Machine Learning for NICA SPD Aerogel Reconstruction

F.Shipilov (1,2), A.Barnyakov (3,4), A.Ivanov (5), F.Ratnikov (1)

(1) HSE, (2) Skoltech, (3) Budker Institute of Nuclear Physics, (4) Novosibirsk State Technical University (5) JINR

fshipilov@hse.ru

LAMBDA • HSE

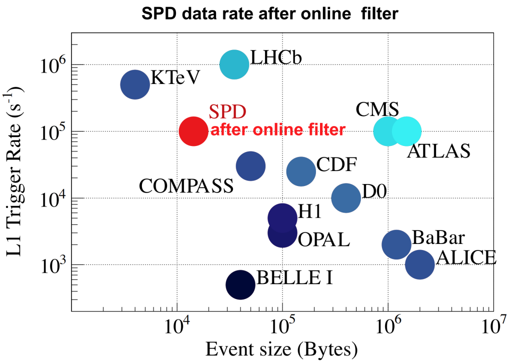
June 21, 2024

Introduction

- The experimental high-energy physics (HEP) objectives require searching for rare signals in background dominated environments
- Machine learning techniques can extract high-level representations from the input data and model complex relations
- This report concerns a case of machine learning application for aerogel reconstruction in the NICA SPD experiment

NICA SPD

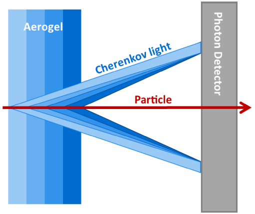
- A universal detector proposed by the SPD collaboration at NICA
- Study the Drell–Yan (DY) processes, J/Ψ production processes, elastic reactions, spin effects in one and two hadron production processes, polarization effects in heavy ion collisions, and more
- The SPD is a medium energy experiment, offering unique possibilities of beam operation and bridging the gap between the low-energy measurements, e.g. ANKE-COSY and the high-energy measurements, such as RHIC
- High luminosity up to 10^{32} cm^{-2}s^{-1} and free-flowing (triggerless) running mode

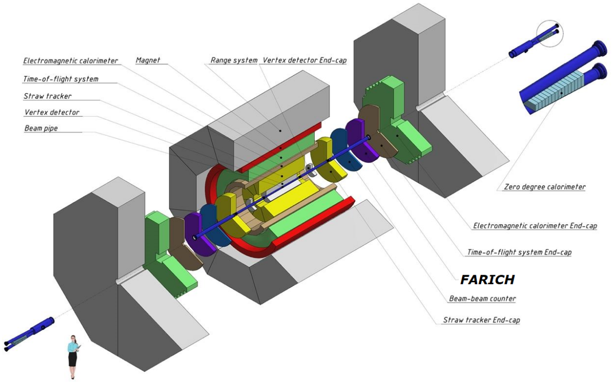


FARICH

Compact particle ID detector

- trade off between resolution and number of Cherenkov photons
- improve resolution with discrete focusing capability





https://nica.jinr.ru/projects/spd.php

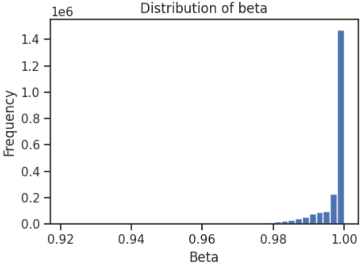
Dataset

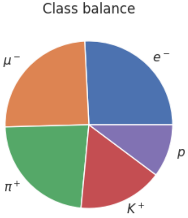
Provided input:

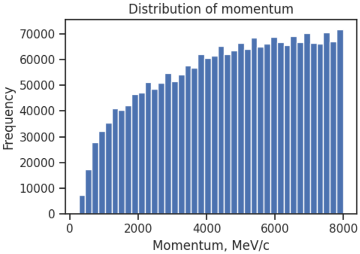
- Track parameters: x_p, y_p, z_p – coordinates of the particle when entering aerogel detector, θ_p, ϕ_p – polar angle and azimuth of the direction of travel;
- Photon hits: x_c, y_c, z_c – coordinates of triggered pixels in the photosensitive matrix.

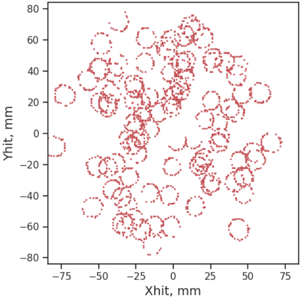
Expected output:

- PID statistics
- Particle type



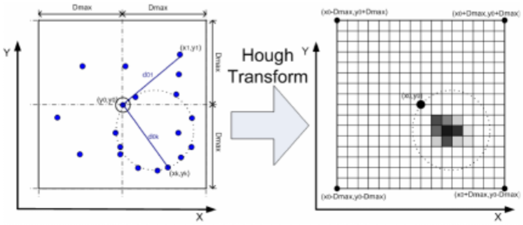






Baseline: Hough Transform

- Based on the RICH reconstruction from the CBM experiment at FAIR;
- Utilizes Hough transform for ellipses, more precisely the Taubin method;
- Outputs estimated parameters of the ellipse, such as a, b – semi-axes, x, y – center coordinates, and ϕ – angle of rotation;
- Does not account for refraction, does not use the information from the straw tracker, such as θ_p and ϕ_p.



Our Methods: Refraction Correction

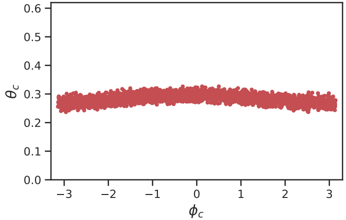
First order approximation

$$\alpha \approx \beta - (n-1)\tan\beta \tag{1}$$

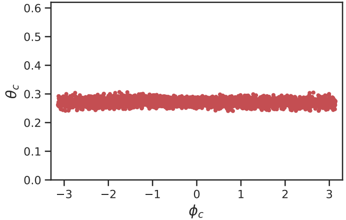
Or fixed-point iteration

$$\sin\alpha_{k+1} = \left(n\sqrt{1+\left(\frac{d}{r-h\tan\alpha_k}\right)^2}\right)^{-1} \tag{2}$$

θ_c before correction



θ_c after correction



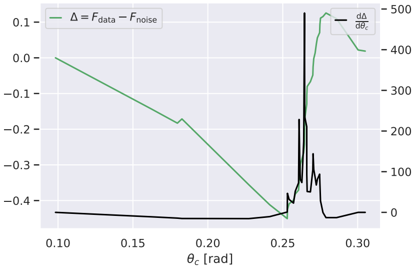
Algorithmic methods

Median:

- Compute θ_c distribution
- Take a median value $\hat{\theta}_c$

MLE:

- Compute θ_c distribution
- Drop unphysical velocities $\beta = v/c \geqslant 1$
- Construct eCDF F_{data}
- Take a numerical derivative and find its peak $\hat{\theta}_c = \arccos(1/n\hat{\beta})$



MLE example

Machine Learning Models

- Algorithmic models output $\hat{\theta}_c$ that can be converted to the rest mass of the particle using momentum p from the straw tracker
- The underlying task is regression
- The objective of RICH detector is to separate different particle types, in particular π/K separation
- ML end-to-end models are able to skip the intermediate step of computing $\hat{\theta}_c$
- Moreover, the regression task is not well suited for an ML model because of the significant value imbalance

We implement a neural classifier that incorporates both **hit coordinates** x_c, y_c and **momentum** p.

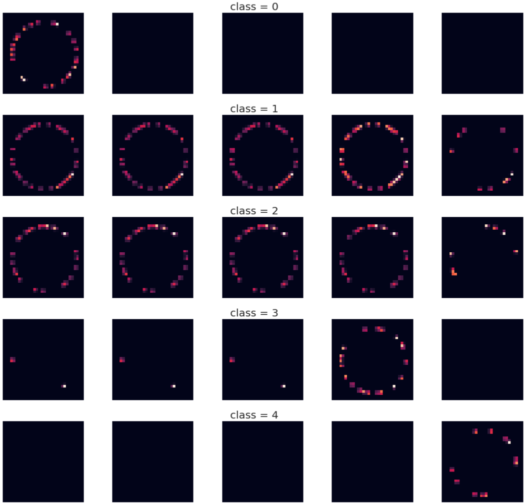
Data Transform

- Option 1: provide the value of p as an extra input channel / pathway
- Option 2: normalize the input using p

The non-linear dependence on p can be disentangled:

$$\cos\theta_c = \frac{1}{n}\sqrt{m_0^2/p^2 + 1} \tag{3}$$

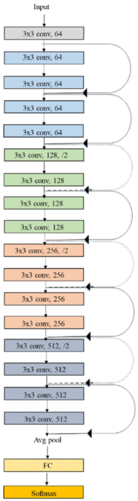
- Use (3) with N mass probes for each of the particle types in the data to transform θ_c into N mass probes
- Convert the probes to 2d images and stack into a tensor



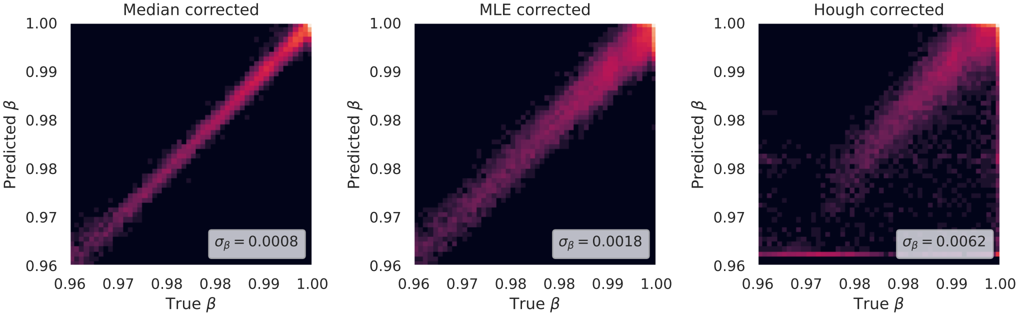
Input tensor example

Procedures: Learning

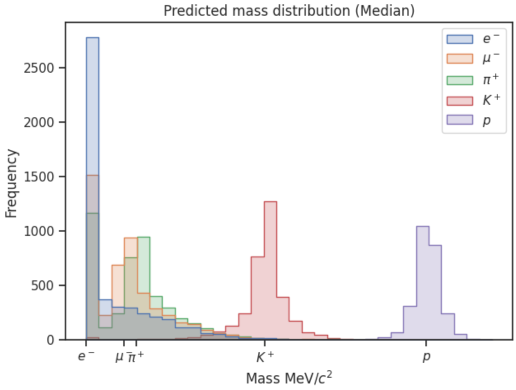
- ResNet-18 CNN architecture
  - Changed input conv, max pooling and classifier head to accommodate input and output data formats
- Input format for NN classifier: $5 \times 32 \times 32$ tensor
- Trained for 200000 samples with Adam optimizer, cosine annealing scheduler, a learning rate of $5 \cdot 10^{-4}$ and a batch size of 128
- Another model trained on π/K specifically for 400000 samples



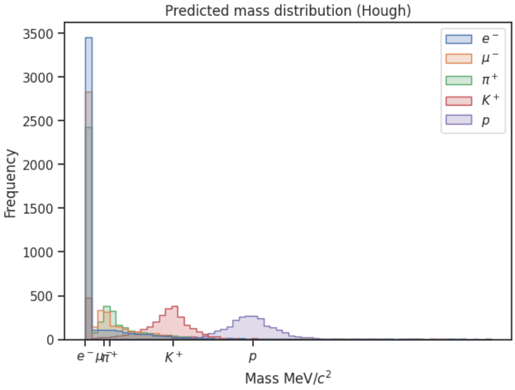
Results


Median corrected
MLE corrected
Hough corrected
Predicted β
True β
1.00
0.99
0.98
0.98
0.97
0.96
0.96 0.97 0.98 0.98 0.99 1.00
$\sigma_\beta = 0.0008$
$\sigma_\beta = 0.0018$
$\sigma_\beta = 0.0062$

Results



Predicted mass distribution (Median)
Frequency
0
500
1000
1500
2000
2500
e^-
μ^-
π^+
K^+
p
Mass MeV/c^2

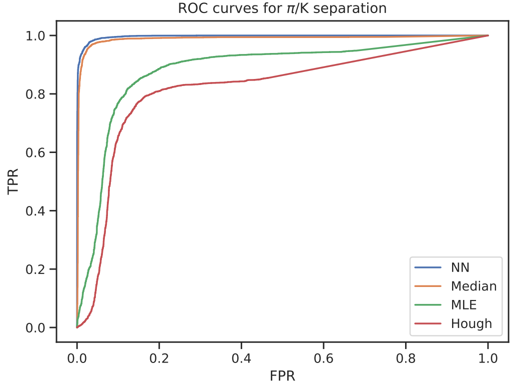


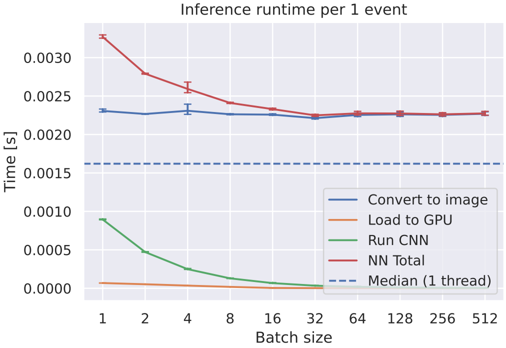
Predicted mass distribution (Hough)
Frequency
0
500
1000
1500
2000
2500
3000
3500
e^-
μ^-
π^+
K^+
p
Mass MeV/c^2

Results

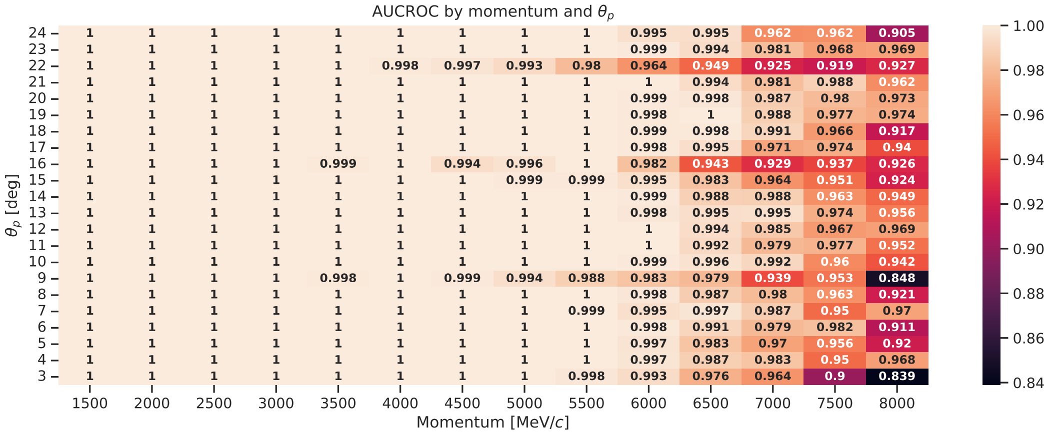
Method	$\mathbb{P}(\text{true } \pi, \text{ pred } K)$	$\mathbb{P}(\text{true } K, \text{ pred } \pi)$	π, K AUROC	Total accuracy	σ_β
NN	$\mathbf{0.016 \pm 0.005}$	$\mathbf{0.010 \pm 0.005}$	**0.997**	**0.65**	**N/A**
Median	0.06 ± 0.02	0.02 ± 0.02	0.989	0.64	0.0008
MLE	0.20 ± 0.02	0.12 ± 0.02	0.883	0.52	0.0018
Hough	0.13 ± 0.02	0.26 ± 0.02	0.817	0.49	0.0062

Results


ROC curves for π/K separation
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0
NN
Median
MLE
Hough


Inference runtime per 1 event
Time [s]
Batch size
0.0000
0.0005
0.0010
0.0015
0.0020
0.0025
0.0030
1
2
4
8
16
32
64
128
256
512
Convert to image
Load to GPU
Run CNN
NN Total
Median (1 thread)

Results



Detailed NN performance

Conclusion and Future Plans

- NN model achieved good performance
- The real time use of ResNet-18 in the SPD DAQ is not possible because of the unavailability of GPUs
- Conversion to tensor is too slow regardless

Future steps

- Architecture optimization to run on CPU
  - knowledge distillation, pruning and quantization
- More efficient conversion or a different input format
- Multi-ring PID, automatic track-ring matching using ML