Politechnika Warszawska

WYDZIAŁ FIZYKI

Praca dyplomowa magisterska

na kierunku Fizyka Techniczna
w specjalności Fizyka i Technika Jądrowa

Application of machine learning methods for particle identification in the CBM experiment

Julian Nowak
Numer albumu: 298198

promotor:
dr hab. inż. Hanna Zbroszczyk, prof. uczelni

WARSZAWA 2023

Streszczenie

Tytuł pracy: Wykorzystanie metod uczenia maszynowego do identyfikacji cząstek w eksperymencie CBM

Uczenie maszynowe (ML) to technologia, której popularność i wydajność rośnie dynamicznie. Niniejsza praca bada zastosowanie technik uczenia maszynowego do identyfikacji cząstek w eksperymencie Compressed Baryonic Matter (CBM), który będzie prowadzony na terenie FAIR w Darmstadt, w Niemczech. Jako że w eksperymencie CBM dochodzić ma do nawet 10 milionów zderzeń na sekundę, wydajna metoda identyfikacji cząstek jest niezbędna.

Głównym celem pracy jest stworzenie pakietu ML do identyfikacji cząstek dla eksperymentu CBM. W pierwszej kolejności omówiono rozwój pakietu do konwersji danych, który ułatwia integrację danych symulacyjnych eksperymentu CBM z Pythonem. Praca podkreśla zalety podziału danych na przedziały w zależności od pędu, co przekłada się na poprawę wydajność i czystości zidentyfikowanych hadronów. Przeprowadzono również porównanie różnych podejść do ważenia klas, omawiając zalety i wady jednolitego i "zrównoważonego" ważenia. Interpretacja modeli ML została omówiona poprzez dostarczenie różnych narzędzi wizualizacyjnych, takich jak histogramy m^2 i wykresy SHAP. Przeprowadzono porównanie z tradycyjną metodą "TOF" wykorzystującą dopasowanie gaussowskie, pokazując lepszą wydajność identyfikacji opartej na ML, po wyeliminowaniu niepoprawnie zrekonstruowanych cząstek.

Praca kończy się sugestiami dotyczącymi przyszłych badań, w tym włączenia danych z dodatkowych detektorów i opracowania ulepszonego algorytmu rekonstrukcji. Integracja modeli ML z oprogramowaniem eksperymentu CBM przy użyciu ekosystemu ONNX jest podkreślana jako obiecująca droga do wdrożenia tychże. Wyniki zostały przygotowane przy użyciu danych z modeli Monte Carlo przepuszczonych przez symulowaną konfigurację eksperymentu CBM w GEANT4.

słowa kluczowe:
uczenie maszynowe, identyfikacja cząstek, zderzenia ciężkich jonów, CBM, FAIR, GSI

(podpis opiekuna naukowego) (podpis dyplomanta)

Abstract

Title of the thesis: Application of machine learning methods for particle identification in the CBM experiment

Machine learning (ML) is a technology whose popularity and performance is dynamically growing. This thesis explores the application of ML techniques for particle identification in the Compressed Baryonic Matter (CBM) experiment, which will be hosted at FAIR in Darmstadt, Germany. As the planned interaction rate of the CBM experiment is up to 10 million collisions per second, an efficient particle identification method is indispensable.

The central focus of the work is the creation of an ML package for particle identification for the CBM experiment. The development of a data conversion package that facilitates the integration of CBM experiment simulations data into Python is discussed first. The thesis highlights the advantages of dividing data into momentum-divided bins, improving efficiency and purity of hadron identification. The comparison between different approaches to class weighting, discussing the benefits and drawbacks of uniform and "balanced" weighting, is also performed. Interpretability of ML models is addressed by providing various visualization tools, such as mass-squared plots and SHAP plots. A comparison with the traditional "TOF" method using Gaussian fitting is performed, showcasing the superior performance of ML-based identification when the reconstruction mismatches are eliminated.

The thesis concludes with suggestions for future research, including the incorporation of data from additional detectors and the development of an improved reconstruction algorithm. The integration of ML models into the CBM experiment software using the ONNX ecosystem is highlighted as a promising avenue for implementation. The results have been prepared using data from Monte Carlo models passed through simulated CBM experiment setup in GEANT4.

Keywords:
machine learning, particle identification, heavy-ion collisions, CBM, FAIR, GSI

Oświadczenie o samodzielności wykonania pracy

Politechnika Warszawska

Julian Nowak
298198
Fizyka Techniczna

Oświadczenie

Świadomy/-a odpowiedzialności karnej za składanie fałszywych zeznań oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie, pod opieką kierującego pracą dyplomową.

Jednocześnie oświadczam, że:

- niniejsza praca dyplomowa nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 roku o prawie autorskim i prawach pokrewnych (Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.) oraz dóbr osobistych chronionych prawem cywilnym,
- niniejsza praca dyplomowa nie zawiera danych i informacji, które uzyskałem/-am w sposób niedozwolony,
- niniejsza praca dyplomowa nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadawaniem dyplomów lub tytułów zawodowych,
- wszystkie informacje umieszczone w niniejszej pracy, uzyskane ze źródeł pisanych i elektronicznych, zostały udokumentowane w wykazie literatury odpowiednimi odnośnikami,
- znam regulacje prawne Politechniki Warszawskiej w sprawie zarządzania prawami autorskimi i prawami pokrewnymi, prawami własności przemysłowej oraz zasadami komercjalizacji.

Warszawa, dnia (data) (czytelny podpis dyplomanta)

Oświadczenie o udzieleniu Uczelni licencji do pracy

Politechnika Warszawska

Julian Nowak
298198
Fizyka Techniczna

Oświadczenie studenta w przedmiocie udzielenia licencji Politechnice Warszawskiej

Oświadczam, że jako autor / ~~współautor~~* pracy dyplomowej pt.

Implementation of machine learning algorithms for particle identification in the CBM experiment

udzielam / ~~nie udzielam~~* Politechnice Warszawskiej nieodpłatnej licencji na niewyłączne, nieograniczone w czasie, umieszczenie pracy dyplomowej w elektronicznych bazach danych oraz udostępnianie pracy dyplomowej w zamkniętym systemie bibliotecznym Politechniki Warszawskiej osobom zainteresowanym. Licencja na udostępnienie pracy dyplomowej nie obejmuje wyrażenia zgody na wykorzystywanie pracy dyplomowej na żadnym innym polu eksploatacji, w szczególności kopiowania pracy dyplomowej w całości lub w części, utrwalania w innej formie czy zwielokrotniania.

Warszawa, dnia (data) (czytelny podpis dyplomanta)

* - niepotrzebne skreślić

Acknowledgements

I would like to extend my thanks to Dr. Ilya Selyuzhenkov, who supervised me, but also helped me a lot during my stay at GSI, sharing his experience. I would also like to thank Dr. Shahid Khan, and Oleksii Lubynets, without whom I would not be able to finish my thesis in time. I would not be able to meet them without the support from the Get_Involved programme at FAIR, and Dr. Pradeep Ghosh, so danke sehr! Finally, the biggest thanks go to my supervisor, prof. Hanna Zbroszczyk, for dedicated time, and lots of help.

Contents

Introduction

Machine learning (ML) is one of the most fascinating recent technologies. Its popularity is booming, and ML-based tools are now being recognized by general public, especially thanks to generative models, such as ChatGPT [1].

ML has been applied in the field of heavy-ion collisions for a few years already; probably the biggest success of this field of physics – identification of the Higgs boson – would not be possible without ML [2]. In the planned Compressed Baryonic Matter (CBM) experiment ML is going to be used for i.e. identification of short-lived particles [3].

The main goals of this work is to prepare ML-based particle identification method for the CBM experiment, and compare it with the traditional approach.

In the first chapter, the physical motivation is described. The second chapter introduces the CBM experiment. The third one presents traditional method of particles identification. In the next chapter, ML concepts and tools are discussed. The following chapter shows the created package for ML-based identification of hadrons; the next one provides a comparison between traditional and ML identification. In the last chapter the obtained results are discussed, and the outlook for further development is presented.

Physical motivation

2.1 Standard Model

The Standard Model (SM) is a theory formulated in the 1970s to describe the elementary particles and the interactions between them. It combines the theory of the elementary particles, quantum mechanics, quantum chromodynamics (strong interactions), and electroweak forces (unified description of the weak and electromagnetic forces). Most of its assumptions were confirmed experimentally in the 1980s. The most recent one, (and a great success of high-energy physics) - the confirmation of the existence of the *Higgs boson*, dates from 2013. [4]

With the SM two Nobel Prizes in Physics can be associated - the one from 1979, awarded "for contributions to the theory of the unified weak and electromagnetic interaction between elementary particles(...)" [5], and the second one from 2013, awarded "for the theoretical discovery of a mechanism that contributes to our understanding of the origin of mass of subatomic particles, and which recently was confirmed through the discovery of the predicted fundamental particle, by the ATLAS and CMS experiments at CERN's Large Hadron Collider" [6].

However, the SM is not a complete theory. It fails to account for various phenomena, such as the gravitational force, or the new findings such as the existence of the dark matter, and the muon's magnetism [7]. Nevertheless, the SM effectively describes the majority of interactions between matter and remains a valuable tool in understanding its basic constituents.

According to the SM, there are two main classes of elementary particles: **fermions**, which constitute matter, and **bosons**, which carry interactions. They are presented on Figure 2.1 and described in the next paragraphs.

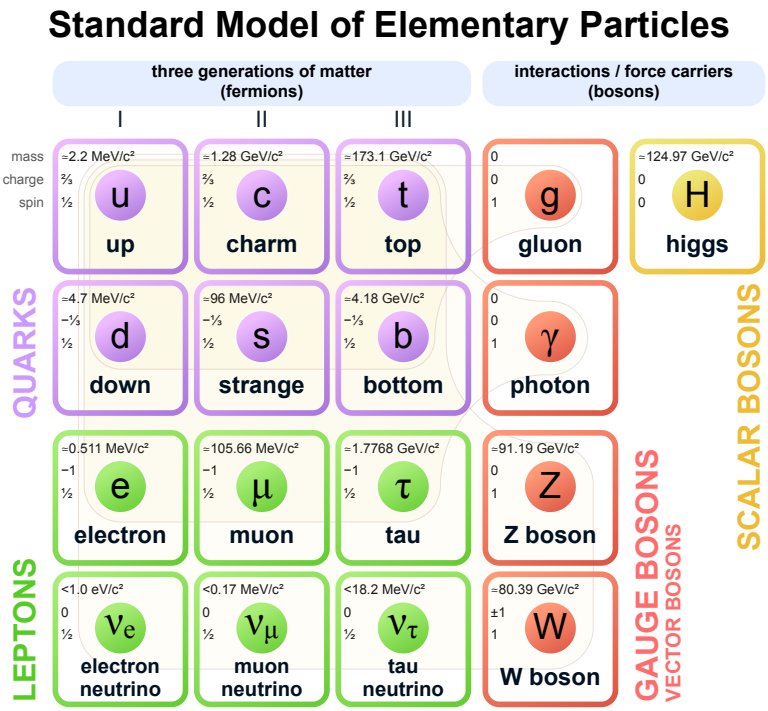


Figure 2.1: Standard Model of Elementary Particles [8]

Fermions follow the Fermi-Dirac statistics (hence the name), they have half odd integer spin and thus follow the **Pauli exclusion principle**. They can be divided into two groups: **leptons** and **quarks**. Quarks, unlike leptons, do not have an integer charge number; quarks (contrary to leptons) have the **color charge**, so they engage in strong interactions. There are three generations of fermions:

I. quarks: *up (u)* and *down (d)*; leptons: *electron* (e^-) and *electron neutrino* (ν_e)

II. quarks: *charm (c) and* strange (s) ; leptons: *muon* (μ^-) and *muon neutrino* (ν_μ)

III. quarks: *top (t)* and *bottom (b)*; leptons: *tau* (τ^-) and *tau neutrino* (ν_τ)

The particles which follow Bose-Einstein statistics are named Bosons, they have an integer spin. There are five elementary bosons carrying:

- strong interactions: *gluons* (g)
- weak interactions: bosons $W^{\pm}$ and Z^0
- electromagnetic interactions: *photons* (γ)
- mass: *Higgs boson* (H)

2.2 Quantum chromodynamics

Quantum chromodynamics (QCD), a part of the Standard Model, is a theory which describes strong interactions between quarks and gluons. It provides explanation for various phenomena such as quark confinement (described later). [4]. According to this theory:

- there are three color charges: *Red*, *Green*, and *Blue* which are exchanged between the quarks via *gluons*, which are bosons themselves
- gluons interact with both quarks and other gluons
- the strong interaction has a pulling character, its potential called *color potential* can be described by the formula:

$$V(r) = -\frac{4}{3}\frac{\alpha_s}{r} + kr \tag{2.1}$$

where α_s and k - coupling constants, r - distance.

According to the formula 2.1, the magnitude of the force grows with distance, unless the latter is bigger than a certain threshold (as shown on Figure 2.2) when a new particle-antiparticle couple is created. This effect, called **quark confinement**, explains why the quarks cannot be separated and form **hadronic matter**. However, if the distance r is very small, and the energy of the system is big enough, there exists another state of matter called **quark-gluon plasma (QGP)** for which the quarks reach asymptotic freedom (the strong interactions still do not allow the existence of *free* quarks, hence the asymptotic freedom).

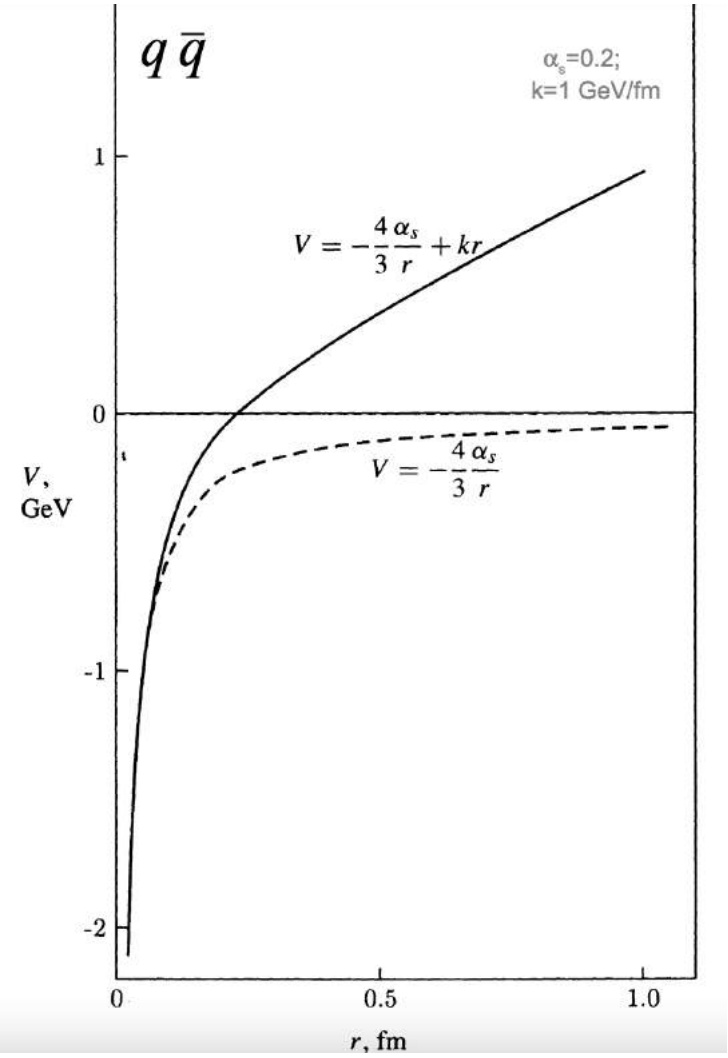


Figure 2.2: Dependence of the color charge potential and the distance between the quarks and gluons. At long distances, the binding energy is too high and a new particle-antiparticle pair is created [9]

The matter can exist in both *hadronic matter* and *QGP* states, so there should be a phase transition between the two. The phase diagram of the matter, according to the QCD, can be created (with net baryon density on the *x*-axis and temperature on the *y*-axis). The Figure 2.3 is an example of such a diagram. The investigation of the QCD phase diagram is one of the main goals of the high-energy physics experiments.

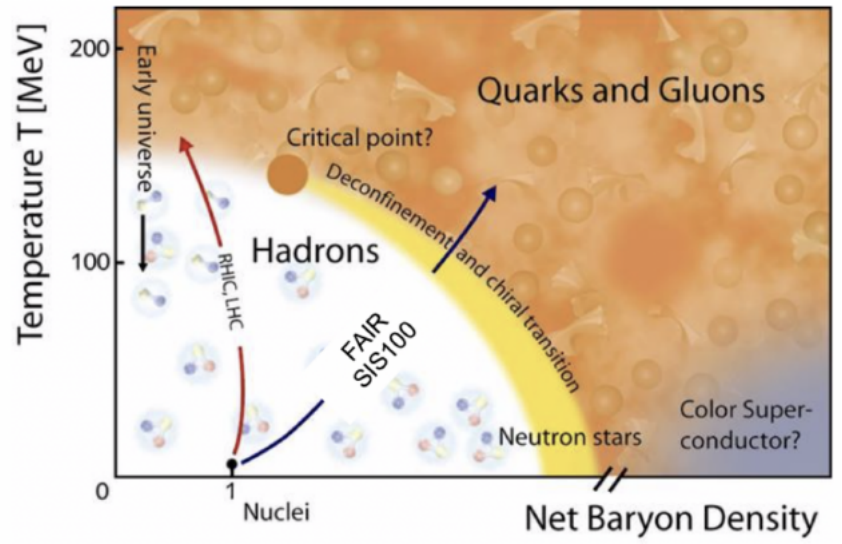


Figure 2.3: Phase diagram of the QCD matter [10]

2.3 Heavy-ion collisions

Heavy-ion collisions at relativistic energies allow achieving high enough temperature and net baryon density, to consequently create QGP for a short time. There are two main types of heavy-ion experiments, depending on the type of collision they use. In the *fixed-target* experiments a heavy-ion is aimed at e.g., a foil - their setup is less complicated and allows for e.g., higher interaction rate; in the *collider* experiments two heavy-ions are accelerated and then collided, consequently allowing to achieve higher energies.

Besides the type of setup, the high energy physics experiments also differ from one to another by i.e. the energy of the collision, interaction rate (number of collisions per unit of time), atomic number of the collided ions, etc. The most important, current experiments are shown on the Figure 2.4.

As can be seen on both Figure 2.3, and Figure 2.4, the experiments at the Large Hadron Collider (LHC) in CERN aim for bigger collision energies and hence temperatures [12]. The experiments performed (or planned) on SIS accelerators (fixed-target) in FAIR aim for bigger interaction rates and consequently bigger net baryon density [13].

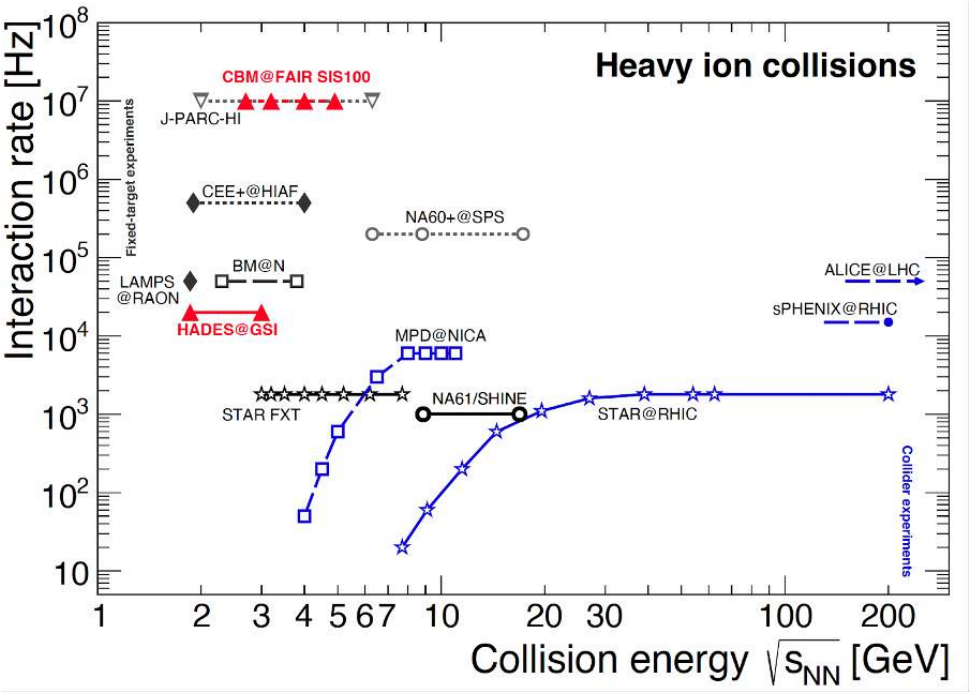


Figure 2.4: The "map" of the heavy-ion collision experiments [11]

2.4 Neutron stars

Neutron star is formed in a collapse of the core of a massive star. Astrophysicists categorize them as compact stars, along with the white dwarfs, and the black holes. Typical neutron star masses are about 1.4 solar masses, $M_\odot$, while the radii are in the range of 10 to 14 km. The central density can reach more than five times the nuclear saturation density. It is why the properties of such stars are largely determined by the equation of state (EoS) of highly compressed baryonic matter, which is also a point of interest of the CBM experiment (more details in Chapter 2). The temperatures, which can reach a few keV in the interior, are small on nuclear scales. Therefore super fluid and/or superconductive states of matter are expected to appear inside such stars.

As mentioned before, compact stars are a result of collapse of massive progenitor stars ($8 < M/M_\odot < 25$). The initial collapse of the core is halted by the stiffening of the equation of state at densities beyond nuclear saturation, which generates an outward travelling shock wave. In this process temperatures of about 50 MeV can be reached. The proto-neutron star cools by emitting a lot of neutrinos first, and later by radiating photons. During this process it shrinks to its final size. The masses of neutron stars are limited by stability against gravitational collapse: above a certain central density the pressure of high-density matter is not sufficient to withstand the gravitational attraction; the star collapses to a black hole. If a lighter star collapses, however, a white dwarf might be created.

The structure of a compact star (presented on Figure 2.5) is as follows:

- Surface - covered by a thin atmosphere, which is relevant for the energy transport and for the shape of the observed electromagnetic spectrum.

- Crust - contains ionized nuclei, arranged in a lattice and embedded in a sea of electrons. At the surface of the star, the most stable nuclei, ^{56}Fe and neighboring, are found. With increasing density, the proton number of the nuclei remains moderate, while the neutron number grows substantially.
- Core - with homogeneous distributions of nucleons, electrons and muons. As the crust, with a thickness of 1-2 km, contributes only a few percent to the total mass of the star, the bulk properties of a compact star depend primarily on the characteristics of compressed matter at densities beyond nuclear saturation.

The transition from the outer to the inner core is specified by the appearance of exotic phases, e.g., matter containing additional particle species such as hyperons, pion or kaon condensed matter, or deconfined quark matter (also important for the physics of the CBM experiment).

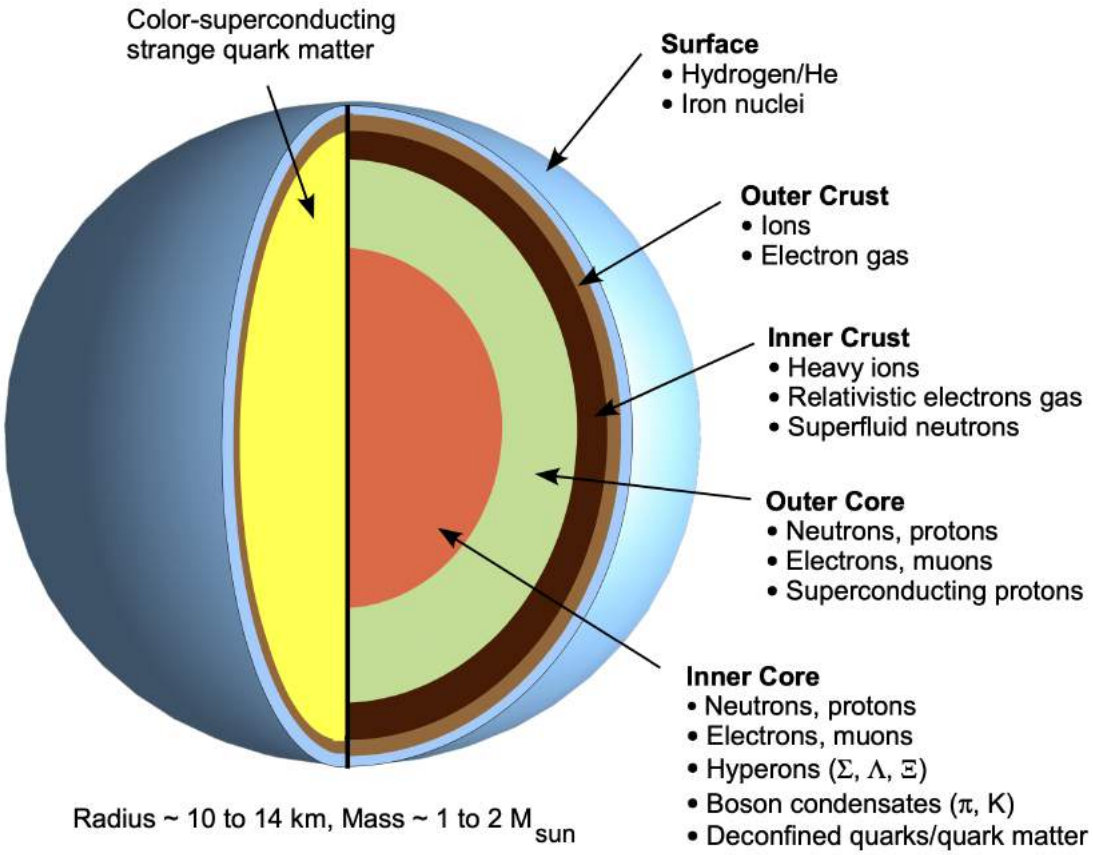


Figure 2.5: Composition of a compact star [14]

The predominance of neutrons (> 90%) in the conventional picture of a compact star with a hadronic core motivated the term "neutron star" itself. More exotic scenarios, where a transition to quark matter takes place inside the core, or stars composed almost completely of strange quark matter, are termed "hybrid stars" and "strange stars", respectively. [14]

FAIR and CBM

3.1 FAIR

FAIR (*Facility for Antiproton and Ion Research*) [15] is an international accelerator facility for the research with antiprotons and ions which is being developed and built in cooperation with various international partners, including Poland. FAIR is being built at the GSI Helmholtzzentrum für Schwerionenforschung in Darmstadt, Germany. The existing GSI accelerators will become a part of the future FAIR facility and serve as the first acceleration stage [15]. The new accelerator, SIS 100, is foreseen to become operational in 2028 [16]. The map of the existing facilities of GSI, and FAIR (which is still under construction) is shown on Figure 3.1. The following experiments are planned to be conducted initially at FAIR:

- APPA (Atomic, Plasma Physics and Applications)
- HADES (High Acceptance Di-Electron Spectrometer)
- NUSTAR (Nuclear Structure, Astrophysics and Reactions)
- PANDA (Physics with High Energy Antiprotons)
- CBM (Compressed Baryonic Matter)

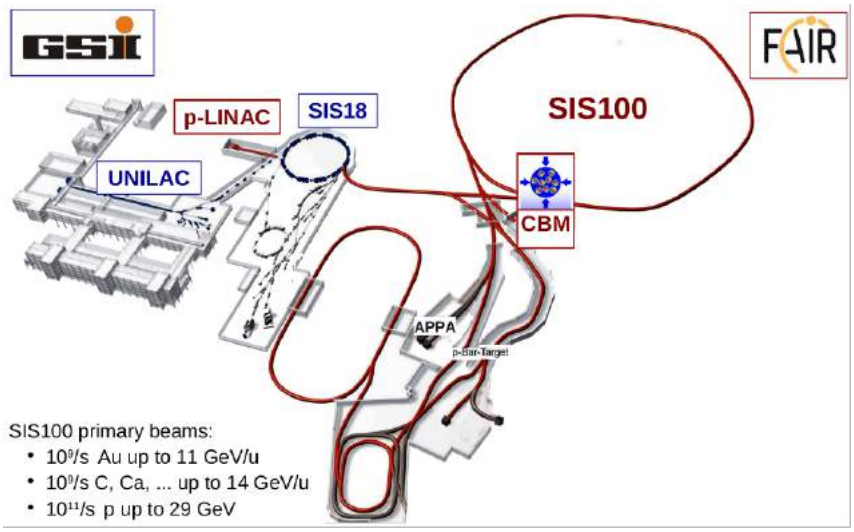


Figure 3.1: Map of FAIR[15]

3.2 CBM experiment

The CBM experiment will be held at the FAIR. Its goal is the exploration of the QCD phase diagram in the region of high net baryon densities and moderate temperatures. It will allow i.a. studies of the equation-of-state of nuclear matter at neutron star core densities. The measurements will be performed at reaction rates up to 10 MHz. The latter requires highly efficient reconstruction and identification setup. [17]

3.2.1 CBM setup

In order to identify the particles coming from the collisions, the setup of 8 detectors is under development in FAIR (Figure 3.2).

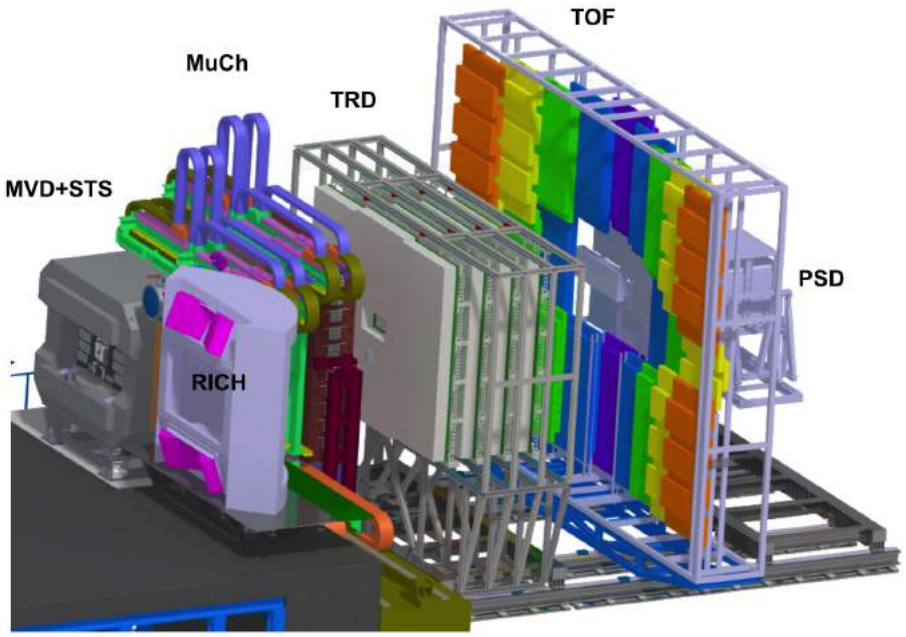


Figure 3.2: CBM setup [16]

The CBM detectors setup will consist of [16]:

- STS - Silicon Tracking System
- MVD - Micro Vertex Detector
- One of the two detectors:
  - RICH - Ring Imaging Cherenkov Detector (in the electron setup)
  - MUCH - Muon Chamber System (in the muon setup)
- TRD - Transition Radiation Detector
- TOF - Time of Flight Detector
- PSD - Projectile Spectator Detector.

In the next subsections the elements of the electron setup (containing the RICH detector) are described.

STS and MVD

Silicon Tracking System (STS) and Micro-Vertex Detector (MVD) are both silicon-based detectors. They are parts of what can be called the tracking system of the setup.

The MVD is placed closest to the primary vertex, its main goal is primary and secondary vertex reconstruction with high precision. The MVD requires very light detector stations equipped with highly granular and thin pixel sensors adapted to hostile running conditions. This is why a specific monolithic CMOS pixel sensor, called MIMOSIS, is being developed for MVD.

The STS comprises 8 tracking stations, and is located inside the 1 Tm dipole magnetic field. The tracking stations are built from carbon fibre supporting ladders on which 876 silicon sensor modules will be placed. Its main goals are track, vertex and momentum reconstruction. [16]

RICH

The Ring Imaging Cherenkov Detector (RICH) will have the dimension of 2 × 4 × 5 m. Cherenkov photons are produced if a particle moves through the medium with a velocity higher than the velocity of light in this medium. RICH will use Multi-Anode Photo-Multipliers (MAPMTs) for Cherenkov photon detection. It will also have 80 trapezoidal glass mirror tiles arranged in two half-spheres for focusing the photons. The CO_2 at atmospheric pressure will be used as the radiator. RICH is especially useful for separating pions from electrons, as the two types of particles have similar mass-squared, but pions are less likely to produce Cherenkov radiation. [18]

TRD

The Transition Radiation Detector (TRD) will be have 4 layers, each constructed from Multi-Wire Proportional Chambers (MWPC), filled with either mix of Xe and CO2 (electron setup) or Ar and CO_2 (hadron setup). Each particle passing through TRD will depose energy, the value of deposed energy can be used for identification. This is in particular important for the separation of, e.g, nuclear fragments, such as deuterons and ^{4}He. [19]

TOF

The time-of-flight wall is used for identifaction of hadrons in the angular range covered by the STS detector (2.5○ -25○). It has an active area of about 120 m^2 (9 m high x 13.5 m wide). To distinguish kaons reasonably well from pions and protons full-system the time resolution of at least 80 ps is needed. The individual detection efficiency should reach at least 95%. Due to these challenges, a Multi-gap timing Resistive-Plate Counters, MRPCs were chosen as its material, which is an affordable and efficient enough solution. It also needs to be moved in the distance between 6m and 10m from the collision vertex for different beam energies.

The difference between the time of entering the MVD detector and reaching the TOF wall, the so-called *time of flight* is useful for identification of particles in a method known as the time

of flight method, described in the next chapter. It causes another challenge, which is a correct matching between the reconstructed particle in the tracking system, and its "hit" in the TOF detector. [20]

PSD

The Projectile Spectator Detector can be used for determination of centrality and impact factor of the collision. In 2022, a new project of Forward Wall, scintillator-based forward detector system with a Silicon photo-multiplier (SiPM) readout, was proposed, which should replace the PSD. Its main goal and function, however, will remain the same. [16]

3.2.2 Data Simulation

The main CBM accelerator, the SIS100, will not start functioning sooner than in 2028, so the Monte Carlo (MC) models are handy in simulating the possible results and planning the actual setup of the experiment [16].

The majority of the simulations performed by the CBM Collaboration are performed using two MC-based simulation packages: **URQMD** (Ultra relativistic Quantum Molecular Dynamics) [21], and **DCM-QGSM-SMM** (Dubna Cascade Model and Statistical Multifragmentation Model) [22]. The production of new particles via the formation and fragmentation of specific colored objects, strings, are simulated in both models. The differences between the two arise on different stages of a string formation and fragmentation. [22] A simulation of heavy-ion collision is shown on Figure 3.3.

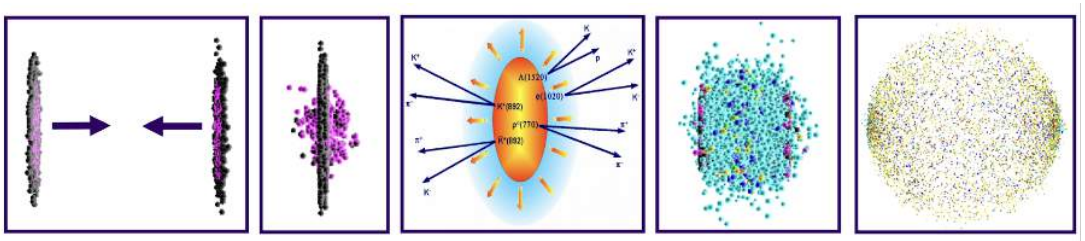


Figure 3.3: Visualisation of heavy-ion collisions in the URQMD model[23]

The data from the MC models is later passed through the CBM setup simulation in **GEANT4** – a toolkit for simulating the passage of particles through matter [24]. The CBM setup is simulated in it, allowing the recreation of behaviour and work of different detectors and the influence of the construction elements. Finally, the simulated MVD+STS, RICH, TRD, TOF, and PSD hits are reconstructed into tracks and clusters and saved into a special data format, called *Analysis Tree* [25].

Traditional methods of particles identification

In this chapter, the traditional methods of particles reconstruction and identification are introduced. Most importantly, the time-of-flight method, which is the main hadron identification method in the CBM experiment, is explained. Also, the Bethe-Bloch formula, which allows for identification of electrons, is described.

4.1 Particle identification using the time-of-flight method

The time-of-flight method (TOF) is a multi-step method used for the identification of hadrons such as pions, kaons, and protons. First, hits in the STS detector are identified, and matched with a certain particle (track reconstruction). It gives information about the momentum and charge sign. These tracks are extrapolated to the TRD stations where the TRD hits are included in the track reconstruction. In the last step, these reconstructed tracks are matched to the nearest hit in the TOF detector. Finally, the mass-squared of a particle can be determined using the formula:

$$m^2 = \frac{p^2}{c^2} \cdot \left(\frac{c^2 t^2}{L^2} - 1 \right) \tag{4.1}$$

where c – velocity of light in vacuum, p – reconstructed momentum, L – reconstructed distance from the first hit in the STS to the TOF (due to magnetic field it is not linear, so it is not a simple distance between the STS and TOF detectors), t – time-of-flight.

A so called "TOF plot" can be created, showing m^2 on the *y*-axis, and $p \cdot q$ (or simply p) on the *x*-axis, as in the upper left corner of the Figure 4.1. To further differentiate between the groups of the particles, Gaussian functions can be fitted to the distributions of mass-squared. This task is getting increasingly complicated, as the momenta of the particles are higher, and the tails of the distribution of each particle overlap (as shown in the Figure 4.1).

For example, the measurement of event-by-event particle ratio fluctuations requires kaon identification with high purity (shown on Figure 4.2). The requirement of a kaon purity of 99% restricts the efficiency to laboratory momenta below to about 3.5 GeV (using traditional identification methods). , 38% of the generated kaons are geometrically accepted, and 18.4% of the emitted kaons can be reconstructed and identified with a purity of 90%. (as presented on Figure 4.3) [14]

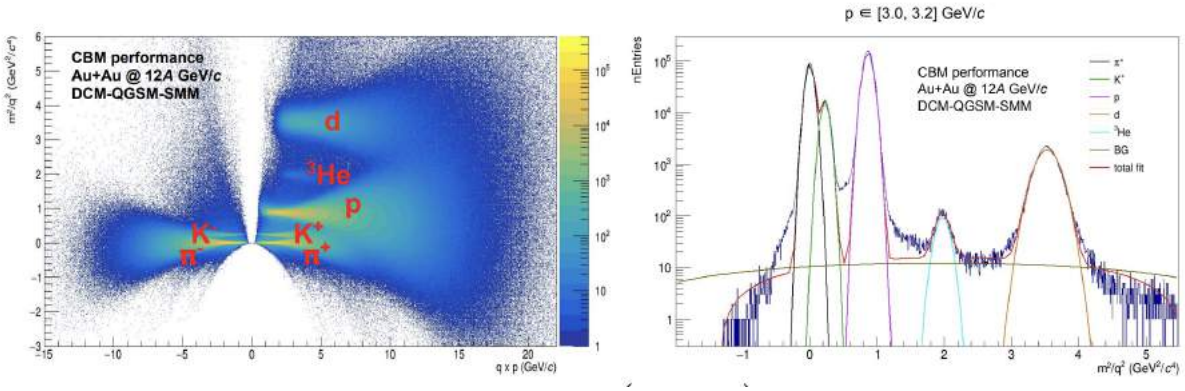


Figure 4.1: Left: TOF plot for simulated particle species in the CBM experiment at 12AGeV/c. Right: Gaussian fits applied to particle in the momentum bin $p = (3.0, 3.2)$ GeV/c [26]

One of the drawbacks of this method is that it is limited to only two variables: mass-squared and momentum. While several solutions to increase the efficiency and purity of this method exist, such as multidimensional fitting (using more reconstructed variables) exist, Gaussian fitting gets increasingly complicated with each new variable (new dimension). Also, the "TOF" method requires manual selection of Gaussian fitting parameters for each momentum bin; the optimal fit parameters must be selected again for each collision energy, detector setup, etc., which makes this method inefficient.

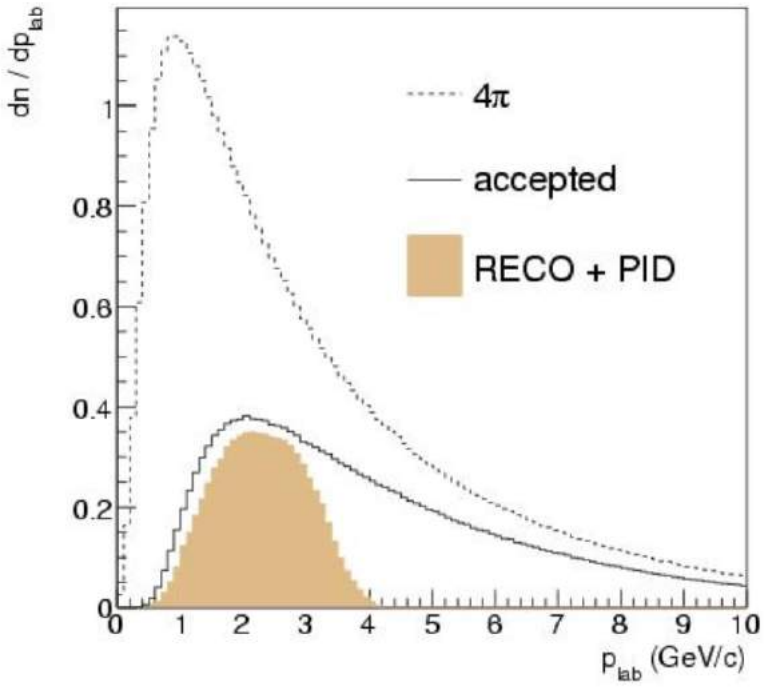


Figure 4.2: Distribution of generated, accepted, and identified kaons (99% purity) as a function of laboratory momentum [14]

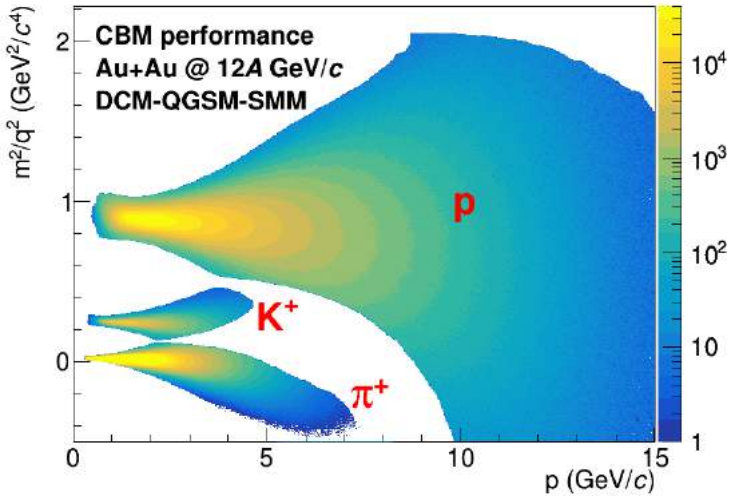


Figure 4.3: Proton, K+ and positive pion selection with 90% purity requirement [26]

4.2 Bethe-Bloch formula

The TRD detector is useful for track reconstruction, as presented in the previous section, but it also provides information about the deposited energy of a particle in each of its 4 stations. The specific energy loss can be described usin the relativistic Bethe-Bloch formula:

$$-\left\langle \frac{dE}{dx} \right\rangle = K\frac{Z}{A}\frac{q^2}{\beta^2}\left[\frac{1}{2}\ln\left(\frac{2m_e c^2\beta^2\gamma^2 T_{max}}{I^2}\right) - \beta^2 - \frac{\delta(\beta\gamma)}{2}\right] \tag{4.2}$$

where $K = 4\pi N_A r_e^2 m_e c^2 = 0.307$ MeV*mol*cm^2, Z – atomic number, A – mass number, q – charge of the particle, m_e – invariant mass of the electron, r_e – mean radius of electron, c – velocity of light in vacuum, β – ratio of velocity of a particle to c, γ – Lorentz factor, T_{max} – the maximum kinetic energy possible to be deposited on a free electron in an elastic scattering process, I – mean excitation energy, $\delta(\beta\gamma)$ – density correction form.

This formula could be used for identification of hadrons, but only for low momenta, as e.g., in the HADES experiment (as shown on Figure 4.4), as the distributions of hadrons overlap for momentum $p > 1$ GeV/c.

However, in the CBM experiment the information about the deposited energy could be used to differentiate between pions and electrons, as their mass-squared is similar, and the two types of particles are impossible to distinguish using only the TOF method. Electrons, which are lighter, and thus travel faster, will leave more energy in the TRD stations (as seen on Figure 4.5).

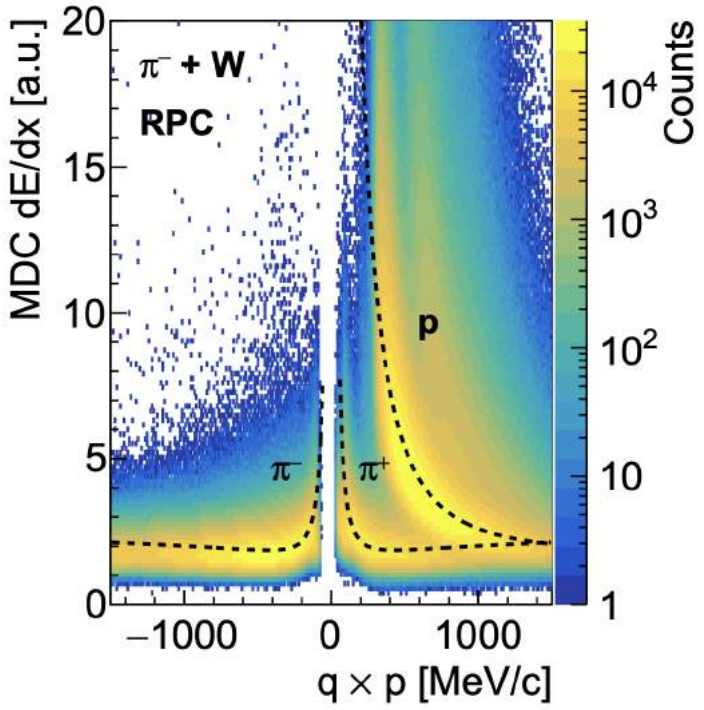


Figure 4.4: Histograms of particle's specific energy loss vs. momentum measured by the energy loss detector of the Hades experiment (RPC). The black dashed lines represent theoretical distributions for protons and pions. The overlap of the two lines, above 1000 MeV/c (1 GeV/c) makes the identification of the two particle types impossible [27]

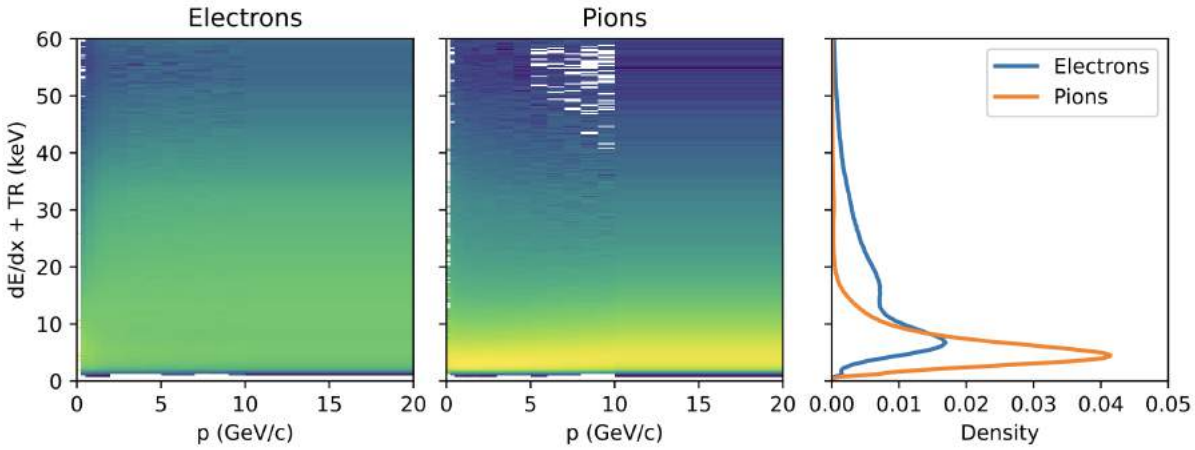


Figure 4.5: Left: Column-wise normalized histogram of the simulated energy output to the TRD for different momenta of the electrons and pions (middle), respectively. Right: Momentum integrated histograms. Electrons have more energy output to the TRD due to the generated transition radiation. [28]

Machine learning

Machine learning (ML) is a branch of **artificial intelligence (AI)** and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. [29]. The term "Machine Learning" was first introduced in a paper from 1959, where the computer was trained to play the game of checkers [30]. In this chapter different types of machine learning algorithms are presented, with an emphasis on the **XGBoost** which is used in this work.

5.1 Types of ML alghoritms

ML can be divided into four primary categories based on how they learn:

- **supervised learning** - using labeled datasets to train algorithms that classify data or predict outcomes accurately.
- **unsupervised learning** - using machine learning algorithms to analyze and cluster unlabeled datasets.
- **semi-supervised learning** - it uses a smaller labeled data set to guide classification and feature extraction from a larger, unlabeled data set.
- **reinforcement learning** - similar to supervised learning, but the algorithm is not trained using sample data. This model learns using the trial and error method.

Division can also be drawn based on the task of the ML model:

- **Classification** - model that assigns input into one of the predefined classes.
- **Regression** - function that maps input data into continuous output values.
- **Clustering** - data are divided into groups with certain common traits, without knowing the different groups beforehand.
- **Generation** - building a model to generate data that are akin to a training dataset in both examples and distributions of examples.

An useful illustration of these categories is shown on Figure 5.1

Another common source of misunderstandings is the difference between the **deep learning** and "classical" ML. The way in which deep learning and machine learning differ is in how each algorithm learns.

- **Deep machine learning** doesn't necessarily require a labeled dataset. Deep learning can ingest unstructured data in its raw form (e.g., text or images), and it can automatically determine the set of features which distinguish different categories of data from one another. This eliminates some of the human intervention required and enables the

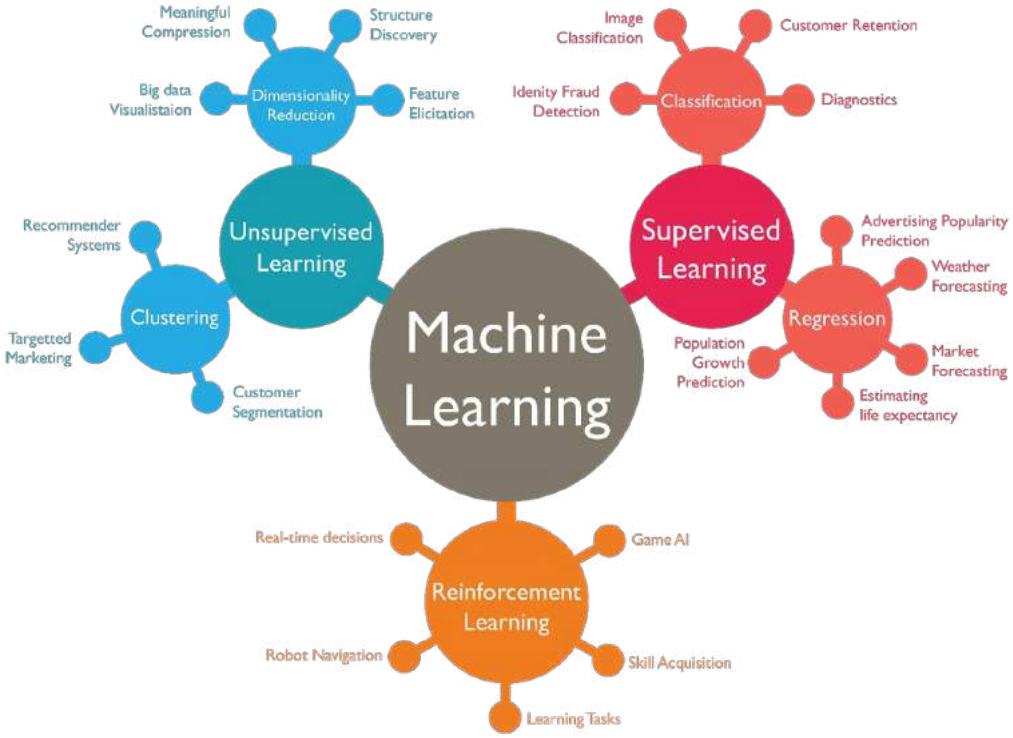


Figure 5.1: Type of ML algorithms [31]

use of larger data sets. On the other hand, these algorithms are computationally more expensive; *interpretability* of their results is also more challenging

- **Classical**, or "non-deep", machine learning is more dependent on human intervention to learn. Human experts determine the set of features to understand the differences between data inputs, usually requiring more structured data to learn. They are, however, faster to train; the decision-making process and its source is also less vague. [29]

An illustration of the difference between the two types of ML alghoritms is shown on Figure 5.2

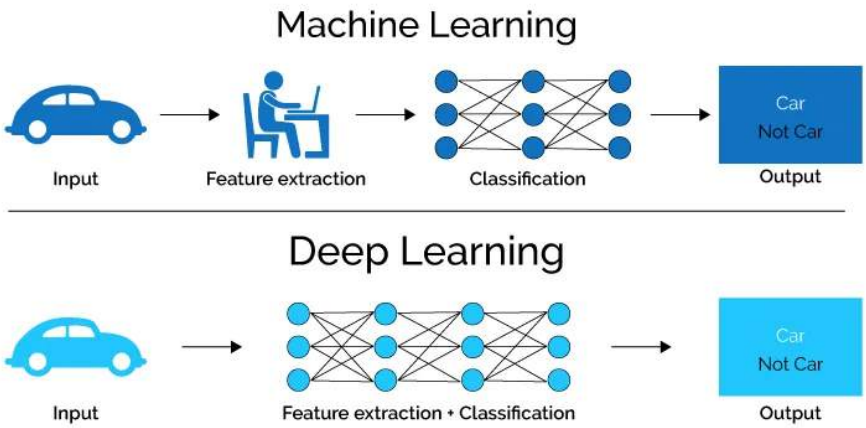


Figure 5.2: Comparison between classical ML and deep learning [32]

5.2 XGBoost

XGBoost is a **decision-tree-based, supervised** ML algorithm that uses gradient boosting, available as an open-source package. It has unique tree structure and it offers parallel processing and regularization parameters. [33]. It is highly efficient when used with e.g., tabular data. It combines four elements (also shown on Figure 5.3):

- **Decision trees** - Graphical representation of possible decisions based on certain conditions
- **Random Forest** - Selecting a random subset of features from multiple decision trees, and **bagging** - making a decision based on the majority of their predictions
- **Gradient Boosting** - Applying gradient descent algorithm to minimize the errors from random forests to train the algorithm [34]

5.2.1 Decision trees

Decision trees (DT) are key to understanding the logic behind the XGBoost alghoritm. DT are a machine learning technique first developed in the context of data mining and pattern recognition. [35] The basic principle consists of extending a simple cut-based analysis by applying multiple variables at the same time. Most object which are to be classified do not have all the clear characteristics of a particular class. The concept of a decision tree is therefore to not immediately reject events that fail a criterion, and instead to check whether other criteria may help to classify these events properly. [36]

A brief description of the binary DT algorithm, in this example seperation between signal, and background events, is following:

Consider a sample of signal (s_i) and background (b_j) events, each with weights w_i^s and w_j^b, respectively, described by a set $\overrightarrow{x_i}$ of variables. This sample constitutes the root node of a new decision tree. Starting from this root node:

1. If the node satisfies any stopping criterion, declare it as terminal (a leaf) and exit the algorithm

2. Sort all events according to each variable in $\overrightarrow{x}$

3. For each variable, find the splitting value that gives the best separation between two groups, one with mostly signal events, the other with mostly background events. If the separation cannot be improved by any splitting, turn the node into a leaf and exit the algorithm.

4. Select the variable and splitting value leading to the best separation and split the node in two new nodes (branches), one containing events that fail the criterion and one with events that satisfy it.

5. Apply recursively from step 1 on each node. [36]

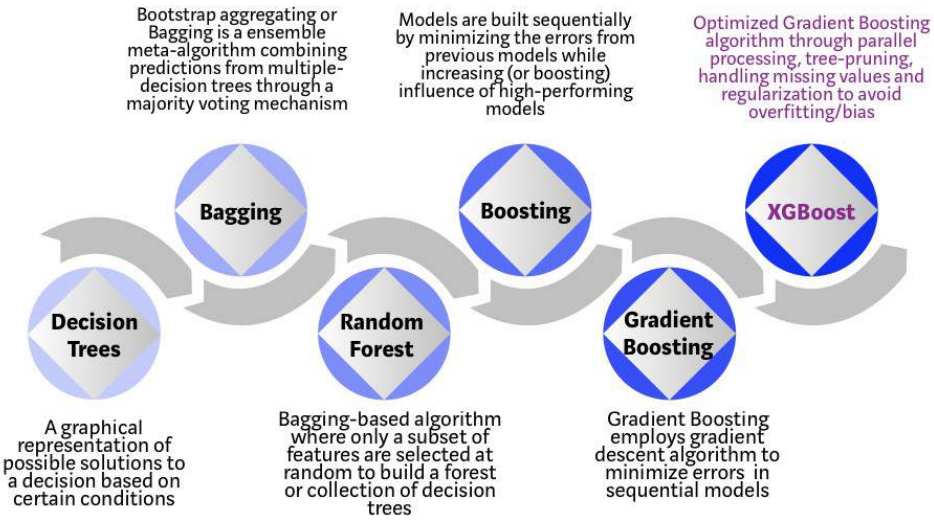


Figure 5.3: Evolution of XGBoost Algorithm from Decision Trees [34]

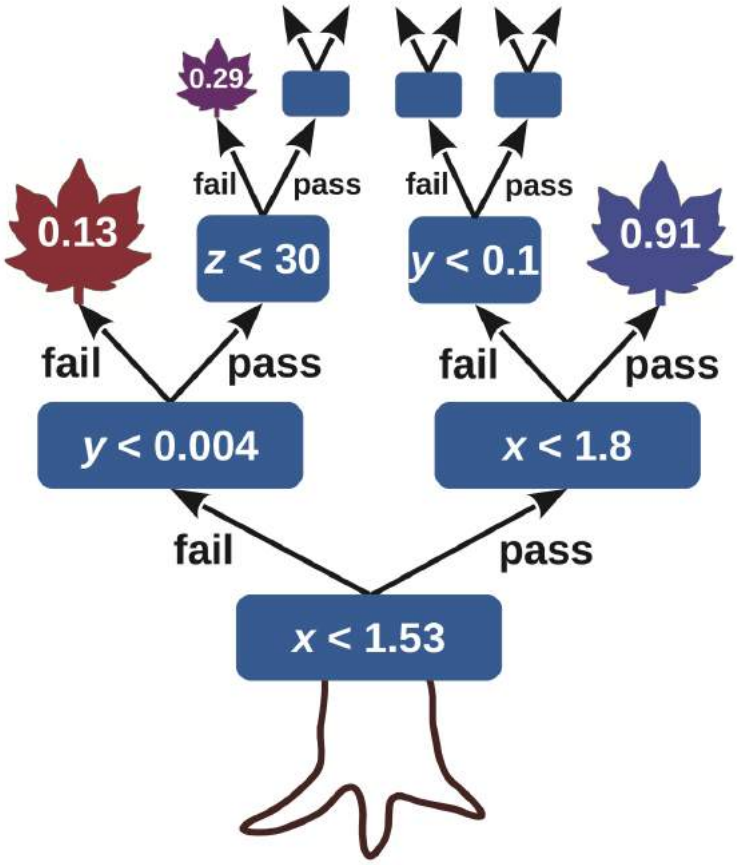


Figure 5.4: Graphical representation of a decision tree. Blue rectangles are internal nodes with their associated splitting criterion; leaves are terminal nodes with their purity [36]

A graphical representation of such a DT is shown on figure 5.4

Contrary to classical DT, the XGBoost combines multiple DT, and returns the **probability** of a certain decision instead of a binary decision (thanks to *random forest*, and *bagging* methods) and trains itself given the error of its predictions (using the previously mentioned *gradient boosting*).

5.2.2 Hyperparameters

The XGBoost (and in general almost all ML models) has **hyperparameters** - parameters that describe the model itself. They have a strong influence on the efficiency of the model. There are many hyperparameters which can be fine-tuned, but the ones which will be set in this work are:

- **n_estimators** [300, 1200] - number of decision trees.
- **max_depth** [2, 12] - depth of a decision tree, with every level the data is split.
- **learning_rate** [0. 01, 0.1] - shrinks weights of new features.
- **subsample** [0.3, 0.9] - samples randomly a ratio of dataset in each boosting round.
- **alpha** [0, 5] - L1 regularization, shrinks parameters' weights.
- **lambda** [0, 10] - L2 regularization, shrinks parameters' weights.
- **gamma** [0, 10] - min. loss reduction required to make a further partition on a leaf node of the tree. [37]

The values in square brackets are ranges in which each parameter would be set (they are not necessarily limited to these values, but these were chosen experimentally for this work). There are multiple ways to choose the optimal configuration (values), notably *Grid Search*, where each combination of hyperparameters is tested. This approach, however, is highly inefficient, as it takes too much time to find the optimal configuration. In this work, the **Optuna** software will be used, which is an open-source software, providing combination of efficient searching and *pruning* (which monitors the intermediate result of each trial and kills the unpromising trials prematurely in order to speed up the exploration) alghoritms. [38]

5.2.3 Unbalanced datasets

Another challenge in classification ML is correctly dealing with unbalanced datasets, in which some of the classes are more present than others. For example, the number of produced protons in a heavy-ion collision is significantly higher than the number of produced kaons. This overrepresentation of protons would result in a model, which is more likely to classify a particle as a proton than a kaon (as the XGBoost aims to statistically have the most accurate decision, so a "ground probability" of a particle belonging to some type depends on the overall number of each particle type in the training dataset. While this could be beneficial for, e.g., higher purity of the selected kaons, it is possible to force the model to treat each particle type equally. Some of the solutions include:

- **Undersampling** - randomly removing samples from the majority class to reduce its dominance.
- **Oversampling** - increasing the number of samples in the minority class by duplicating or synthesizing new instances.
- **Class weighting** - adjusting the class weights during training to give more importance to the minority class.

The first approach is undesirable in this work, as it would decrease the number of protons, hence removing some of the edge cases, in which e.g., there are some protons in the

mass-squared region close to the one of kaons. The second approach is difficult to implement, as it would require e.g., another generative ML network to correctly simulate more rare particles. The third approach is the easiest to implement, e.g, using the *scikit learn compute_class_weight* method [39], which assigns a different weight to each class i by the formula:

$$w_i = \frac{n_{\text{samples (all classes)}}}{n_{\text{classes}} \times n_{\text{samples (class i)}}} \tag{5.1}$$

5.2.4 Missing data

Another frequent challenge for ML is dealing with missing data. In heavy-ion collisions, some particles often do not leave traces in some of the detector's subsystems. The simplest solution is to simply reject particles with missing data, which would result in lower overall efficiency, though. Some other solutions for these problems were proposed, e.g., an adaptation of the AMI-Net architecture, used for medical diagnosis [40], or recreating missing data, using e.g., k-nearest neighbors algorithm, to complete missing values. Comparing these more sophisticated methods goes beyond the scope of this thesis; the XGBoost internal methods for dealing with missing data are compared with the simplest approach of rejecting particles with missing values.

5.3 Interpretability

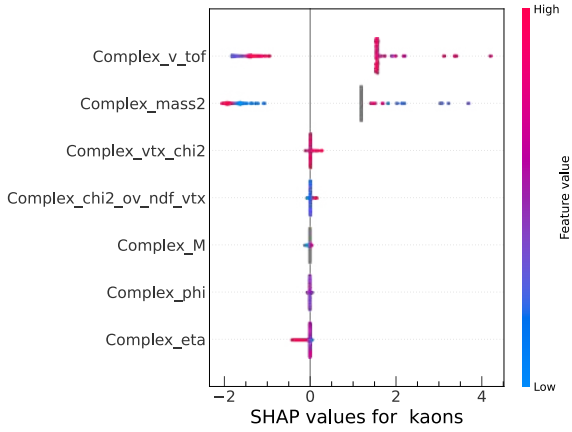
Interpretability is a crucial aspect of machine learning models, especially when dealing with complex algorithms. As applying a "black box" model for heavy-ion research could result in some unpredicted results, understanding how exactly the ML models make their decisions is an important step before implementing them in the experiment software. SHAP (SHapley Additive exPlanations) plots are a powerful tool that enhances model interpretability by providing insights into feature importance and the impact each feature has on individual predictions. SHAP plots combine game theory principles with local explanations to offer a comprehensive understanding of model predictions. [41]

Three main types of SHAP plots are used in this work, each created separately for each class (particle type):

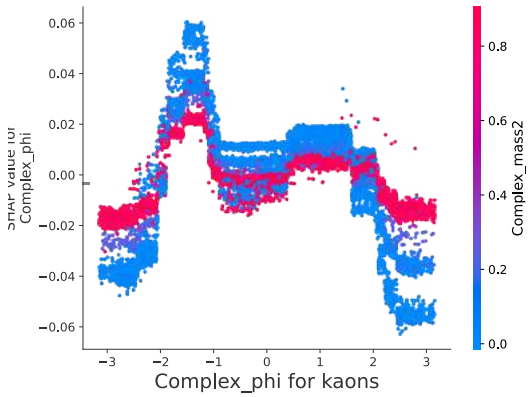
1. **Summary plots** (presented on Figure 5.5a) - features are sorted based on their importance, with the most influential features at the top. The horizontal position of each feature's bar indicates the magnitude of its effect on predictions (SHAP value). Features associated with positive SHAP values suggest that a particle could belong to a given class.

2. **Scatter plots** (presented on Figure 5.5b) - each data point is represented as a single dot. The x-axis represents the feature's value for that data point, while the y-axis represents the corresponding SHAP value. The color of each dot indicates the value of another feature. It can be helpful in understanding interactions between features and their impact on the SHAP values.

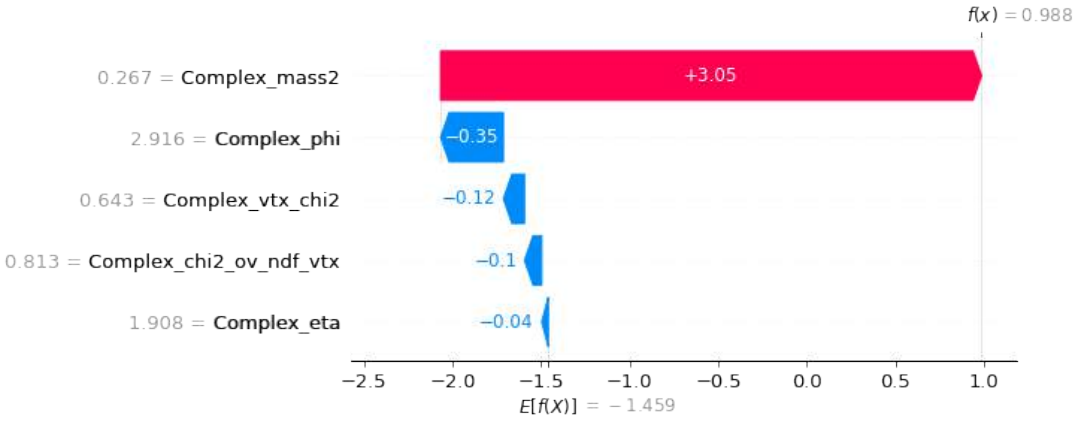
3. **Waterfall plots** (presented on Figure 5.5c) - created for a selected point, e.g., a specific particle. They are useful for understanding the relative importance and direction of feature contributions. By analyzing a waterfall plot, one can identify the key drivers behind specific predictions and gain insights into the decision-making process of the model. It can e.g., help understanding why several particles in a specific TOF plot region are mismatched as another type of particle, as for a specific value of each feature it returns its exact SHAP value. They also show the *expected value* of a prediction for each class ($E[f(x)]$), which can help understadning e.g., "ground probability" connected with overall number of each particle type (as described in "Missing data" section).



(a) Summary plot



(b) Scatter plot (here for ϕ angle of a kaon vs. its mass-squared)



(c) Waterfall plot for a particle (here a probability of a particle being a proton is estimated)

Figure 5.5: Three types of the SHAP plots

Identification of charged hadrons using ML

In this chapter, the application of the ML algorithms for charged hadron identification is discussed. Opposed to the traditional "TOF" method, instead of Gaussian fitting, the data from MVD+STS and TOF detectors (in the electron setup) will be provided directly to the ML model. The aim is to differentiate between the three groups of charged hadrons:

- protons
- kaons
- pions (in the third group, positrons and muons are included as well, as their indistinguishable without using data from other detectors as mentioned in the chapters before). Positive muons are, however, very rare so not shown on this plot (for only 500k events).

The mass-squared of all the particles to be distinguished is shown on Figure 6.1.

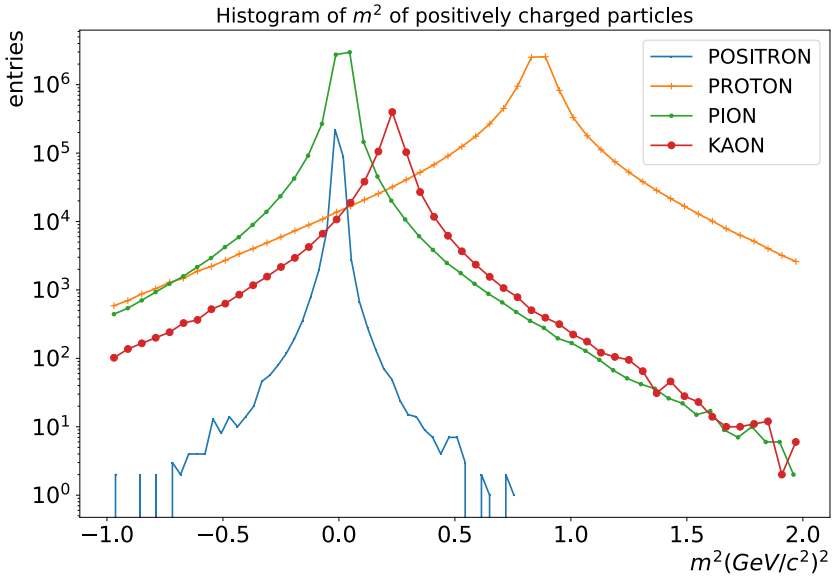


Figure 6.1: Histograms of mass-squared of each particle class (differences of quantity of each particle type can also be observed)

6.1 Data preparation

6.1.1 Converting data

First step is converting data, from previously mentioned *AnalysisTree* format, to *PlainTree* format, which can be loaded using Python. In this aim, the **ml-tree-plainer** package was created. [42]

The list of available variables depends on the detectors to be included. In this chapter following variables are considered:

- From the TOF detector:
  - m^2 calculated using Formula 4.1
  - distance l, and time-of-flight t
  - $v_{tof} = \frac{l}{t}$
- From tracking detectors:
  - reconstructed momentum p, and its components p_x, p_y, p_z
  - reconstructed momentum in terms of transverse momentum p_T, pseudorapidity η, and azimuthal angle ϕ
  - *vtx*$_{\chi^2}$ (called *vtx_chi2* later)- primary vertex fit χ^2
  - $\frac{\chi^2}{NDF}$ *(vtx)* (called *chi2_ov_ndf)* later - χ^2 of the fit to the primary vertex divided by the number of degrees of freedom.
  - reconstructed charge sign q
- M - reconstructed multiplicity of the event
- Information about MC simulated particle:
  - pid - PDG code of the particle [43]
  - MC-true momentum p, and MC-true pT, η, ϕ

6.1.2 Data cleaning

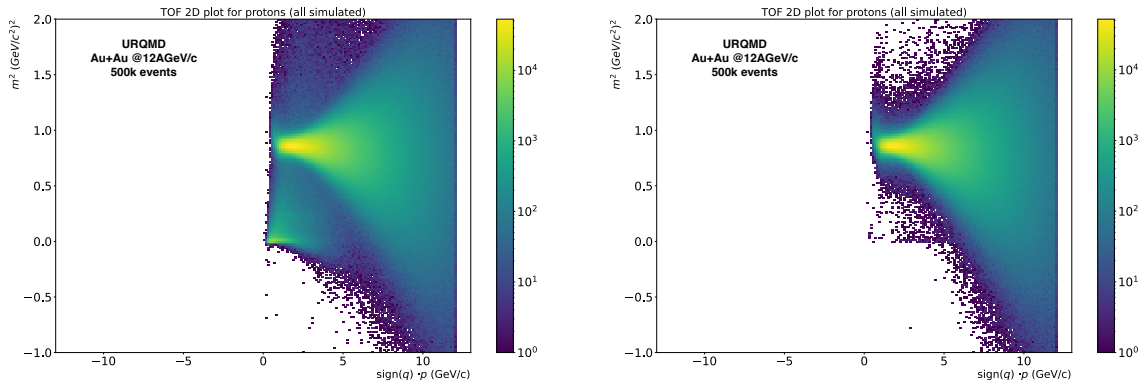
To reject the numeric values of parameters that do not have physical sense, but are a result of a mismatch or computational error, some selection criteria are applied before the beginning of the model training. The following preselection is applied:

- $-1(GeV/c^2)^2 < m^2 < 2(GeV/c^2)^2$
- $pT < 2GeV/c$
- $0 < \eta < 6$
- $0 <$ *vtx_chi2* < 4000
- $0 <$ *chi2_ov_ndf* < 4000

6.1.3 Reconstruction mismatches

During earlier work [44] the problem of mismatches, so particles that are incorrectly identified during the reconstruction process, were found. As the ML algorithm learns by analyzing examples, the incorrect training dataset will likely result in incorrect predictions.

In this work, a possible solution was proposed: In the *AnalysisTree* format each TOF hit is associated with a particle associated with a MC simulated particle; each reconstructed particle in the tracking detectors is also associated with a MC-true particle; each TOF hit is later matched with a corresponding reconstructed particle (if such a particle exists; if not, the particle is not being taken into consideration at all). However, the MC-true particle associated with the reconstructed particle might not be the same as the MC-true particle associated with the TOF hit of the same particle. In this case, we reject such input, so that we double check if not only there exists a match between the tracking detectors, and the TOF detector, but also if the matched particle is exactly the same MC simulated particle. A result of this "double-check" is shown on a "TOF" plot on Figure 6.2.

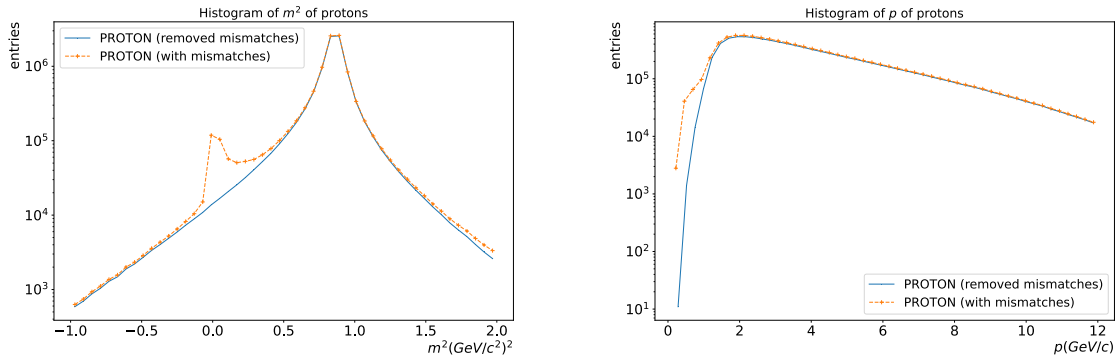


(a) Mismatched protons

(b) Protons with removed mismatches

Figure 6.2: "TOF" plot for protons

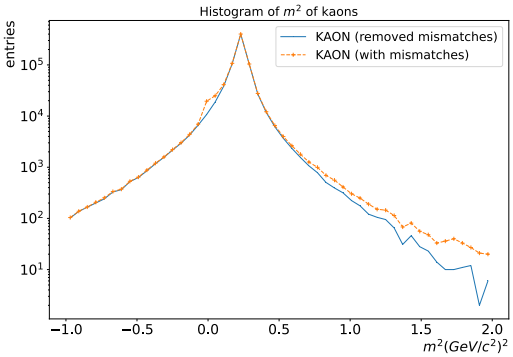
In this process about 17% of all reconstructed particles are lost, the exact differences in distribution of mass-squared and momentum are shown on different plots for each particle type, i.e. on Figure 6.3 for protons, on Figure 6.4 for K+, on Figure 6.5 for $\pi+$, and on Figure 6.6 for positrons.



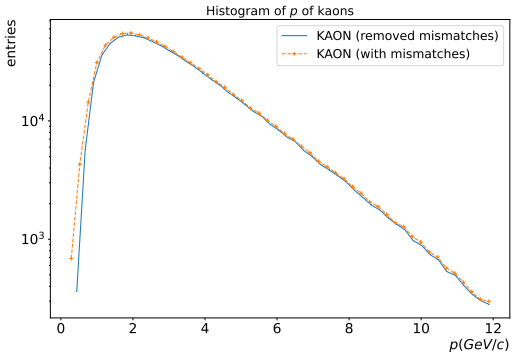
(a) Mass-squared distribution

(b) Momentum distribution

Figure 6.3: Mismatched protons

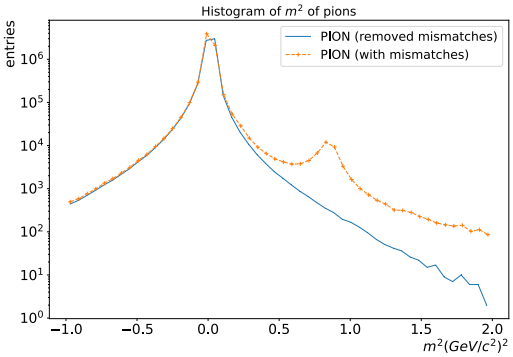


(a) Mass-squared distribution

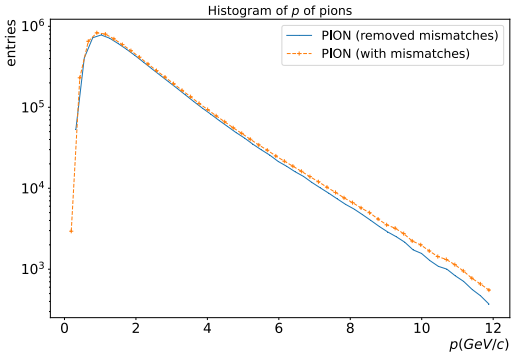


(b) Momentum distribution

Figure 6.4: Mismatched kaons

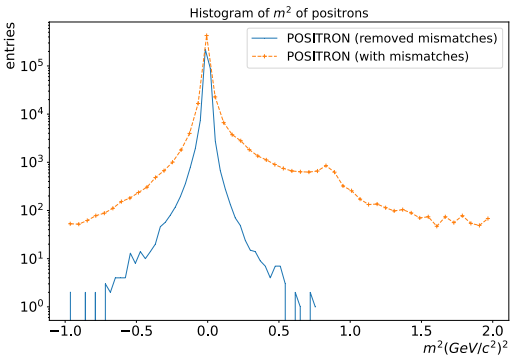


(a) Mass-squared distribution

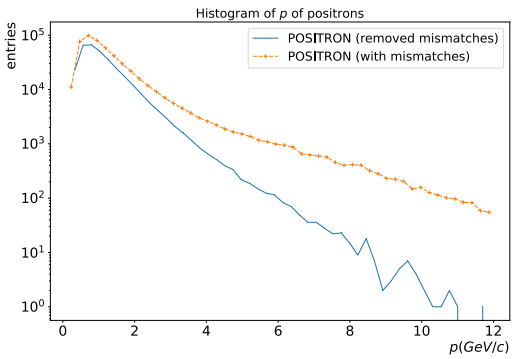


(b) Momentum distribution

Figure 6.5: Mismatched pions



(a) Mass-squared distribution



(b) Momentum distribution

Figure 6.6: Mismatched positrons

6.1.4 Training dataset

The final training dataset consists of 2M events generated in DCM-QGSM-SMM, passed through the CBM experiment setup in Geant4. For the training dataset only primary particles are selected. In this chapter, only positively charged particles are selected for both training, and validation, as the negative particles identification is a slightly different problem, as it would

consist of the binary classifier which would distinguish pions from kaons, as antiprotons are too rare. The "TOF" plot are shown on Figure 6.7 for protons, Figure 6.8 for kaons, and Figure 6.9 for pions. While the 5σ mass-squared preselection was tested, finally all particles are used to keep the training dataset as broad as possible.

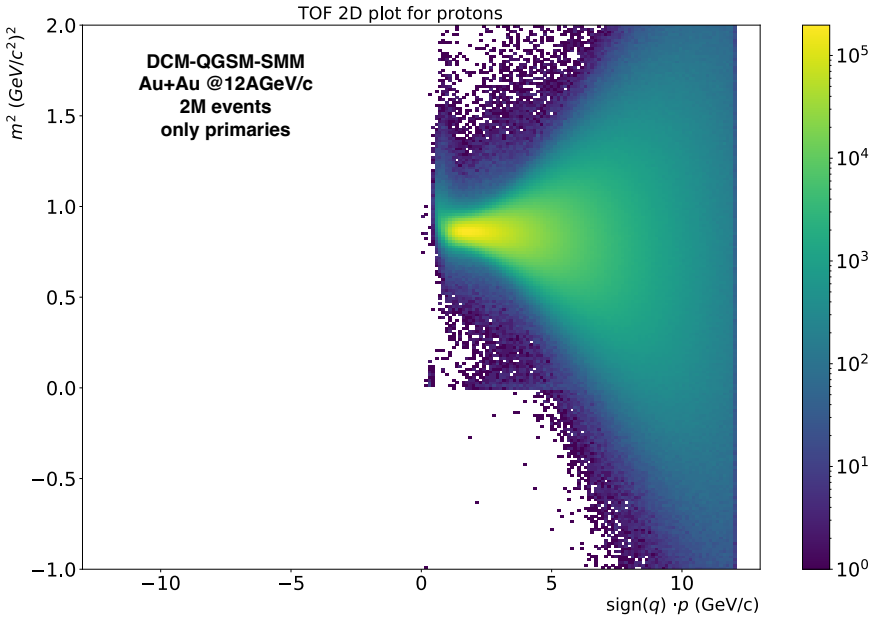


Figure 6.7: Protons used for training

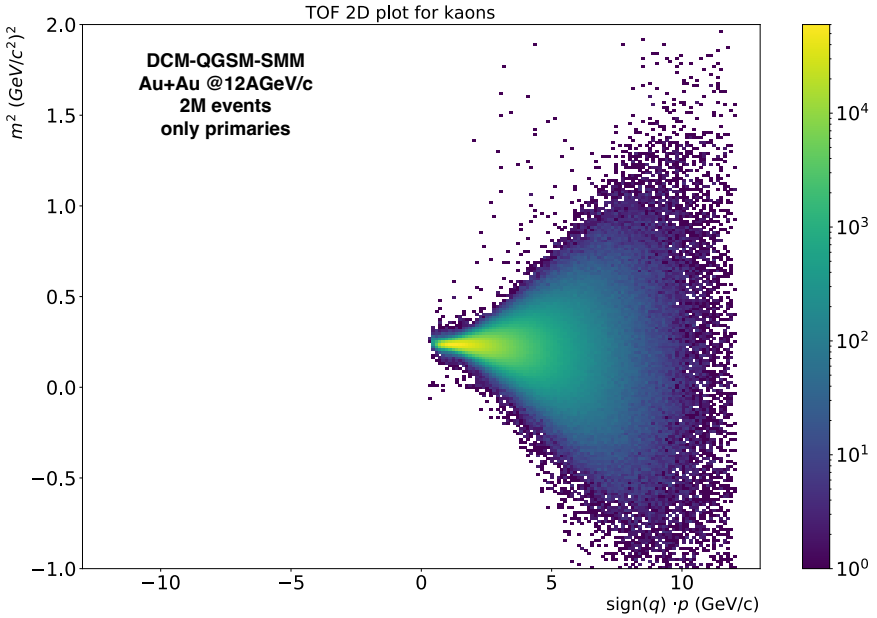


Figure 6.8: Kaons used for training

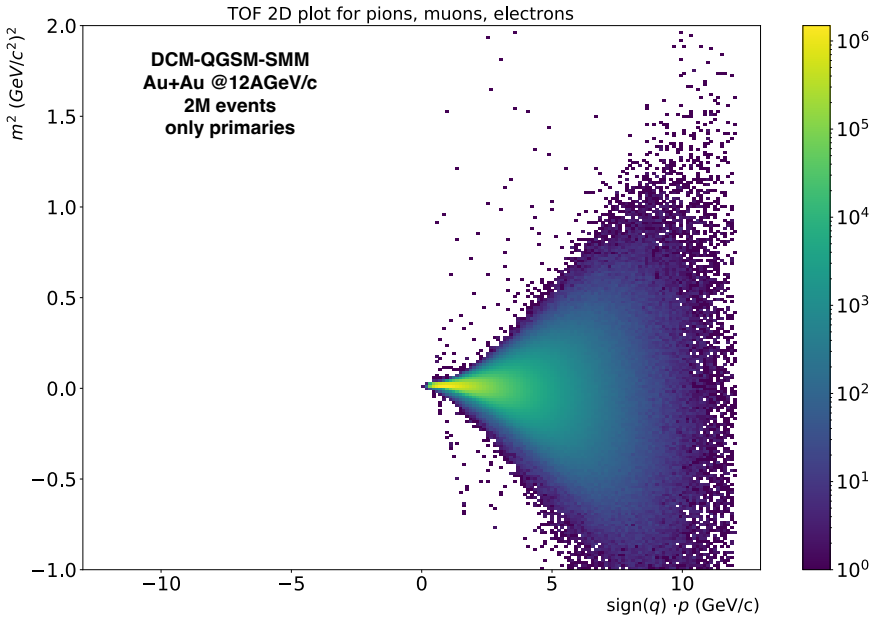


Figure 6.9: Pions used for training

6.2 Model preparation

The training, and validation of ML models is being accomplished using created **ml-pid-cbm** package [45], which bases many of the functionalities on the **hipe4ml** package [46]. The ml-pid-cbm package was created using **Python 3.8**, with standard **pandas**, **numpy**, and **matplotlib** libraries. As mentioned before, the ML algorithm used in this work is **XGBoost** in version 1.5.0, optimization of hyperparameters is done using the mentioned before **optuna** package. The created package is optimized for multithreading, so the **fastreeshap** library was selected for creating SHAP plots, as it uses i.a. multithreading and more efficient algorithm than the original shap package [47]. The used version of the fasttreeshap package was 0.1.5., of which the author of this work is a co-author. [48]

6.2.1 Selection criteria

The XGBoost for each particle returns its probablity of belonging to one of the classes, i.e. probaility of being a proton, kaon, or pion. In this work it is called Boosted Decision Tree (BDT) score. A fair comparison between different configurations can only be done, if the selection criteria of the minimal probability value are universal. It means the particle is classified as the class which has the highest probability, as long as the probability is bigger than the selected *BDT cut*, a minimal value of the accepted probability. If the BDT score is lower than the BDT cut, a particle is classify as *background* (bckgr).

The following alghoritm is proposed:

- The minimal purity value p_{min} is selected by the user
- The algorithm tests multiple BDT cut and chooses:

1. BDT cut with the highest efficiency, for which purity is higher than p_{min}
2. If there is no BDT cut with purity higher than p_{min}, it selects the BDT cut with the highest purity score.

where efficiency (often called *recall* in ML) is defined as:

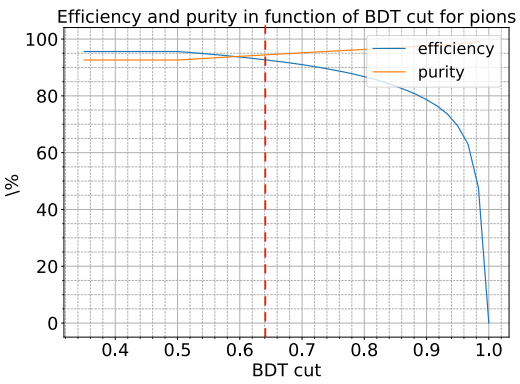
$$\text{efficiency} = \frac{\text{correctly identified particles}(BDT > BDT_{cut})\text{of type X}}{\text{all simulated particles of type X}} \tag{6.1}$$

and purity (often called *precision* in ML) is defined as:

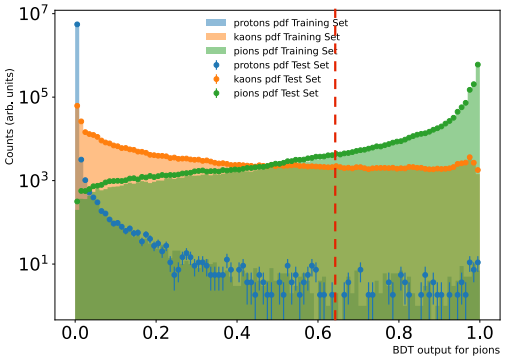
$$\text{purity} = \frac{\text{correctly identified particles}(BDT > BDT_{cut})\text{of type X}}{\text{all particles identified as type X}(BDT > BDT_{cut})} \tag{6.2}$$

It is important that the efficiency defined in this way is only the efficiency of the identification; it doesn't take into consideration the rejected particles due to mismatches, reconstruction, and particles which are not accepted due to the geometry of the CBM experiment (as explained in the section 4.1.).

The minimal purity selected for this work is 90%. For each particle the efficiency and purity depending on the BDT cut can be plotted, such as on Figure 6.10a. The corresponding distribution of each particle for each BDT score is shown on Figure 6.10b



(a) Purity/Efficiency based selection of the BDT cut (red line)



(b) BDT distribution for pions. The red line is the BDT cut from Figure 6.10a

Figure 6.10: Visualization of the BDT selection algorithm

6.3 One variable model

The first tested model in this work is based only on the m^2 variable, to prepare well the package, and understand correctly how the most simple model works.

6.3.1 Validation dataset

In the simplest model, the validation dataset is also tested on the simplest case, so only with the primary particles. The validation dataset is 1M events generated in URQMD this time, which could be replaced with real data once the CBM experiment starts. The aim is to test if the ML model works independently from simulation model, and it could work with the real data.

The "TOF" plots for validation of data are shown below, for protons on Figure 6.11, for kaons on Figure 6.12, and for pions on Figure 6.13.

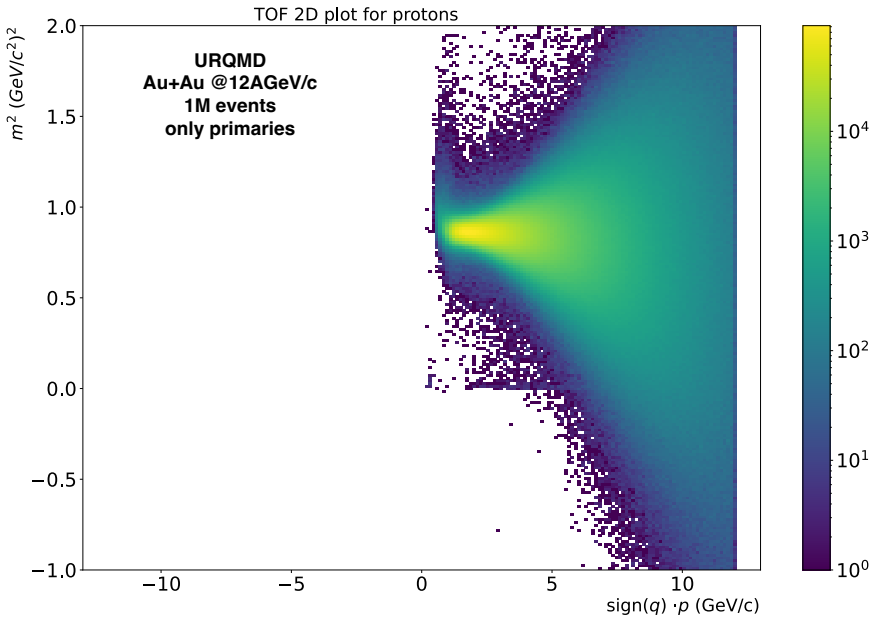


Figure 6.11: Protons used for validation

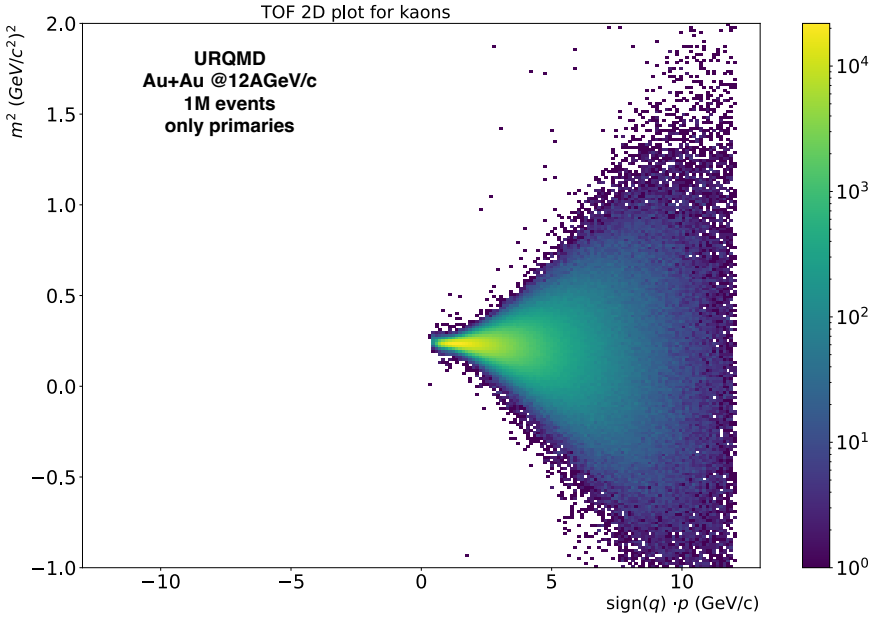


Figure 6.12: Kaons used for validation

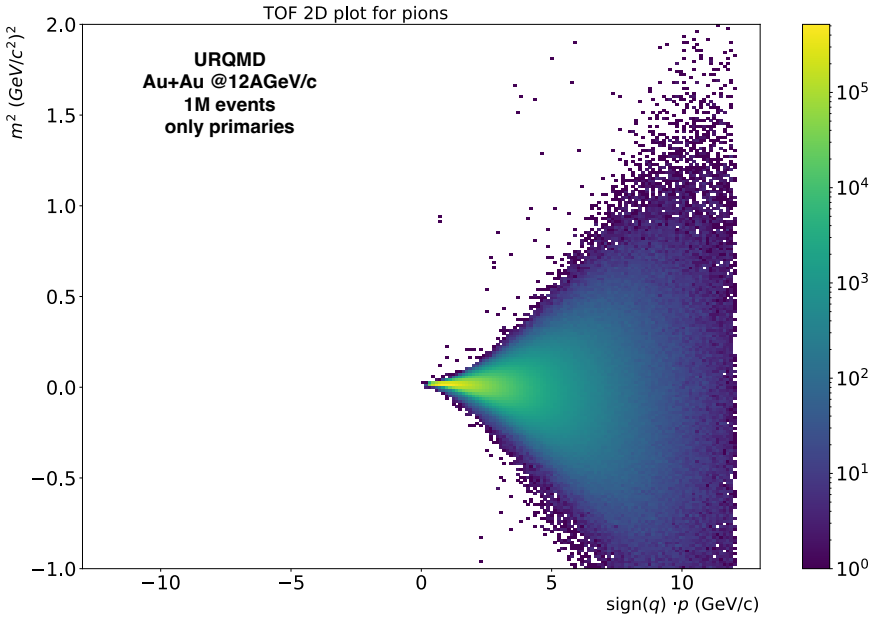
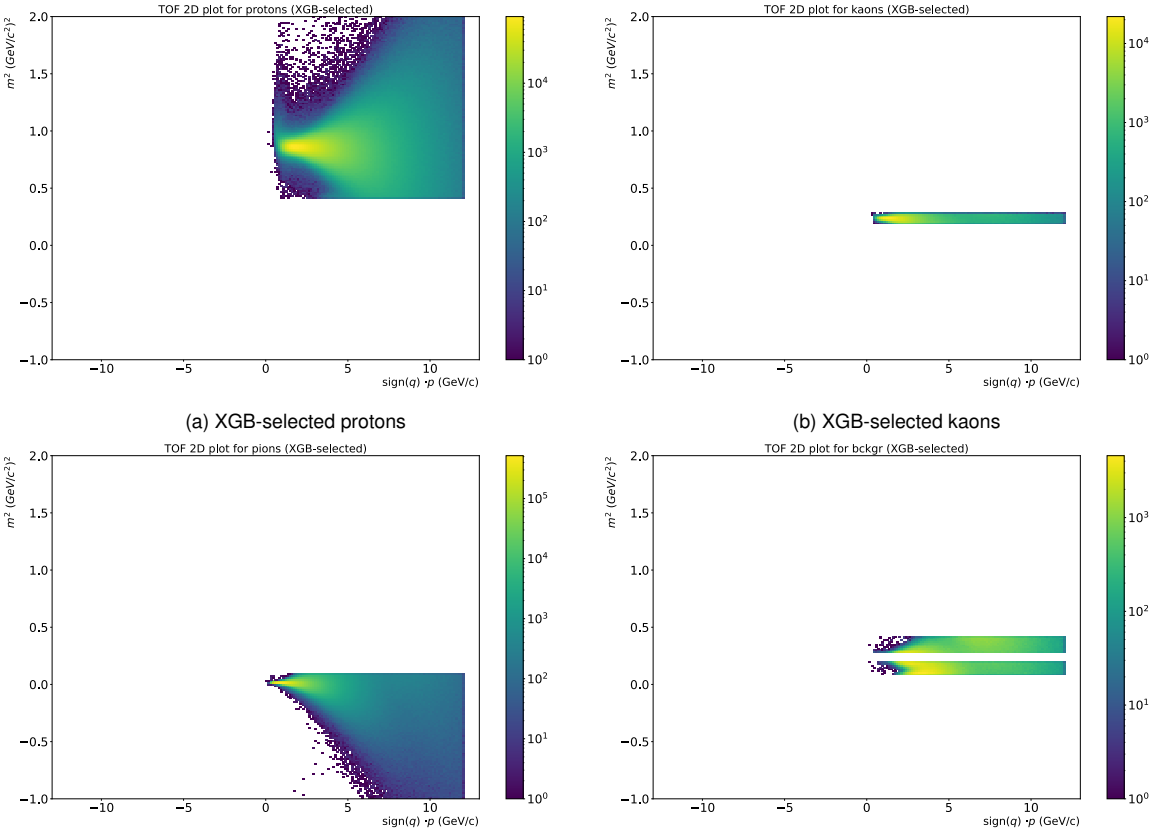


Figure 6.13: Pions used for validation

6.3.2 Single model for all momentum bins

The first test consisted on training the ML model on data from all momentum bins, in the range $p = 0 - 12$ GeV/c. "TOF" plots of XGB-selected particles are shown on Figure 6.14. The table presenting efficiencies and purities for each particle type is shown in Table 6.1

As most reconstructed particles have low momentum, the model select cuts such that work well for particles with lower momenta, rejecting the ones with higher momenta, and



(a) XGB-selected protons

(b) XGB-selected kaons

(c) XGB-selected pions

(d) XGB-selected background

Figure 6.14: XGB-selected particles in the single model

Table 6.1: Efficiency and purity for each particle type for a single ML model

Efficiency and purity of selection for a single ML model					
Protons		Kaons		Pions	
96.5	99.6	61.2	87.3	96.8	97.5
Efficiency	Purity	Efficiency	Purity	Efficiency	Purity

mass-squared away from the mean value. Also the number of rejected particles (bckgr) is quite high. As the aim so to train the model that includes the value of the momentum, but without providing this variable explicitly, dividing the data into momentum bins (as in the traditional "TOF" method) could be performed. It also increases efficiency of the training, as the loaded subsamples take less disk space, decreasing the number of needed RAM memory, and increasing the training speed. During the validation step, the BDT cut selection procedure is also done separately for each bin, thus ensuring that in each bin a given purity value should be achieved.

6.3.3 momentum divided bins

Selection of the momentum values

The momentum bins are chosen according to the number of kaons (the most rare hadron type). Dividng data into four bins with equal number of kaons, selected values were chosen (roudning to the first decimal point):

- $p = 0 - 1.6$ (GeV/c)
- $p = 1.6 - 2.3$ (GeV/c)
- $p = 2.3 - 3.4$ (GeV/c)
- $p = 3.4 - 12$ (GeV/c)

The last selected bin, however, is still very broad. When testing a ML model for these momenta, the same problem was faced as for the single model, that the cuts were selected to work well only with the particles with lower momentum values. The "TOF" plot of selected kaons in the broad bin $p = 3.4 - 12$ (GeV/c) is presented on Figure 6.15.

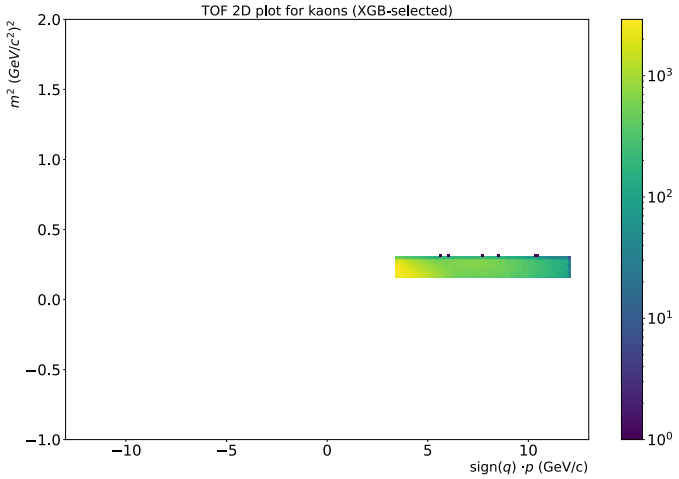


Figure 6.15: XGB-selected kaons in bin $p = 3.4 - 12$ (GeV/c)

By dividing the least bin into 3 bins, $2/3$ of data in this bin has momentum between 3.4-5 GeV/c, and $1/3$ of data has momentum above 5 GeV/c. In the next section the data divided into these 5 bins is used, the distribution of kaons in each bin in the training dataset is presented on Figure 6.16.

The final selected bins are:

- $p = 0 - 1.6$ (GeV/c)
- $p = 1.6 - 2.3$ (GeV/c)
- $p = 2.3 - 3.4$ (GeV/c)
- $p = 3.4 - 5$ (GeV/c)
- $p = 5 - 12$ (GeV/c)

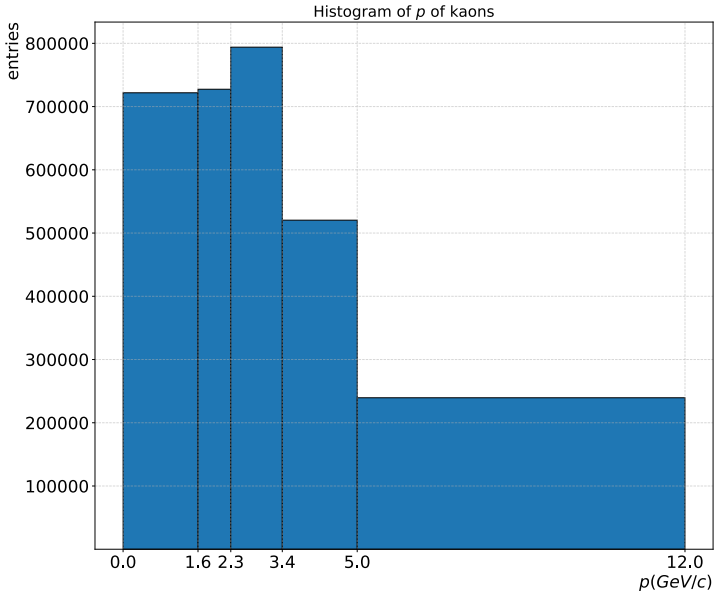
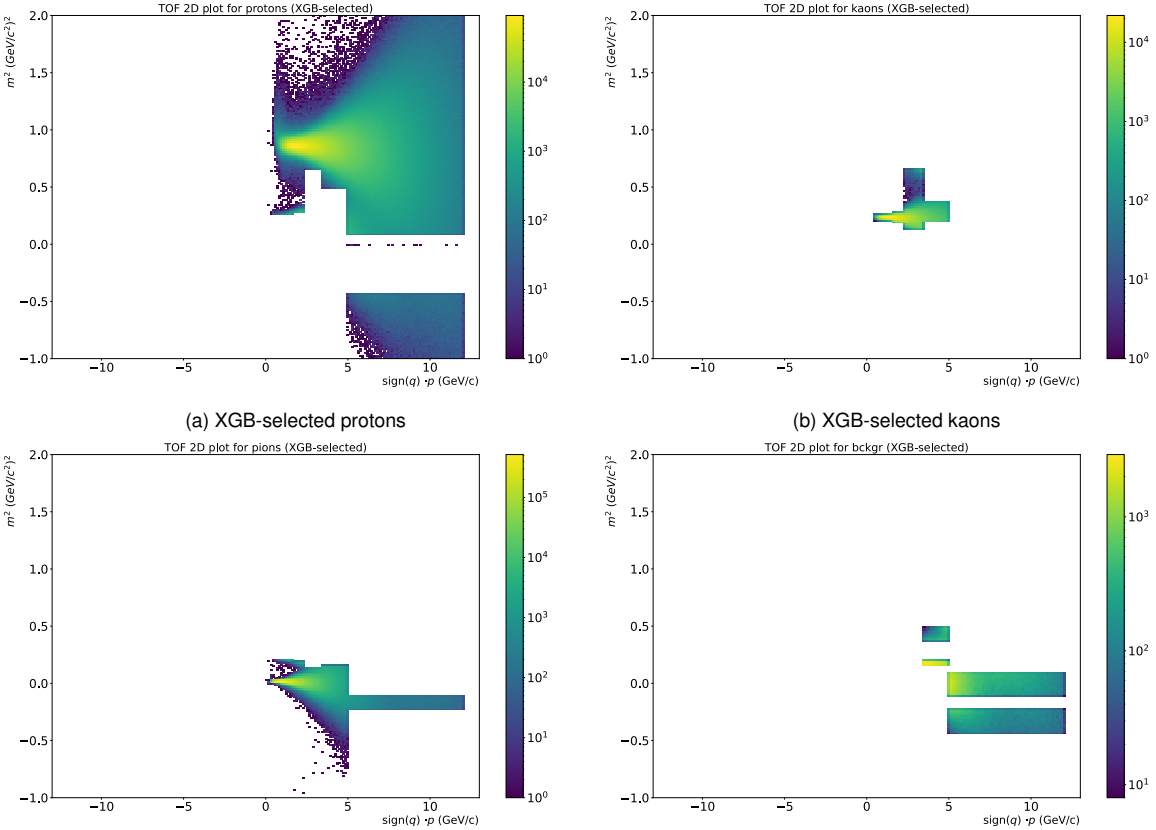


Figure 6.16: Kaons in bins (training dataset)

Results for momentum-divided bins

This time each ML model is trained and validated separately for each momentum bin. The results are then brought together; XGB-selected particles are shown on the Figure 6.17.



(a) XGB-selected protons

(b) XGB-selected kaons

(c) XGB-selected pions

(d) XGB-selected background

Figure 6.17: XGB-selected particles in 5 bins

Efficiency and purity for each class (for all momentum bins together) is also collected in the Table 6.2.

The results are significantly better than for the single model, but there is room for improvement. It is especially visible in the "TOF" distributions, where selection "cuts" are sharp and don't take into consideration the overlap between the tails of the distributions of mass-squared.

Table 6.2: Efficiency and purity for each particle type for a 5 momentum-divided ML models

Efficiency and purity of selection for 5 momentum-divided ML models					
Protons		Kaons		Pions	
99,0	98.0	71.6	97.2	95.9	98.6
Efficiency	Purity	Efficiency	Purity	Efficiency	Purity

6.4 Multiple variables model

The main advantage of the ML particle classification is the ability to use multiple variables. It creates a counterpart of Gaussian multi-dimensional fitting, but is done automatically and fast. In this section the selection of variables for training, and comparison between different weights selection methods, is performed.

6.4.1 Selection of variables

The criteria for selection of variables for training is to provide the model with new features, hence new information which could be useful for classification of particles. Also, there's no need to use variables which are highly correlated with the ones already used, as they do not provide much additional information. The correlation matrix was created for this purpose (shown on Figure 6.18). The correlation is defined as the Pearson correlation coefficient (which shows linear correlation) of each variable; it is calculated following this formula:

$$\rho = \frac{\mathrm{COV}(X,Y)}{\sigma_X \times \sigma_Y} \tag{6.3}$$

where the covariance of both variables is defined as:

$$\mathrm{COV(X,Y)} = \mathrm{E}\left[(X - \mathrm{E}\left[X\right])\left(Y - \mathrm{E}\left[Y\right]\right)\right] \tag{6.4}$$

and the standard deviation of a variable is defined as:

$$\sigma_X = \sqrt{\mathrm{E}\left[\left(X - \mathrm{E}\left[X\right]\right)^2\right]} \tag{6.5}$$

As the mass-squared variable is already chosen, and the variables correlated with momentum should also be omitted, the following five variables are selected: m^2, η, ϕ, **vtx_chi2, chi2_ov_ndf_vtx**.

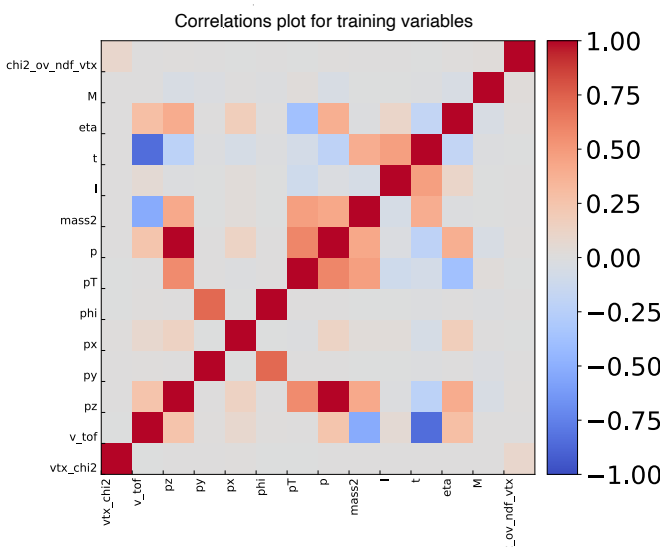


Figure 6.18: Correlations plot

6.4.2 "Balanced" vs. uniform weights

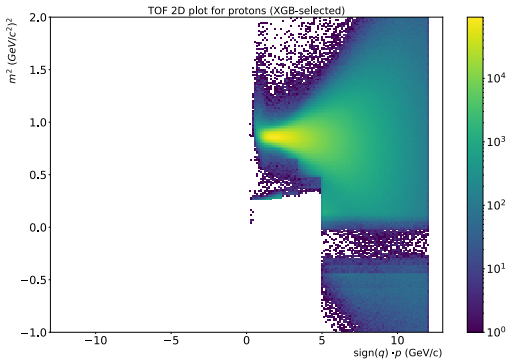
The selected particles for five variables model, in five momentum-divided bins are shown on Figure 6.19 for uniform weights, and on Figure 6.20 for "balanced" weights, as defined in Formula 5.1. The comparison of efficiency and purity of selected particles in both methods is shown in Table 6.3.

The introduction of "balanced" weights has cerrtain effects on the identification of particles. While these weights slightly improve the efficiency of identifying kaons, they lead to a decrease in the purity of the selected sample. Additionally, the efficiency of identifying protons and pions is also reduced.

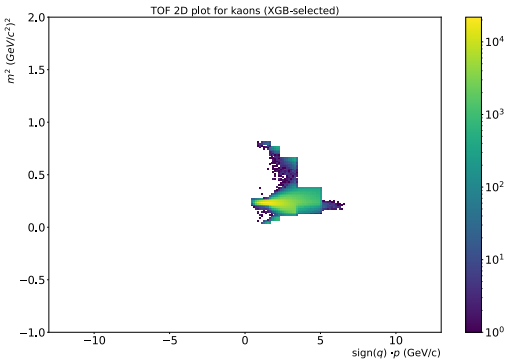
In particular, at higher momenta, the number of pions is much lower compared to protons. Consequently, the weights assigned to pions are significantly higher than those for protons. In result, protons with high momentum and low mass-squared values are not selected. This region overlaps with the region where pions are also present, resulting in neither protons nor pions being selected. As a result, this region is mostly classified as background, as illustrated on Figure 6.20d.

A potential solution would be to adjust the weights assigned to different particle types based on their relative abundances. By lowering the weight assigned to a particle type that is significantly less abundant, the differences between the weights would be less significant. However, developing an optimal weight selection algorithm is not a trivial task.

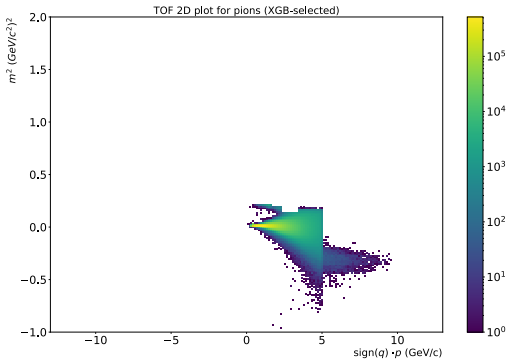
To simplify the process, and achieve reasonably good results, uniform weights are used in this work.



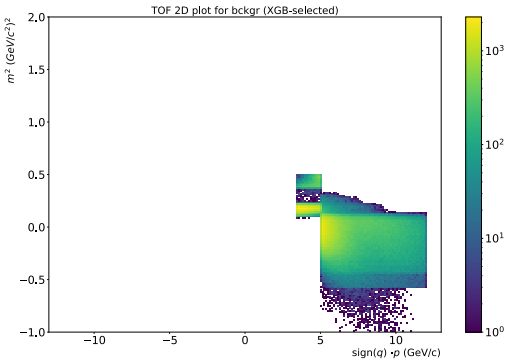
(a) XGB-selected protons



(b) XGB-selected kaons

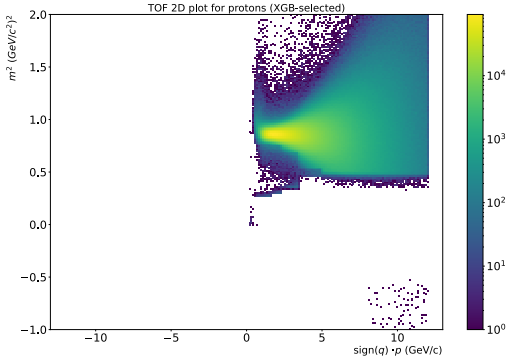


(c) XGB-selected pions

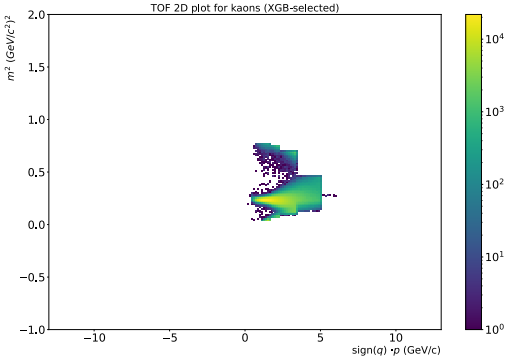


(d) XGB-selected background

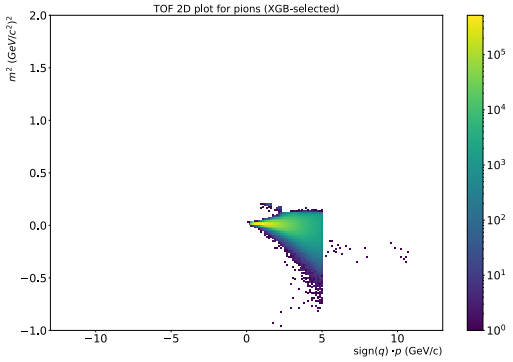
Figure 6.19: XGB-selected particles for 5 variables and uniform weights



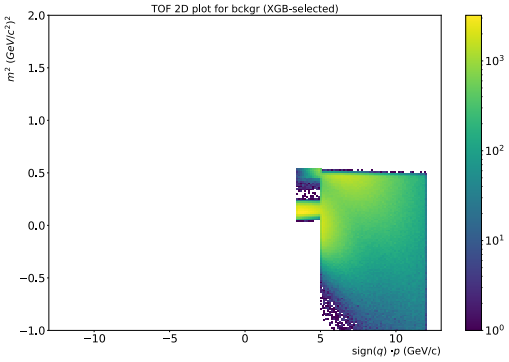
(a) XGB-selected protons



(b) XGB-selected kaons



(c) XGB-selected pions



(d) XGB-selected background

Figure 6.20: XGB-selected particles for 5 variables and "balanced" weights

Table 6.3: Comparison between efficiency and purity for uniform, and "balanced" weights model

Efficiency and purity of selection for five variables ML model in all momentum bins						
	protons		kaons		pions	
uniform weights	99.0	98.2	72.7	96.5	95.0	99,1
"balanced" weights	95.3	99.8	75.4	93.0	94.1	99,6
	efficiency	purity	efficiency	purity	efficiency	purity

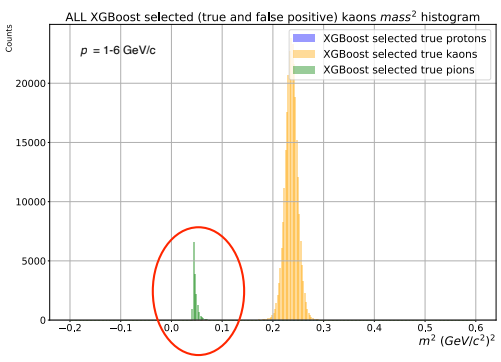
6.4.3 Misidentified particles in the tails of mass-squared distributions

Another issue that was investigated in this step of creating the ML model for particle identification, was misidentification of particles in the tails of mass-squared distributions. This challenge is also present in the Gaussian fitting procedure, but in ML-based approach it could be more easily solved.

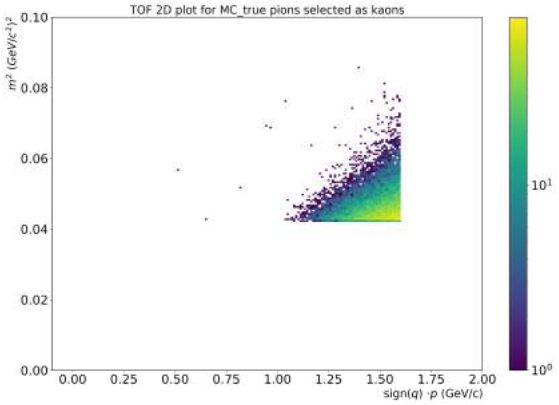
Notably, it could easily be solved for the particles with low momenta, i.e. in the first momentum bin. It is observed in both models with uniform, and "balanced" weights. It is more visible in the second approach, and ML model for the bin $p = 0 - 1.6(GeV/c)$ with "balanced" weight; for this reason results for this model are used for visualisation of this issue in this section.

On Figure 6.21a XGB-selected kaons in the first momentum bin are shown. MC-true pions are circled in red, a "TOF" plot of these falsely identified pions is shown on Figure 6.21.

For the higher momenta, particles which "overlap" are more common; they are less present for the lower momentum values. In effect, the model is more likely to classify them incorrectly.



(a) Mass-squared histogram of XGB-selected kaons. MC-true pions circled in red



(b) "TOF" plot of MC-true pions selected as Kaons

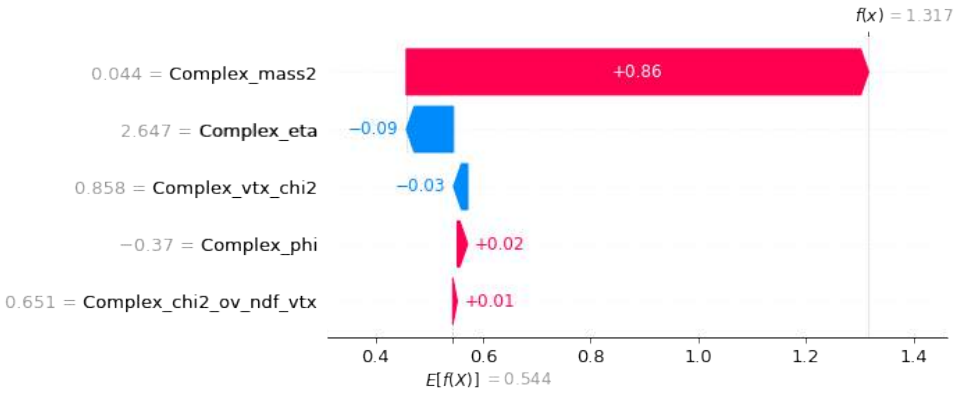
Figure 6.21: Pions mismatched as pions in the first momemtum bin.

This hypothesis can be checked using the SHAP waterfall plots, for randomly selected particles in this region.

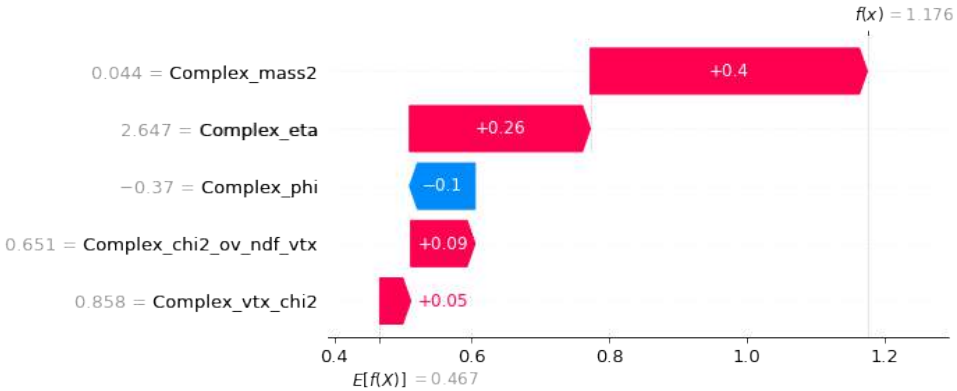
Fore example, SHAP score for a pion identified as a kaon (the higher the SHAP score, the higher the probability that a particle belongs to a certain class) can be shown this way. On figure 6.22a the SHAP score of identification of this particle as a kaon is shown, along with the

input from each feature of the model (exact values of each variable the model is trained on). Figure 6.22b presents the same waterfall plot for identification as a pion.

The overall SHAP values for both cases are close, but the one (false one) that this particle is a kaon is higher. As the decision is mainly based on the mass-squared value, it is really difficult to remove these mismatches using only the five selected variables.



(a) SHAP score for classification as kaon



(b) SHAP score for classification as pion

Figure 6.22: Waterfall shap plot for single misidentified pion. The similar shap values results in probability of being kaon equals 54%, of being pion equals 46%.

While investigating the mass-squared distribution of all particles in this momentum bin (shown on Figure 6.23), it is observed that the particles present in the overlaps between the distribution are quite rare. One solution would be to simply increase the size of the dataset, but it is not an ideal solution.

Another one would be to e.g., add another variable, v_{tof}, which has a different distribution than mass-squared (as presented on Figure 6.24). The results for the model trained using this additional variable are presented in the next section.

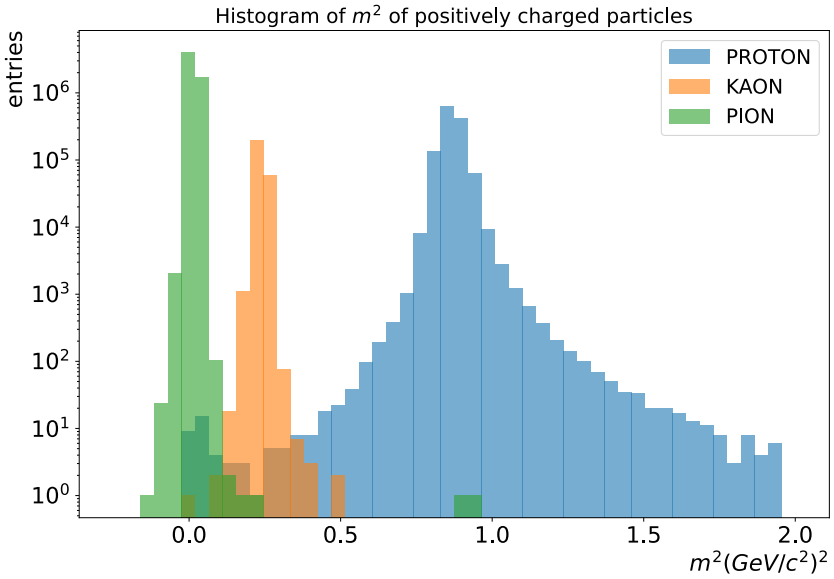


Figure 6.23: Histogram of mass-squared in the first momentum bin. Note a low number of entries in the tails of distributions of each particle type.

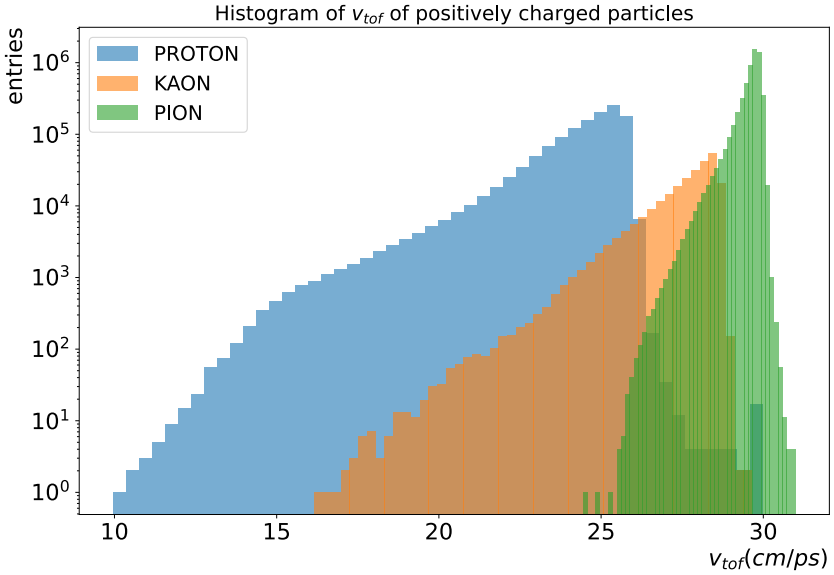


Figure 6.24: Histogram of v_{tof} in the first momentum bin.

6.5 Final "TOF" ML model

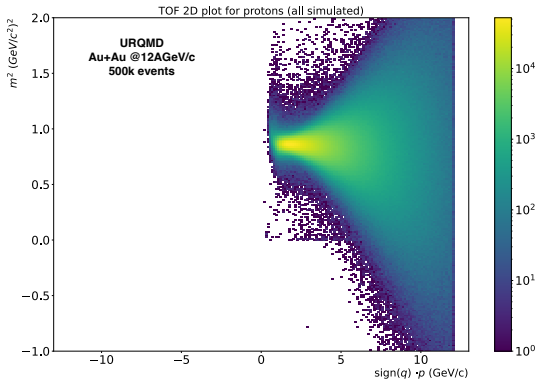
The final model trained on data from tracking, and "TOF" detectors uses the following variables: m^2, v_{tof}, η, ϕ, **vtx_chi2, chi2_ov_ndf_vtx**.

In the following subsections, each element of the created package is presented and described.

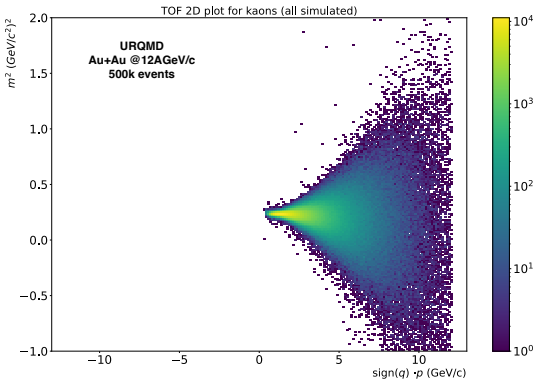
6.5.1 Validation dataset

The training dataset is the same as in previous steps, but the validation is performed using all particles, not only primaries (with the same preselection cuts as before).

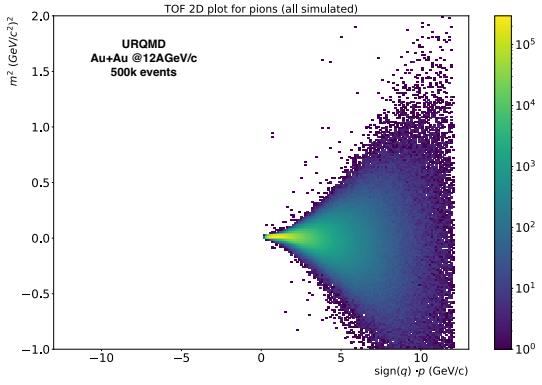
As including not only primary particles increases the size of the dataset, validation is performed using not 1M, but 500k events. The "TOF" plots of validation dataset is shown on Figure 6.25.



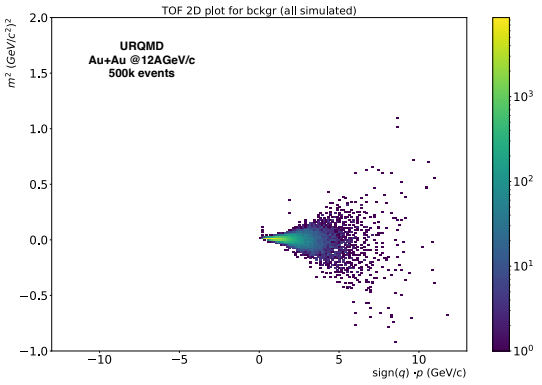
(a) Protons used for validation



(b) Kaons used for validation



(c) Pions used for validation



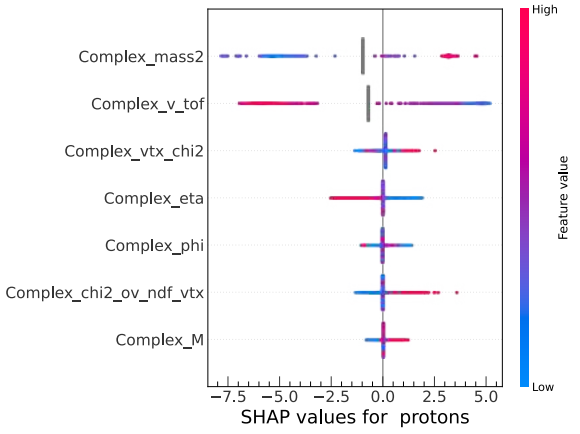
(d) Background used for validation (MC-true negative muons and electrons)

Figure 6.25: Validation Particle Selections

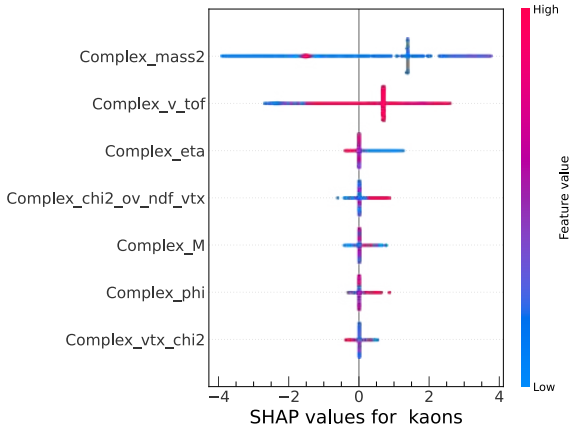
6.5.2 SHAP plots

Preparing SHAP plots is an important step in creating "interpretable" ML model, and thus an important part of the created package.

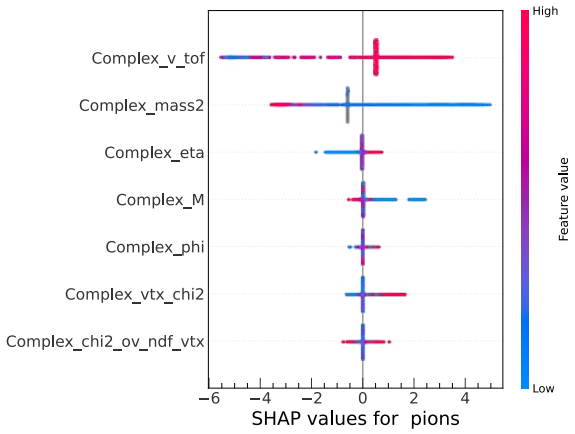
For each momentum bin a seperate ML model is created, so SHAP plots also differ from bin to bin. The innspection of SHAP plots allows to better understand the decisions taken by each model, for particle type. As an example, SHAP summary plots created for momentum bin $p = 2.3 - 3.4$ (GeV/c) are shown on Figure 6.26. A closer look is possible using scatter plots. For example, the relations between mass-squared, multiplicity, and presudorapidity, and their impact on identification of kaons are presented on Figure 6.4).



(a) Summary SHAP plot for protons

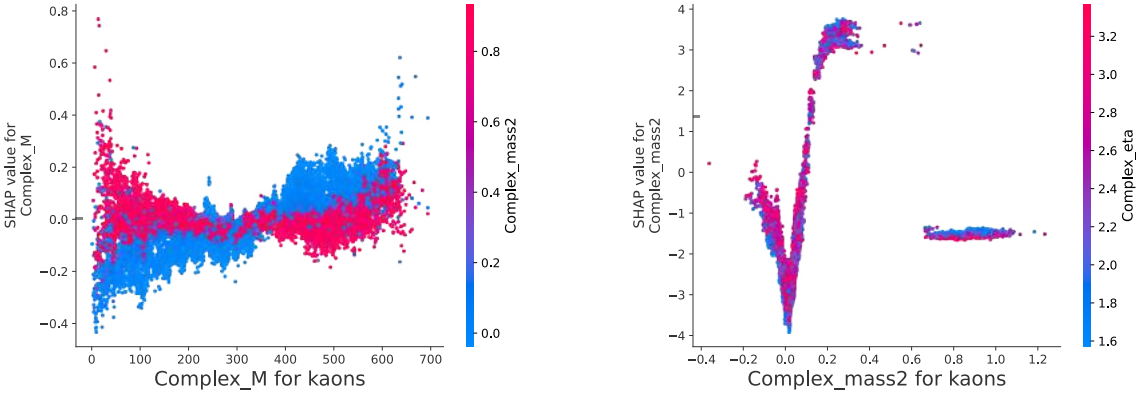


(b) Summary SHAP plot for kaons



(c) Summary SHAP plot for pions

Figure 6.26: Summary SHAP plots in momentum bin $p = 2.3 - 3.4$ (GeV/c)



(a) Scatter plot for M and mass-squared for kaons

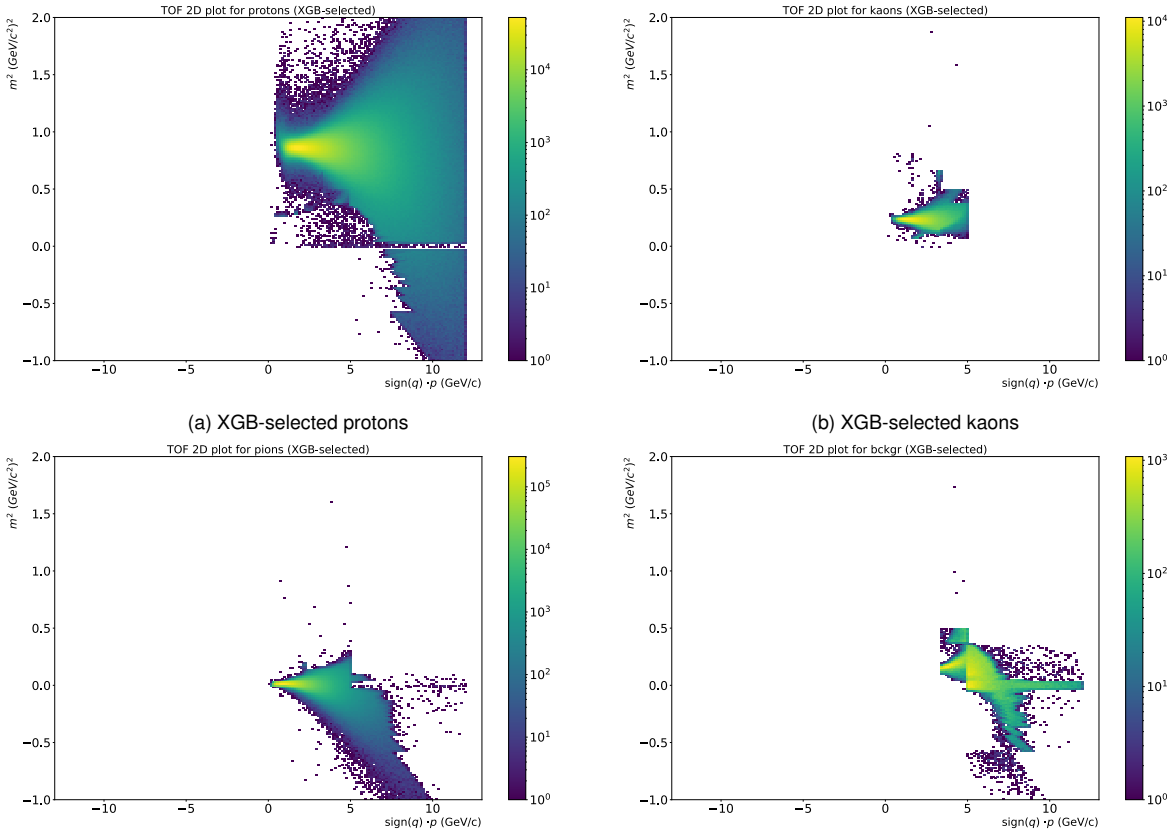
(b) Scatter plot for mass-squared and η for kaons

Figure 6.27: Scatter plots for kaons in momentum bin $p = 2.3 - 3.4$ (GeV/c)

6.5.3 Results

As in previous sections, the results can be presented using "TOF" plots (Figure 6.28), and table with efficiencies and purities (Table 6.4).

The increase of efficiency for pions, and kaons, with a slight decrease of purity can be observed. Also, the misidentified particles in the tails of distribution in the lowest momentum is also visible on the "TOF" plot.



(a) XGB-selected protons

(b) XGB-selected kaons

(c) XGB-selected pions

(d) XGB-selected background

Figure 6.28: XGB-selected particles in the final "TOF" model

Table 6.4: Efficiency and purity for each particle type for the final "TOF" model

Efficiency and purity for each particle type for the final "TOF" model					
Protons		Kaons		Pions	
99,7	98.8	75.1	97.1	97.2	97.9
Efficiency	Purity	Efficiency	Purity	Efficiency	Purity

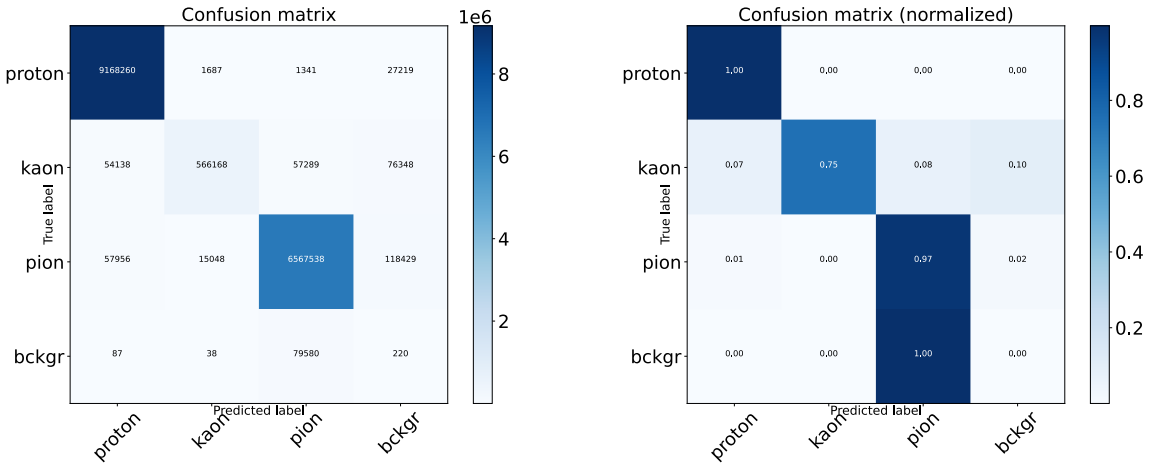
6.5.4 Analysis of the results

More in-depth representation and analysis of the results is possible using other plots created in the package.

Confusion matrix

The confusion matrix is a useful tool for evaluating the accuracy of a classifier. In binary classification, it provides information about the classification results by showing the counts of true negatives ($C_{0,0}$), false negatives ($C_{1,0}$), true positives ($C_{1,1}$), and false positives ($C_{0,1}$) [49].

To visualize the confusion matrices of created models, they are plotted on Figures 6.29a and 6.29b. Figure 6.29a displays the confusion matrix without normalization, showing the actual number of observations in each category. On the other hand, Figure 6.29b presents the normalized confusion matrix, where the values are expressed as percentages, showing the efficiency of classification, as defined in this chapter.



(a) Confusion matrix

(b) Confusion matrix (normalized)

Figure 6.29: Confusion matrix for results from all bins

ROC plots

Receiver Operating Characteristic (ROC) illustrates the diagnostic ability of a binary classifier. Threshold on the ROC (Receiver Operating Characteristic) curve which maximizes

Approximate Median Significance

$$\text{AMS} = \sqrt{2}[(tpr + fpr)\log(1 + tpr/fpr) - tpr] \tag{6.6}$$

(where $t(f)pr$ is true (false) positive rate) on the test sample is the best threshold. In multi-class approach it compares the ability to distinguish between each combination of particle types, as presented on Figure 6.30.

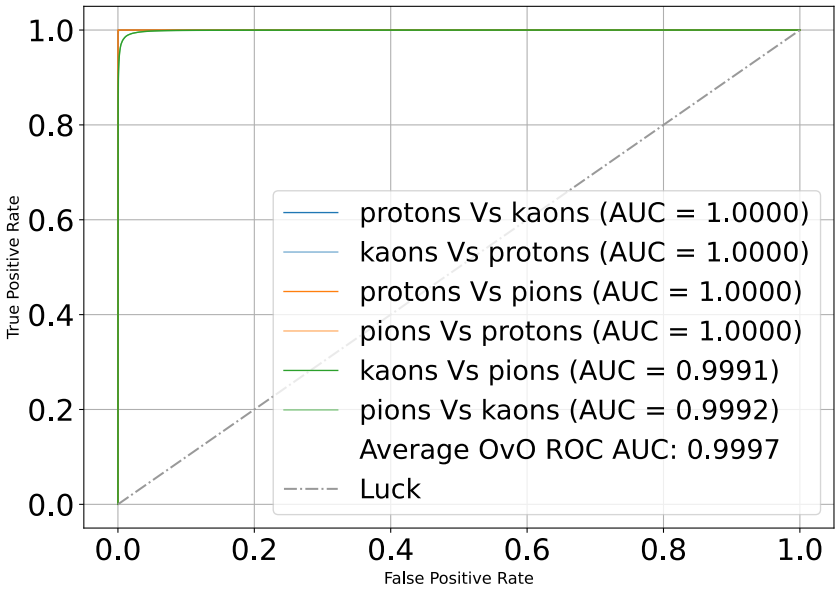


Figure 6.30: ROC plot for momentum bin $p = 2.3 - 3.4$ (GeV/c)

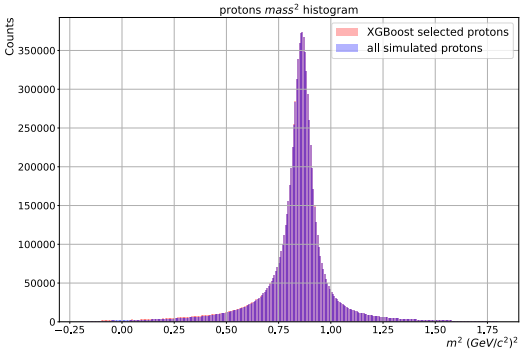
It helps understanding if the model works correctly, and which particles types are most often mistaken.

Mass-squared histograms

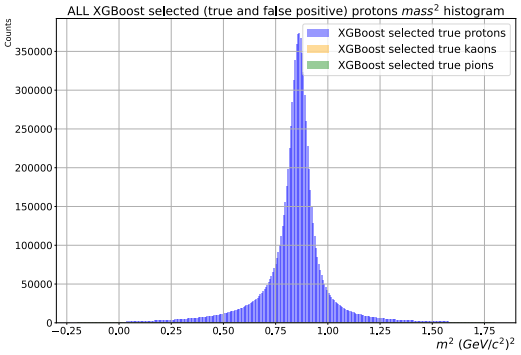
Mass-squared histograms are an intuitive tool in analysis of the results. The type of them are created in the package, and shown in this section.

The first one is the XGBoost selected particle type vs. all simulated particles of this type, shown consequently for protons, kaons, and pions on Figures 6.31a-6.33a. This type allows to check which in mass-squared region particles are not classified, or are too often classified as a given particle type.

Another type is MC-true particles type selected as consequently protons, kaons, and pions on Figures 6.31b-6.33b. This type allows to i.e. check which mass-squared values particles are mismatched, and what the sources of the impurities are.

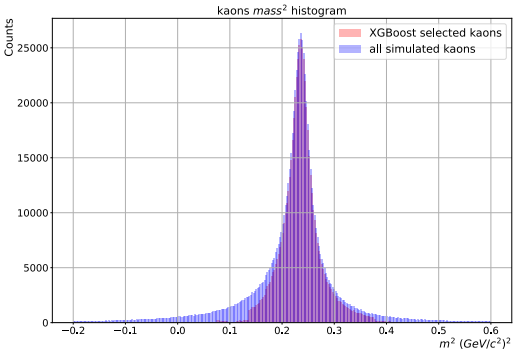


(a) XGBoost-seleceted vs all simulated protons

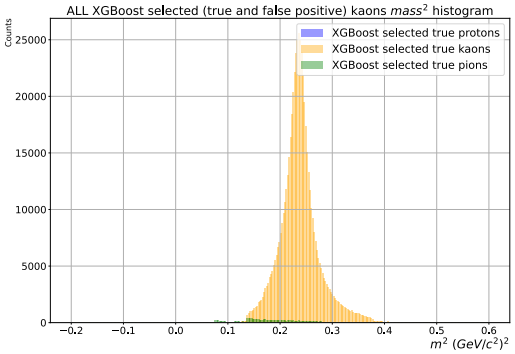


(b) MC-true particles selected as protons

Figure 6.31: Mass-squared histogram of selected protons

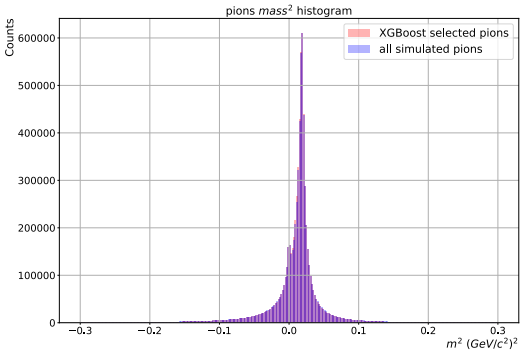


(a) XGBoost-seleceted vs all simulated kaons. Low number of identified kaons with low mass-squared value is visible.

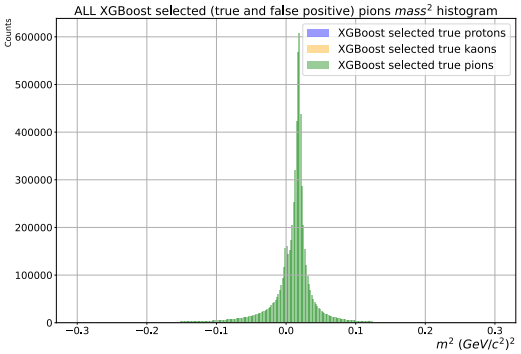


(b) MC-true particles selected as kaons. MC-true pions are visible (in green).

Figure 6.32: Mass-squared histogram of selected kaons



(a) XGBoost-seleceted vs all simulated pions



(b) MC-true particles selected as pions

Figure 6.33: Mass-squared histogram of selected pions

p_T-rapidity graphs

2D histograms showing p_T, and rapidity are often used in high-energy physics. They are also created in this package, showing efficiency of identification, and total number of particles in each histogram bin, for each particle type. An example of such plots for all momentum bins

together is shown on Figure 6.34. They help understanding how the geometry of the event influences the identification efficiency.

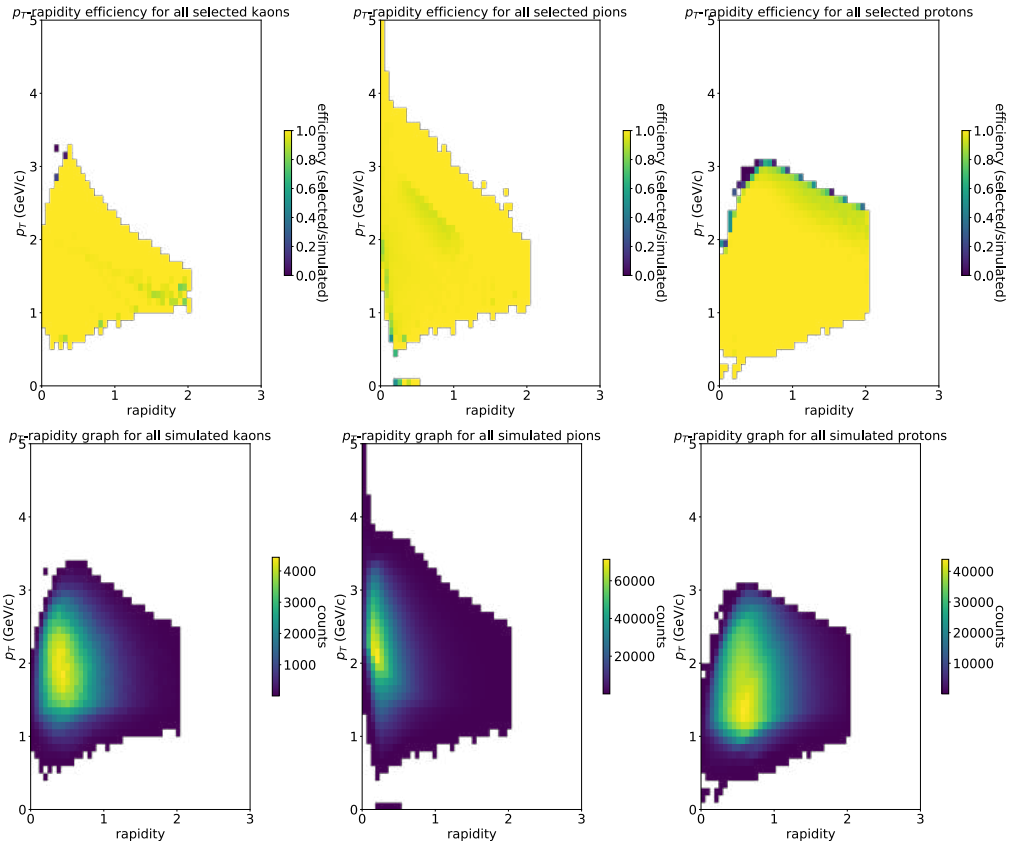


Figure 6.34: pT-rapidity graphs for all particles types

Comparison between ML and traditional identification

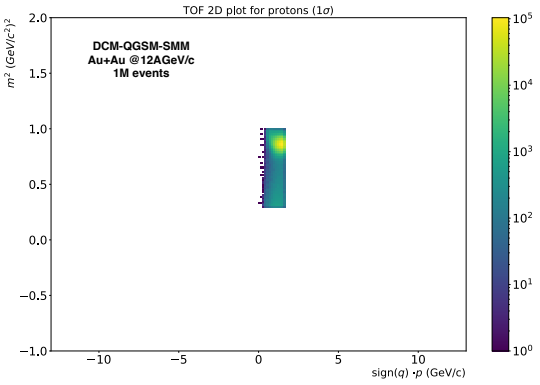
In this chapter, the comparison between Gaussian fitting (traditional identification method) and ML is performed.

The results for the traditional method were obtained using data in *AnalysisTree* format, with already identfied particles (using Gaussian fitting package [50]), was performed. As this package also returns information about the probability of a specific particle belonging to a given class, the same probability cut procedure was performed as for the ML model results. The validation dataset (and traditional particle identification results) consists of 500k events generated in URQMD, without removing mismatches.

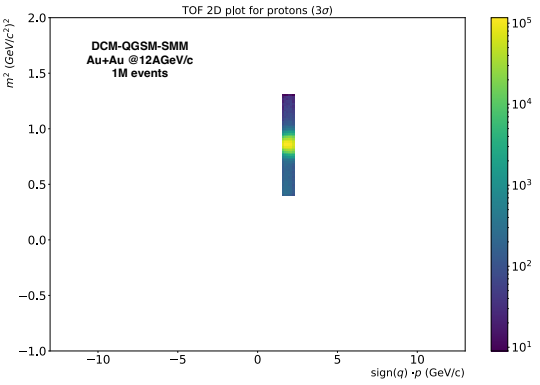
7.1 Training dataset

As the mismatches weren't removed in the available dataset with the traditional identification method applied, the models are trained on 1M events generated in DCM-QGSM-SMM model, also with mismatches present.

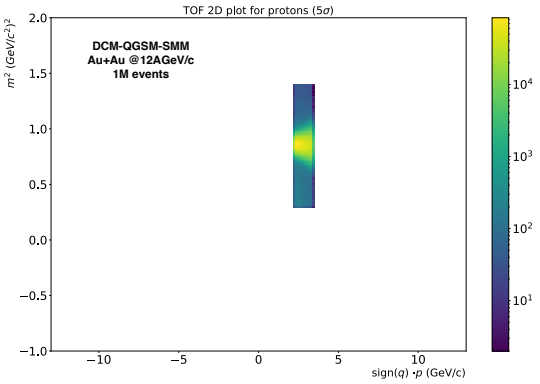
However, to omit training the model on clearly invalid data (e.g., protons with mass-squared value close to zero), $n - \sigma$ selection of the mass squared values was performed for each momentum bin. The value of n is selected individually for each particle type, and each bin. The example of selected protons training dataset is shown on Figure 7.1



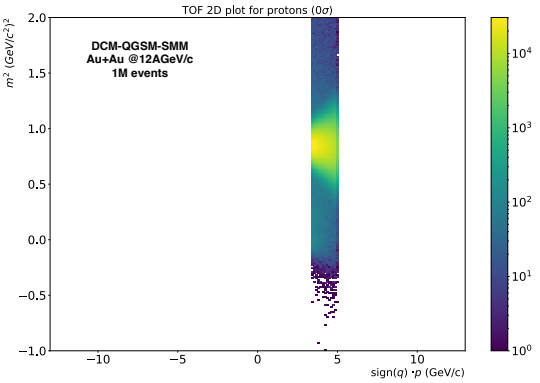
(a) Selected protons in 1σ region in bin $p = 0 - 1.6$ GeV/c.



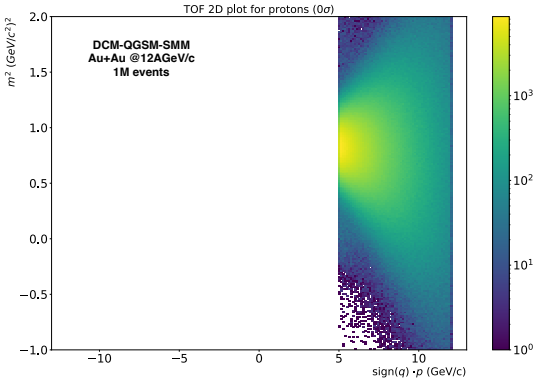
(b) Selected protons in 3σ region in bin $p = 1.6 - 2.3$ GeV/c



(c) Selected protons in 5σ region in bin $p = 2.3 - 3.4$ GeV/c



(d) Selected protons without σ selection (called "0 sigma" in the plot) in bin $p = 3.4 - 5$ GeV/c



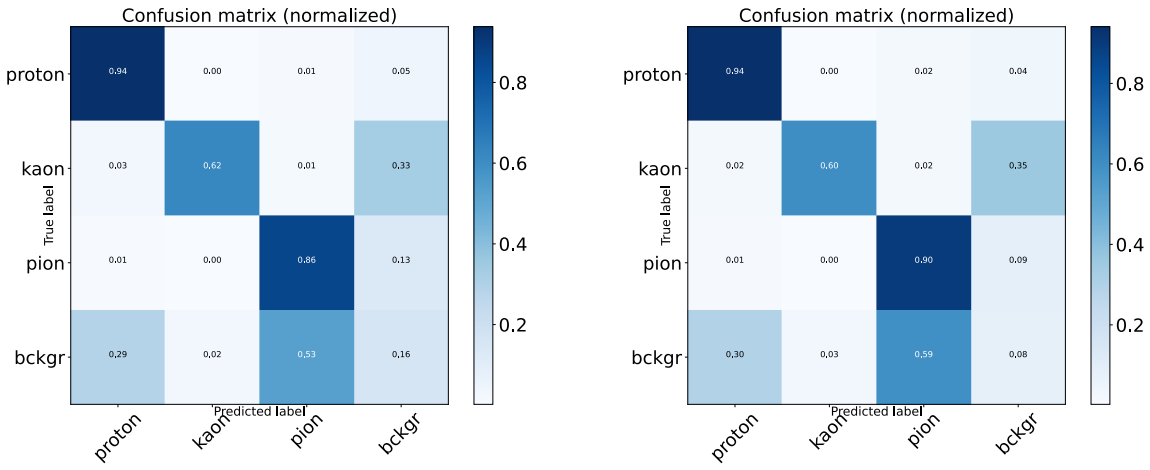
(e) Selected protons without σ selection (called "0 sigma" in the plot) in bin $p = 5 - 12$ GeV/c

Figure 7.1: Selected protons in the training dataset

7.2 Results

The results of both methods are shown on confusion matrices (Figure 7.2).

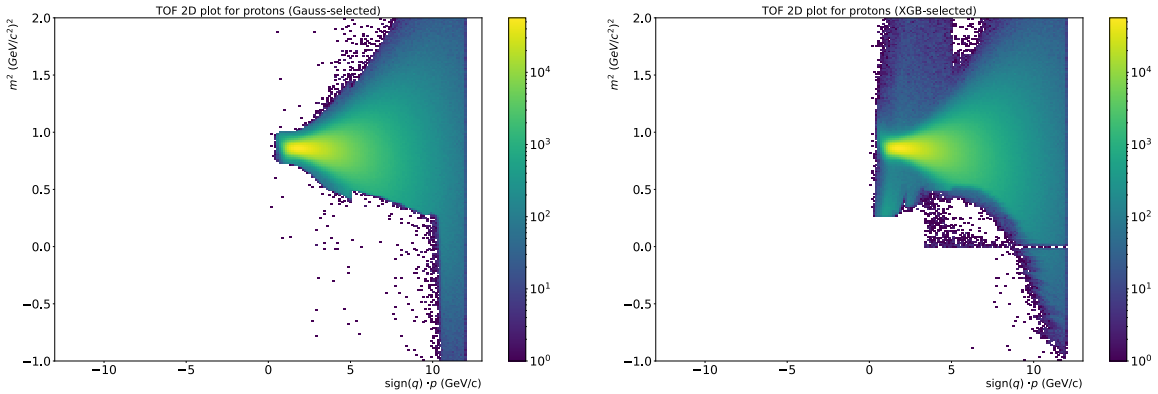
The "TOF" plots for traditionally selected particles, and XGBoost-selected are shown for protons, kaons, and pions, consequently on Figures 7.3-7.5.



(a) Gaussian fitting

(b) XGBoost-selected

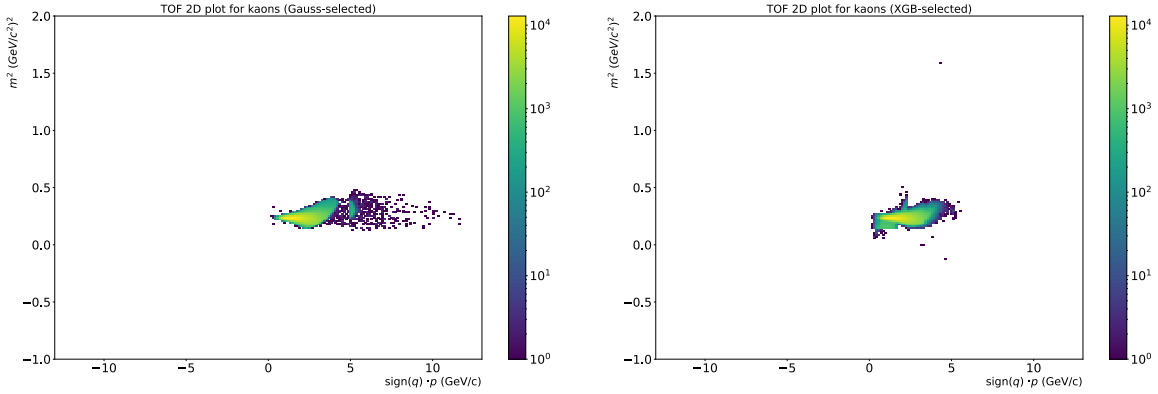
Figure 7.2: Confusion matrices (normalized) for results from all bins



(a) Gaussian fitting

(b) XGBoost-selected

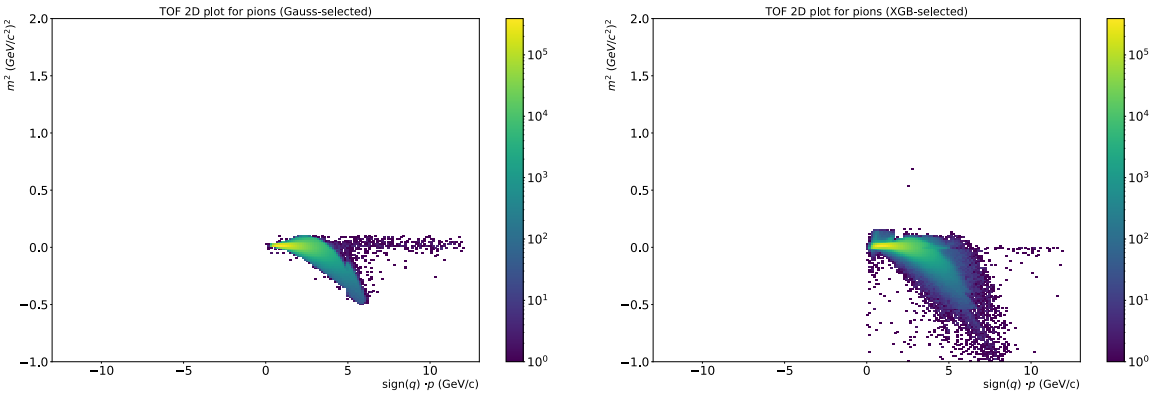
Figure 7.3: "TOF" plots for selected protons



(a) Gaussian fitting

(b) XGBoost-selected

Figure 7.4: "TOF" plots for selected kaons



(a) Gaussian fitting

(b) XGBoost-selected

Figure 7.5: "TOF" plots for selected pions

7.3 Comparison

An in-depth comparison between efficiency, and purity for ML-selected particles, Gaussian selected particles, and ML-selected particles with mismatches is performed for each momentum bin, and particle type seperately.

The results are shown on Figures 7.6-7.8. It is important to note again that the total number of particles with removed mismatches is different, as explained in section 6.1.3; the efficiencies presented for the ML approach with removed mismatches don't take this into consideration while calculating the efficiency.

It is complicated what the MC-true particle type of a mismatched particle should be decided as, but it is worth noting that the total number of particles which could be identified differs in the first two approaches, and in the third one.

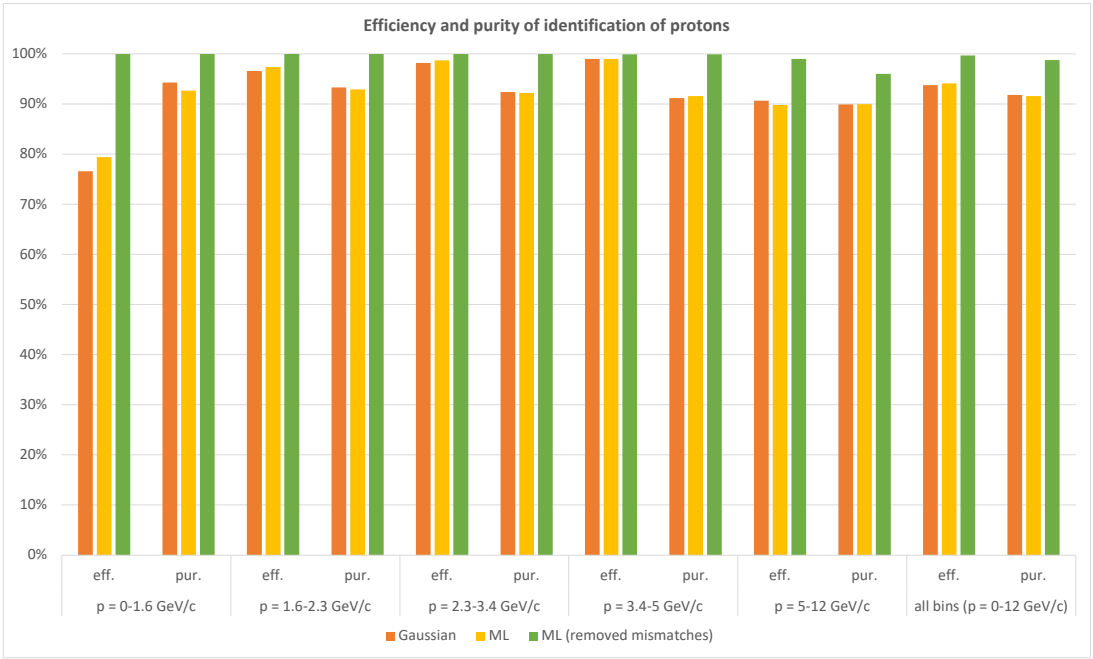


Figure 7.6: Efficiency and purity of identified protons

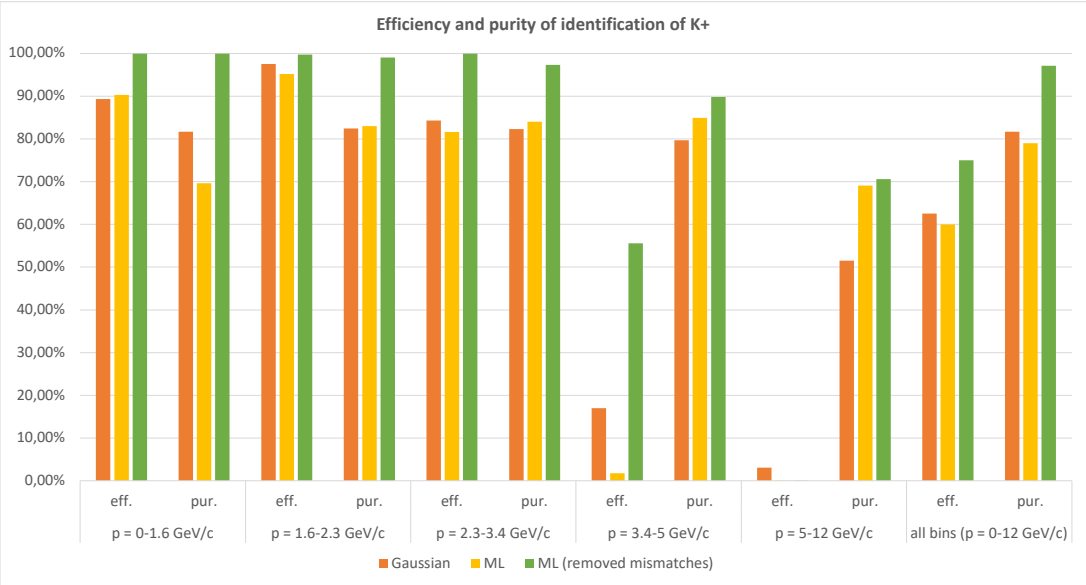


Figure 7.7: Efficiency and purity of identified positive kaons

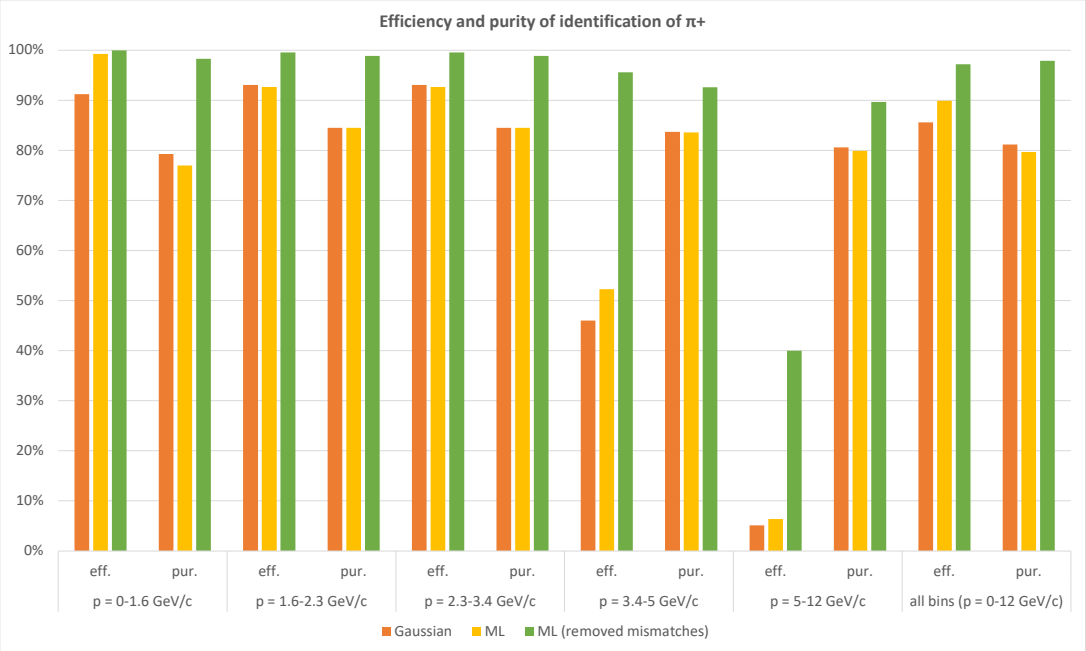


Figure 7.8: Efficiency and purity of identified positive pions

Discussion and summary

In this thesis, the investigation of the use of ML techniques for particle identification in the CBM experiment was performed.

Created "converter" package allows for transforming the data from the internal *AnalysisTree* format, into *PlainTree* format, which can be then easily loaded into Python. Data coming from STS+MVD, and TOF detectors was investigated in this work, but the created package can be easily adapted (as it was tested outside the scope of this thesis) to include data from other detectors.

The created ML package for particle identification, the main part of this thesis, could easily be used for further investigation of the optimal configuration, and setups. Selected data was also narrowed down to particles which are not a result of a mismatch between the tracking detectors, TOF detector, and simulation. It allowed to obtain much cleaner data sample, resulting in much better results, as shown in the Chapter 7.

It was shown that dividing data into multiple momentum-divided bins increases the efficiency and purity of the identification of hadrons (positive protons, kaons, and pions), as shown in Tables 6.1-6.2. It also decreases the needed memory, allowing to increase the efficiency of training of the models, and using data from more events, hence improving the classification. Another advantage is that the minimal purity set by the user is selected for each momentum bin seperately, resulting in better quality of selected data for further analysis.

Selection of classes weights was also discussed in Table 6.3. The uniform approach gives higher purity of the least represtended hadrons, kaons, and higher efficiency of identification of protons and pions. The "balanced" approach, which treats each particle type as if their number was equal, is not optimal. However, an algorithm which is a compromise between the two approaches could be developed in the future, and its results easily tested with the created package.

The interpretability of a ML model is one of the hot topics in AI, and a necessary one when it comes to implementing the ML classifiers in real-world experiments such as CBM. Created package provides the user with multiple types of plots, which can help understand the model's decision, better than simple "TOF" plots, such as mass-squared plots, presented on Figures 6.31-6.33. It also implements SHAP plots (created using an efficient *fasttreeshap* package), a popular tool in ML community. For example, they help understanding the source of misidentification of particles in the tails of mass-squared distributions for the lowest momenta,

as presented in the Section 6.4.3. Furthemore, diving the data into momentum bins also allows for better interpretation of the decisions made by ML, as they are narrowed down to smaller momentum range for each created ML model.

A comparison with the traditional "TOF" method using Gaussian fitting was performed in Chapter 7. A quite complicated and time-consuming process of selecting optimal fit parameters can be omitted by using ML, which results in comparable results, if the mismatches are not removed. On the other hand, removing mismatches, and using ML increases the efficiency and purity of selected particles distinctly, as shown on Figures 7.6-7.8. If the model is trained, and validated on data without mismatches, it often selects particles in the tails of the mass-squared distribution as well (shown on Figure 6.28), which is not possible with traditional Gaussian fitting. Creating an improved reconstruction algorithm should be a priority now, especially as the ML learns by analyzing examples; if not trained on correct data, the appearance of incorrect results is likely.

In the future, the development of the ML model which includes data from RICH, and TRD detectors could be interesting to perform, especially as it is possible to implement with the existing software. Possible training variables were presented in [28]; this work focuses solely on the identification of electrons, but a multi-class ML model's performance could be increased using data from all available detectors too. Another challenges, such as dealing with missing data would arise, though; possible solutions were explained in the Section 5.2.4. The implementation of ML models (mostly in Python) in the experiment's software (written in C++) can be done using the ONNX ecosystem, as discussed in [40]. This is one of the advantages of using DT-tree based package (e.g., XGBoost presented in this wokr), as their functioning and implementation is quite straightforward.

To summarize, using machine learning for particle identification in the CBM experiment is a promising solution. In this thesis, it was tested on the three most abundant hadrons, but the created software could easily be adapted for identification of other particles, such as electrons, using data from more detectors. Nonetheless, dealing with reconstruction mismatches emerges as an important area of focus for achieving accurate and reliable results. With skyrocketing popularity of AI-based tools, ML cannot be (and has never been) overlooked in the heavy-ion experiments, but elements such as, notably, interoperability, and uncertainty-awareness must be further developed.

Bibliography

[1] openai.com/blog/chatgpt, accessed 21.06.2023

[2] P. Calafiura, D. Rousseau, and K. Terao, Artificial intelligence for high energy physics. Singapore, Singapore: World Scientific Publishing, 2022.

[3] S. Khan et al., “Machine Learning application for hyperon reconstruction in CBM at FAIR,” EPJ Web Conf., vol. 259, p. 13008, 2022.

[4] Hanna Paulina Zbroszczyk. ”Eksperymentalne aspekty badania korelacji femtoskopowych w zderzeniach relatywistycznych ciezkich jonów”. Oficyna Wydawnicza Politechniki Warszawskiej, 2019, pp. 12, 24, 29.

[5] nobelprize.org/prizes/physics/1979/summary/, accessed 15.05.2023

[6] nobelprize.org/prizes/physics/2013/summary/, accessed 15.05.2023

[7] D. Castelvecchi, “What’s next for physics’ standard model? Muon results throw theories into confusion,” Nature **593** (2021) no.7857, 18-19 doi:10.1038/d41586-021-01033-8

[8] wikipedia.org, accessed 8.05.2023

[9] Grebieszkow K.:"Fizyka zderzeń ciężkich jonów", Lectures at Faculty of Physics of WUT, www.if.pw.edu.pl/ kperl/HIP/hip.html, accessed 8.05.2023

[10] "Compressed Baryonic Matter Experiment at FAIR, Progress Report 2020" Senger, Peter et. al (CBM Collaboration) doi:10.15120/GSI-2021-00421.

[11] CBM Collaboration, EPJA 53 3 (2017) 60 T.Galatyuk, NPA982 (2019), update (2022 NuPECC)

[12] K. Aamodt *et al.* [ALICE], “The ALICE experiment at the CERN LHC,” JINST **3** (2008), S08002 doi:10.1088/1748-0221/3/08/S08002

[13] T. Ablyazimov *et al.* [CBM], “Challenges in QCD matter physics –The scientific programme of the Compressed Baryonic Matter experiment at FAIR,” Eur. Phys. J. A **53** (2017) no.3, 60 doi:10.1140/epja/i2017-12248-y [arXiv:1607.01487 [nucl-ex]].

[14] B. Friman, C. Hohne, J. Knoll, S. Leupold, J. Randrup, R. Rapp and P. Senger, Lect. Notes Phys. **814** (2011), pp.1-980 doi:10.1007/978-3-642-13293-3

[15] fair-center.eu, accessed 8.05.2023

[16] "Compressed Baryonic Matter Experiment at FAIR, Progress Report 2022" Senger, Peter et. al (CBM Collaboration) doi:10.15120/GSI-2023-00384].

[17] fair-center.eu/for-users/experiments/cbm.html, accessed 8.05.2023

[18] C. Höhne, Ed., 'Technical Design Report for the CBM Ring Imaging Cherenkov Detector', 2013.

[19] C. Blume, C. Bergmann, and D. Emschermann, Eds., 'The Transition Radiation Detector of the CBM Experiment at FAIR: Technical Design Report for the CBM Transition Radiation Detector (TRD)', Facility for Antiproton and Ion Research in Europe GmbH, Darmstadt, 2018. doi:10.15120/GSI-2018-01091

[20] N. Herrmann, Ed., Technical Design Report for the CBM Time-of-Flight System (TOF). Darmstadt: GSI, 2014, p. 182 S.

[21] S. A. Bass, M. Belkacem, M. Bleicher, M. Brandstetter, L. Bravina, C. Ernst, L. Gerland, M. Hofmann, S. Hofmann and J. Konopka, *et al.* "Microscopic models for ultrarelativistic heavy ion collisions," Prog. Part. Nucl. Phys. **41** (1998), 255-369 doi:10.1016/S0146-6410(98)00058-1 [arXiv:nucl-th/9803035 [nucl-th]].].

[22] M. Baznat, A. Botvina, G. Musulmanbekov, V. Toneev and V. Zhezher, "Monte-Carlo Generator of Heavy Ion Collisions DCM-SMM," Phys. Part. Nucl. Lett. **17** (2020) no.3, 303-324 doi:10.1134/S1547477120030024 [arXiv:1912.09277 [nucl-th]].

[23] Słodkowski M.:"Modelowanie Procesów Jądrowych", Lectures at Faculty of Physics of WUT, efizyka.if.pw.edu.pl/MPJ

[24] S. Agostinelli *et al.* [GEANT4], "GEANT4–a simulation toolkit," Nucl. Instrum. Meth. A **506** (2003), 250-303 doi:10.1016/S0168-9002(03)01368-8

[25] D. Blau, O. Golosov, E. Kashirin, V. Klochkov and I. Selyuzhenkov, "Performance studies for collective flow measurements with CBM at FAIR," J. Phys. Conf. Ser. **1390** (2019) no.1, 012027 doi:10.1088/1742-6596/1390/1/012027

[26] O. Golosov. "Performance of the CBM experiment at FAIR for measurement of charged hadron anisotropic flow". 3rd workshop on "Physics performance studies at FAIR and NICA" (FANI-2021). 2021.

[27] J. K. Wirth. "(Strange) Meson Production in Pion-Nucleus Collisions at 1.7 GeV/c". PhD thesis. Technische Universität München, 2019, pp. 33, 34

[28] H. Schiller. "Application of Machine Learning to Particle Identification or Dielectron Analysis in CBM", Masther thesis. WWU Münster, 2022. p. 12

[29] ibm.com/cloud/learn/machine-learning, accessed 7.06.2023

[30] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," in IBM Journal of Research and Development, vol. 3, no. 3, pp. 210-229, July 1959, doi: 10.1147/rd.33.0210.

[31] 7wdata.be/visualization/types-of-machine-learning-algorithms-2/, accessed 7.06.2023

[32] semiengineering.com/deep-learning-spreads/, accessed 7.08.2023

[33] Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016. [arXiv:1603.02754]

[34] towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she -may-rein-edd9f99be63d, accessed 7.08.2023

[35] L. Breiman, J. H. Friedman, R. A. Olshen and C. J. Stone, Classification and Regression Trees (Wadsworth, 1984).

[36] Y. Coadou, “Boosted decision trees,” doi:10.1142/9789811234033_0002

[37] xgboost.readthedocs.io/en/stable/parameter.html, accessed 8.06.2023

[38] Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M., “Optuna: A Next-generation Hyperparameter Optimization Framework”, 2019. doi:10.48550/arXiv.1907.10902

[39] scikit-learn.org/stable/modules/generated/sklearn.utils.class _weight.compute_class_weight.html, accesed 8.06.2023

[40] M. Kasak. "Particle identification with neural networks", Eng. thesis. Warsaw University of Technology, 2023.

[41] S. M. Lundberg et al., “From local explanations to global understanding with explainable AI for trees,” Nature Machine Intelligence, vol. 2, no. 1, pp. 56–67, 2020, doi: 10.1038/s42256-019-0138-9.

[42] J.Nowak, ml-tree-plainer, 2022, github.com/julnow/ml-tree-plainer

[43] pdg.lbl.gov/2019/reviews/rpp2019-rev-monte-carlo-numbering.pdf, accessed 16.06.2023

[44] J. Nowak. "Implementation of machine learning algorithms for particle identification in the CBM experiment", Eng. thesis. Warsaw university of Technology, 2022.

[45] J.Nowak, ml-pid-cbm, 2023, github.com/julnow/ml-pid-cbm/

[46] Barioglio, L. et al. (2022). hipe4ml/hipe4ml (v0.0.14). Zenodo. doi.org/10.5281/zenodo.7014886

[47] Yang, J., “Fast TreeSHAP: Accelerating SHAP Value Computation for Trees”, 2021. doi:10.48550/arXiv.2109.09847.

[48] github.com/linkedin/FastTreeSHAP/releases/tag/v0.1.5, accessed 16.06.2023

[49] scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html, accessed 18.06.2023

[50] V. Klochkov et al., "Particle Identification Framework", 2018, github.com/HeavyIonAnalysis/Pid

List of Figures

List of Tables