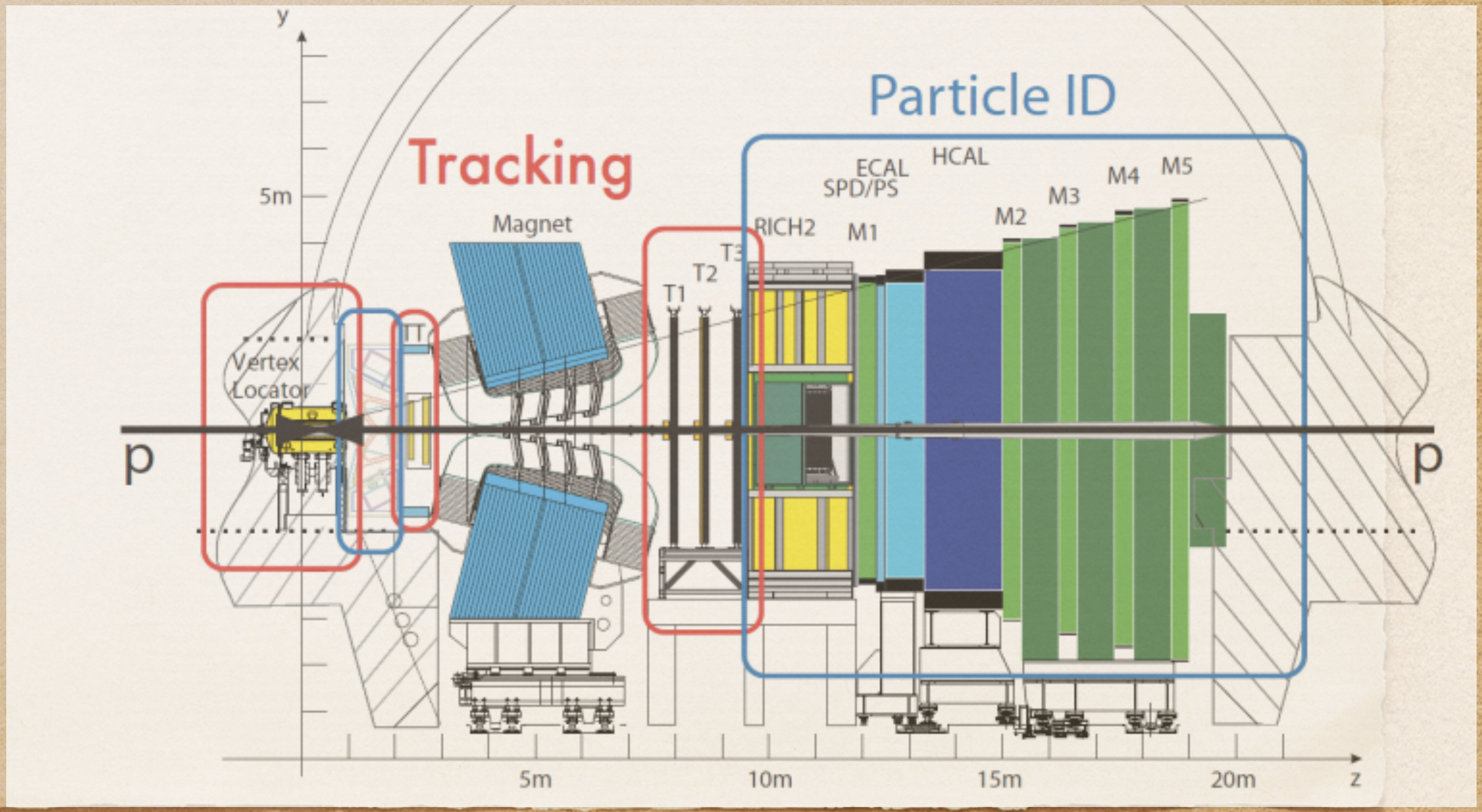
Introduction

The LHCb detector

$$\eta \equiv -\ln\left[\tan\left(\frac{\theta}{2}\right)\right],$$



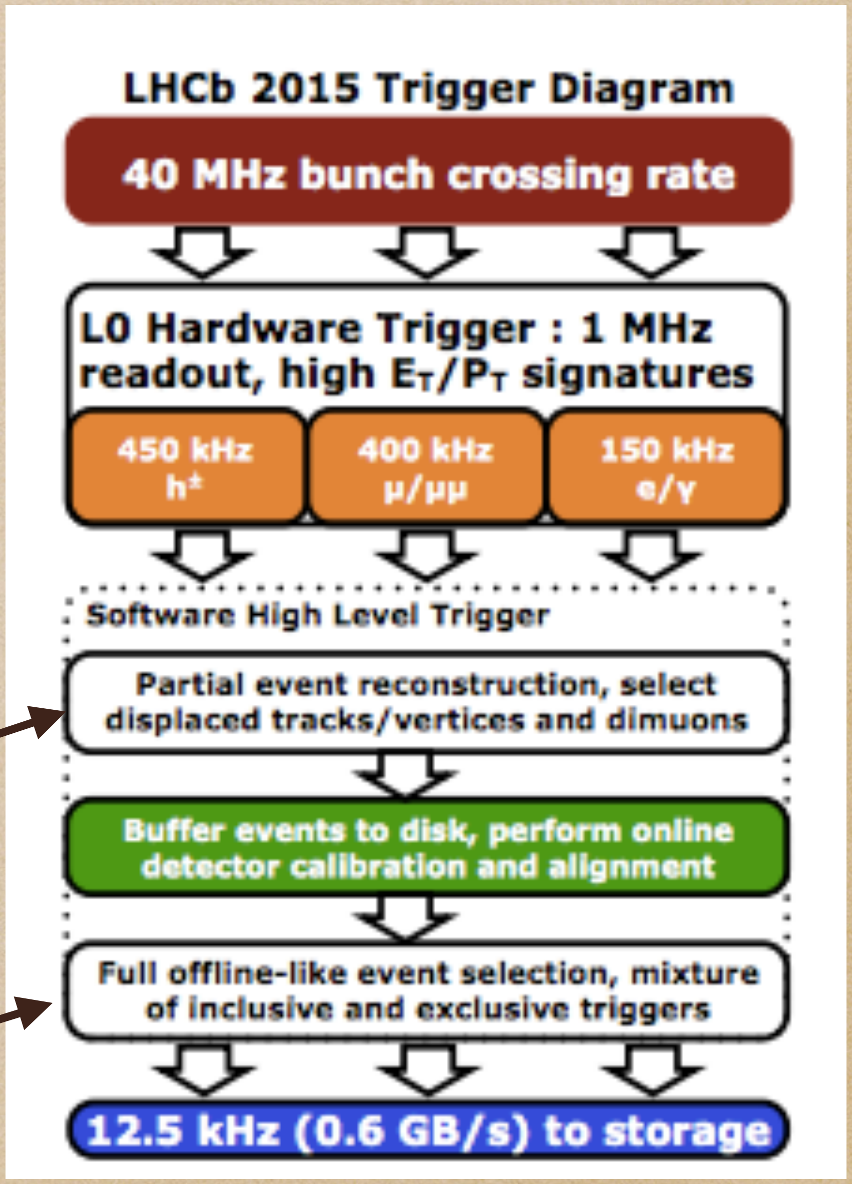
- The LHCb detector is a general purpose single arm forward spectrometer. Heavy hadrons are produced boosted in the forward region: $2<\eta<5$
- The main aim of this detector is to focus on studying CP violation in beauty and charm decays as well as rare decays of b and c hadrons
- To achieve above goals excellent detector performance is expected:
  - track reconstruction efficiency > 95 %,
  - Momentum resolution dp/p~ 0.5 - 1%
  - decay time resolution ~ 45 fs
  - Excellent particle identification:
    - ε(K->K) ~ 95%.
    - ε(μ->μ)~ ~ 97%.



[Int. J. Mod. Phys. A 30, 1530022]
[JINST 3 S08005]

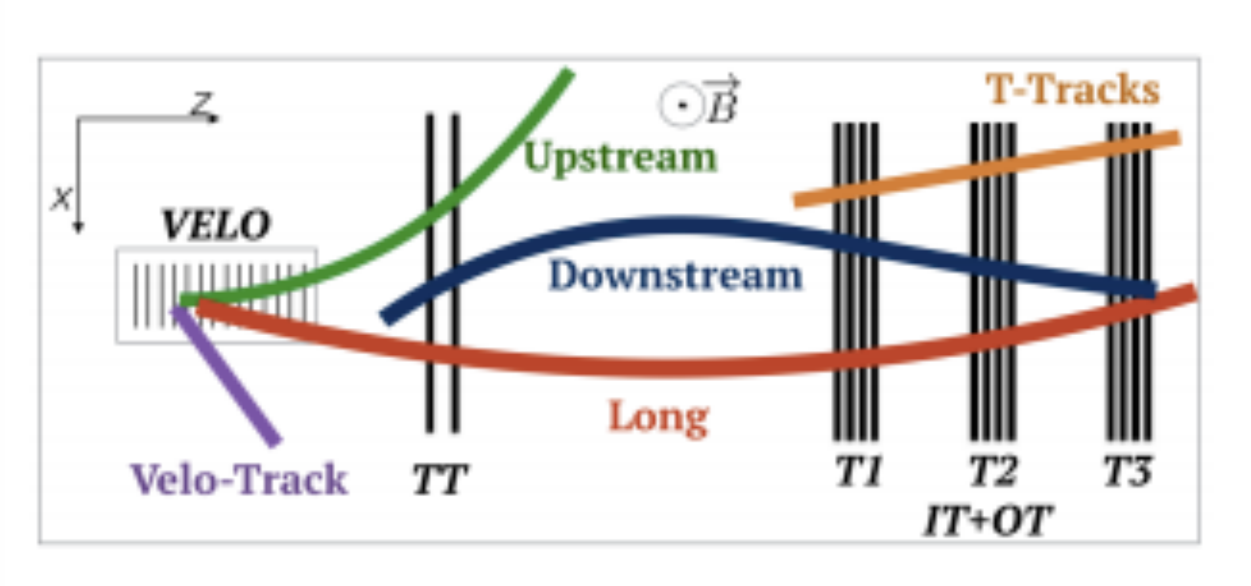
LHCb in Run II

- LHCb moved to real time reconstruction, alignment and calibration setup in Run II
- This allows to achieve the offline quality of the event selections performed at trigger level.
- The track reconstruction had to be made identical online and offline.
- We need to make track reconstruction faster without performance loss.
- Reconstruction in two stages
  - Fast stage (HLT1) for long tracks with pt>500 MeV and tighter track quality requirements.
    Time per event ≈ 40 ms
  - Full stage (HLT2) achieves offline efficiency and precision. Time per event ≈ 800 ms



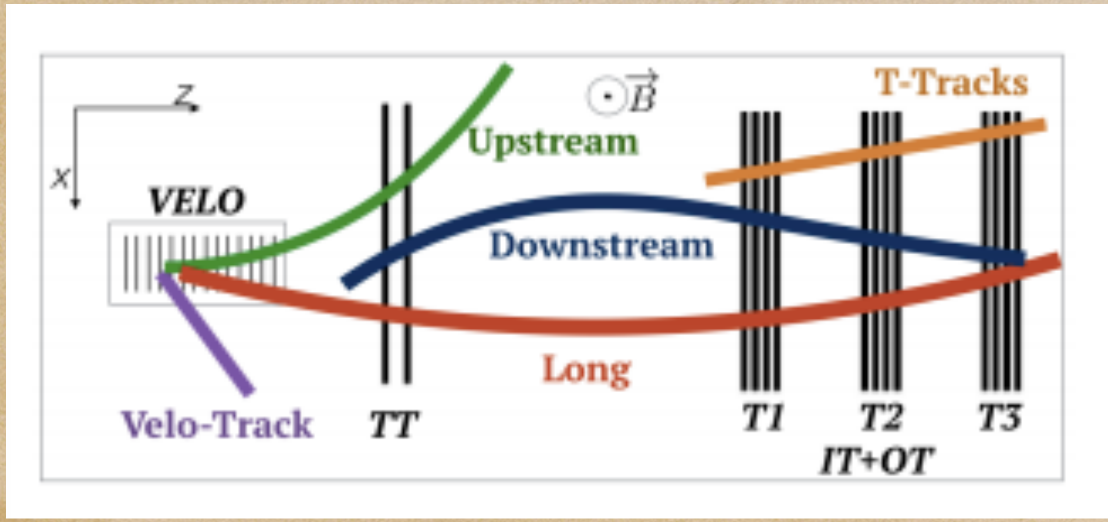
Track reconstruction at LHC b

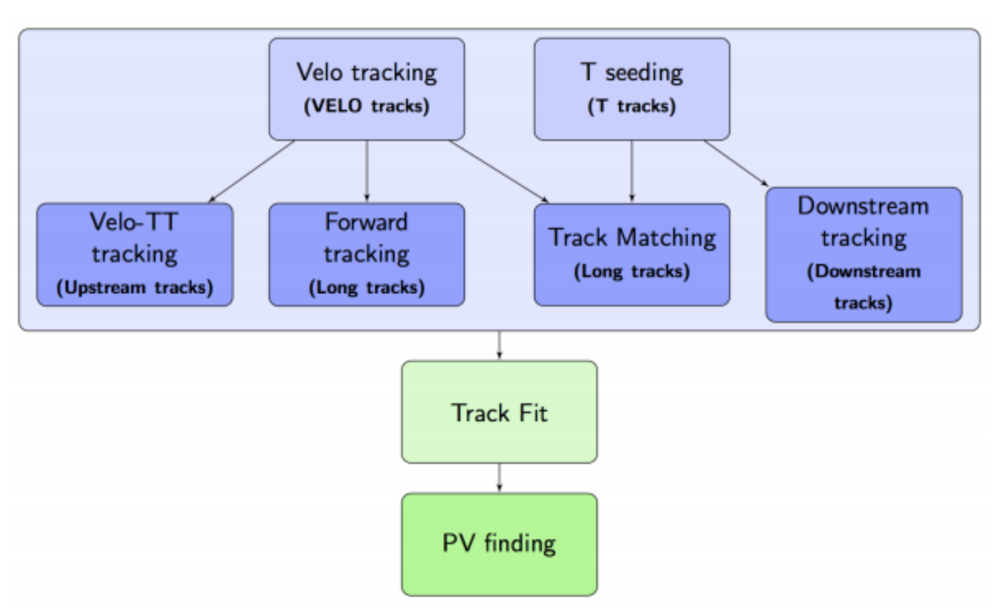
- We distinguish two most important track types (from the physics analysis point of view):
- Long Tracks
  - Hits in VErtex LOcator, Inner Tracker and/or Outer Tracker and can have hits in TT
  - Used in majority of analyses
- Downstream Tracks
  - Hits in TT and IT/OT
  - Tracks from daughters of long lived particles
- The LHCb tracking contains two steps:
  - Track finding - pattern recognition algorithm
  - Track fitting - using Kalman filter. Used for track parameters estimation taking into account multiple scattering and energy losses.
- The tracking has to be fast and efficient keeping low fake rate!



LHCb tracking pattern recognition algorithms

- Pattern Recognition algorithms find individual hits that compose a track
- VELO Tracking: A stand-alone search is made for straight line track segments in the VELO
- T seeding: A stand-alone search is made for track segments in T stations
- Forward tracking: Starting from seeds in the Velo, tracks are searched in T stations
- Track matching: Starting from a set of tracks reconstructed in the Velo and a second set reconstructed in T stations, track matching attempts to match the tracks in the two sets to one another.
- Downstream tracking: Using tracks in T stations, algorithms implementing this strategy search for matching TT hits.

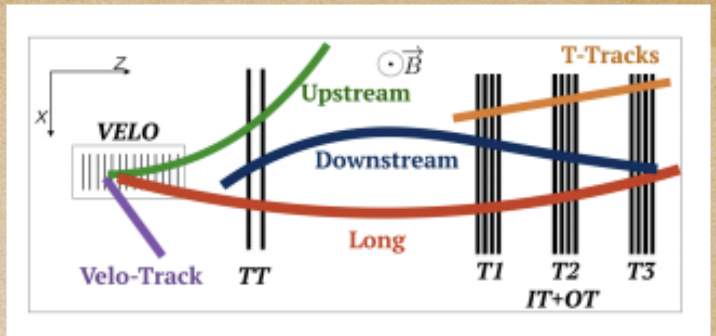




Part two

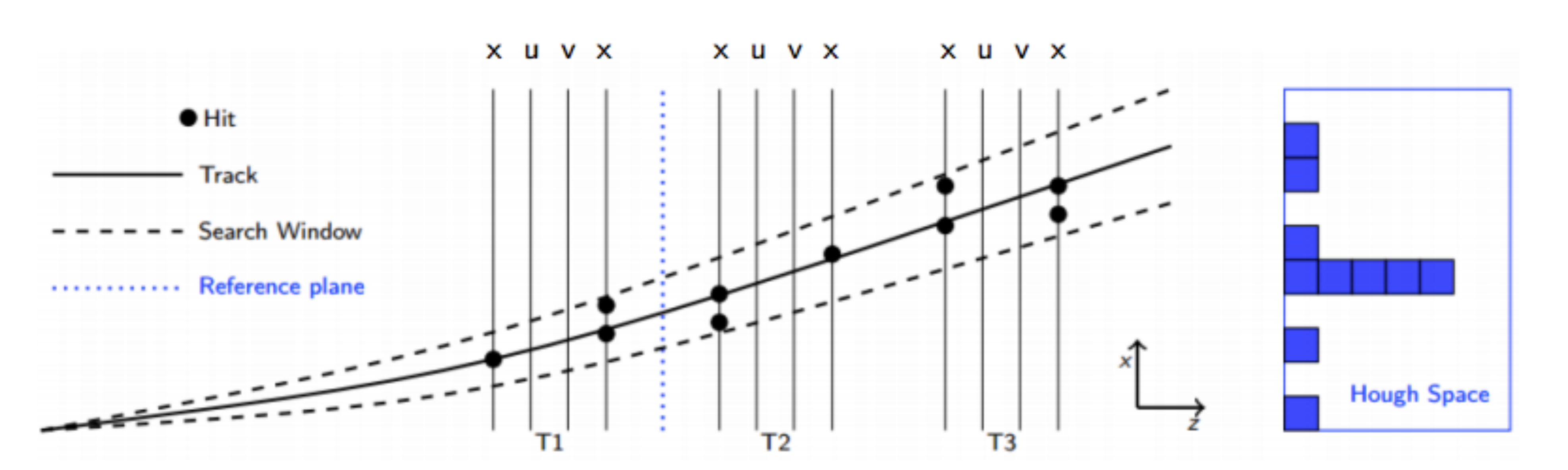
Machine Learning in Long Track Reconstruction

Fake long tracks rejection

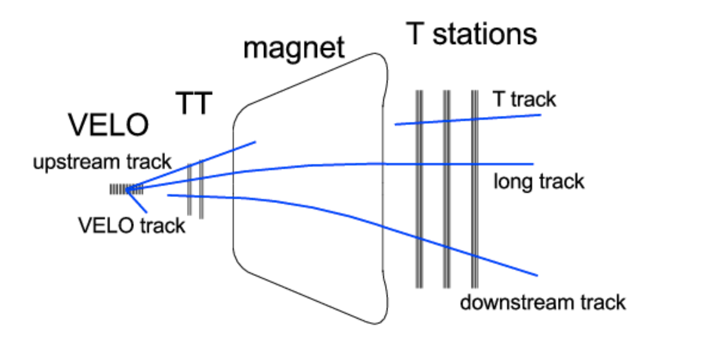


- Fake long tracks mainly come from wrong matching between VELO and T stations.
  - There is no tracking detector in the magnet section, very long lever arm.
- Remaining fake tracks from Kalman filter χ^2/dof cut ≈22%
- The aim of the project is to reject fake track at stable efficiency in early stages of processing (forward tracking)

Machine Learning in forward tracking



- Forward tracking algorithm contains following steps:
  - Search window in T station defined by VELO track estimate and minimal pt
  - Project x-hit into reference plane (create clusters) - Hough transformation
  - Fit x-cluster and remove outliers
  - Add and fit track with stereo hits
  - Recovery loop in HLT2 for track candidates with hits in only 4 x-layers



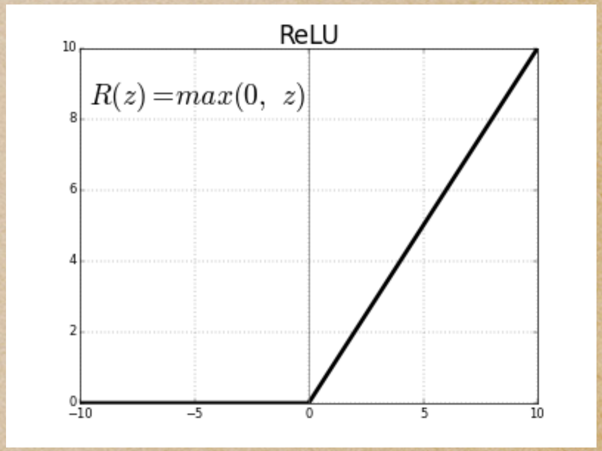
- We trained two neutral networks
  - First of them is tuned for rejection of bad 4-layer-x-clusters in recovery loop
  - Second one is trained for candidates selection after stereo fit

Deep neural networks architecture

- The models used in forward tracking are two Feed Forward Neural Networks.

$$\mathcal{L}(y,\hat{y}) = -y\log\hat{y} - (1-y)\log(1-\hat{y})$$

- As a lost function cross-entropy
  - Measures the similarity between expected and predicted value
  - One way to interpret cross-entropy is to see it as a (minus) log-likelihood for the data y under a model y'
  - Models parameters are optimized to minimize this function (gradient descent algorithm)
  - C-E is chosen as estimator, since it mostly leads to fast convergence and good results in terms of classification error

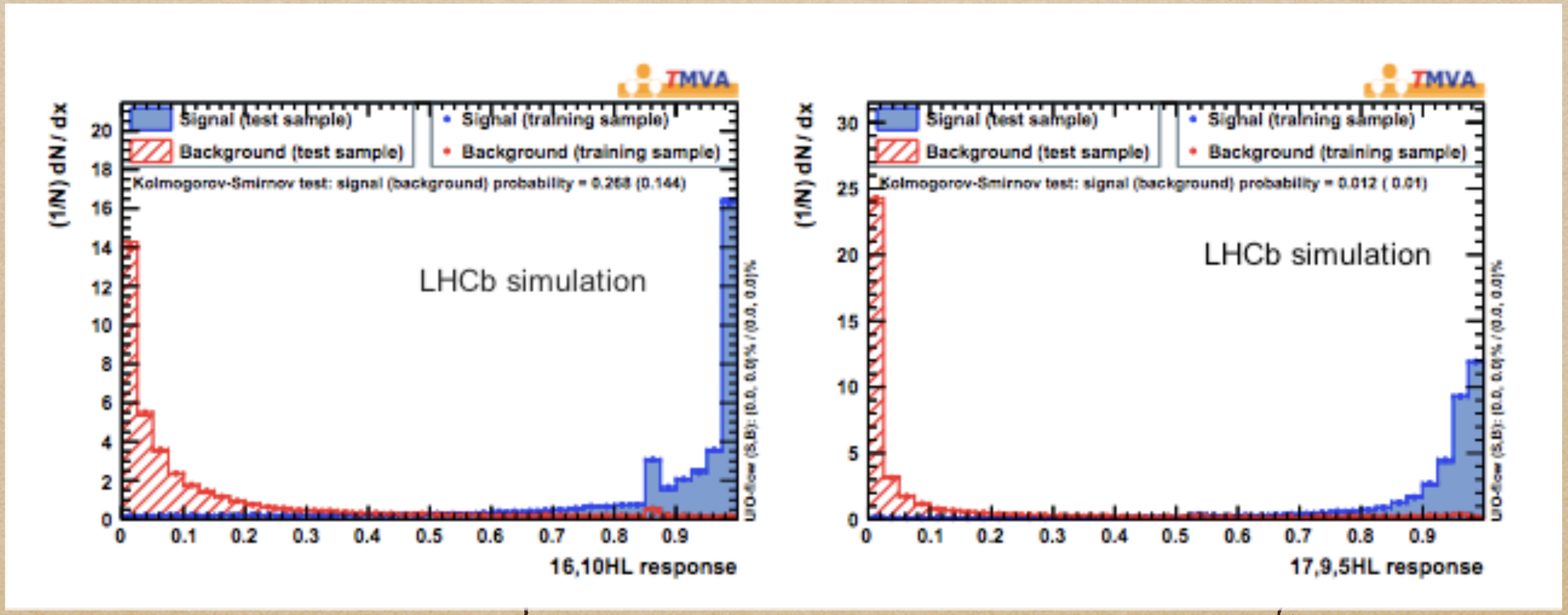


- As an activation unit the ReLu has been chosen.
  - efficient gradient propagation
  - Fast computation: Only comparison, addition and multiplication

Deep neural networks architecture

- The model were trained using TMVA library
- The architectures (depth and hidden unit per layer number) was the hardest hyper-parameter to tune
  - NN in recovery loop : 9 input nodes, 2 hidden layer (16 and 10 unit)
  - NN track selection: 16 input nodes, 3 hidden layer (17,9,5 units)

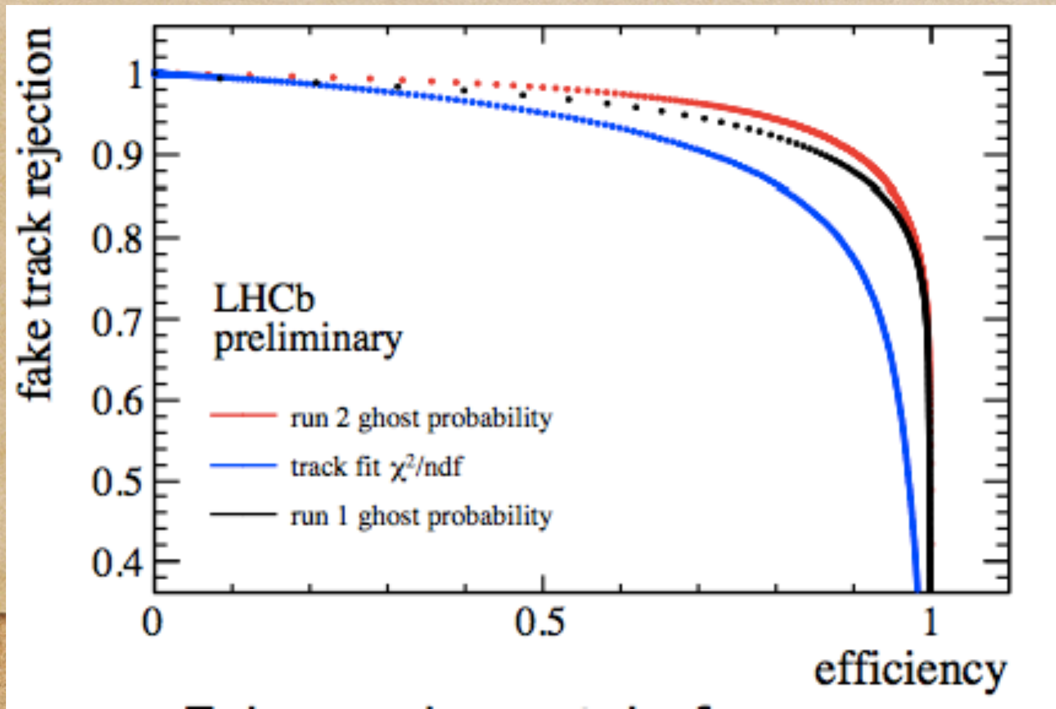
Classifier response for test and training sample



HLT2 recovery loop Stereo fit

Ghost probability

- For Run I, a quantity called ghost probability, based on a neural network output was used offline, such that the analyst could use it as a cut variable.
- In Run II the ghost probability is calculated by the second Neural Networks - online.
- According to this number algorithm decide whether accept or reject the track candidate - Final track selection
- This approach allows to decrease ghost rate from 22% to 14%



Performance of Deep Neural Networks for forward tracking

- Neural Networks were trained with MC and minimum bias events
- Results:
    - Increased efficiency
    - Reduced fake rate considerably
    - Both Deep Neural Networks contribute to 2% (HLT2) and 0.5% (HLT1) to timing of forward tracking algorithm

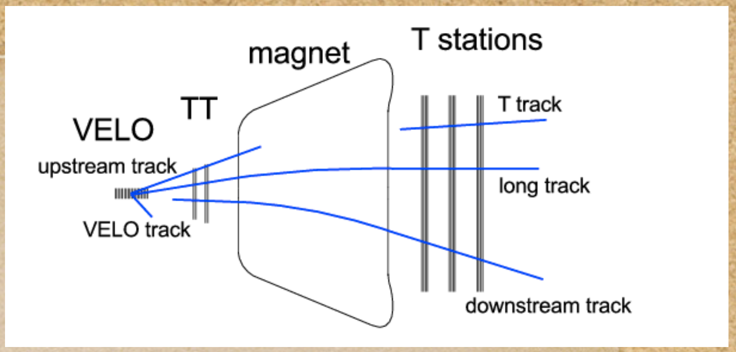
MC performance 2016 w.r.t. 2015	ν =1.6 w/ RL	ν =1.6 w/o RL
timing HLT1	± 0 %	
timing HLT2	$+4$ %	-38 %
fake rate	-27 %	-35 %
fake rate HLT1	-15 %	
ε long	$+0.5$ %	$+0.1$ %
ε long from B	$+0.2$ %	-0.2 %
ε_{HLT1} long from B $p > 3, p_T > 0.5$ GeV	$+0.1$ %	

LHCb-TALK-2016-362

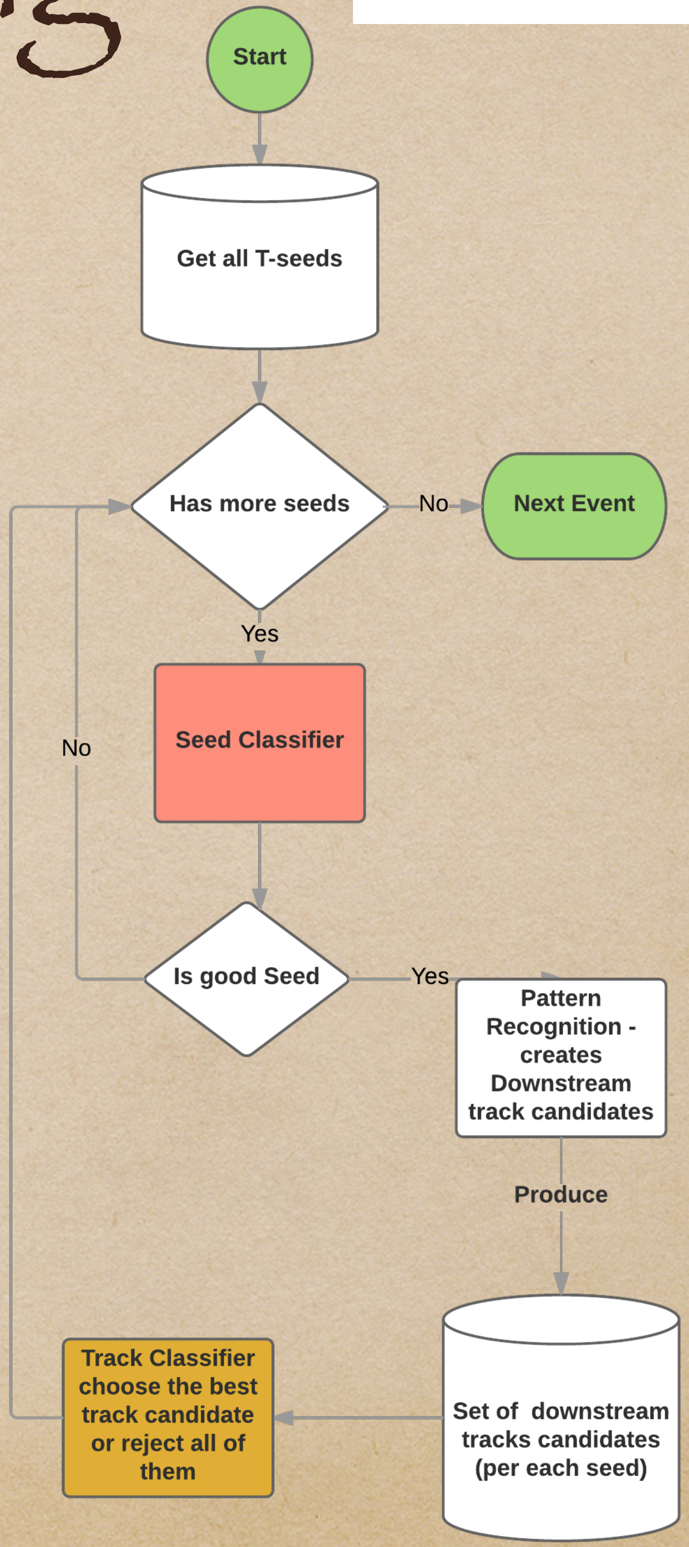
Part three

Machine Learning in Downstream Tracking

Downstream Tracking



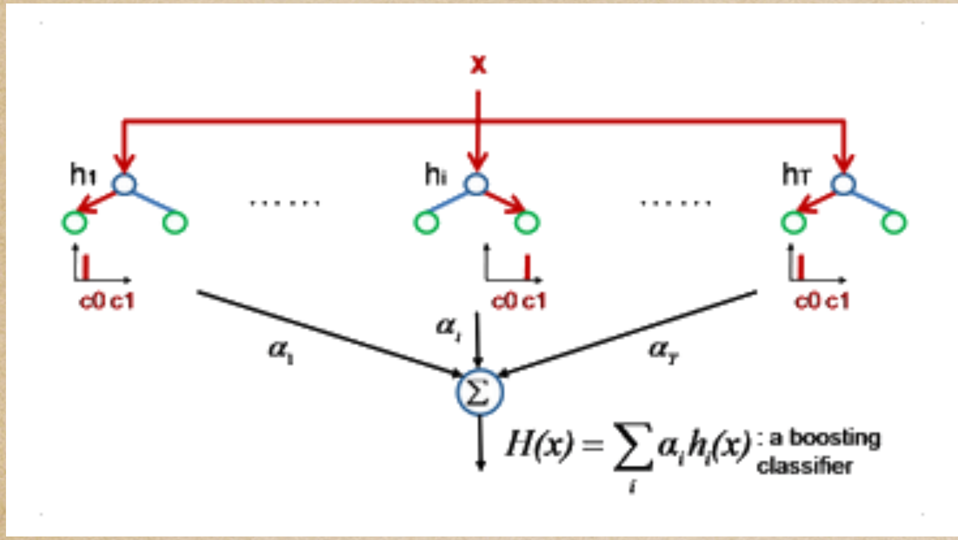
- The downstream tracking algorithm contains following steps (listed only the most important ones from the ML point of view):
    - The algorithm is seeded by tracks reconstructed in T stations.
    - Find matching TT hits
    - Accept downstream tracks candidates



T-Seed classifier

- Downstream tracking contains two classifiers.
- The first is designed to reject as much fake T-Seeds as possible
    - Tracks that cannot be reconstructed by downstream algorithm , due to material interaction, etc.
- The classifier constrains:
    - Keep the efficiency and purity of the selected signal tracks as high as possible
    - CPU performance
- We trained a number of models including linear models, Deep Neural Networks and BDT.
- The training procedure contains model selection, hyper parameters tuning, feature engineering and overfitting checking
- The best model according to the area under ROC curve score is Boosted Decision Trees.

Bonsai Boosted Decision Trees

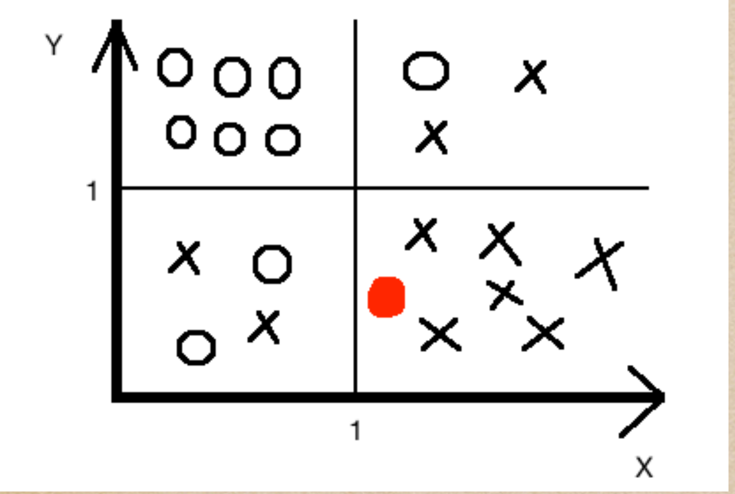


x	y	z	prediction
1.25	2000	3.5	0.88

- Using continuous input features to calculate classifier response takes too much CPU time.
- We had to figure out how to speed-up the calculations.
- The idea is pretty simple. We discretise input features space and for each of bin calculate classifier response.
- Instead of calculating response for each tree we just need to take one number from the lookup table.
- The bBDT evolution time is O(1)!
- The table has very complex structure, no way to fit approximation function.
- This idea comes from previous study on LHCb HLT [doi:10.1088/1748-0221/8/02/P02013]

Bonsai Boosted Decision Trees

How does it work?

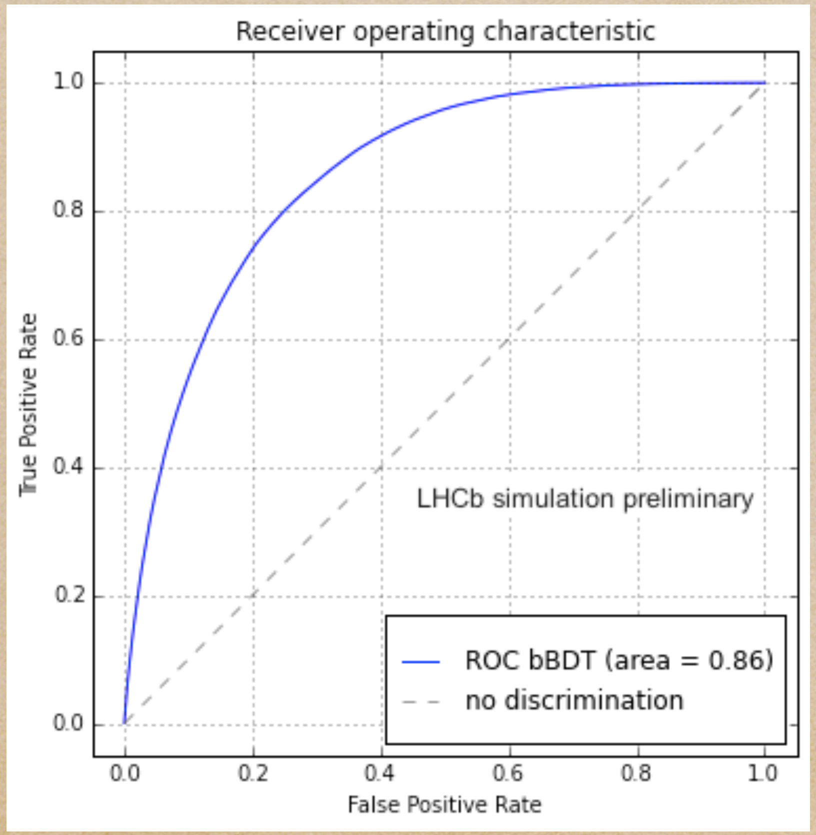


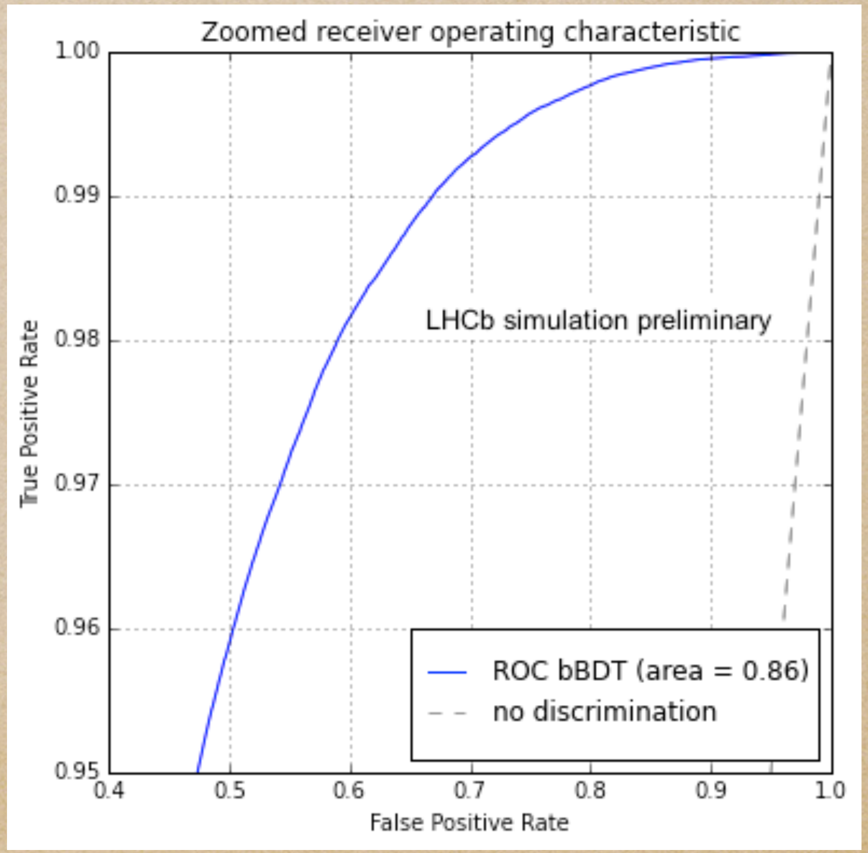
x	y	prediction
0	0	0.5
0	1	0
1	0	1
1	1	0.5

- We don't want to generate very long and complex if-else based function
- Instead we divide feature space into bins
- for each of these bins we calculate classifier response
- To evaluate classifier response we need to find bin indices and take corresponding number from the lookup table

T-Seed classifier performance

- The current version of classifier was trained using 2M track seeds from simulated samples containing B -> J/ψ Ks signal decay.
- The classifier scores ~ 0.86 (ROC auc).
- It corresponds to the rejection of about 40% of fake T-Seeds
- Above results obtained for validation set that constituted 20% of the input sample





Conclusions

- With help of two Deep Neural Networks LHCb reduced its its rate of fake long track by about 40%
- This was obtained without any negative influence on tracking efficiency
- The study on Downstream tracking algorithm also shows promising results
- Currently only the first classifier is implemented as Bonsai Boosted Decision Trees and studies on the second classifier are ongoing

Thank you for attention!