Machine-learning activities in the ALICE experiment

Fabio Catalano[1] on behalf of the ALICE Collaboration

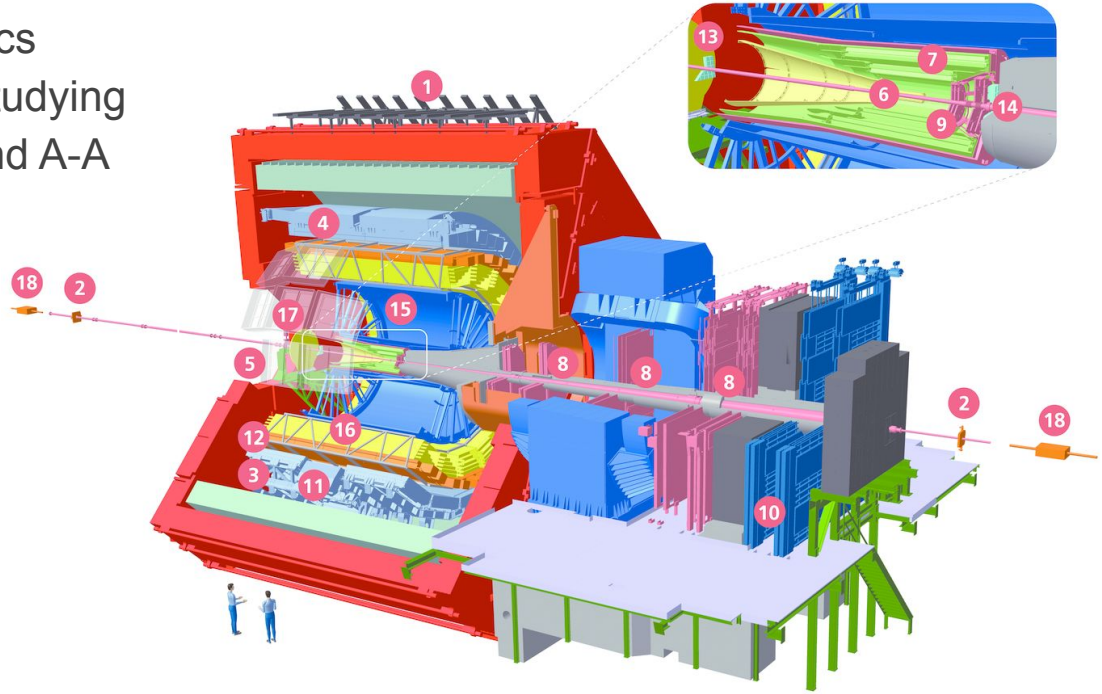
ALPACA ECT* Workshop
21/11/2023

[1]CERN, fabio.catalano@cern.ch

The ALICE experiment

- ALICE experiment designed to study the quark-gluon plasma formed in ultra-relativistic heavy-ion collisions at the CERN LHC
  - quark-gluon plasma ⟶ exotic state of matter in which quark and gluons are deconfined

- Rich physics program studying pp, p-A, and A-A collisions

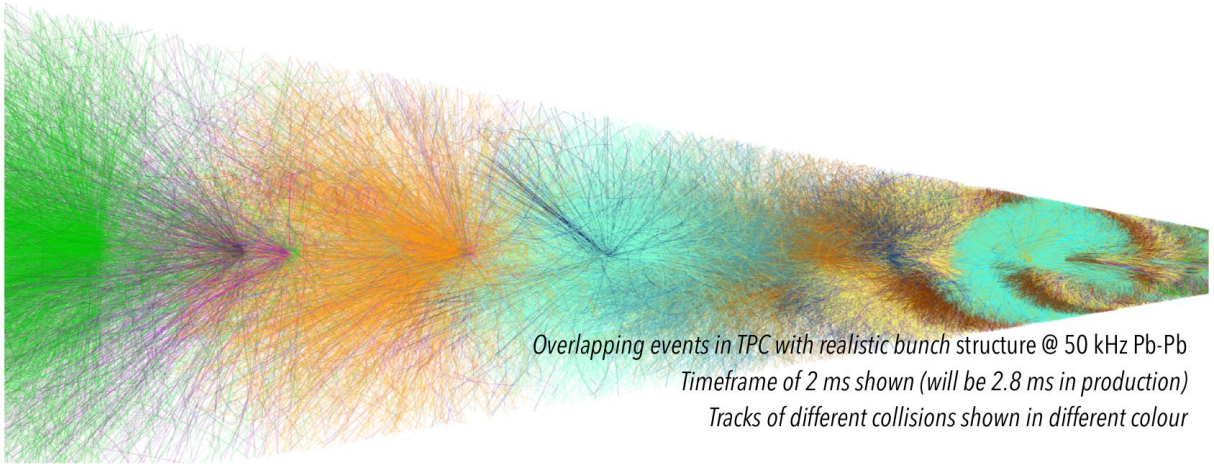


1. **ACORDE** | ALICE Cosmic Rays Detector
2. **AD** | ALICE Diffractive Detector
3. **DCal** | Di-jet Calorimeter
4. **EMCal** | Electromagnetic Calorimeter
5. **HMPID** | High Momentum Particle Identification Detector
6. **ITS-IB** | Inner Tracking System - Inner Barrel
7. **ITS-OB** | Inner Tracking System - Outer Barrel
8. **MCH** | Muon Tracking Chambers
9. **MFT** | Muon Forward Tracker
10. **MID** | Muon Identifier
11. **PHOS / CPV** | Photon Spectrometer
12. **TOF** | Time Of Flight
13. **T0+A** | Tzero + A
14. **T0+C** | Tzero + C
15. **TPC** | Time Projection Chamber
16. **TRD** | Transition Radiation Detector
17. **V0+** | Vzero + Detector
18. **ZDC** | Zero Degree Calorimeter

ALICE upgrades for Run 3

- In preparation for the LHC Run 3 → substantial detector upgrades
  - new ITS, MFT, FIT detectors and new GEM readout chambers for the TPC
  - enable operations at much higher interaction rate than in Run 2
  - improved vertexing and tracking resolution at low transverse momentum

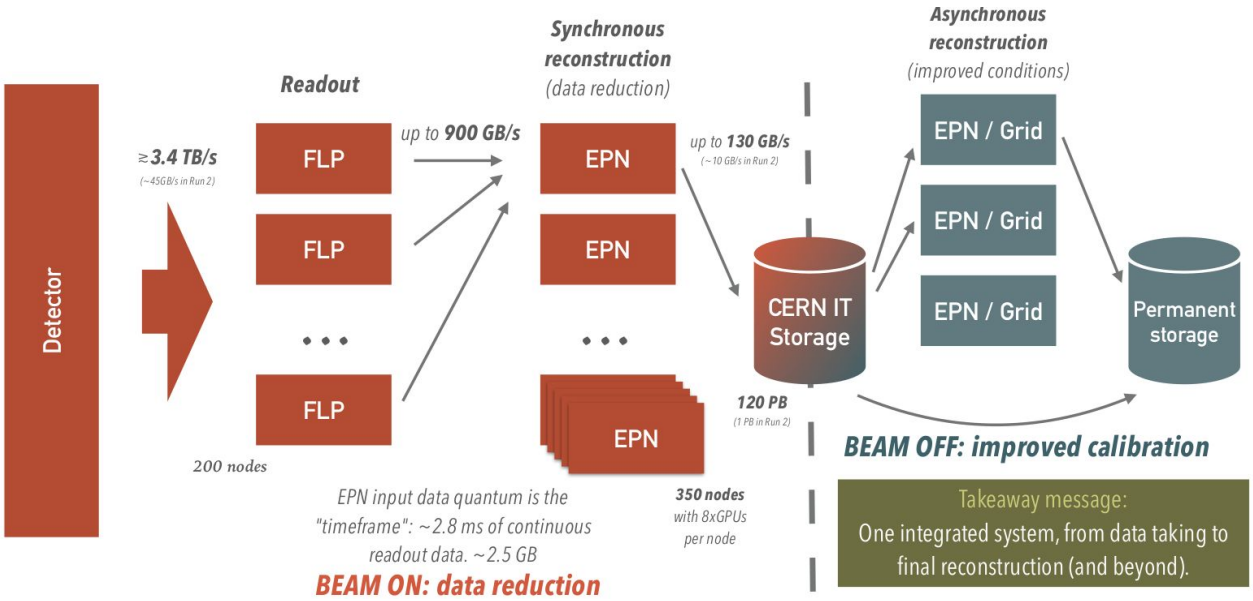
- Reconstruct data in continuous readout, recording time frames instead of events
  - ~500 kHz interactions at pp data taking
  - up to 50 kHz during the Pb-Pb run
  - about x50 increase in statistics for physics observables



Overlapping events in TPC with realistic bunch structure @ 50 kHz Pb-Pb
Timeframe of 2 ms shown (will be 2.8 ms in production)
Tracks of different collisions shown in different colour

Data flow and software

- Completely new software and computing framework (O^2) for synchronous and asynchronous reconstruction (w.r.t. data taking)
  - developed to cope with the increased data volume

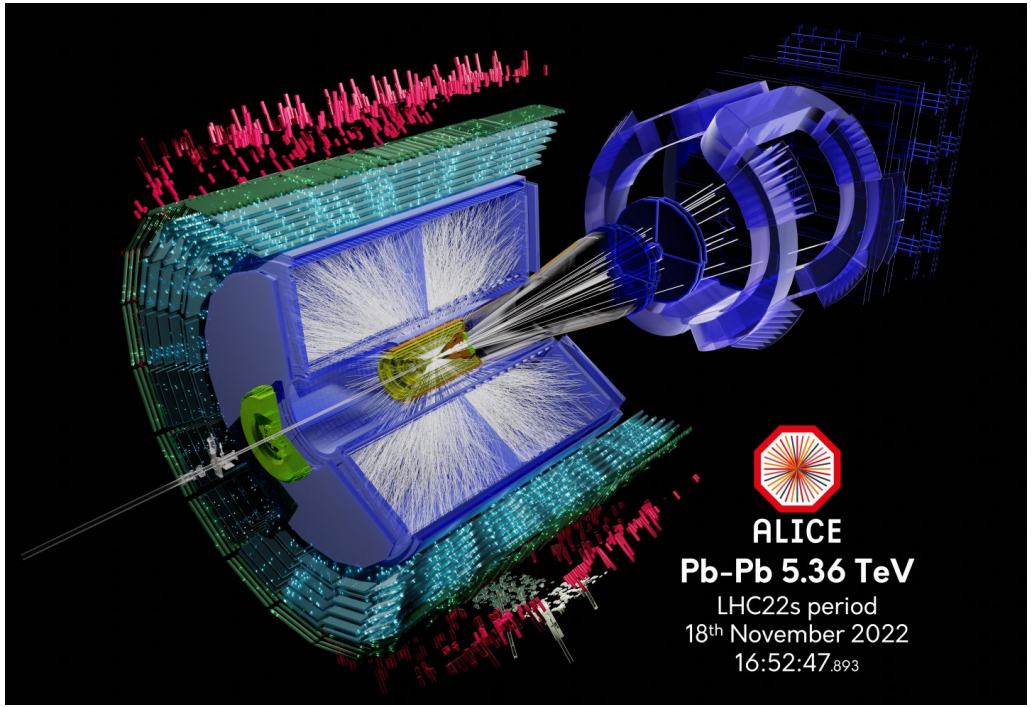


- Analysis software also overhauled to increase throughput
  - data model now based on Apache Arrow tables

From G. Eulisse talk at CHEP 2023

Machine learning in ALICE

- Fundamental tool to
  - maximise the physics potential of the measurements
  - cope with the huge amount of data produced at LHC
- ALICE ML activities involve
  - physics analyses
  - detector calibrations
  - data quality control (QC)
  - Monte Carlo simulations



Ongoing ML activities

Established

Signal-vs-background classification

- BDT and NN replacing “traditional” linear selections in data analysis

Jet p_T reconstruction

- correction for the background from the underlying event using shallow NN

Heavy-flavour hadron trigger

- BDT to trigger on displaced decay-vertex topologies

TPC response calibration

- ML to compute corrections of space charge distortions
- NN for energy-loss (dE/dx) calibration

General framework developments

- common tools and procedures

Particle identification (PID)

- exploit complex relationship between track properties and PID
  - NN to combine info from different detectors
  - PID with ITS2 using BDT regression

MFT-MCH track matching

- NN classification giving the score for a correct match

ML for quality control/assurance

- alert experts quickly and accurately about issues in data-taking

Fast simulation

- ZDC calorimeter simulation with GANs and VAEs

... not a comprehensive list!

Ongoing ML activities

Exploratory

Signal-vs-background classification

- BDT and NN replacing “traditional” linear selections in data analysis

Jet p_T reconstruction

- correction for the background from the underlying event using shallow NN

Heavy-flavour hadron trigger

- BDT to trigger on displaced decay-vertex topologies

TPC response calibration

- ML to compute corrections of space charge distortions
- NN for energy-loss (dE/dx) calibration

General framework developments

- common tools and procedures

Particle identification (PID)

- exploit complex relationship between track properties and PID
  - NN to combine info from different detectors
  - PID with ITS2 using BDT regression

MFT-MCH track matching

- NN classification giving the score for a correct match

ML for quality control/assurance

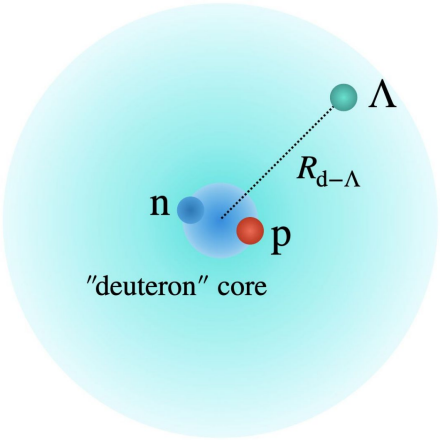
- alert experts quickly and accurately about issues in data-taking

Fast simulation

- ZDC calorimeter simulation with GANs and VAEs

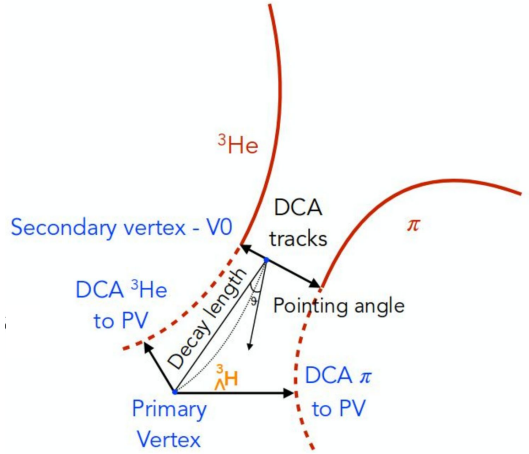
... not a comprehensive list!

ML to improve measurements – Hypertriton



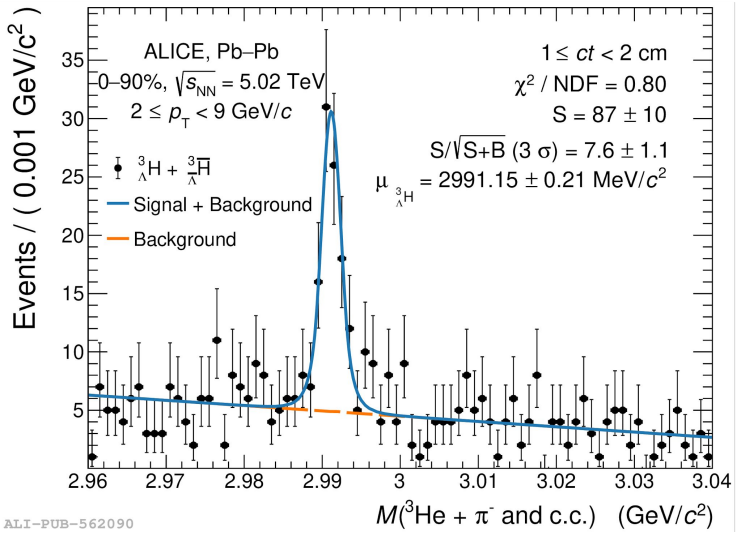
- ${}^{3}_{\Lambda}$H is the lightest known hypernucleus
  - bound state of a neutron, a proton and a Λ
  - could be approximated as a deuteron-Λ bound state with an expected radius of ~10 fm
- Unique probe to study the Λ-nucleus interaction, with strong implications for astro-nuclear physics
  - hyperons expected to be produced in neutron-star inner core

- ${}^{3}_{\Lambda}$H candidates built combining ^{3}He + π^- pairs (and charge conjugates)
  - matching of tracks coming from a common secondary vertex
  - huge combinatorial background

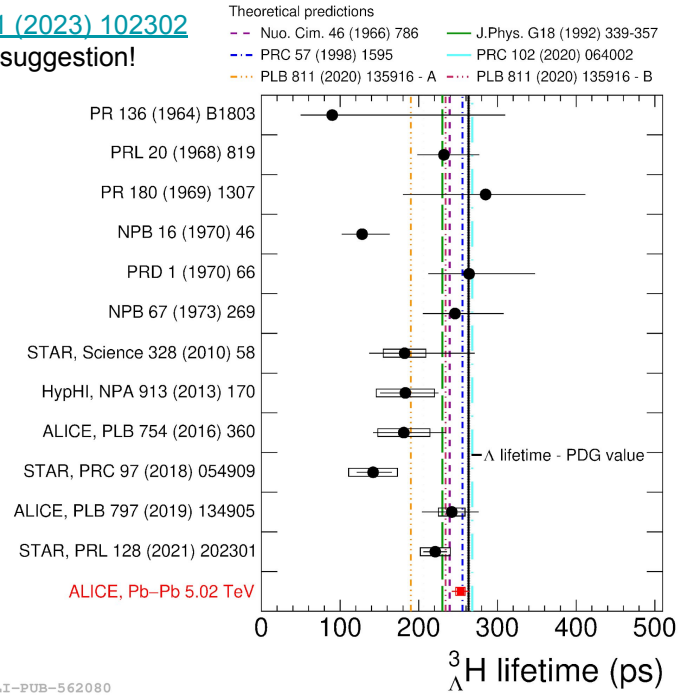


ML to improve measurements – Hypertriton

- XGBoost BDTs for binary classification (signal vs combinatorial background)
- Models trained employing high-level physical variables (decay length, PID, ...)



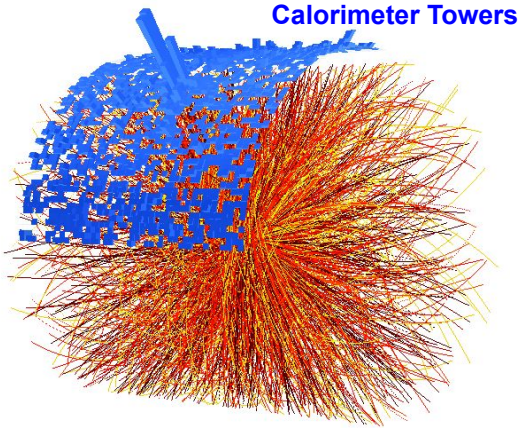
PRL 131 (2023) 102302
Editors' suggestion!



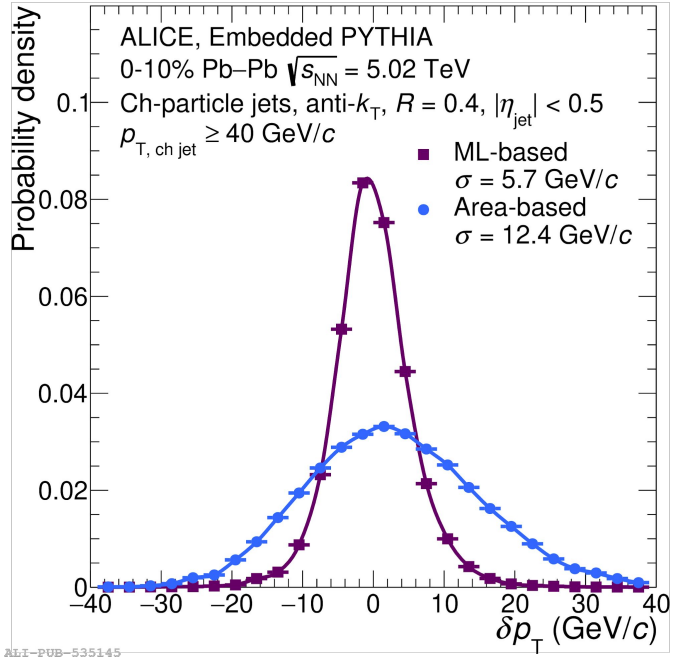
- Most precise measurements of the hypertriton lifetime and Λ separation energy to date

ML to improve measurements – Jet p_T reconstruction

- Reconstruction of inclusive jet p_T in heavy-ion collisions
  - difficult due to large fluctuating background from the underlying event



arXiv:2303.00592



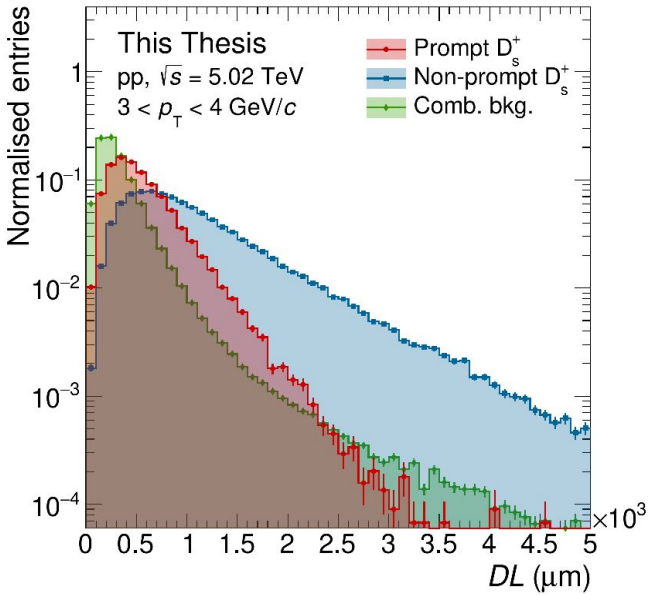
$\delta p_T = p_{T, rec} - p_{T, true}$

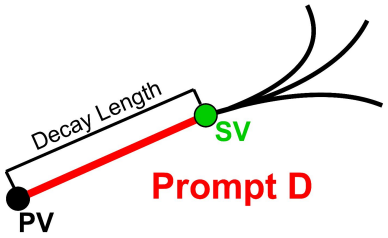
- Shallow NN from scikit-learn to correct the jet transverse momentum
  - jet and constituent (p_T of leading tracks) properties as input to the model
- Improved performance w.r.t. "standard" area based approach
  - narrower δp_T ⟶ reduced residual fluctuations

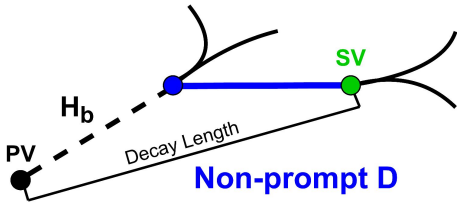
ML to enable new measurements – Non-prompt charm hadrons

- ML tools used to single out charm hadrons produced in beauty-hadron decays
  - indirect access to the beauty sector

CERN-THESIS-2022-046



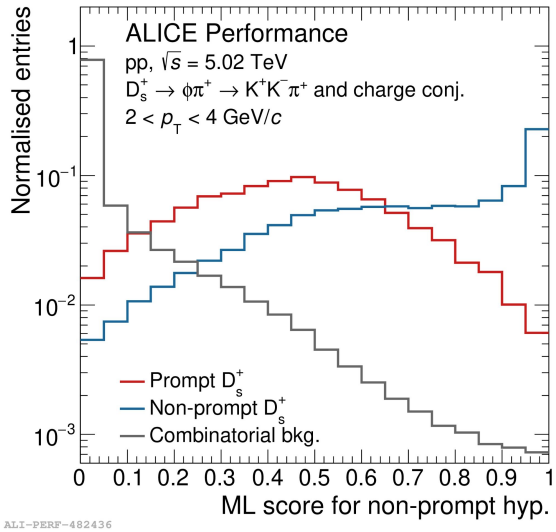




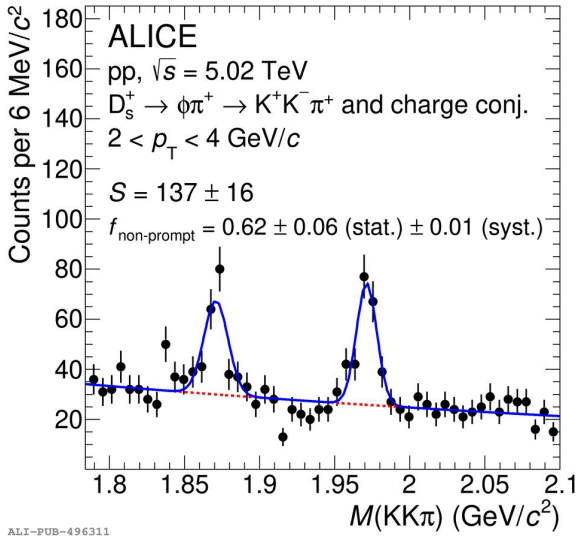
- XGBoost BDTs for multiclass classification, to disentangle
  - two kinds of signal (prompt and non-prompt charm hadrons)
  - combinatorial background
- Models trained using high-level physical variables (decay length, PID, ...)

ML to enable new measurements – Non-prompt charm hadrons

- BDT output related to the candidate probability of being a prompt charm hadron, a non-prompt charm hadron or combinatorial background



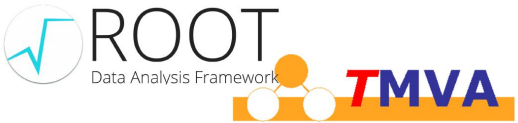
JHEP 05 (2021) 220



- Production of prompt and non-prompt D^0, D^+, D_s^+, and Λ_c^+ hadrons measured separately
  - non-prompt charm-hadron measurements not possible without ML

Software for ML

- ML applications in ALICE based either on
  - ROOT TMVA
    - early applications, now essentially abandoned
  - python software stack (scikit-learn, XGBoost, TensorFlow, PyTorch, …)



✅ Integrated out-of-the-box in ALICE analysis software

❌ Limited selection of ML models and tools

❌ Limited documentation

✅ Widely used outside HEP

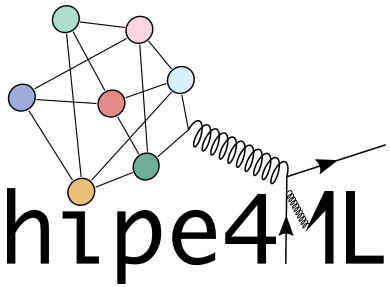
✅ Huge amount of ML models and techniques available

❌ Need interfaces with the ALICE C++ software (uproot, treelite, ONNXRuntime)

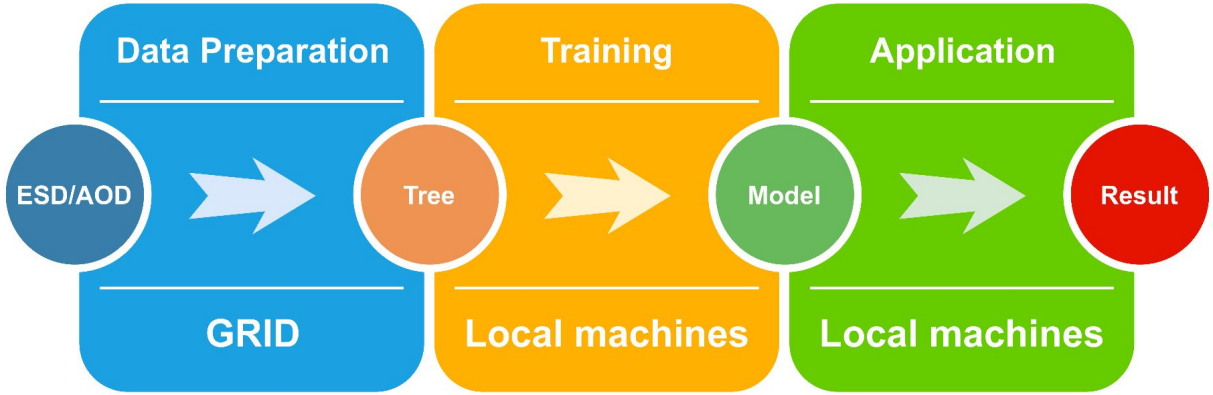
Software for ML

- ML applications in ALICE based either on
  - ROOT TMVA
    - early applications, now essentially abandoned
  - python software stack (scikit-learn, XGBoost, TensorFlow, PyTorch, …)

- Some common software tools have been developed by ALICE members
  - automatise workflows and/or ease typical steps of an analysis
  - useful to kickstart new analysers and to have consistent practices within the Collaboration
  - hipe4ml package (available on PyPI)
    - wrapper around popular ML libraries
    - used also outside the ALICE Collaboration



Typical analysis workflow – Local inference



Data preparation

- Information written from AO2D to ROOT TTree
- Full data and Monte Carlo samples downloaded locally

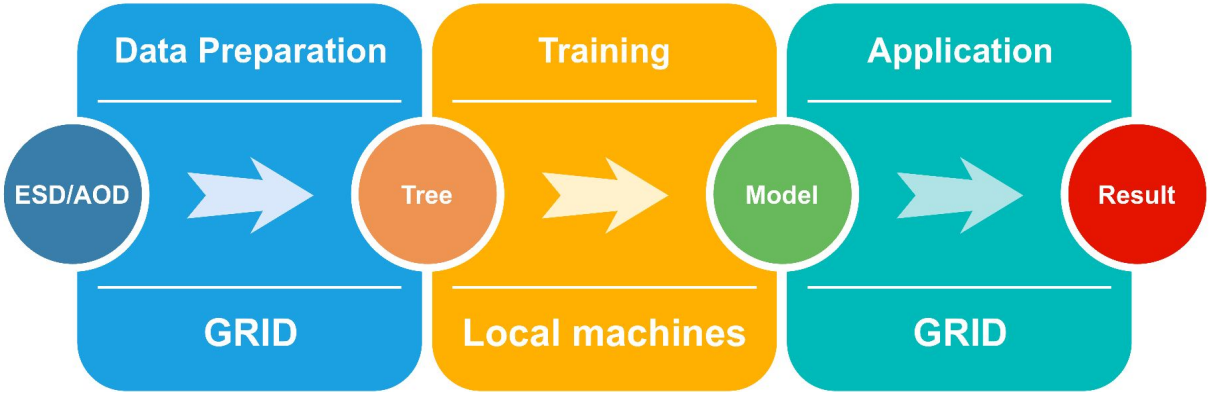
Training and optimisation

- Small fraction of real data and all MC simulations used to train/optimise the model
  - requires a few minutes/hours on a workstation for common use cases

Inference on full data sample

- A few minutes/hours depending on the use case
  - in some cases high-end machine needed to store and process the large amount of data

Typical analysis workflow – GRID inference



Data preparation

- Information written from AO2D to ROOT TTree
- Only data needed for training downloaded locally

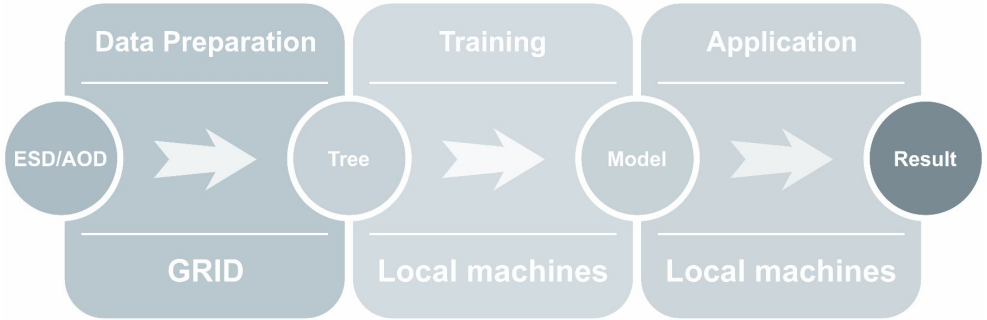
Training and optimisation

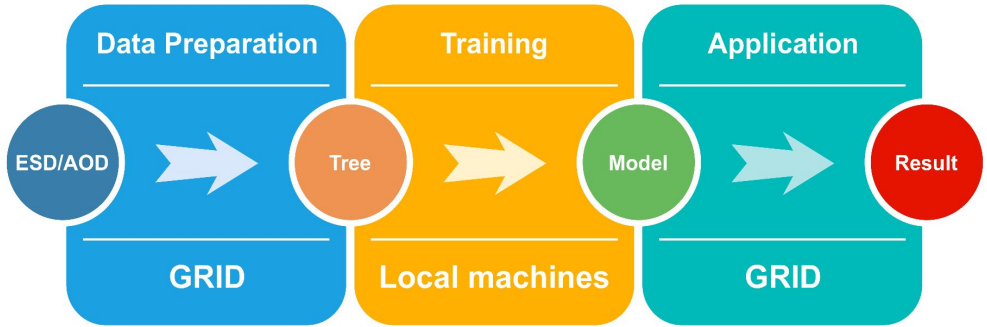
- Small fraction of real data and all MC simulations used to train/optimise the model
  - requires a few minutes/hours on a workstation for common use cases

Inference on full data sample

- A few days on the Worldwide LHC Computing Grid (WLCG)
  - usual time for a “train run” from the user point of view
  - the ML inference step is added to standard analysis tasks

Analysis workflows in Run 3

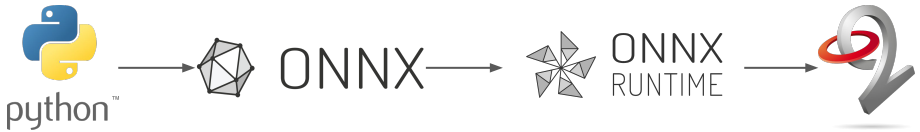




- ALICE is collecting a lot of data in Run 3
  - ~27 pb^{-1} of pp in 2022 and 2023
    - ~4 PB of raw data stored
  - ~1.5 nb^{-1} of Pb-Pb collected this year
    - ~43 PB of raw data stored
- ML model inference on full data samples challenging on local machines
  - even with server-grade machines
- Efficient way to perform ML inference on the GRID implemented. To support:
  - analyses
  - “core” tasks (trigger, calibration, particle identification, …)

Software for ML – Model inference

- Inference of ML models in ALICE Run 3 software implemented via ONNX+ONNXRuntime
  - positive experience so far



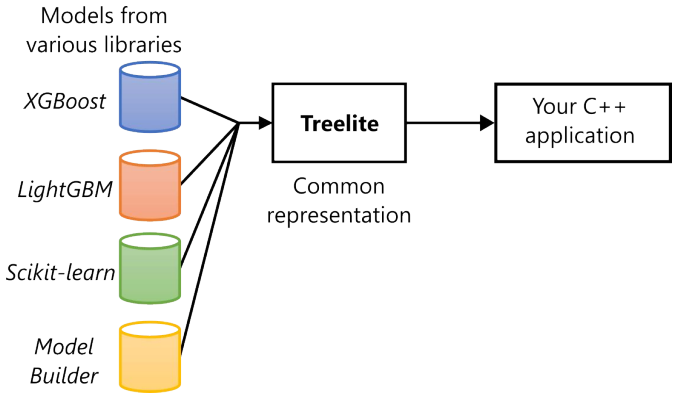
- Models trained with Python software exported to ONNX format
  - supports most ML models (BDT, NN, …) and libraries (XGBoost, PyTorch, TensorFlow, ...)
  - stable format ⟶ good for model preservation
  - industry standard

- Inference of models in ONNX format performed by ONNXRuntime library
  - integrated in ALICE software stack
  - C++ API available, some custom classes developed to simplify usage
  - mainly used on the GRID at the moment
  - ML models stored in database and retrieved at runtime

Software for ML – Model inference

- Under investigation
  - provide data from Apache Arrow tables (ALICE Run 3 data format) to ONNXRuntime efficiently and with flexibility
  - TMVA SOFIE as inference provider
    - experimental tool in ROOT to read and perform inference for ONNX models
    - pros: easy integration, possibly better support for ALICE data format through RDataFrame Arrow backend
    - cons: limited number of ONNX operators supported

- In ALICE Run 1 and Run 2 software, inference of ML models was enabled by
  - treelite
    - project of a XGBoost developer providing a C++ API
    - support for decision tree forests only (e.g. BDTs)
  - various custom classes in C++ developed by analysers



Particle identification using the TPC

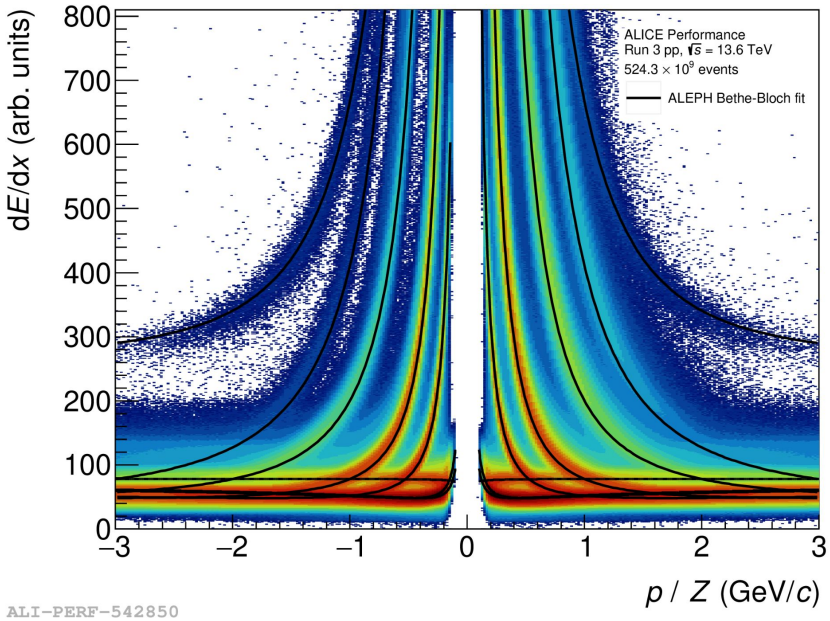
- The Time Projection Chamber (TPC) is one of the main detectors used for particle identification (PID) in ALICE
  - via measurement of the particle energy-loss per unit length (dE/dx) in the TPC gas

- Particle energy loss as a function of momentum described by Bethe-Bloch parameterization
  - parameters determined from a fit to data

$$\langle dE/dx \rangle_{ALEPH} = a_1 * (a_2 - \log(a_3 + (\beta\gamma)^{-a_5})/\beta^{a_4} - 1) * z^{f_z} * f_{MIP}$$

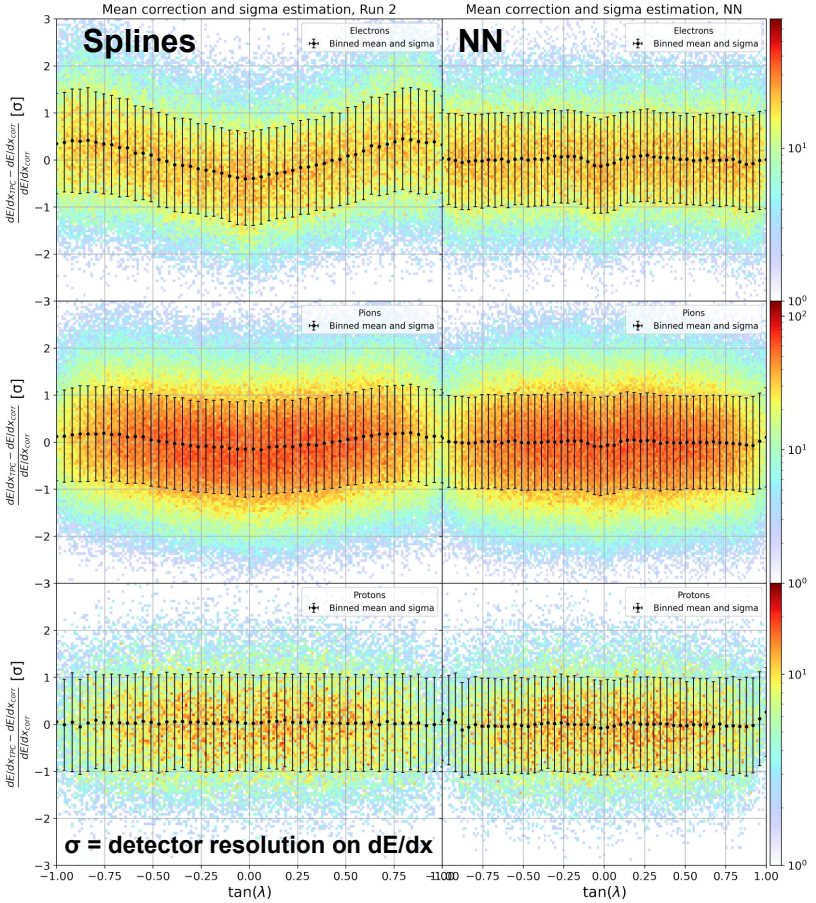
f_{MIP}: dE/dx value for minimum ionising particles

- Assignment of particle species for a track
  - by testing different mass hypotheses and comparing measured energy loss with the parametrization



TPC PID calibration with neural networks

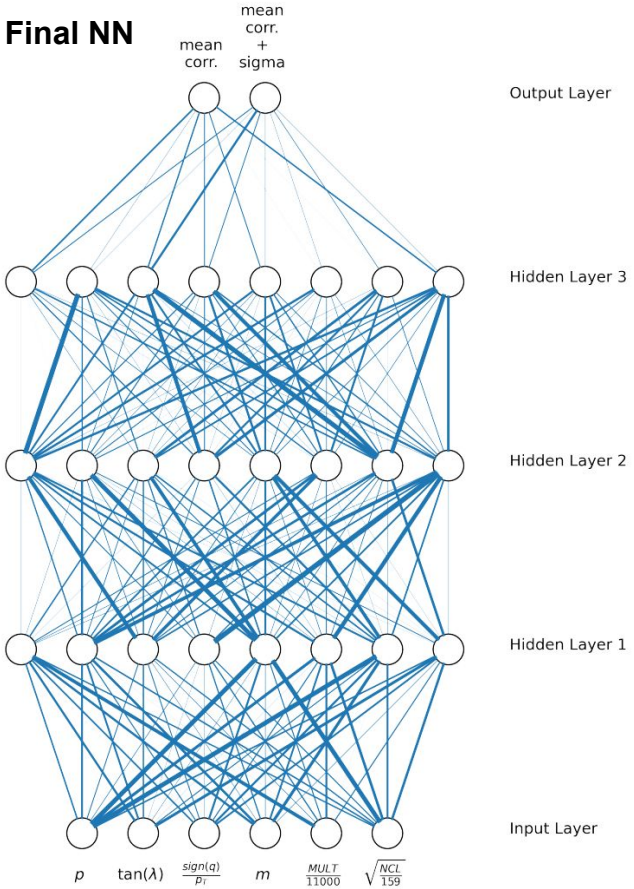
- NN corrections to the Bethe-Bloch parameterization of particle energy loss
  - track information as input (p, tan(λ), N_{CLS}, …)
  - n-dimensional (6D) corrections ⟶ correlations kept into account
  - only one iteration needed
- Replaced the “Spline corrections” used in Run 2
  - per-dimension corrections assuming factorisation
  - multiple iterations to produce
- Performance comparable or better than Splines on Run 2 data
- Fully data-driven NN corrections now available for all Run 3 pp data



Further details: [CERN-THESIS-2022-342]

TPC PID calibration with neural networks

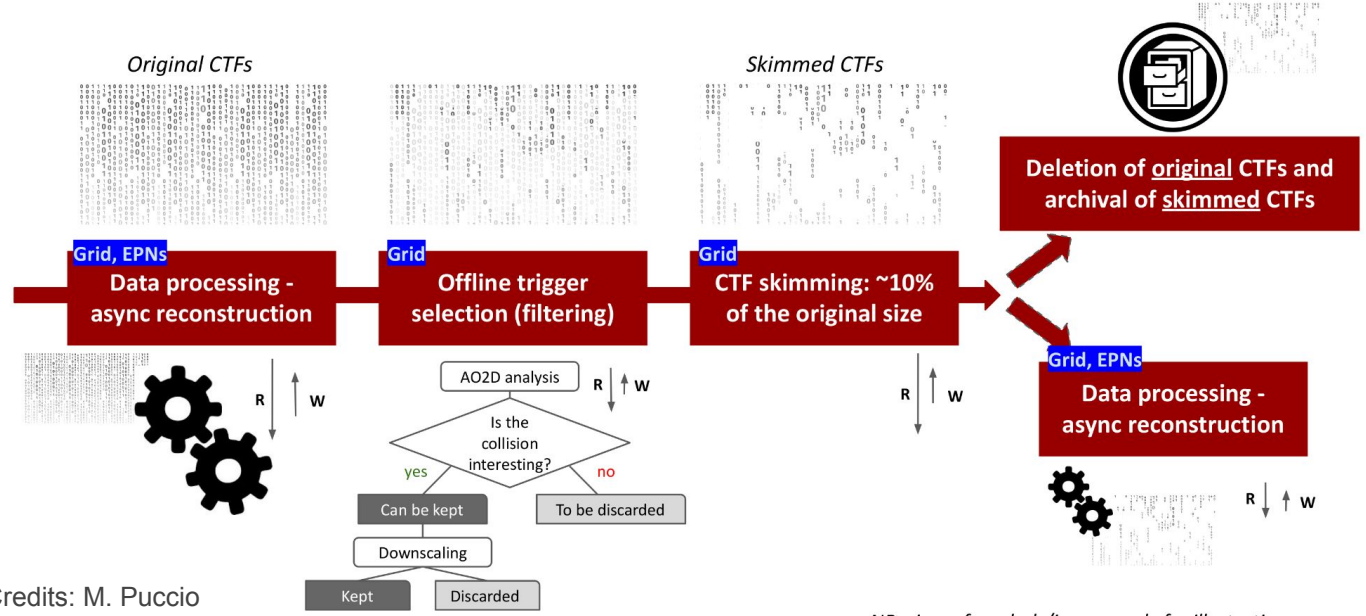
- Fully connected NNs performing a regression
  - PyTorch library used
  - final NN trained on the output of two larger models (12 nodes x 10 layers) for performance reasons at inference time
- Training performed for each data-taking period
  - starting from analysis-object data (AO2D)
  - on farm equipped with GPUs
  - ~7-8 hours of GPU time on Nvidia V100 or AMD MI100
- ~300 hours of training time per data-taking year
- Trained models uploaded to database and accessible for analyses on the GRID
  - model inference based on ONNXRuntime



Further details: [CERN-THESIS-2022-342]

Software trigger for high-energy pp program

- ALICE high-energy pp program aims to collect an integrated luminosity of ~200 pb^{-1} during LHC Run 3 (2022-2025) CERN-LHCC-2020-018
  - based on software trigger running after data reconstruction (similarly to a normal analysis)
  - interesting events selected ⟶ definition of time intervals around triggered collisions to be kept in reduced compressed time-frames (skimmed CTFs) and subsequently re-reconstructed

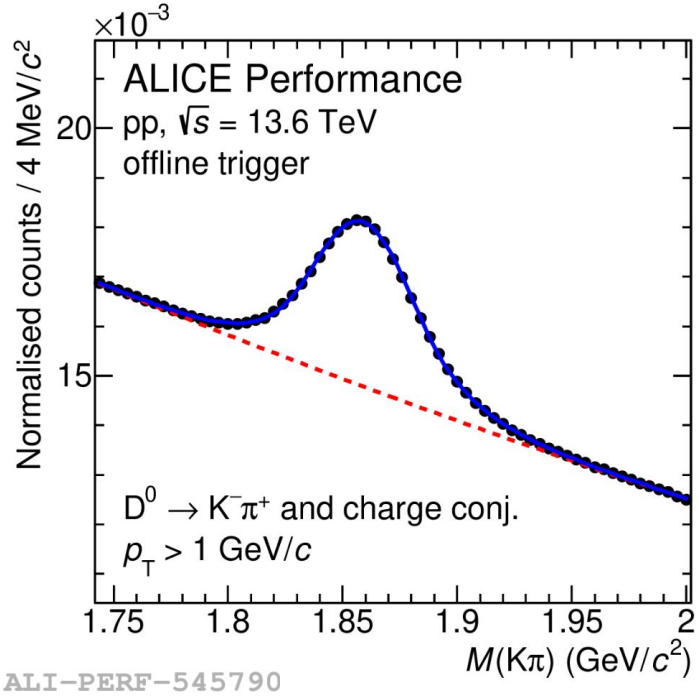


Credits: M. Puccio

NB: sizes of symbols/images only for illustration purposes

Heavy-flavour hadron trigger for high-energy pp program

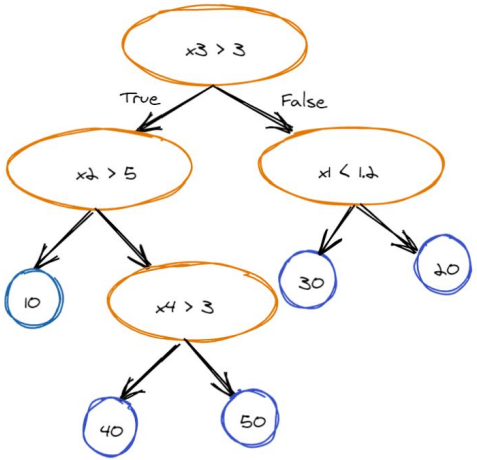
- Trigger dedicated to select interesting events for heavy-flavour (HF) hadron studies
  - selection of signal-like particle candidates reconstructed from track combinatorial
- HF selections exploit XGBoost multi-class BDTs
  - input: few and simple variables based on the HF-hadron displaced decay-vertex topology
  - aim: to be robust against possibly non-optimal reconstruction and calibrations (which are improved iteratively)
- Currently being used in the skimming of all 2022 and 2023 pp data collected by ALICE
  - inference on reconstructed data using ONNXRuntime

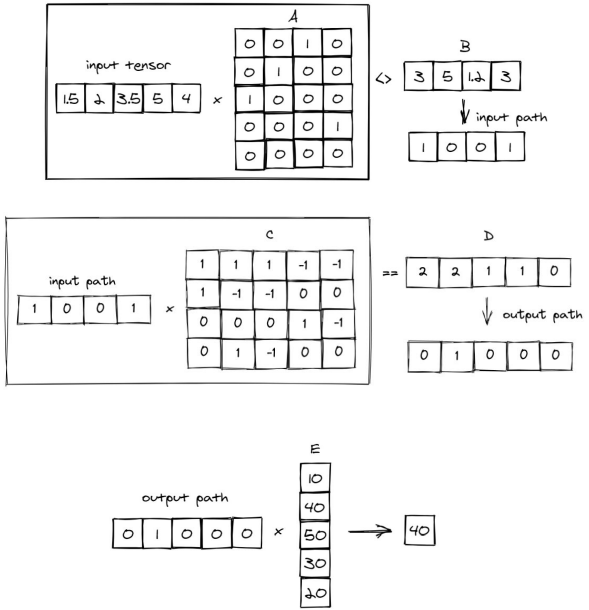


HF-hadron trigger – BDT inference optimisations

- ONNXRuntime is optimised for the tensor computations typical of NNs
  - not so efficient for the inference of BDTs and classical ML algorithms, which are used for the HF trigger and in many other ALICE analyses

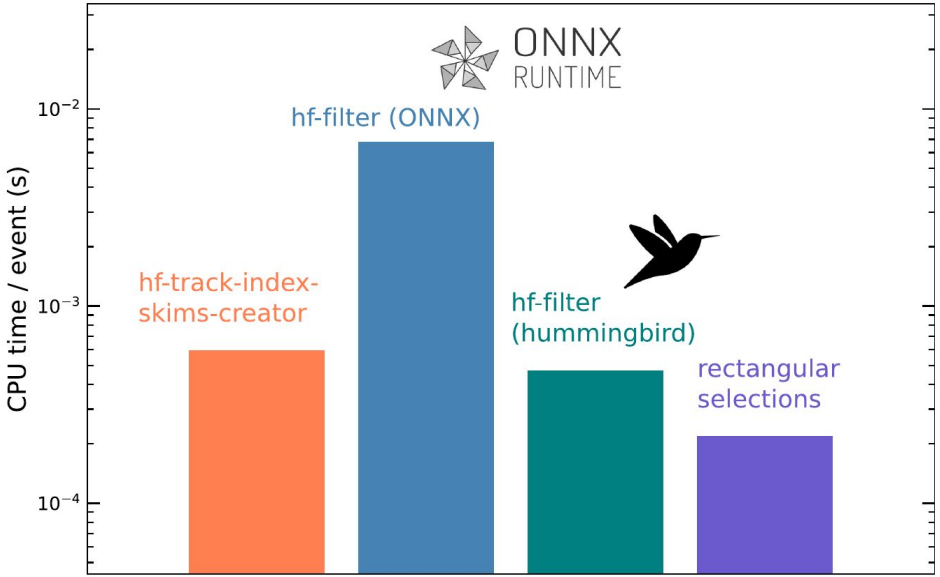
- hummingbird (Python library)
  - converts trained ML models into tensor computation for faster inference





HF-hadron trigger – BDT inference optimisations

- Performance improvement given by humminbird tested in the context of heavy-flavour hadron trigger studies

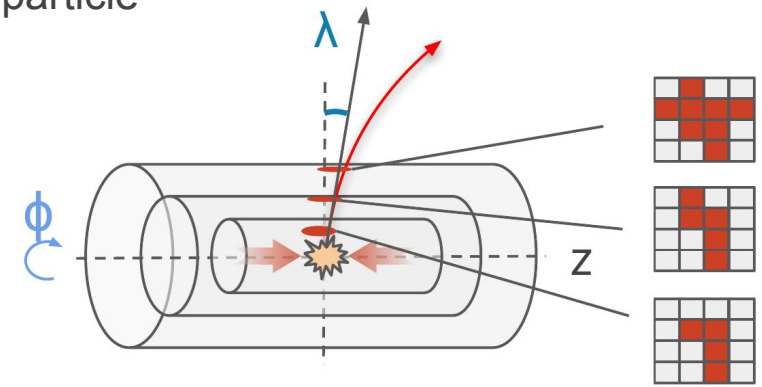


Mind the log scale!

- About 10x speedup compared to same model non converted with hummingbird
- CPU time / event of the task comparable to using simple rectangular selections

Particle identification with the ITS2

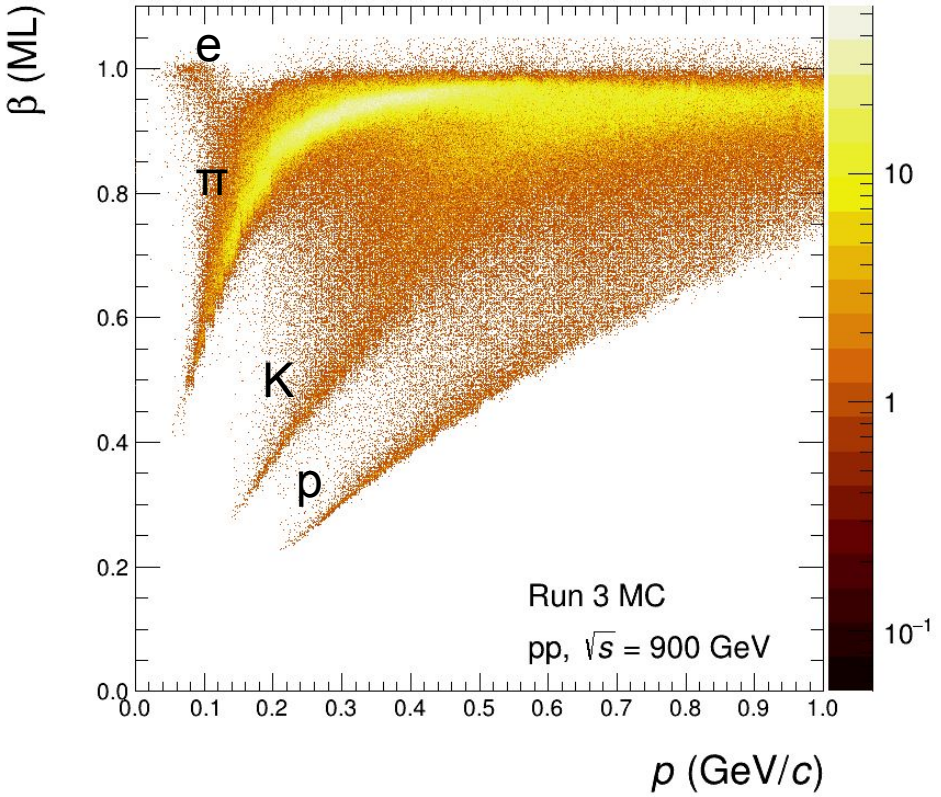
- The new ALICE Inner Tracking System (ITS2) has a binary pixel readout
  - no dE/dx information from deposited charge in the silicon, as present in old detector used during LHC Run 1 and Run 2 ⟶ in principle no particle identification
- Topology of the produced signal (cluster) in the detector layers can be used as a proxy for the energy loss of the particle



- XGBoost BDT regressor to estimate the particle β
  - track information (p, tan(λ)) and properties of clusters (size, shape, …) in the ITS2 layers as inputs to the model

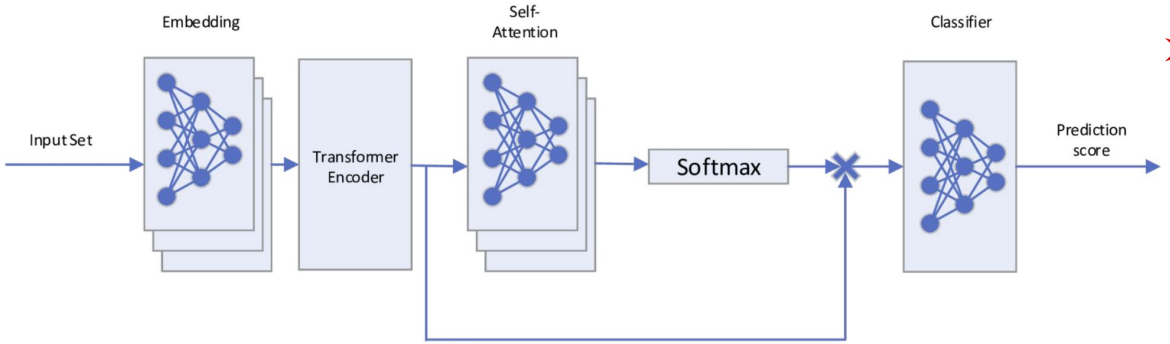
Particle identification with the ITS2

- Training performed using particles tagged in TPC
  - starting from reconstruction output
  - not dependent on data taking period
- Method validated on Run 3 MC
  - good separation between e, π, K, p at low momentum
- Encouraging results on Run 3 data
  - further studies using tagging performed with K_s^0, Λ, Ω decays ongoing
  - training on larger data samples foresee



Combination of detector PID information

- Combine the particle-identification information of different detectors to provide global PID
  - replace hand-crafted combinations and selections
  - aim to provide high purity samples of particles of a given species
- Different NN models trained for each particle species and data-taking period
  - PyTorch library used
  - starting from analysis-object data (AO2D)
  - track information and detector signals related to PID as input



- Information from one or more detector could be missing
  - typical for low p_T particles
  - solution: model based on feature set embedding (FSE) with multi-head self-attention mechanism

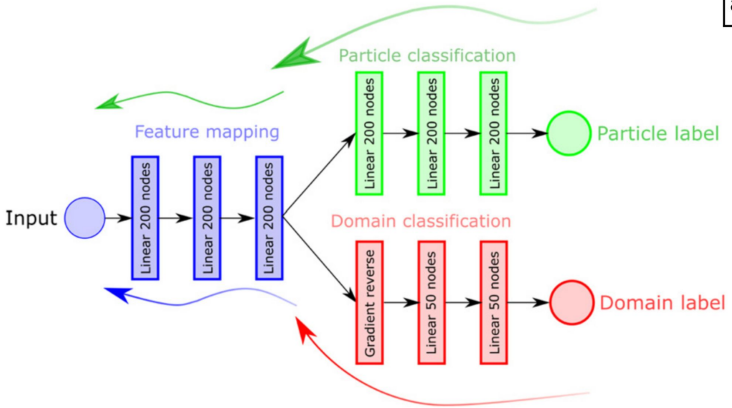
Further details: M. Kabus talk at CHEP 2023

Combination of detector PID information

- On Run 2 pp MC, NN with self-attention + FSE shows better performance than other approaches for incomplete data
  - data imputation
    - mean
    - linear regression
  - NN ensemble

	π			p			K		
model	purity	efficiency	F_1	purity	efficiency	F_1	purity	efficiency	F_1
mean	0.9718	0.9934	0.9825	0.9559	0.8927	0.9232	0.8858	0.8081	0.8452
regression	0.9723	0.9931	0.9826	0.9520	0.8973	0.9238	0.8795	0.8168	0.8470
case deletion	–	–	–	–	–	–	–	–	–
NN ensemble	0.9745	0.9914	0.9829	0.9607	0.8895	0.9237	0.8751	0.8207	0.8470
attention + FSE	0.9734	0.9937	0.9835	0.9648	0.9009	0.9318	0.8841	0.8337	0.8581

best model **2nd best model**



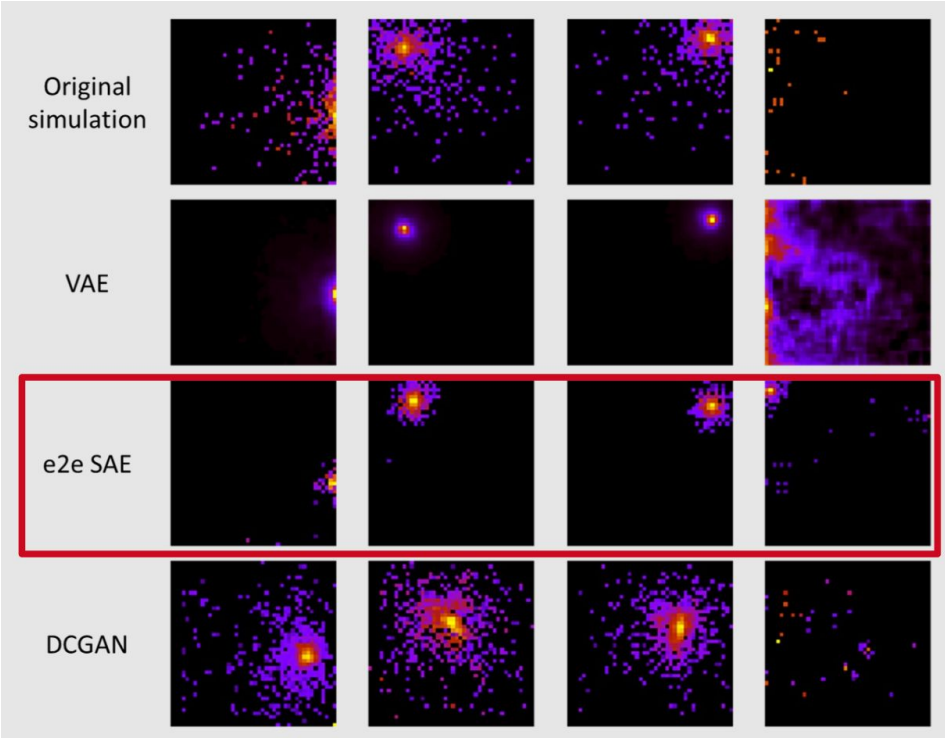
- Further developments
  - address data-to-MC discrepancies using domain adversarial neural networks
  - define approach to systematic uncertainty estimation

Further details: M. Kabus talk at CHEP 2023

Fast simulation for the ZDC

- Generative models to replace full simulation for the Zero Degree Calorimeter (ZDC)
  - ZDC: system of five sampling calorimeters placed at forward rapidity on both sides of ALICE

- Response of the ZDC treated as an image
  - Variational AutoEncoders (VAE) and Deep Convolutional GANs (DCGAN) investigated
  - generation steered by conditional parameters (particle energy, mass, position, …)



e2e SAE → end-to-end Sinkhorn autoencoder [arXiv:1810.01118]

Further details: [J. Dubinski lightning talk at 5th IML Workshop]

Fast simulation for the ZDC

- Various different models evaluated
  - Wasserstein distance to quantify the discrepancy between full and fast-ML simulation
  - best performance provided by conditional DCGAN with modified loss to enhance diversity of generated samples

model	WS MEAN	WS CH1	WS CH2	WS CH3	WS CH4	WS CH5
cond VAE	6.45	4.75	5.03	4.23	4.34	13.72
cond DCGAN	8.25	4.35	5.46	7.28	9.13	14.98
cond end2end SAE	**6.27**	**4.17**	**5.05**	**4.05**	**4.03**	**13.56**
cond DCGAN + auxREG	7.20	4.24	8.42	3.54	4.55	15.19
cond DCGAN + postproc	5.71	2.53	3.92	3.64	5.93	12.55
cond DCGAN + auxREG + postproc	5.16	2.71	4.63	4.89	6.71	8.59
cond DCGAN + selectiv div increase	**4.51**	**2.21**	**4.03**	**4.38**	**6.17**	**8.04**

- Generative models are integrated in ALICE Monte Carlo production workflows
  - about 100x speedup in simulating the ZDC response compared to full simulation
  - some fine tuning still needed to reach physics-ready state

Further details: J. Dubinski lightning talk at 5th IML Workshop

Summary

- ALICE machine-learning activities expanded considerably in the last years
  - ML tools are a staple of data analysis in ALICE, boosting the measurement physics reach
  - from the start of Run 3, ML used in “core” tasks such as:
    - calibration of the TPC particle-identification information
    - software trigger for heavy-flavour studies

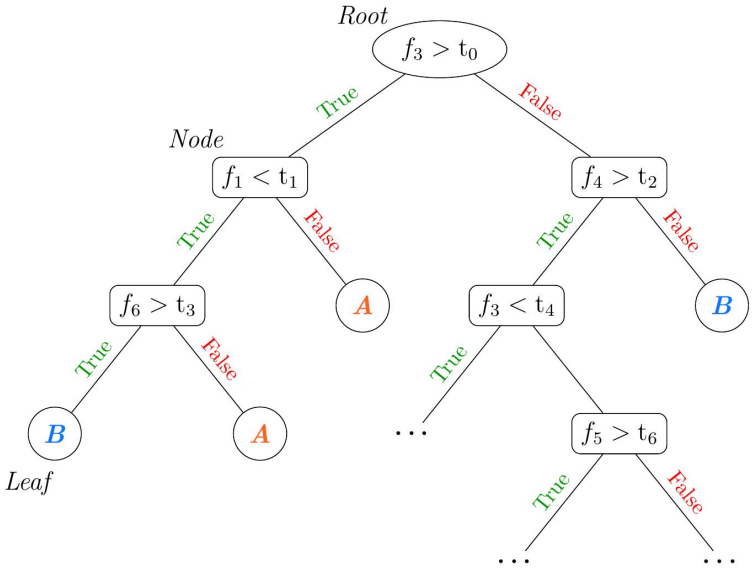
- ML applications based mainly on off-the-shelf Python and C++ libraries
  - with some internal developments to address specific needs and help analysers wanting to dive into the topic

- Large-scale inference of ML models using distributed-computing resources is fundamental to the experiment activities

Backup

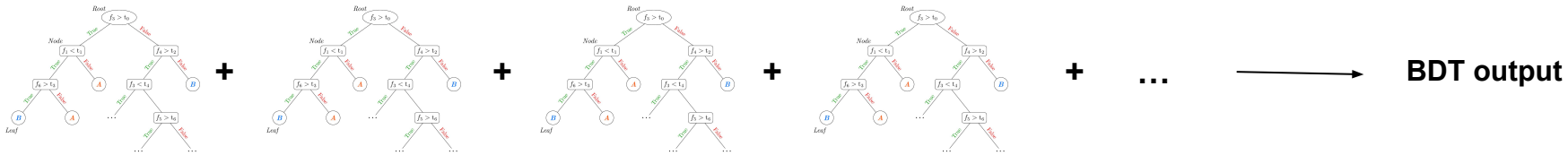
Supervised learning — Boosted Decision Trees

- The building block of Boosted Decision Trees (BDTs) is the decision tree (DT)
- For a binary classification problem
  - DT built recursively utilising the training data
  - at each node the variable and its value that maximize the separation between classes (*A* and *B*) is selected
  - goodness of the separation quantified by a score (Gini index, entropy, . . .)
- A large enough decision tree can perfectly separate the training data
  - however its predictions for unknown data are not so good (poor generalisation)



Supervised learning – Boosted Decision Trees

- Solution: use an ensemble of many small decision trees
  - each DT is built trying to improve the performance of the current ensemble
  - the BDT output is the sum of DT ones



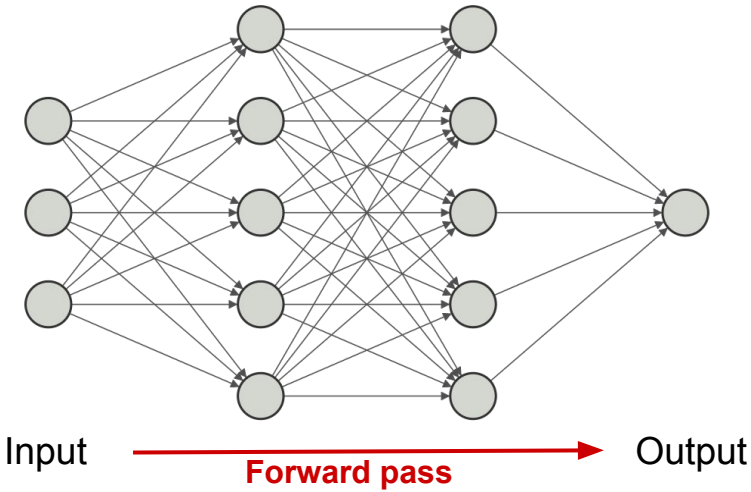
- Modern BDT algorithms are based on a procedure called gradient boosting
  - decision trees are built trying to minimise a target function via gradient descent

Supervised learning – Shallow neural networks

- Loosely inspired by biological neural networks
- Flow of information happens between nodes
  - each node connected to every other node in the subsequent layer
  - each connection has a weight
- Output of a node generally given as

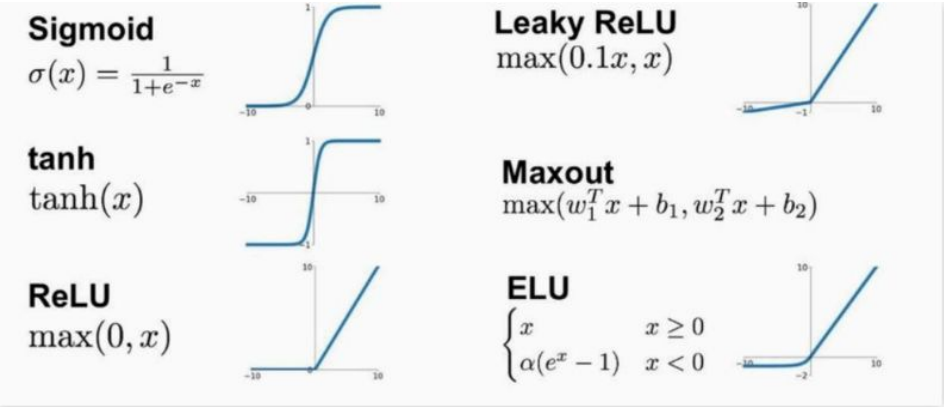
$$g_j = \sigma\left(\sum u_{ij} g_i \right)$$

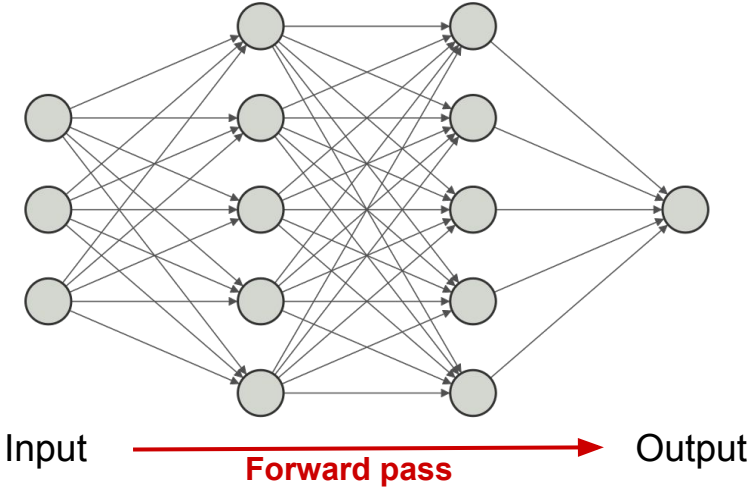
- σ is the so-called activation function



Supervised learning – Shallow neural networks

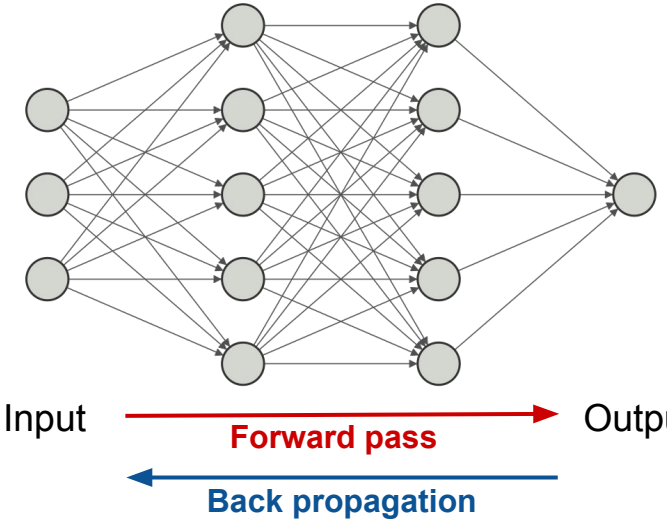
- The activation function introduces non-linearity into the network
  - fundamental to learn complex relations
  - can be any nonlinear function differentiable analytically
- Many possible choices for this!





Supervised learning – Shallow neural networks

- As we calculate each node's output in each layer we are completing one forward pass
- After each forward pass a process called back propagation occurs
  - the error of the network is computed and the weights are updated accordingly
  - the aim is to minimise a target function via gradient descent



(a) compute error on g_j

$$\frac{\partial E}{\partial g_j} = \sum_i \sigma'(h_i) v_{ij} \frac{\partial E}{\partial h_i}$$

should g_j be higher or lower? | how h_i will change as g_j changes | was h_i too high or too low?

(b) for each u_{jk} that affects g_j

(i) compute error on u_{jk}

$$\frac{\partial E}{\partial u_{jk}} = \frac{\partial E}{\partial g_j} \sigma'(g_j) f_k$$

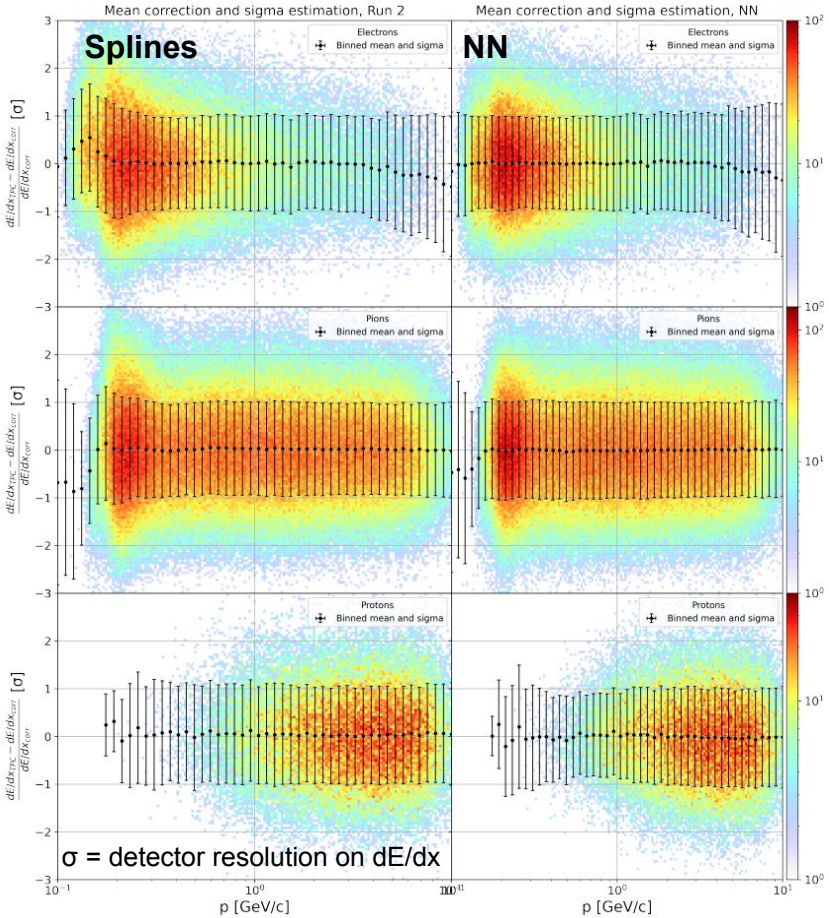
do we want g_j to be higher/lower | how g_j will change if u_{jk} is higher/lower

(ii) update the weight

$$u_{jk} \leftarrow u_{jk} - \eta \frac{\partial E}{\partial u_{jk}}$$

- For each node
  - compute error on output g_j
  - compute error on weights u_{jk}
  - update weights accordingly

TPC PID calibration with neural networks



Further details: [CERN-THESIS-2022-342]

Combination of detector PID information

- Tests of domain adversarial neural networks on Run 2 pp MC

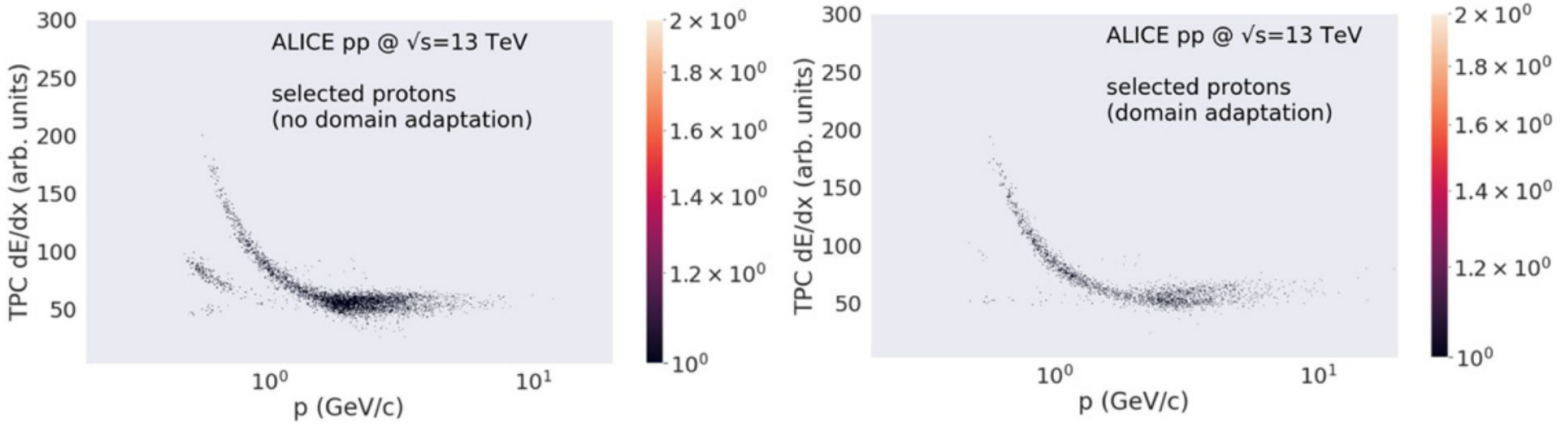
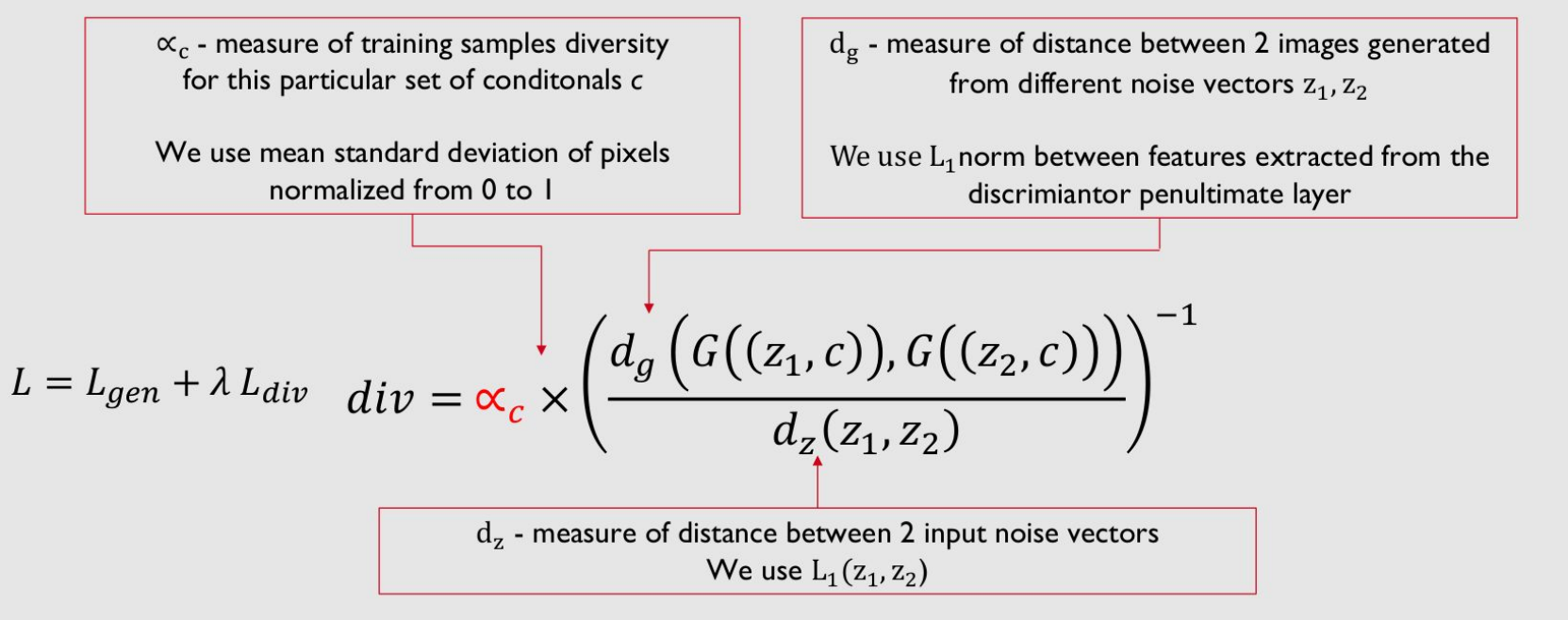


Figure 3. Preliminary result of DANN PID for the TPC detector signal (dE/dx) as a function of particle momentum for particles identified as protons without domain adaptation (left) and with domain adaptation (right).

Further details: JINST 17 C07016 (2022)

Fast simulation for the ZDC

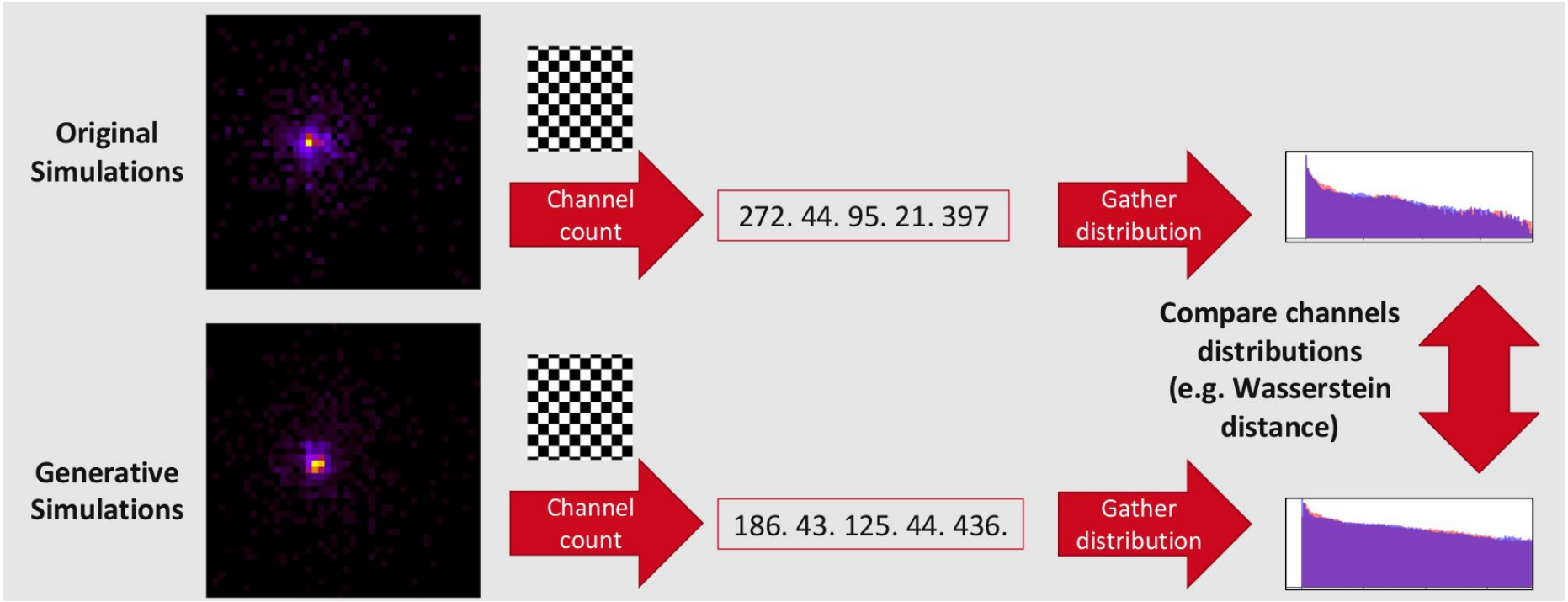
- Selective increase of diversity



Further details: J. Dubinski lightning talk at 5th IML Workshop

Fast simulation for the ZDC

- Validation



Run 2 ML applications at a glance

Signal-vs-background classification

- Boosted Decision Trees (BDTs) and Neural Networks (NN) replacing “traditional” linear selections

Jet p_T reconstruction

- correction for the background from the underlying event
- regression task using shallow NN

Heavy flavor jet tagging

- BDTs and Deep Neural Networks (DNN) to tag heavy-flavour jet topologies

Monte Carlo (MC) reweighting

- improve agreement between data and MC simulations

Data quality assurance (QA)

- K-nearest neighbors and Autoencoders to detect outliers

RootInteractive

- tool for multidimensional statistical analysis
- wrappers for tree-based models and NNs

... not a comprehensive list!