PV-finder

S. Akar[1], E. Kauffman[2], R. Garg[3], M. Peters[1], H.F. Schreiner [2], M.D. Sokoloff[1], W. Tepe[1], L. Tompkins[3]

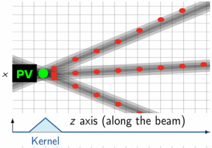
[1]University of Cincinnati, [2]Princeton University, [3]Stanford University

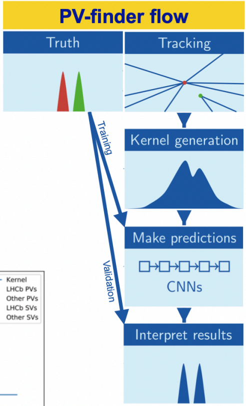
ML4Jets2022, November 4, 2022

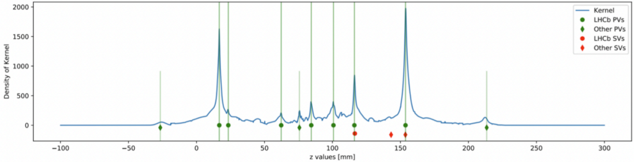
PV-finder History – I

- **Previous PV-finder iterations:**
  - **Using KDEs** (Kernel Density Estimators) **to reduce sparse 3D data (tracks parameters ; 41M pixels) to feature-rich 1D data – kernel densities in z**

$$\mathcal{K}(x,y,z) = \frac{\sum_{\text{tracks}} \mathcal{G}(\text{IP}(x,y)|z)^2}{\sum_{\text{tracks}} \mathcal{G}(\text{IP}(x,y)|z)} - \sum_{\text{tracks}} \mathcal{G}(\text{IP}(x,y)|z)$$





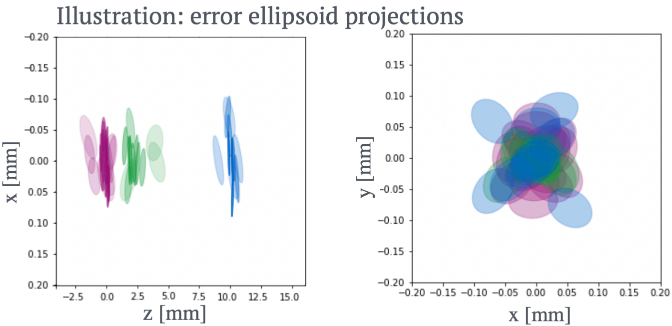


KDE distributions exhibit peaking structures near PV positions

Hand-written KDE computations expensive!

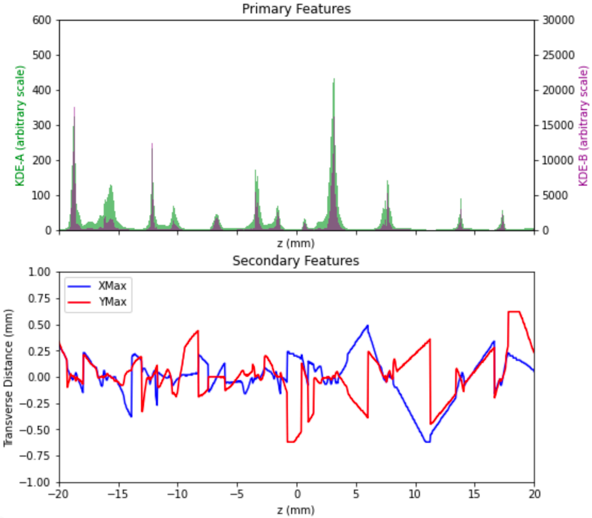
PV-finder History – II

- **Input features updated:**
  - Replaced **input tracks information** from **IP** (impact parameter) to **error ellipsoid at point of closes approach (POCA)** to beamline:



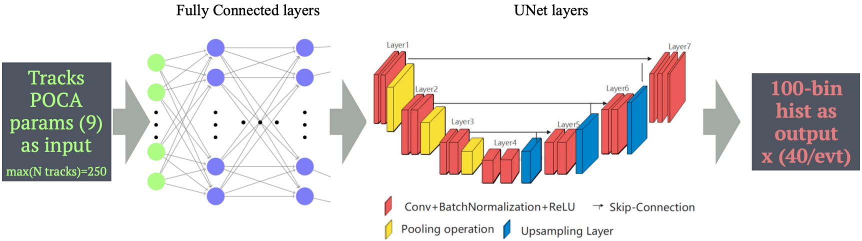
Each track represented from ellipsoid with 9 parameters
defining central position (3 pars.) and volume/uncertainty (6 pars.)

Cartoon KDEs for a GPD



end-to-end DNN, train using 40 $\times$ 10 mm intervals (LHCb simulation)

- **From tracks to PVs:**
  - **<u>Current model</u>**: hybrid DNN

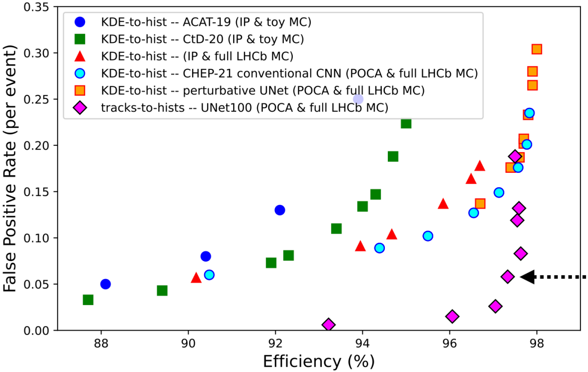


6 Fully Connected layers building (8 x 100) output channels x (40/evt)

UNet layers* summing the 8 contributions per bin to construct final predicted histogram

* current UNet model using 3 layers

- **From tracks to PVs:** **performance evaluation**



Competitive performances of a pure DNN tracks-to-PVs for the first time

Eff. ~ 97%
FP ~ 1%

Specific model used for the next slides detailed performance studies

[ACAT19 - J.Phys.Conf.Ser. 1525 (2020) 1, 012079] [CtD20 - arXiv:2007.01023] [CHEP21 - arXiv:2103.04962]

False positive rate can be reduced by choosing another asymmetry parameter:

Moving Forward

LHCb:

- instantiate existing `tracks-to-hists` inference engine inside `Allen`, the GPU-resident first level trigger, as a proof-of-principle; we hope to use `tensor cores` rather than `CUDA cores`;
- iterate `tracks-to-hists` architecture to improve performance (efficiency vs. false positive rate on one hand, memory footprint and number of calculations/throughput on the other);
- investigate use of "quantization" (`fp16` arithmetic rather than `fp32`); preliminary studies using "toy" MC rather than full LHCb MC indicate that the "same" architecture can achieve the same FP rate with a drop in efficiency of a small fraction of a percent.

ATLAS:

- current status: `kde-to-hists` model implemented for ATLAS data;
  - vertex resolution exceeds that of the default algorithm;
  - the efficiency and false positive rates are comparable to default algorithm;
  - hope first validated results will be public soon.
- would like to optimize architecture for ATLAS;
- plan to try to implement in ACTS (perhaps with GPU implementation).

This work was supported, in part, by the U.S. National Science Foundation under Cooperative Agreement OAC-1836650. All of the machine learning training described here was done in `PyTorch` using `nvidia` GPUs.

Back-up Material

see following pages

Prior Work

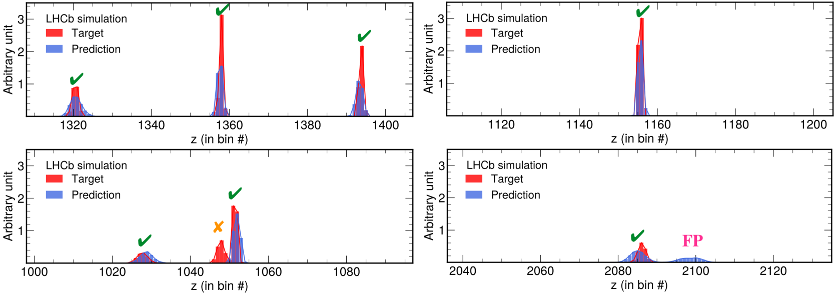
We are indebted to all those who contributed to the of earlier versions of `PV-finder` and the software infrastructure we are using, including Gowtham Atluri, Thomas Beottcher, Sarah Carl, Rui Fang, Marian Stahl, Constantin Weisser, and Michael Williams. In addition, we have used simulated data prepared by LHCb's Real Time Analysis team.

Earlier results have been reported in the following publications:

Rui Fang et al 2020 J. Phys.: Conf. Ser. 1525 012079, ACAT-2019.

S. Akar et al., arXiv:2007.01023, CtD-2020.

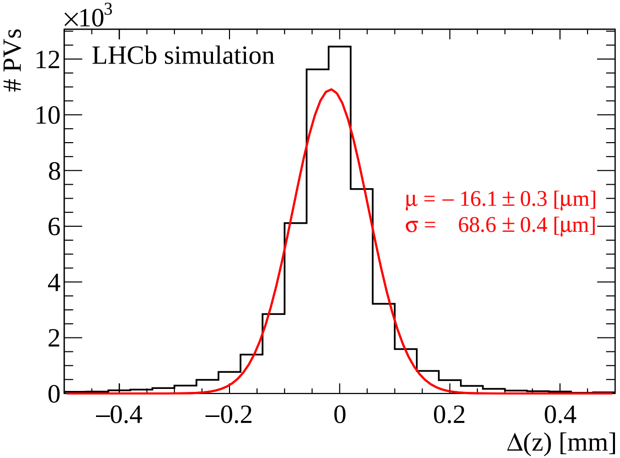
S. Akar et al., EPJ Web of Conferences 251, 04012 (2021), CHEP-2021.



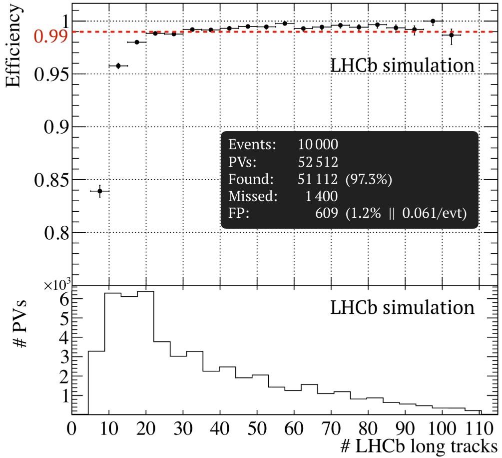
LHCb simulation
Target
Prediction
Arbitrary unit
z (in bin #)
1320
1340
1360
1380
1400
1120
1140
1160
1180
1200
1000
1020
1040
1060
1080
2040
2060
2080
2100
2120
FP

Observed Resolution & Bias

LHCb



Efficiency vs nTracks
LHCb



PV-finder Evolution

PV-finder has evolved over several years. Some of the changes since its first presentation at ACAT-2019 include:

- originally, all target histograms (labels) had the same width, height, and area; now, higher multiplicity PVs have smaller width, greater height and area target histograms;
- originally, a single KDE was calculated from tracks' slopes,intercepts, and their uncertainties; now, two KDEs are used in the KDE-to-hists DNNs and these use POCA-ellipsoid parameters.
- originally, the AllCNN KDE-to-hists DNNs had "flat" architectures; now, the UNet architectures is preferred;
- the original attempt to build a tracks-to-hists DNN, described at CHEP-2021, was trained using all tracks, building target histograms for all 4000 bins at once covering 400 mm longitudinally; now, a similar DNN is trained in intervals of 100 bins covering 10 mm longitudinally, and the results are stitched together.

Training the `tracks-to-hists` Model

The `tracks-to-hists` DNN takes track POCA-ellipsoids as its input features and produces target histograms from which candidate PV positions and resolutions are deduced. The first part of the DNN consists of fully connected layers. The second part is a convolutional neural network (CNN) using a UNet-like architecture. The training builds on domain "expertise" – it repeats the logic of its construction.

- train a fully-connected network (FCN) to predict a single, 100-bin KDE;
- freeze the weights and biases in the first 5 layers of the FCN; replace the 6th layer with 8×100-bin channels; use these 8 channels as the input features of a UNet-like CNN; train to predict the KDE;
- freeze the weights and biases in 6-layer FCN and train the UNet parameters to predict the 100-bin target histograms;
- float all the weights and biases in the FCN and the UNet CNN to predict the target histograms.