UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Dept. of Computer Systems and Computation

Real-time anomaly detection using Reinforcement Learning at LHCb CERN experiment

Master's Thesis

Master's Degree in Artificial Intelligence, Pattern Recognition and Digital Imaging

AUTHOR: Del Pianta Pérez, Lorenzo

Tutor: Alberola Oltra, Juan Miguel

Cotutor: Sánchez Anguix, Víctor

External cotutor: GARCIA PARDINAS, JULIAN

ACADEMIC YEAR: 2022/2023

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Real-time anomaly detection using Reinforcement Learning at LHCb CERN experiment

Master's Thesis

Master's Degree in Artificial Intelligence, Pattern Recognition and Digital Imaging

Author: Lorezo Del Pianta Pérez

Supervisors: Juan Miguel Alberola Oltra, UPV

Víctor Sánchez Anguix, UPV

Julián García Pardiñas, CERN

2022/2023

Resumen

Esta tesis se centra en el desarrollo e implementación de un sistema de detección de anomalías en tiempo real utilizando técnicas de reinforcement learning en el contexto del experimento LHCb (Large Hadron Collider beauty). El experimento LHCb se lleva a cabo en el CERN (Organización Europea para la Investigación Nuclear) y tiene como objetivo explorar el comportamiento de las partículas que contienen quarks del tipo belleza y encanto.

La detección de anomalías juega un papel crucial para asegurar la integridad de los datos recogidos durante las colisiones de partículas, ya que identificar una anomalía significa la presencia de errores en los sub-detectores o en las condiciones de operación del experimento.

Así, esta tesis explora la aplicación novedosa de algoritmos de reinforcement learning para la detección de anomalías en tiempo real con el fin de mejorar la eficiencia y la precisión en la detección de eventos anómalos, contribuyendo de esta manera a la fiabilidad de los datos recogidos.

Palabras clave: Detección de anomalías en tiempo real, Reinforcement learning, Física Experimental de Altas Energías, Inteligencia Artificial, CERN

Abstract

This thesis focuses on the development and implementation of an innovative real-time anomaly detection system using reinforcement learning techniques within the context of the LHCb (Large Hadron Collider beauty) experiment. The LHCb experiment is conducted at CERN (European Organization for Nuclear Research) and aims to explore the behaviour of particles containing beauty and charm quarks.

Anomaly detection plays a crucial role in ensuring the integrity of the data collected during particle collisions since identifying an anomaly means the presence of errors in the sub-detectors or the operating conditions of the experiment.

Thus, this thesis explores the application of reinforcement learning algorithms for real-time anomaly detection to enhance the efficiency and accuracy of detecting anomalous events, ultimately contributing to the reliability of the data collected.

Keywords: Real-time anomaly detection, Reinforcement learning, High-energy physics, Artificial Intelligence, CERN

General Index

CHAPTER 1 .... **10**

1.1 CERN OVERVIEW 10

1.1.1 CERN's accelerators *11*

1.1.2 CERN's Experiments *12*

1.2 THE NEED FOR AN AUTOMATIZATION OF THE MONITORING TASK 14

1.3 OBSERVED REQUIREMENTS 18

1.4 PROPOSED SOLUTION 19

1.5 MAIN AIMS 19

CHAPTER 2 .... **21**

2.1 LARGE HADRON COLLIDER OVERVIEW 21

2.2 LHCB DETECTOR OVERVIEW 23

2.2.1 Run 3 LHCb update *24*

2.2.2 Data classification automatization attempt at LHCb and Run3 context *25*

2.3 REINFORCEMENT LEARNING WITH HUMAN IN THE LOOP OVERVIEW 25

2.3.1 Key components of RLHF *27*

2.3.2 Related applications of RLHF *27*

2.3.3 Advantages and limitations of RLHF *31*

CHAPTER 3 .... **32**

3.1 RAY RLLIB 32

3.2 PROXIMAL POLICY OPTIMIZATION ALGORITHM (PPO). 33

CHAPTER 4 .... **35**

4.1 REAL DATA COLLECTION 35

4.2 SYNTHETIC DATASET FOR THE EXPERIMENTS 36

4.2.1 Data explanation *36*

4.2.2 Dataset hyperparameters overview *40*

4.2.3 Dataset samples *41*

CHAPTER 5 .... **43**

5.1 ENVIRONMENT SETUP AND AGENT WORKFLOW 43

5.2 RL AGENT'S HYPERPARAMETERS TUNING 44

5.3 TRAINING OVERVIEW 46

5.4 HUMAN ERROR SIMULATION AND MACHINE-ASSISTED HUMAN DECISION 49

5.4.1 Human error implementation *49*

5.4.2 Machine-assisted human decision implementation *49*

5.4.3 Workflow of the experiment. *51*

CHAPTER 6 .... **52**

6.1 EXPERIMENTS WITH DIFFERENT LEVELS OF NOISE 52

6.2 MACHINE-ASSISTED HUMAN DECISION 54

CHAPTER 7 .... **58**

7.1 MAIN CONCLUSIONS 58

7.2 FUTURE STEPS 59

Figure Index

Figure 1. Map showing the location of the LHC tunnel across France and Switzerland. 10
Figure 2. CERN accelerator complex 11
Figure 3. CERN experiments. In clockwise order are ATLAS, ALICE, LHCB, and CMS. 13
Figure 4. The LHCb control room. 14
Figure 5. Example of correct data from sub-detector. Red histogram reference, blue histogram data received. 16
Figure 6. Example of anomalous data from sub-detector. Red histogram reference, blue histogram data received. The histograms have different shapes thus there is an anomaly. 17
Figure 7. LHC ring sketch with beam pipes (dark grey) in which proton bunches (red & blue dots) are circulated in opposite directions and collide in the experiments (yellow) [22]. 22
Figure 8. Side view of the LHCb detector Layout. Particles collide in the Vertex Locator. 23
Figure 9. Reinforcement Learning algorithm scheme [28]. 26
Figure 10. RLHF algorithm working scheme [29]. 26
Figure 11. Reconstruction example of a one-dimensional autoencoder trained with the proposed solution of the CMS collaboration. 30
Figure 12. Diagram of the RL iterative learning process in Ray RLlib. 33
Figure 13. Normal distribution histogram sketches with statistical noise limits per each bin. 37
Figure 14. Good histograms with different levels of statistical noise. First ±1%, second ±5% .. 37
Figure 15. Standard normal distribution curve with standard deviation indicators and different label zones for bin classification. 39
Figure 16. Synthetic dataset histograms samples 42
Figure 17. Human render mode during agent training. 47
Figure 18. First agent training mean reward evolution in test and training. 47
Figure 19. Anomaly classification evaluation through the human mode agent render. 48
Figure 20. Smoothed Heaviside step function with threshold of 0.5 used to establish the distrusting probability of the human in the algorithm. 50
Figure 21. Evolution of mean rewards achieved by humans and agent with different levels of noisy labels. Y axe represents the mean rewards, the x axe the iterations. 53
Figure 22. Example of wrong human assistance due to a low threshold. (y axe represents the mean reward values, x the iterations) 54
Figure 23. Example of good human assistance due to a better threshold. (y axe represents the mean reward values, x the iterations) 55
Figure 24. Example of wrong human assistance due to a high threshold. (y axe represents the mean reward values, x the iterations) 55
Figure 25. Human assistance training overview with human error of 20%. (y axe represents the mean reward values, x the iterations) 56

Table Index

Table 1. Synthetic dataset hyperparameters resume. *41*
Table 2. Hyperparameter values are used in the tuning process. *45*
Table 3. Best algorithm hyperparameter values found after the tuning process. *45*
Table 4. Dataset parameters used in the training phase. *46*
Table 5. Comparative between the Agent results and human results considering noisy labels. *52*
Table 6. Optimal thresholds depend on human error. *57*

Glossary

ALICE: A Large Ion Collider Experiment. It is a particle physics experiment at CERN that focuses on studying the properties of matter under extreme conditions, particularly the quark-gluon plasma.

ATLAS: A Toroidal LHC ApparatuS. It is one of the major particle detectors at CERN and It is designed to observe a wide range of physics phenomena.

CERN: The Conseil Européen pour la Recherche Nucléaire, or European Organization for Nuclear Research, is a globally renowned laboratory for scientific research and collaboration in the field of particle physics.

CCC: CERN Control Center. It is the main control centre and combines the control rooms of the Laboratory's eight accelerators as well as the operation of cryogenics and technical infrastructures.

CMS: Compact Muon Solenoid. It is a CERN general-purpose detector designed to investigate various aspects of particle physics.

DQ: Data Quality Shifter. Role responsible for reviewing and analysing data quality, identifying anomalies or issues in experimental data, and making decisions about data usability.

DM: Data Manager Shifter. Role responsible for monitoring and overseeing real-time histograms and data quality within a particle physics experiment, identifying any anomalies, and taking appropriate actions.

Ground Truth: A reference or standard against which experimental or observed data is compared to determine accuracy or correctness.

HIL: Human-in-the-Loop. An approach where human feedback and guidance are incorporated into an AI or automation process to enhance performance and decision-making.

LHCb: Large Hadron Collider beauty. A scientific experiment at CERN focused on understanding the asymmetry between matter and antimatter in the universe through the study of 'beauty quarks' or 'b quarks.'

LHC: Large Hadron Collider. It's a circular tunnel designed to accelerate particles to extremely high speeds and collide them, enabling the study of fundamental particle physics.

NMF: Non-negative matrix factorization, is a group of algorithms in multivariate analysis and linear algebra where a matrix V is factorized into (usually) two matrices W and H, with the property that all three matrices have no negative elements.

Particle Accelerator: A facility that accelerates particles, such as protons or electrons, to high speeds and collides them for various research purposes.

Quark-Gluon Plasma: A theoretical state of matter that existed just after the Big Bang, consisting of liberated quarks and gluons.

RL: Reinforcement Learning. A machine learning approach where an artificial intelligence system learns through interactions with its environment, receiving feedback and adjusting its actions to achieve desired outcomes.

RLHF: Reinforcement Learning with Human in the Loop. A variant of reinforcement learning where human feedback is integrated into the learning process to improve the AI's performance.

Run: Several-year period in which collisions are made inside the LHC with a certain overall configuration (typically the same collision energy). Do not confuse it with “run” (with lowercase), the few-hour periods where collisions are being maintained at the LHCb experiment.

SL: Shift Leader. It is responsible for the experiment operations and ensures that the daily program is respected.

Super-Human State: A state where the combined decision-making of an AI algorithm and a human, results in better outcomes than each entity acting independently.

TeV: Tera electron Volts. An electron volt is a unit of energy, particularly used in atomic and nuclear processes. It is the energy given to an electron by accelerating it through 1 volt of electric potential difference.

Toy Dataset: A simplified, and artificial dataset created for experimentation or testing purposes to mimic real-world data patterns.

VELO: Vertex Locator. It is the inner tracker of the LHCb located around the proton collision which performs highly precise track and vertex reconstruction of the particles after the collision.

Chapter 1

Introduction

This first chapter sets the context in which the thesis will be developed and introduces the problem of data monitoring in the LHCb experiment. Finally, it explains the proposed solution and the objectives to be achieved.

1.1 CERN overview

CERN, situated in the Northwest suburbs of Geneva on the Franco-Swiss border (Figure 1) [1], is renowned as one of the world's largest and most esteemed laboratories for scientific inquiry. The abbreviation "CERN" stands for "Conseil Européen pour la Recherche Nucléaire," signifying European Nuclear Research [1]. Currently, CERN boasts 23 member states, primarily situated in Europe, alongside Japan and the United States of America, holding Observer status.

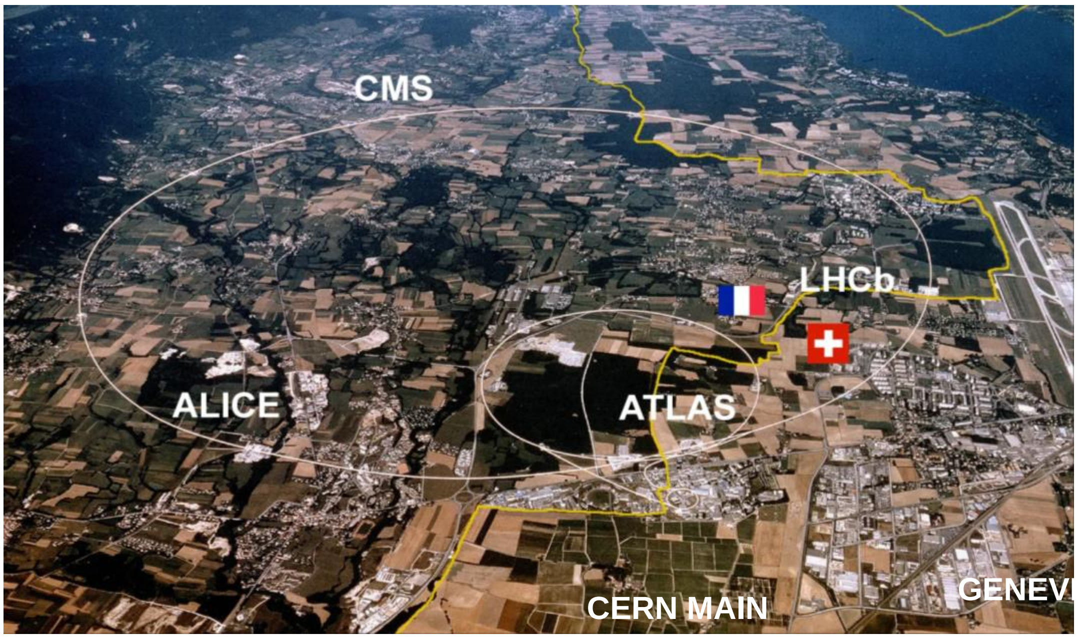


Figure 1. Map showing the location of the LHC tunnel across France and Switzerland.

The organization's international character and numerous collaborations attract scientists from around the globe, rendering CERN one of the most culturally diverse and multicultural entities worldwide. Established in 1954, CERN has remained at the forefront of scientific exploration, leading to significant discoveries, including the inception of the World Wide Web and Nobel physics prizes, among others [3].

In 2021, 3.459 employed members of personnel collaborated with 16.190 associated members of personnel coming from over 40 countries to conduct fundamental research [5]. Besides the fundamental research, it is CERN´s mission to train a new generation of physicists, engineers, and technicians by uniting people from all over the world. CERN is furthermore committed to being a politically neutral voice for science and building links with the industry in terms of transferring knowledge [6].

CERN propels research to the cutting edge of human knowledge through its distinctive array of particle accelerator facilities, which drive particles like protons or electrons to high velocities, close to the speed of light, causing collisions either with other particles or with a target. These collisions of 2 particles could be made by accelerating one particle clockwise and the other particle counterclockwise.

Once the particles are sufficiently energetic, a phenomenon takes place that exceeds human imagination. At the point of the collision, the energy is transformed into matter in the form of new particles. This phenomenon follows Einstein´s famous equation, which basically states that matter is a concentrated form of energy and the two are interchangeable. In this way, the particle accelerators at CERN can explore the fundamentals of matter and contribute to our general understanding of the universe. [7]

1.1.1 CERN's accelerators

CERN is known for the world's largest particle accelerator the Large Hadron Collider (LHC). Located near Geneva, the LHC lies in a ring-shaped tunnel structure at an average of 100 meters underground. The 27-kilometre-long structure can create a collision energy of 13 TeV per beam [8]. Despite the popularity of the LHC, the accelerator complex consists of a succession of machines that accelerate particles to increasingly higher energy levels. The beam is currently accelerated in a total of 5 accelerators before its collision [9] (Figure 2). Hereby, a beam describes a package of particles that is accelerated.

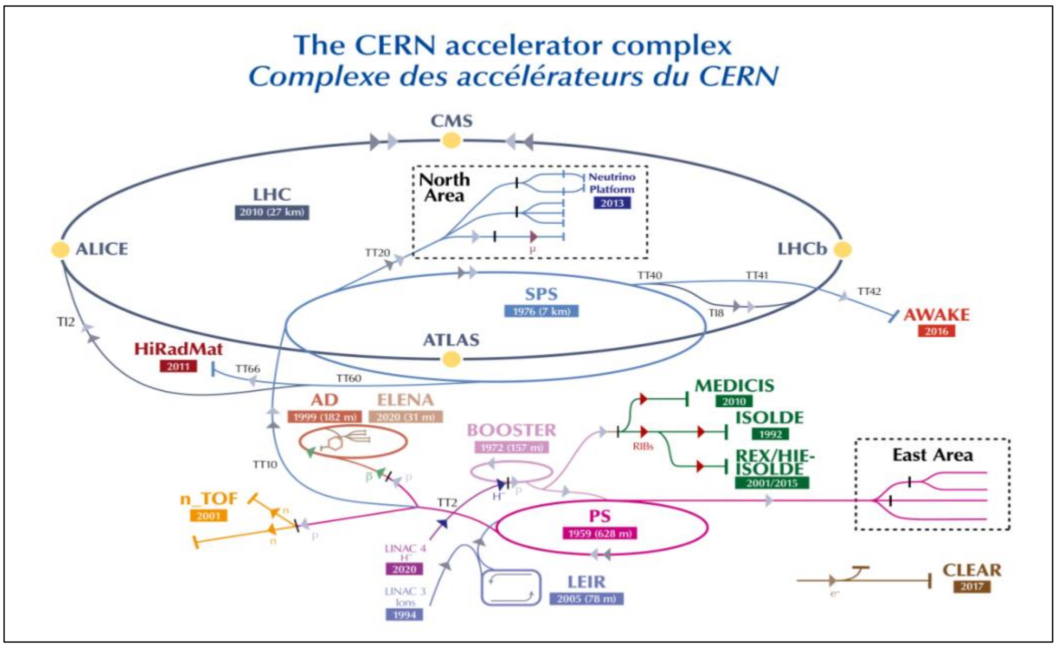


Figure 2. CERN accelerator complex

Initially, negative hydrogen ions are introduced into the Linear Accelerator 4 (LINCAC 4), which serves as the first accelerator in a chain of four stages to increase the particle's energy to 160 MeV [9]. Once the desired energy level is attained, the particle beam is further directed into the Proton Synchrotron Booster (PSB), where the energy is elevated to 2 GeV [10].

The Proton Synchrotron further amplifies the particle energy up to 26 GeV. Spanning 628 meters in length, this circular accelerator houses 277 conventional magnets, including 100 dipoles responsible for bending the beam around the ring [11].

In the subsequent stage, the particles are directed to the Super Proton Synchrotron (SPS). Over its 7 km circumference, protons are accelerated to an energy of 450 GeV using 1317 conventional magnets [12]. In 1983, the SPS experiment led to the discovery of two new particles, which played a pivotal role in Sheldon Glashow, Steven Weinberg, and Abdus Salam being awarded the Nobel Prize in Physics. [13].

Finally, the protons are guided into the two beam pipes of the Large Hadron Collider (LHC). The beams travel in opposite directions, and both pipes are maintained under ultrahigh vacuum conditions. Electromagnets constructed from superconducting electric cables enable the beams to be accelerated close to the speed of light before the collision. These magnets are cooled to an astonishingly low temperature of -271.3 °C, colder than the temperature in outer space. The particles are so minuscule that an analogy is made to firing two needles 10 kilometres apart with such precision that they meet in the middle. Nonetheless, the LHC creates an astounding 4 billion collisions at its detectors every second [14]. With a total cost of 43.32 billion CHF and an energy consumption of 750 GWh, the LHC stands as the most complex machine in the world [8].

1.1.2 CERN’s Experiments

The trajectory of the particles is directed in such a way that they are brought to collision at the center of one of the four corresponding experiments that are arranged around the LHC ring every 25ns. Each experiment has its own task and is designed for a different measurement spectrum:

- **ATLAS** (A Toroidal LHC ApparatuS): ATLAS is a general-purpose detector designed to explore a wide range of physics, including the search for the Higgs boson, extra dimensions, and particles that could make up dark matter. It played a crucial role in the discovery of the Higgs boson in 2012.[15]
- **CMS** (Compact Muon Solenoid): Like ATLAS, the CMS experiment is a general-purpose detector, designed to investigate a wide range of physics phenomena. Its design is different from ATLAS, and it uses different technical solutions and a different magnet-system design. The two experiments independently confirm each other's findings [16].
- **ALICE** (A Large Ion Collider Experiment): Unlike the other detectors, ALICE is designed specifically to study the quark-gluon plasma, a state of matter

believed to have existed just after the Big Bang. It does this by colliding lead ions together at very high energies [16].

- **LHCb** (Large Hadron Collider beauty): The LHCb experiment is focused on understanding the asymmetry between matter and antimatter in the universe by producing and studying the properties of particles that contain so-called ‘beauty’ (b) and ‘charm’ (c) quarks. The layout of the detector is optimized for observing decays of those types of particles with very high precision [17].

  Since the particles of interest only get produced in a small fraction of the collisions, and recording every collision would be prohibitive in terms of data size, the LHCb experiment utilizes a sophisticated trigger system to select the most interesting events in real time. Out of the tens of millions of collisions that occur every second, this system is designed to prioritize events that are likely to contain the particles of interest [17].

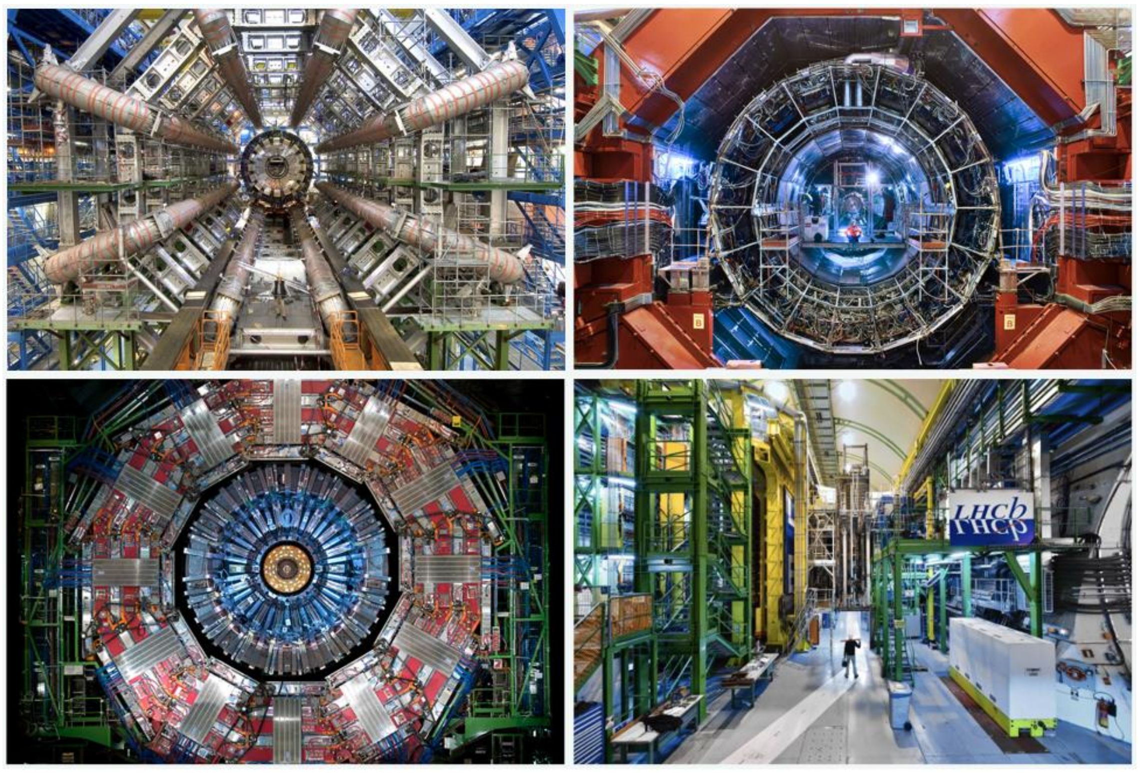


Figure 3. CERN experiments. In clockwise order are ATLAS, ALICE, LHCB, and CMS.

1.2 The need for an automatization of the monitoring task

Every CERN experiment has its own control room, and this serves as the nerve centre of the experiment. Here, operators, engineers, and scientists oversee and manage the experiment in real time thanks to advanced graphical user interfaces (GUIs) and control software. These provide a comprehensive view of both the LHC and the experiment's status and allow operators to interact with various components.

People inside the control room (Figure 4) are called shifters, the name comes from the fact that their work schedule is organized into 3 shifts of 8:30 hours each, every day of the year, except punctual periods when the machine is stopped. Their role is to try to maximise the time that the detector is taking data in good conditions, which will translate into larger amounts of useful data, and in turn higher precision of the physics measurements [18].

The role of the shifters is performed by volunteering non-expert physicists working at the LHCb experiment (a pool of a few hundred physicists is needed), who are trained in devoted sessions to do the role, such that each of them does around 5 shifts per year. Thus, this implies large needs in terms of person power and training resources, and some level of inefficiency in the process due to (infrequently but possibly) random human errors and potential loss of expert knowledge due to the naturally imperfect human-human communication.

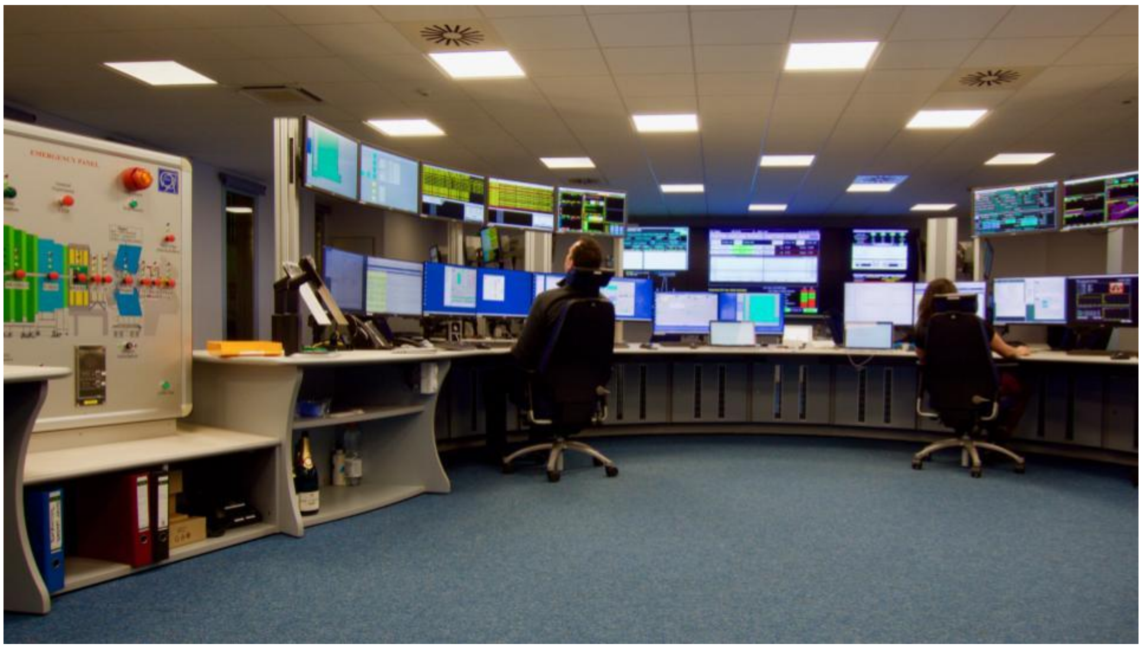


Figure 4. The LHCb control room.

The LHCb central Shift Crew is made up of:

- Shift Leader (SL): Is responsible for safety, supervision of activities, supervision of control system and experiment conditions, contact with experts and with the CERN Control Centre (CCC)
- Data Manager (DM): Is responsible for monitoring the quality of data, in particular checking that the report plots are being filled, evaluating the data and reports, or flagging any anomaly. Moreover, replaces SL if needed.

Most of the quantities produced by the monitoring tasks are histograms: 1D, 2D or trend plots as a function of time and those represent different features, for example, the number of particles that hit that detector, how much energy they deposited in the detector and other physics properties that allow recording each collision Every sub-detector has its series of histograms corresponding to current data, accumulating over the past 10 minutes at maximum (i.e., the histograms are reset every 10 minutes and filled again from empty plots, to spot potential new problems).

The tasks of the DM are crucial for the running of LHCb. Data with malfunctioning detectors will most certainly be discarded for posterior analysis. The DM tasks are executed when the experiment is running and is acquiring data. This phase is called “online”.

The particle collisions are produced and recorded continuously in separate intervals of typically a few hours, called “runs”. Additional histograms corresponding to the full duration of each run are produced and stored away. After the data has been collected, in a so-called "offline” phase, a third person examines those histograms to finally label the data as good (usable for research) or bad.

The person in charge of that task is the so-called Data Quality shifter (DQ), which is also a non-expert volunteer. The DM and DQ roles are complementary. The DM can spot problems in real-time and hence lead to a change in the conditions that can fix the situation.

The DQ cannot change the way a certain dataset was collected but has much more time to do a finer inspection and identify previously hidden problems. The DQ has a high responsibility in labelling the data correctly: bad runs labelled as good can produce wrong measurements, and good runs labelled as bad will mean loss of valuable data. In the following paragraphs, the procedure will be described in more detail.

In the online phase, the DM checks real-time histograms over a predefined and selected list of plots. Specifically, they have a reference of what we can define as a ground truth histogram for each plot, this histogram describes in the current operative conditions of the machine what can be considered as a relevant result that can be used for further physic analysis (Figure 5).

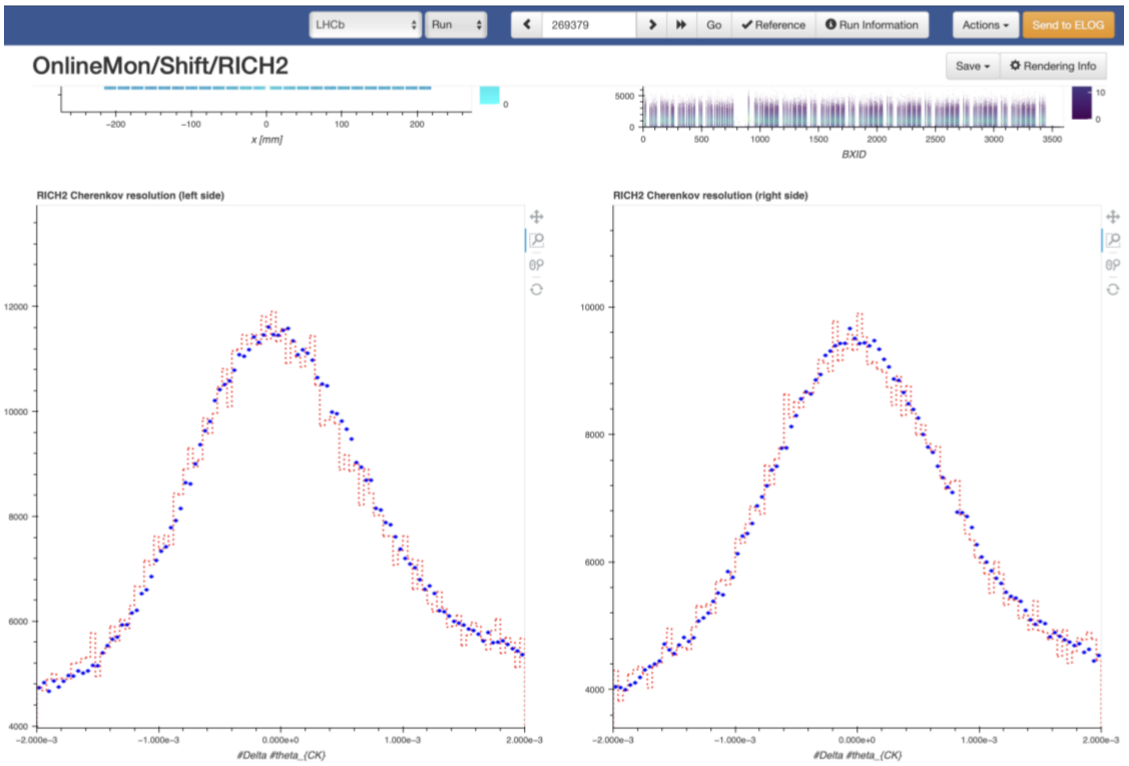


Figure 5. Example of correct data from sub-detector. Red histogram reference, blue histogram data received.

Those Reference Histograms (see Figure 5) have to be defined by detector experts every time at the start of the machine “run” and can be changed across time since the conditions inside the accelerator are constantly changing due to different factors such as the density and speed of the particles, number of bunches (group of particles) per beam, etc.

If one of the sub-detectors has a problem, this will result in an altered histogram (the type of deformation not typically known a priori). This is illustrated in Figure 6 where the reference (red histogram) is completely different from the data acquired (blue histogram).

It is important to outline that keeping histogram references up to date is a huge challenge for the experts, as not only the operation conditions but also the definition of what is acceptable changes with time. For example, the experts may discover a transitory problem in the readout of one of the subdetectors, which is producing altered histograms (e.g., a fraction of them is empty), but not substantially affecting the quality of the data. In such a case, either the reference histograms need to be modified to include the deformation or the DM needs to be instructed not to pay attention to the affected sub-set of histograms.

Keeping reliable and timely references is currently a very labour-intensive human task, which if not done adequately can lead to a loss in data-taking efficiency. That inefficiency combines with the stochastic human error of the DM that needs to compare a large number of histograms with their references.

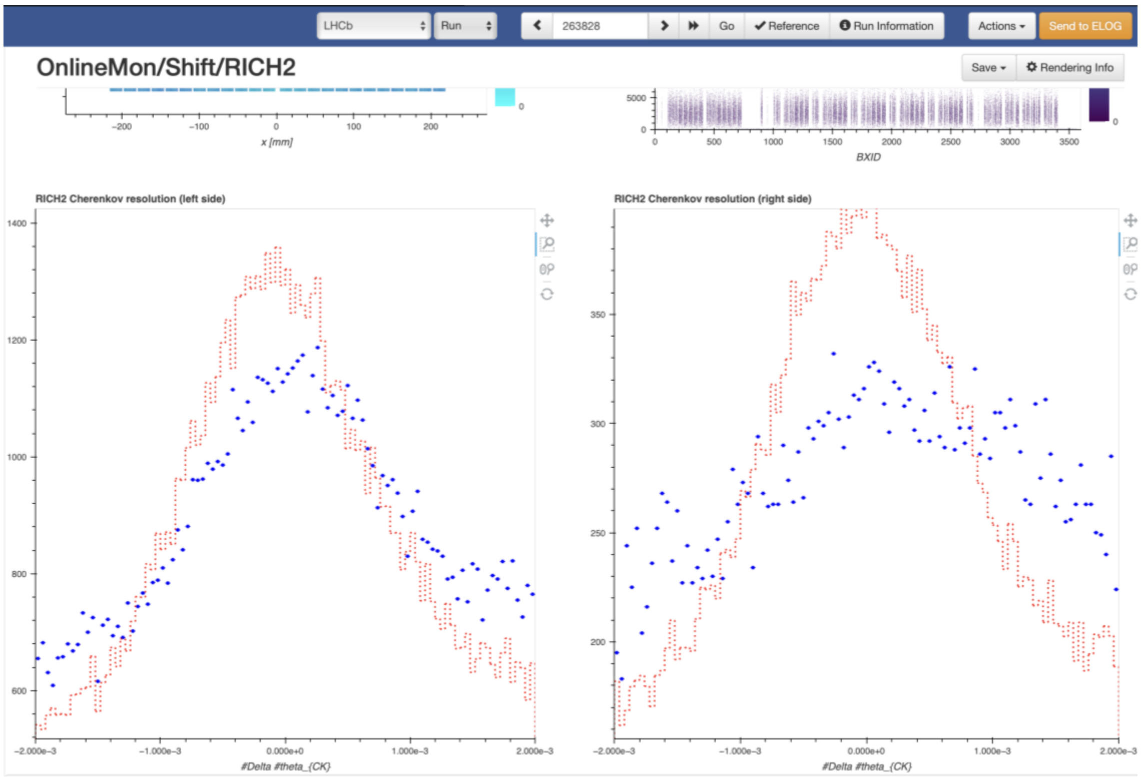


Figure 6. Example of anomalous data from sub-detector. Red histogram reference, blue histogram data received. The histograms have different shapes thus there is an anomaly.

In the offline phase, DQ review the histograms of that data that apparently was labelled as good during the online phase and performs an extended analysis checking more parameters and looking for possible flaws or errors that were not identified in the online phase due to the real-time conditions.

If the DQ after evaluating the plots does not find any issue, they will mark the data as correct, and thus this will be used by physicists. Otherwise, if an error is found the data will be marked as an anomaly, and this cannot be used. However, neither the best DQ nor DM can guarantee that when a run data is classified, either as an anomaly or as correct, that decision is 100% correct.

We can resume the tasks of the DM and DQ described in the above paragraphs:

Online phase:

1. Reference histograms are created by the detector experts every time the working conditions change, adapting these to the status of the machine.
2. DM monitor the histograms received and compares these with the reference.
3. If they see relevant incongruences, they flag the issue and call the experts to analyse the problem and fix it as soon as possible, or alternatively adjust the ground truth reference if the deviation is harmless.
4. If the signal received is compatible with the reference no action is required by the DM.

Offline phase:

1. DQ do an extended analysis of the data acquired during the runs checking all the relevant histograms.
2. If there are no anomalies the data of that run is labelled as good so it can be used in future analysis by physicists.
3. If there are anomalies the data of that run is labelled as anomaly, and it cannot be used by the physicists in future analysis.

Finally, data monitoring is a time-consuming repetitive task but at the same time crucial to guarantee the quality of future physics analysis. These characteristics force CERN and the experiments to:

- Have properly trained people to acquire the knowledge needed for the task expending resources and time in training courses.
- Organize shifts (covering the 24-hour period) for monitoring every phase of the data acquisition and cover in real-time the machine operations.
- Involve a very large number of physicists in the process that could be used for doing specialized tasks such as future result analysis.

1.3 Observed requirements

As described in the previous paragraphs, automating the data monitoring task could greatly benefit freeing up resources that are otherwise busy in a time-consuming and repetitive task, as well as potentially increasing the data-collection efficiency by reducing human errors of a statistical nature (from the DM and DQ sides) and possible problems of loss of information through inadequate references (detector response at a given time not fully understood by experts, human miscommunication, failure to update the references in time, etc.). However, we need to meet the following requirements and take into consideration a few key characteristics:

- The solution must be used in real-time and needs to evolve within time depending on the working conditions of the detector and the accelerators.
- The ground truth reference is not known in advance, neither the experts can forecast it and they will know only once they observe a trend during the machine operation.
- Experts can correctly classify histograms, but they are not flawless and are subject to an error rate caused by natural human behaviour and task peculiarity.

- The system must be robust, which can be done (at least in the short-medium term) if the aim of the algorithm is not to directly flag each dataset instead of the human but to suggest a flag that a human can then use in his decision.

1.4 Proposed solution

The conditions explained in the previous paragraph make reinforcement learning with human-in-the-loop (RLHF) techniques a good candidate to be able to solve the issue of data monitoring in this scenario. RLHF is a subset of reinforcement learning that incorporates human input to improve the learning process. It's an approach where an artificial intelligence (AI) agent learns from both its interactions with the environment and feedback from humans, effectively creating a human-in-the-loop AI system [20].

Thus, the project consists of the implementation of a real-time anomaly detection system using Reinforcement Learning algorithms that given a histogram as an input can determine if this is an anomaly or a correct signal and will give as an output the label to the shifter.

The general idea at the base of the framework would be to try to "emulate an expert and non-misleading (random noise elimination) shifter", which learns by observing data and interacting with humans, and the training time is the whole data-taking period. Moreover, experts through the RLHF techniques will be able to transfer their knowledge and teach the algorithm in real-time, supervise it until it is able to provide correct results and from that point, they mostly need to follow the suggestions of the algorithm.

This solution has the advantage that does not need to be paused and trained again when the reference changes, unlike other ML solutions such as supervised algorithms. Even though those alternative approaches could a priori be implemented, in practice defining the moment when a retraining should be triggered and implementing some sort of continuity between before and after setups (that would allow some preservation of the overall knowledge of the system) would require a lot of careful human tuning during all the data-taking process. This would significantly impact the overall potential usefulness of such an automatization.

1.5 Main aims

The main objective of this thesis is to explore the application of RLHF techniques in the field of real-time anomaly detection within the framework of the LHCb experiment in a closed environment that re-creates the characteristics of the real context. In detail, the following sub-objectives will be met:

1. Investigating the current state-of-the-art techniques for anomaly detection in high-energy physics experiments, specifically focusing on the CERN experiments.

2. Exploring the principles and concepts of reinforcement learning and understanding its applicability to anomaly detection problems.

3. Build a toy dataset to recreate different possible types of signals with relevant characteristics that came from the detector of the experiment as well as generating a set of anomalies.

4. Design and develop a reinforcement learning-based anomaly detection algorithm tailored to the real-time nature of the LHCb experiment and its requirements.

5. Evaluate the performance of the system using the toy dataset using the accuracy and efficiency rates and optimise as needed.

6. Explore the possibility of reaching a "super-human" state where the combination of the algorithm plus the knowledge of the human can perform better rather than using them separately.

7. Discussion of the results and statement of future steps to be taken to continue further research.

One step further, regarding objective number six, a simplified modelling of the time-dependent change in the human response when assisted by the algorithm's prediction (with a modelled subjective level of "trust" in the algorithm) will be implemented, to investigate the obtention of super-human performance from a conceptual viewpoint. Finally, this project aims to prove with simplified simulated experiments that an average human who gets access to the decision provided by the RL algorithm before making their own decision can achieve better results than the nominal case in which they make their decision independently.

Chapter 2

Theoretical Background

In this chapter, the theoretical background of the study is explained. A more detailed overview of the LHC and the LHCb experiment will be given, focusing on what are the issues caused by the last upgrade and what attempts have been made to solve the anomaly detection problem.

Moreover, the reinforcement learning approach will be described, outlying the key components, and analysing the differences between RL and RLHF. Similar cases to this study where the technique has been relevant will be explained.

2.1 Large Hadron Collider overview

The LHC, situated nearly 100 meters below the ground close to Geneva, Switzerland, stands as the largest and most potent particle accelerator globally. It operates by colliding particles and consists of two superconducting magnet rings that stretch 26.7 kilometres in circumference. These rings store and propel two particle beams moving in opposite directions, leading to collisions at four points where the massive experiments that have been introduced before as ATLAS, CMS, ALICE, and LHCb are set up. These collisions are produced thanks to a bunch of crossings.

A bunch crossing refers to the moment when groups of particle clusters pass through each other at the interaction points, as illustrated in Figure 7. In theory, this occurs approximately every 25 nanoseconds, corresponding to a frequency of 40 megahertz. However, these bunches are arranged in clusters called bunch trains, and these clusters are spaced more than 25 nanoseconds apart. This arrangement lowers the overall collision frequency. Apart from the collisions between protons, the LHC also facilitates collisions between heavy ions like lead and xenon, as well as collisions involving protons and ions.

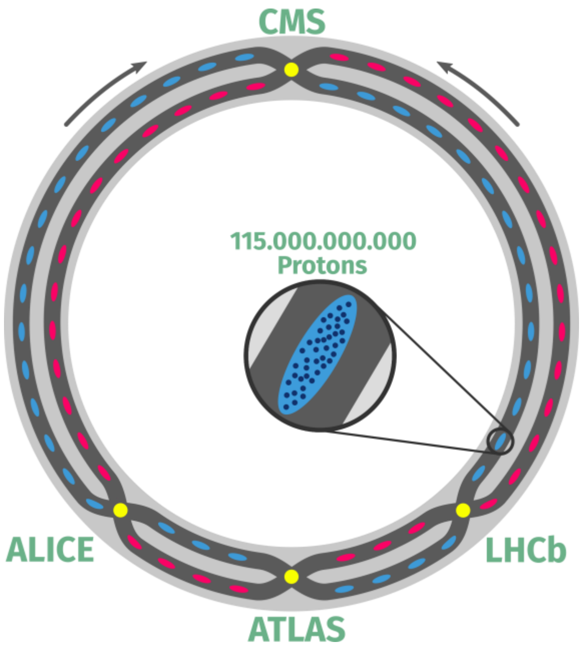


Figure 7. LHC ring sketch with beam pipes (dark grey) in which proton bunches (red & blue dots) are circulated in opposite directions and collide in the experiments (yellow) [22].

The four main detectors located in the LHC ring are designed to monitor the particles produced by the collisions and convert the results into usable data. The data collected by these detectors is then sent to a computer network, known as the LHC Computing Grid, which researchers use to interpret and process the data [23]. The Grid, which combines about 1.4 million computer cores and 1.5 exabytes of storage, is essential in handling the massive amounts of data generated by the LHC and is made up of several data centres spread over 42 countries [23].

The LHC was designed to reach a maximum energy level of 14 TeV for proton collisions, with a luminosity (particle collision density) of around 10^{34} per square centimetre per second. The LHC first ran its test operation on September 10, 2008, and has since undergone several periods of shutdown for upgrades and maintenance [21].

Its operation is organized into distinct periods called "Runs." The first Run spanned from 2009 to 2013, followed by Run 2 from 2015 to 2018. Run 3 commenced in 2022 and is scheduled to run until 2026, during which proton collisions are executed at an energy level of 13.6 TeV, the highest energy ever reached, thus roughly 9,600 magnets are being used to be able to control the particles travelling with that energy.

To conclude, the main aim of the LHC is to help scientists understand The Standard Model of particle physics, a theory developed in the early 1970s that describes the fundamental particles and their interactions, but is incomplete and leaves many questions open, such as “What are dark matter and dark energy?” or “What is the origin of mass?” Questions that the LHC will help to answer [24].

2.2 LHCb detector overview

The LHCb detector, with is weight of 5600 t, 21 m long, 10 m high and 13 m wide, is one of the four major experiments conducted at LHC. Its configuration includes an array of specialized subdetectors meticulously engineered to precisely measure various properties of particles generated during proton-proton collisions. The overall layout of the LHCb detector has been optimized specifically for the meticulous observation and measurement of the decay processes of ’beauty’ and ‘charm’ hadrons, which are particles containing the quarks mentioned before, with an exceptionally high degree of precision. [15]

These specific particles are emitted predominantly in a forward direction as a result of the collision and can fly significant distances before their decay, thus for that reason the LHCb experiment functions as a single-arm forward spectrometer that contrasts with the approach of enclosing the entire collision point within a detector. The first subdetector is positioned in proximity to the collision point, and subsequently, a series of additional subdetectors are arranged consecutively over a distance spanning 20 m [15].

A crucial aspect of the LHCb experiment is its ability to accurately reconstruct the positions of particles' displaced decay points (points where a particle transforms into one or more different particles or products), distinguishing them from the primary proton-proton collision locations. This capability is central to the LHCb's objectives and is primarily facilitated by its first sub-detector, known as the VELO (Vertex Locator).

Other particles, such as protons and electrons, as well as particles with a longer lifespan like pions, and muons, continue their journey through various components of the detector. This journey includes the first Ring-imaging Cherenkov detector (RICH1), the Upstream Tracker (UT), the magnetic field region, the Scintillating Fibre tracker (SciFi), the second RICH detector (RICH2), the electromagnetic calorimeter (ECAL), and the hadronic calorimeter (HCAL). Muons, due to their characteristics and very high life, reach the four muon stations (M2-M5) located at the far end of the detector setup. (Figure 8)

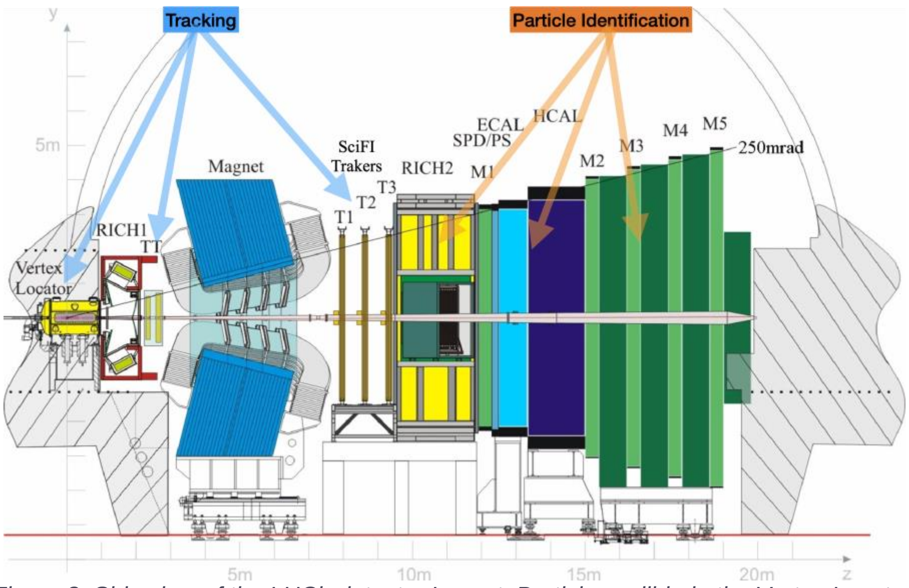


Figure 8. Side view of the LHCb detector Layout. Particles collide in the Vertex Locator.

In this thesis, we are not entering into detail the role of each one of the sub-detectors mentioned before. However, we can summarize that the RICH detectors, the calorimeters, and the muon stations make up the particle identification (PID) system, where their main aim is to collect information from the particle such as the energy among other data that will allow the identification of the particle itself.

The latter system is made up of the VELO, UT, SciFi tracker and the magnet and is the LHCb's tracking system. As its name suggests its main aim is to register and collect the particle data to be able to reconstruct the trajectory of the particle.

The correct functioning of all those sub-detectors is the one to be monitored by the DM and the DQ. When each of the particles produced in the collisions hits them, an individual response is generated in the detector and immediately read by a data acquisition system. By accumulating information from a large number of hits (over a period of minutes or hours), the response of each detector can be visualised in the form of occurrence histograms in a set of variables relevant to each specific sub-detection system.

Many variables are inspected, from the raw output of each of the individual readout channels in each subdetector to the processed measurements of the particles' properties. This leads to a total of around 100,000 different histograms that could be inspected each time the data needs to be monitored. The DM and DQ look continuously at a subset of those histograms, that has been preselected by experts.

2.2.1 Run 3 LHCb update

The LHCb upgrade was first outlined in 2008, proposed in 2011 and approved the following year at a cost of about 57 million Swiss francs. The collaboration started dismantling the current detector just before the end of 2018 and the Run 3 is the first with the upgraded detector [25]

New sub-detectors capable of sustaining up to five times the instantaneous luminosity seen at Run 2 have been installed. Moreover, thanks to other software updates, LHCb is enabled to process signal data in an upgraded CPU farm at the rate of 40 MHz and at an immense rate of 4 TB/s, data will travel from the cavern, straight from the detector electronics via some 9000 300 m-long optical fibres, into front-end computers located in a brand-new data centre [23].

It has just been substantially redesigned and modified to be capable of taking data at much higher particle collision rates. All the changes made must now be tested during collisions, correcting errors, and adapting the detector as necessary to eventually perform optimally. This data collection period, therefore, constitutes a non-stationary regime, in which both the type of errors and the nominal operating conditions change as a function of time.

2.2.2 Data classification automatization attempt at LHCb and Run3 context

The issue of the data classification has already been tried to be solved at LHCb, using a directly supervised classification based on observations of past runs in 2017. M Adinolfi developed an algorithm where a vector of Kolmogorov-Smirnov (KS) distances between the histograms and their references is created. With such vector and a flag for each run, the model predicts the run flag based on its KS-distances vector [26].

However, this type of classification is based on the existence of a fixed reference in time of what is correct data, so it can only be applied to the case where the data-taking conditions do not change. At detector commissioning time, (which is going to take a few years after we have almost completely changed the LHCb detector for Run 3), the regime is dynamic, so it cannot be used in this context.

The dynamism comes from multiple sides since the detector operative conditions are changing as well as the LHC beam conditions (different collision conditions are tested to test various possibilities of operation). Moreover, the detector software configuration changes with time improving it, and it can directly change the physicists' knowledge of the system, or the strategy to follow. For example, if a part of a detector is failing often the shifter can decide to ignore the signals coming from that sub-detector and considering the rest, an otherwise bad run can be given a clean bill of health.

Comparing the supervised approach with RL, it is possible to affirm that the latter offers two advantages. RL by construction is ideal for learning about an environment that is evolving over time, and it allows training without explicit references, only with the response of humans at a given time as to whether a run should be considered good or bad.

This gives a flexibility that may allow ML to be used in times of commissioning, to condense in a (dynamic) algorithm the knowledge that humans are acquiring of the system little by little. Not only does the algorithm give a way to store that knowledge, being able to tremendously reduce the time to train new shifters and to pass the knowledge between humans, but the loop between humans and algorithm allows both to be trained at the same time, potentially achieving super-human performance. To the best of our knowledge, ours is the first application of RL for data quality and/or anomaly detection in particle physics experiments.

2.3 Reinforcement learning with human in the loop overview.

Reinforcement Learning with Human in the Loop (also called with human feedback) constitutes a specialized domain within machine learning, where human interactions are seamlessly integrated into the reinforcement learning paradigm. This approach synergizes the computational prowess of reinforcement learning algorithms with the cognitive capabilities of human involvement, with the overarching goal of enhancing the efficacy and efficiency of the learning process in diverse applications [27].

In the conventional framework of reinforcement learning, an autonomous agent endeavours to optimize cumulative rewards by executing actions within a given environment. The agent dynamically interacts with the environment, making decisions based on its present state, and subsequently reaps rewards or incurs penalties in response [28]. (Figure 9)

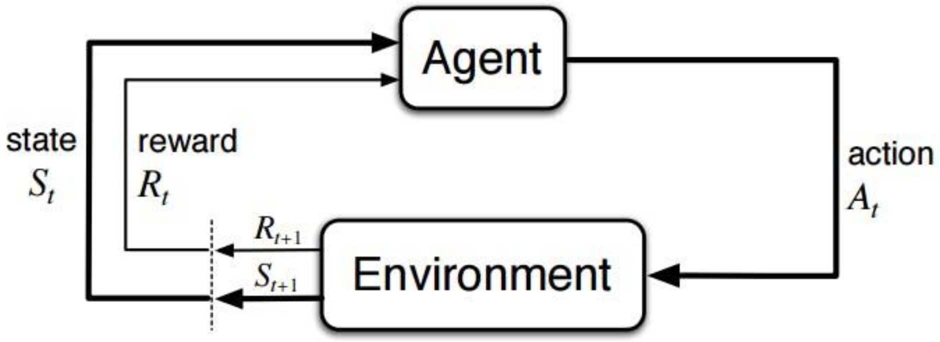


Figure 9. Reinforcement Learning algorithm scheme [28].

In the RLHF framework, a human element becomes an integral part of the learning loop, as you can see in Figure 10. Humans assume various roles in this context, including providing task demonstrations, intervening to avert catastrophic actions, and assessing the fidelity of the policy's performance. The modes of interaction between humans and reinforcement learning algorithms can be flexibly orchestrated and combined to bolster sample efficiency, thereby enabling real-time reinforcement learning in diverse scenarios [27].

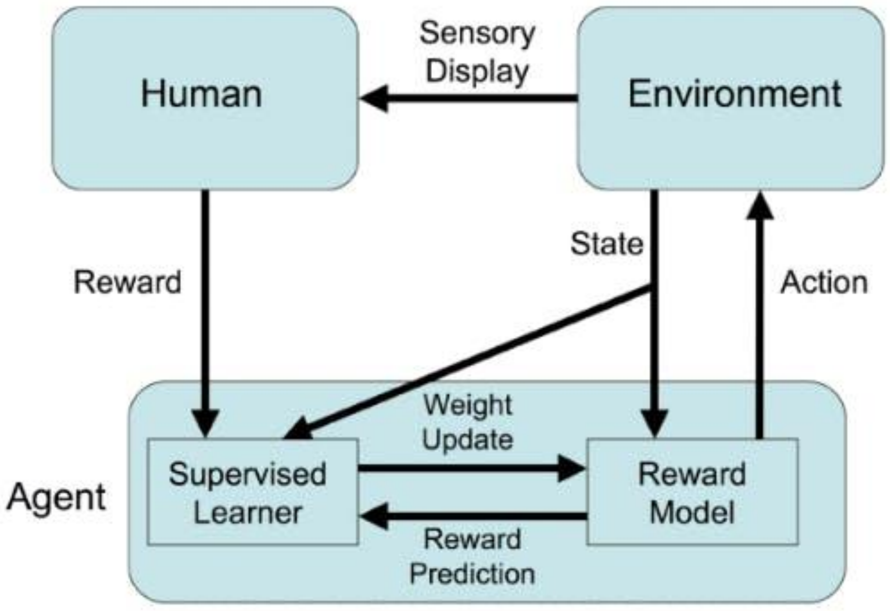


Figure 10. RLHF algorithm working scheme [29].

One of the key motivations behind RLHF is the resolution of the challenge of sample inefficiency, which is pervasive in conventional reinforcement learning. Contemporary reinforcement learning methodologies often necessitate extensive datasets, comprising thousands or even millions of samples. Furthermore, these methods are susceptible to catastrophic failures during training. Incorporating human expertise into the learning process seeks to ameliorate these limitations. Human guidance, informed by knowledge and experience, aids in curtailing the number of requisite samples for effective learning [30] [27].

Within the realm of RLHF, human experts can proffer advice to the agent when deemed necessary, thus the agent then translates this guidance into action, expediting its learning curve [30]. In our project, the human interaction manifests through the rewards the agent receives, that are used to train the algorithm in real time. In turn, the humans are able to see the predictions made by the algorithm, having the chance to optimise their decisions taking that extra information into consideration.

Complementary, when the shifters are informed by the experts of changes in the operation conditions that lead to effective changes in what is considered as good or bad, the new labelling by the shifters will impact the training of the algorithm, to adapt naturally to the updated regime of operation.

2.3.1 Key components of RLHF

The Agent is one of the main components inside an RLHF algorithm, it is an entity that engages with an environment to acquire a set of behaviours aimed at optimizing a reward signal. This agent essentially assumes the roles of decision-maker and learner within the framework of RLHF. The primary objective of the agent revolves around the acquisition of a policy, a rule or strategy, which establishes a mapping between different states within the environment and corresponding actions. The overarching aim of this policy is to effectively maximize the cumulative reward garnered over a period of time.

Another key component of a RLHF algorithm is the action space, it refers to the set of all possible actions that an agent can take in response to the observations it receives from the environment. It can assume either a discrete or continuous form, contingent upon the specific nature of the task at hand.

Within a discrete action space, the agent's choices are confined to a finite and predefined assortment of actions. For instance, these actions might involve moving in distinct directions, such as left, right, up, or down. In contrast, a continuous action space endows the agent with an extensive spectrum of possible actions, essentially allowing for an infinite range of choices. An illustration of this could be the continuous and uninterrupted act of kicking a ball towards a goalpost with varying degrees of force and direction.

Finally, it is important to mention the “model”, this term alludes to the agent's inner depiction or representation of the environment it engages with. This model serves a dual purpose: It can be employed to forecast the forthcoming state of the environment based on the prevailing state and the action taken, or it can simulate a series of conceivable future states and the associated rewards that may ensue from those states. Essentially, the model allows the agent to anticipate and plan for potential outcomes within its interaction with the environment.

2.3.2 Related applications of RLHF

Reinforcement Learning with Human in the Loop (RLHF) has been applied in various domains, such as autonomous driving, robotics, and data quality, solving complex problems by integrating human expertise and machine learning algorithms.

For example, in the study titled "Human-in-the-Loop Deep Reinforcement Learning with Application to Autonomous Driving," a method known as Human-Guidance-Based Deep Reinforcement Learning (Hug-DRL) was developed to facilitate policy training for autonomous driving. This innovative approach introduced a mechanism enabling seamless interaction and control transfer between human operators and automated systems.[31]

Importantly, it empowered humans to intervene in real time and rectify the actions undertaken by the autonomous agent during the model training phase. This strategic integration of human guidance yielded tangible improvements in both the efficiency and performance of deep reinforcement learning, a noteworthy advancement in the field.[31]

To validate the effectiveness and practical utility of the Hug-DRL method, a series of human-in-the-loop experiments was conducted, involving the active participation of 40 subjects. The outcomes of these experiments compellingly demonstrated that the Hug-DRL methodology indeed holds the capacity to significantly enhance the training efficiency and overall performance of deep reinforcement learning algorithms when they operate under the judicious guidance of human expertise and oversight [31].

Another example is the study titled "DQN-TAMER: Human-in-the-Loop Reinforcement Learning with Intractable Feedback" which used RLHF to overcome challenges related to exploration in reinforcement learning, a large obstacle in applying RL to robotics. The researchers proposed an RL method called DQN-TAMER, which efficiently uses both human feedback and distant rewards. [32]

They found that DQN-TAMER agents outperformed their baselines in Maze and Taxi simulated environments. Furthermore, they demonstrated a real-world human-in-the-loop RL application where a camera automatically recognizes a user's facial expressions as feedback to the agent while the agent explores a maze [32].

In the field of anomaly detection, it is possible to mention the following studies, In the former one called "Toward Deep Supervised Anomaly Detection: Reinforcement Learning from Partially Labelled Anomaly Data"[33], the authors tackled the problem of anomaly detection with a small set of partially labelled anomaly examples and a large-scale unlabelled dataset, a common scenario in many important applications.

They proposed a deep reinforcement learning-based approach that enables an end-to-end optimization of the detection of both labelled and unlabelled anomalies. This approach learns the known abnormality by automatically interacting with an anomaly-biased simulation environment, while continuously extending the learned abnormality to novel classes of anomaly (i.e., unknown anomalies) by actively exploring possible anomalies in the unlabelled data.

This is achieved by jointly optimizing the exploitation of the small, labelled anomaly data and the exploration of the rare unlabelled anomalies. The results showed that their model outperformed five state-of-the-art competing methods.

The second study, which has a similar context to the thesis project, is called “Outlier Detection with Reinforcement Learning for Costly to Verify Data" by Michiel Nijhuis and Iman van Lelyveld, published in Entropy in 2023 [34], presents a novel approach to outlier detection in data.

The authors recognize that outliers in data, which could be either data errors or informative deviations from the norm, are common. While these outliers can often be verified, the process is usually manual and therefore expensive. Furthermore, the underlying issues causing these data errors can change over time, necessitating a dynamic approach to outlier detection.[34]

To address these challenges, the authors propose an approach that combines reinforcement learning with statistical outlier detection. Specifically, they employ an ensemble of tried-and-tested outlier detection methods and use a reinforcement learning approach to tune the ensemble's coefficients with every additional piece of data. This enables the system to continually learn and adapt to changes in the data.

The authors demonstrate the effectiveness of this approach using granular data reported by Dutch insurers and pension funds under the Solvency II and FTK frameworks. They show that the ensemble learner can successfully identify outliers and that the reinforcement learner can further improve the results by optimizing the coefficients of the ensemble learner [34].

Despite the fact of several projects of RLHF in the anomaly detection field, to the best of our knowledge, the present project is the first application of reinforcement learning for data quality and/or anomaly detection in particle physics. However, other experiments at CERN have explored different ML models.

For example, the CMS collaboration in their study " Machine Learning applications for Data Quality Monitoring and Data Certification within CMS" by Vichayanun Wachirapusita, published in Iopscience in 2023 [35], discusses the use of ML techniques, such as Non-negative Matrix Factorization (NMF) and one-dimensional autoencoders, for anomaly detection in histograms. NMF is used to decompose histograms into smaller components, allowing for the reconstruction of good histograms and the identification of bad histograms with errors. Autoencoders are trained with known good histograms to reconstruct input histograms and detect anomalies based on the reconstruction error.

In Figure 11 it is possible to see a reconstruction example of a one-dimensional autoencoder trained with data recorded in 2017. Left: histogram from a good sample; and right: histogram from an anomalous sample. Each plot shows different types of histograms or monitoring elements. The blue histograms represent histograms that are used to train an autoencoder. The red histogram is an anomalous reconstruction of the autoencoder when compared to the original histogram in black.

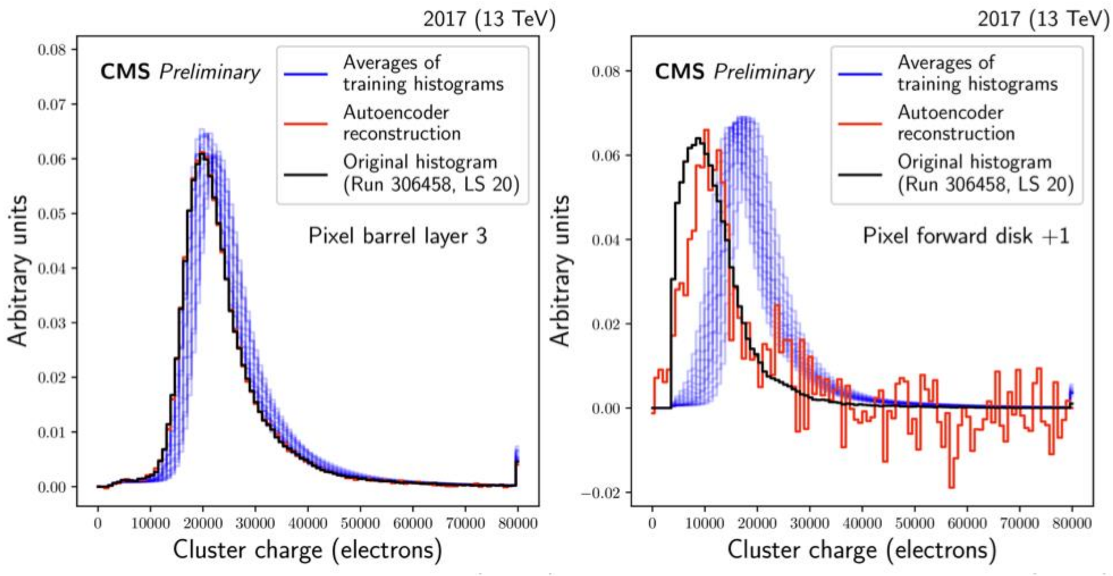


Figure 11. Reconstruction example of a one-dimensional autoencoder trained with the proposed solution of the CMS collaboration.

However, the proposed method suffers from the same problems studied in Chapter 1, specifically the training data suffers from a class imbalance problem, as most anomalies are removed by human experts during data-taking. Moreover, Human-based labels may also be incorrect, and ML algorithms may miss anomalies that have not been seen before. Therefore, the ML system is intended to assist humans rather than replace them entirely [35].

2.3.3 Advantages and limitations of RLHF

RLHF offers a powerful methodology for refining AI systems. However, like any other approach, it comes with both distinct advantages and potential challenges.

Considering the benefits of RLHF is important to mention the adaptability of this technique since it is a dynamic learning method capable of adapting to received feedback. This flexibility renders it suitable for a wide range of tasks, allowing it to fine-tune its behaviour based on real-time interactions [36].

Besides, RLHF has the potential to mitigate model biases. By incorporating diverse and thoughtfully chosen human feedback, these models can learn from a more comprehensive and representative perspective, minimizing overgeneralization and biases inherited from initial training data. Finally, through ongoing interactions and feedback from users, RLHF models possess the ability to continually enhance their performance in both functionality and user experience over time [36].

Despite its advantages, RLHF has some limitations. In fact, one major challenge is the scalability and cost of human feedback Scaling it to cater to larger or more complex tasks can be both resource-intensive and time-consuming. Another problem comes from the nature of this method since it relies on human feedback. Thus, the effectiveness of RLHF is dependent on the quality of this feedback and If lacks impartiality or is inconsistent or incorrect, the model may become biased.

Nevertheless, it's crucial to highlight that there exist viable methods for alleviating these biases. Selecting a diverse group of evaluators, implementing consensus-based assessments, fine-tuning evaluator judgments, conducting ongoing assessments of both the feedback process and agent performance, and integrating feedback from multiple origins all have the potential to diminish bias effects in RLHF [36].

Chapter 3

RLHF tools and algorithms

This chapter describes the library and the RLHF algorithms used in the experiments. In particular, the configuration and main features are described.

3.1 Ray RLlib

To build our RLHF algorithm we are using RLlib, an open-source library for RL built on Ray, a popular framework for distributed computing, that offers a comprehensive set of features, including support for multiple deep learning frameworks such as TensorFlow and PyTorch, highly distributed learning, vectorized and remote environments. It also provides support for a wide range of RL algorithms for both model-free and model-based RL, and multi-agent RL [36].

Moreover, RLlib allows users to customize all aspects of their training and experimental workflows. This includes coding custom environments, providing custom TensorFlow/Keras or Torch models, defining custom policy and loss definitions, and defining custom exploratory behaviour [36].

Regarding the monitoring aspect, Ray provides a web-based dashboard for debugging its applications. The visual representation of the system state allows users to track the performance of applications and troubleshoot issues. This monitoring tool will be complemented with a TensorBoard dashboard.

TensorBoard enables tracking experiment metrics like loss and accuracy, visualizing the model graph, projecting embeddings to a lower dimensional space, and in our case, it will help us to track in real-time the custom metrics that we will establish.

An RLlib environment consists of an action space, which is basically the codification of all possible actions that the agent can take, a complete description of the environment, where all the data is visible, that is called state space and is different from the observation space, the part of the current state that the agent can see in each episode. Finally, a reward is the only feedback the agent receives per action [36].

The decision-making model that tries to maximize the expected sum over all future rewards is called a policy. The policy is a function mapping the environment's observations to an action to take, usually written $\pi\,(s(t))$ à $a(t)$. The simulation iterations, described in Figure 12, of action → reward → next state → train → repeat, until the end state, is called an episode, or in RLlib, a rollout [36].

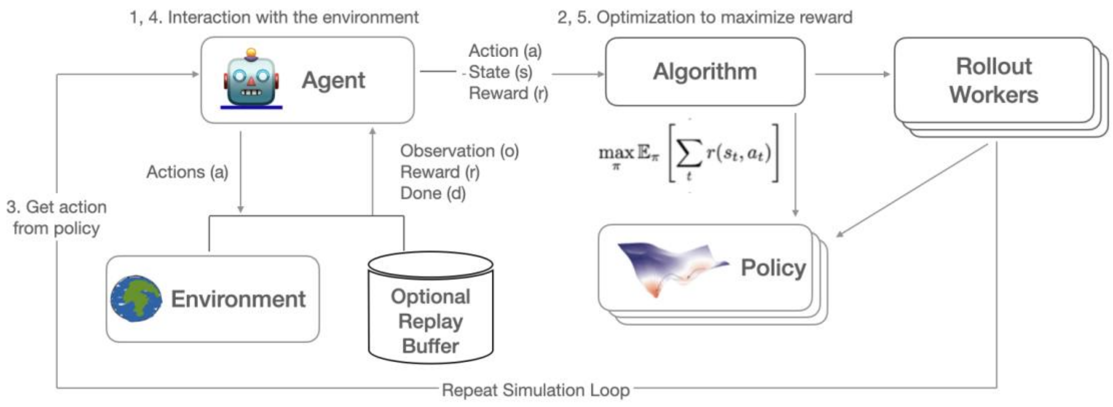


Figure 12. Diagram of the RL iterative learning process in Ray RLlib.

In Figure 12, we see another element of Ray RLlib, a rollout worker. This is a process that interacts with an environment and collects experiences, or samples, which are then used to update the policy of the agent. Rollout workers can run in parallel, each interacting with its own copy of the environment. This allows RLlib to scale the data collection process over multiple cores or even multiple machines [36].

3.2 Proximal Policy Optimization algorithm (PPO).

The algorithm that will oversee optimizing the mean reward will be the Proximal Policy Optimization (PPO). This is a type of Reinforcement Learning (RL) algorithm that has gained popularity due to its balance between performance and comprehension. It's considered a state-of-the-art RL algorithm.

PPO's main strength lies in its ability to balance the trade-off between exploration (trying new actions) and exploitation (sticking with known good actions). It does this by utilizing a novel objective function that minimizes the cost function while ensuring the deviation from the previous policy is relatively small [37].

The PPO algorithm works by sampling a batch of data by playing the policy in the environment for a given number of steps. It then performs a given number of optimization steps with random sub-samples of this batch. This puts a pessimistic bound on the loss: lower return estimates are favoured compared to higher ones [38].

After a comparison with another popular RL algorithm, Deep Q-Network (DQN), the PPO has been chosen due to the fact that it tends to be more sample efficient than DQN. This means that PPO can learn effectively with fewer interactions with the environment than DQN. In detail, PPO's improved sample efficiency is due to its use of multiple epochs of stochastic gradient ascent to perform each policy update, which better optimizes the policy at each iteration [37].

What’s more, PPO is generally more stable and less sensitive to hyperparameters than DQN. In PPO, the policy update is constrained to be close to the current policy, which prevents the policy from changing drastically from one update to the next and helps to maintain stable learning [37].

Finally, PPO is simpler and easier to implement than DQN. DQN requires the use of a replay buffer to store and sample past experiences, which can be complex to manage. PPO, on the other hand, does not require a replay buffer, making it simpler and less memory intensive. [36]

The PPO) algorithm has several important parameters that control its behaviour, the most relevant in our context are the following:

- **Gamma (γ)**: This is the discount factor used in the calculation of returns. It determines the present value of future rewards: a factor of 0 makes the agent "myopic" (only considering current rewards), while a factor approaching 1 will make it strive for a long-term high reward. It's a way of balancing immediate and future rewards [39].

- **Kl coefficient**: This is the coefficient for the KL divergence in the loss function. KL divergence measures how one probability distribution diverges from a second, expected probability distribution. In PPO, KL divergence is used to ensure the new policy doesn't deviate too much from the old one. The Kl coefficient parameter controls the extent to which this deviation is allowed [39].

- **Clip param**: This is the clipping parameter ε, used to limit the policy update step to avoid too large updates. The policy update is clipped to be in the range of $[1-\varepsilon, 1+\varepsilon]$, which helps to keep the new policy close to the old one, preventing harmful large policy updates [39].

- **Lr (Learning Rate)**: This is a hyperparameter that determines the step size at each iteration while moving toward a minimum of a loss function. In other words, it controls how much to change the model in response to the estimated error each time the model weights are updated. Choosing the learning rate can be challenging as a value too small may result in a long training process that could get stuck, whereas a value too large may result in learning a sub-optimal set of weights too fast or an unstable training process [36].

Chapter 4

Data Collection and Pre-processing

In this fourth chapter we will analyse what type of data is available to use in the training of the framework and outline the reasons why we need to create a synthetic dataset, Moreover, the hyperparameters that rule the synthetic dataset and other key features will be described.

4.1 Real data collection

The data acquisition of Run 3 is currently happening, thus there is not a proper amount of data that we can use for training the algorithm. There could be the option to use the data collected in the previous Run (Run 2) that has already been labelled. However, as explained in the previous paragraph both the LHC and experiments have suffered from major upgrades and improvements.

The former one can now deliver more particles per bunch at higher energies and most of the sub-detectors of the latter have been replaced with new ones that can collect more and with higher precision data, thus achieving a better performance. That also implies that they are more sensitive and since that, the nature of the anomalies can be different. For example, a signal that before was not captured or not considered now can be detected and thus can originate a new kind of anomaly or not.

Besides, the available dataset containing the data from Run 2 has a total amount of 229 elements that are divided into 63 well-labelled histograms and 166 flagged as anomalies, therefore working with a small and unbalanced dataset such this introduces several challenges and can lead to poorer model performance. In fact, When the dataset is small, there is a higher risk of overfitting, the model might perform well on the training data but poorly on new, unseen data.

Also, it is important to mention that models trained on imbalanced data frequently encounter challenges in accurately categorizing the minority class. In this case could happen that not all the anomalies are represented, and they could not be accurately detected in the real case scenario.

Nevertheless, the data coming from the Run 2 can be used as data test in a future step to properly validate the model. If the algorithm is able to properly classify the signals there will be a high chance that it will be able to do the same task with the data of Run 3.

4.2 Synthetic dataset for the experiments

We will create a synthetic dataset that will be used to train and test the model, and to study the shifter-algorithm interaction. In detail, it will contain all the statically relevant anomaly types that can be found in the real-case scenario. Finally, to create such dataset we must understand what the data is made up of and what it means.

The main reason to start with this kind of dataset is that this is a novel approach, so we want to develop and study the algorithm in a simplified scenario where we have full control of the nominal and anomalous histograms and are not limited by the statistics of the training sample.

4.2.1 Data explanation

Each histogram represents the dataset distribution over a certain variable. The histogram is made up of n points called bins; every bin contains the number of occurrences (for example particle hits in a sub-detector) that resulted in a value within a certain range (the bin limits) for the specific variable. Not only the value of specific bins but also the shape of the overall distribution gives information about the dataset.

In working conditions, if for example we measure the energy of the particles and we make a histogram with such information we expect one that may be expected to follow a normal distribution with mean μ and standard deviation σ. The registered value of each bin will not be always the same, since collisions happen randomly, and the number of occurrences will statistically fluctuate around the mean values. To emulate this effect in our dataset, we artificially introduce a fixed level of statistical noise (from $\pm 1\%\ to \pm 5\%$ approximately), around all the desired distribution shapes.

In Figure 13, you can observe a representation of what it looks like; for each bin, we have an average value $\pm$ the statistical noise, where the statistical noise itself follows a normal distribution (dark green curved line in y axe). The resulting value will be still considered that belong to the normal distribution in a working scenario if it is inside certain limits. The tolerance limits are represented with the green dot line.

Independently of the statistical noise, it is the underlying shape of the distribution that defines if a dataset is good or bad. In this case, since the reference distribution is considered to be a certain Gaussian function, a dataset will be good if it follows the same Gaussian function, and bad otherwise. The underlying distribution for anomalous datasets can potentially be of any kind. In order to generate the data, we need both elements, the underlying distribution shape (anomaly or nominal), and a random function of a certain type (here a Gaussian function is considered) to add some level of statistical noise independently for each bin.

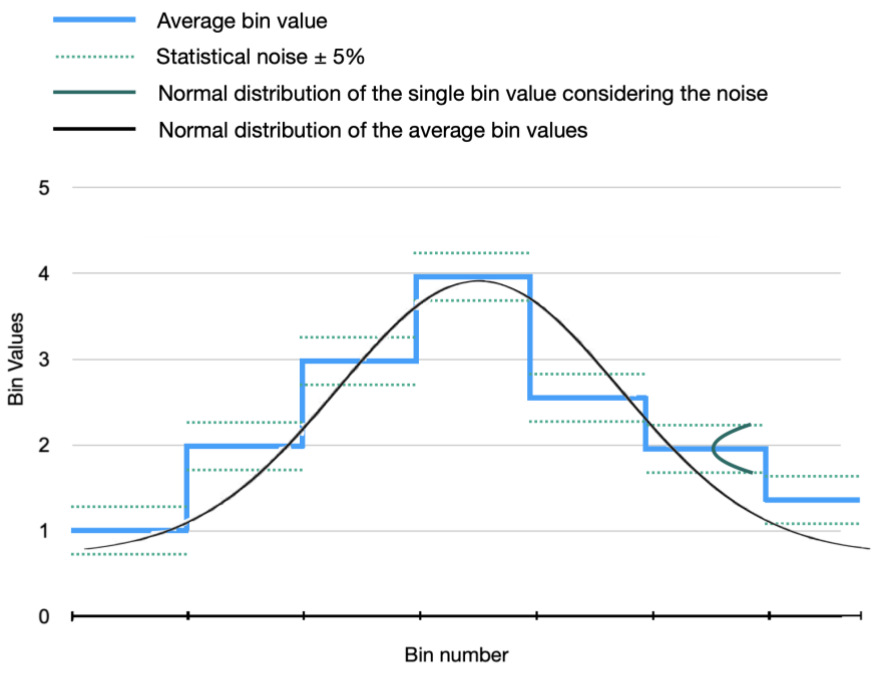


Figure 13. Normal distribution histogram sketches with statistical noise limits per each bin.

The lower the statistical noise, the easier it is to identify the nominal function, and the higher the statistical noise, the harder it is to identify the nominal function. In Figure 14, you can appreciate the difference between two good histograms that follow normal distributions with the same mean and standard deviation, but with different levels of statistical noise, the former one has $\pm$ 1% and the latter $\pm$ 5%.

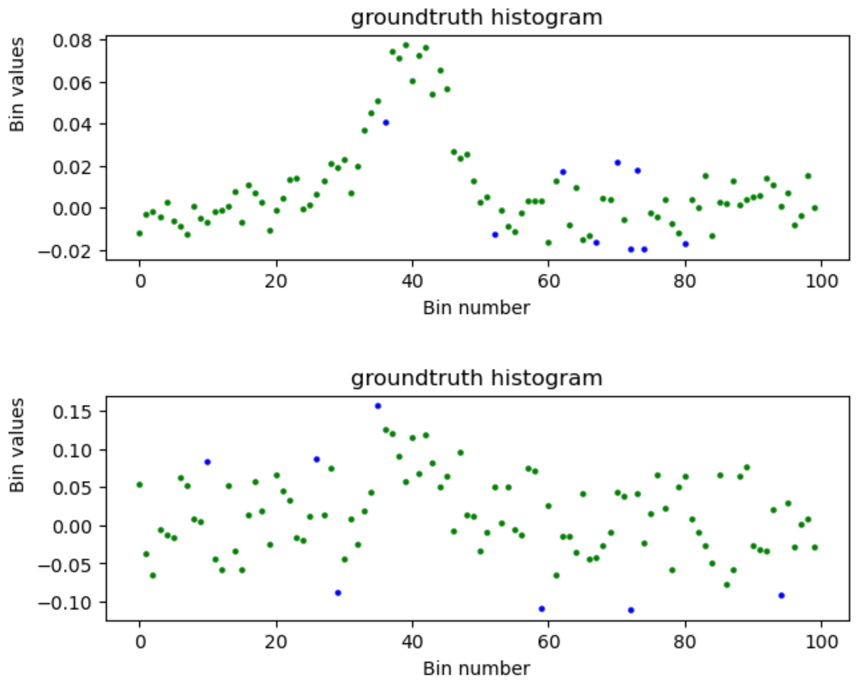


Figure 14. Good histograms with different levels of statistical noise. First $\pm$1%, second $\pm$5%

In the case that we have an anomalous signal, for example, the energy counter gets lower values due to errors in the detector, we will have an anomalous distribution but still, the value in each bin will have the statistical noise.

It is possible to have other kinds of anomaly distributions caused by different factors, such as one or more counters dropping to zero, this will generate “holes” in the histogram. In this case, the shape will be close to the nominal one but in fact, is an anomaly. Another case could be that there is an error in the software that reads the data, and we have a flat histogram since all the values are the same.

In summary, we identify what is the nominal distribution, but we do not know what the anomalies will be, and depending on the values of the intrinsic statistical noise the recognition task could be more difficult or easier. Moreover, the difficulty of the recognition task will also depend on how similar the anomalies will be to the nominal distribution.

Since statistical noise will always be present in data, determining if a dataset is good or bad is not a clear-cut task. One needs to define some sort of threshold on how similar two datasets need to be to consider them as coming from the same underlying distribution. The thresholds can either be defined mathematically over the bin values or be implicitly present in the intuition of the shifters, that try to judge whether a slightly deformed histogram is really an anomaly or just the result of relatively unlikely but possible statistical fluctuations.

In our dataset, even though we are classifying the histograms and not the single bins, there are also mathematically defined thresholds over the bin values for a bin classification. When a nominal distribution is generated, the values of the bin that correspond to the limits of the distribution confidence intervals are obtained (Figure 15). This is also done for the distribution that generates the statistic noise. The noise values are added to the bin values and this result is compared with the nominal distribution confidence level limits.

If the value is inside the area limited from plus or minus two standard deviations from the mean (green area of Figure 15), then even though it is sampled from another distribution, is statistically compatible with the nominal one (could have been sampled from the nominal) and this bin is labelled as good. If the value belongs to the area between $\mu - 2\sigma$ and $\mu - 3\sigma$ or $\mu + 2\sigma$ and $\mu + 3\sigma$ (Figure 15 blue areas), is still possible that this bin could have been sampled from the nominal distribution, but the probability is very low, so it will be labelled as an outlier. If the value is below $\mu - 3\sigma$ or above $\mu + 3\sigma$ (Figure 15 red areas), it will be considered as an anomaly bin.

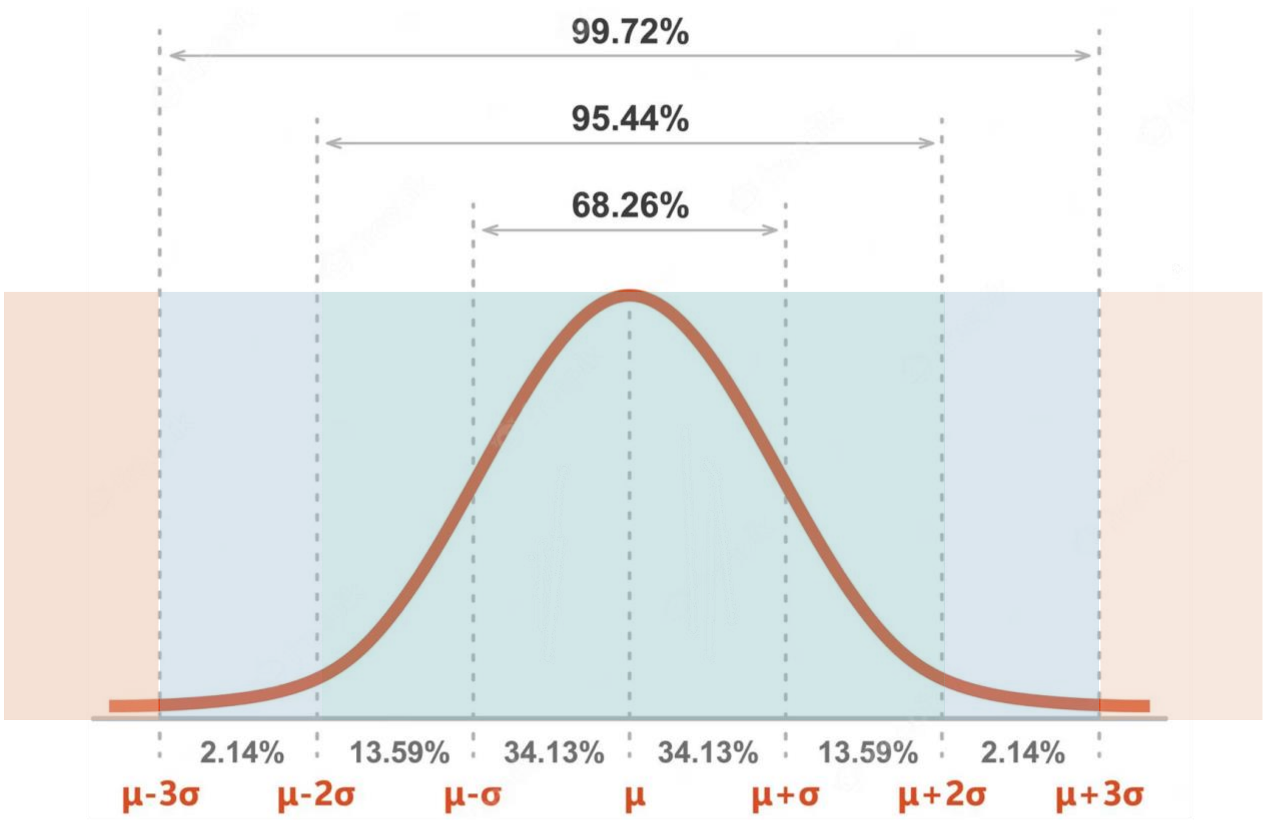


Figure 15. Standard normal distribution curve with standard deviation indicators and different label zones for bin classification.

4.2.2 Dataset hyperparameters overview

The synthetic dataset is governed by several hyperparameters; modifying one or more of them will alter the resulting dataset samples. The following paragraphs will provide a brief and concise explanation of them.

First, we establish how many bins will be contained in all generated histograms. Once we have decided this value, we need to set how many histograms will be contained in our dataset, (both values must be a positive integer) and which percentage of them will be a nominal type. E.g., it is possible to establish that 50% of the histograms will be good data or any other percentage.

Another parameter to set is the types of anomalies we wish to generate and the distribution they should follow. To encompass the widest possible range of real-world scenarios, we can adopt one of two modes. In the former, the anomalous values will follow a single available distribution from the ones explained below:

- $bin\_value \sim Uniform(a, b)$. Value sampled from a uniform distribution where a and b are the parameters representing the minimum and maximum values of the distribution. These values can be a fixed number or can be set as a boundary tuple, the algorithm itself will choose every time a random value that is contained between the upper and lower limit.

- $bin\_value \sim Exp(\lambda)$. Value sampled from an exponential distribution where λ is the parameter of the exponential distribution, which is the rate parameter (the average number of events in a unit interval). Also in this case can be a fixed number or a boundary tuple with the upper and the lower limit.

- $bin\_value \sim Gaussian(\mu, \sigma^2)$. Value sampled from a Gaussian distribution with mean μ and variance σ^2. Both parameters can be chosen randomly between defined limits or assigned directly.

So, we will only have one type of anomaly. In the latter, the anomaly type can be randomly selected from a list of possible distributions with equal probabilities. For example, if we include all the possible types, we have the following probability:

$$P\big(x \sim Uniform(a, b)\big) = P(x \sim Exp(\lambda)) = P\big(x \sim Gaussian(\mu, \sigma^2)\big) = \frac{1}{3}$$

It is worth noting that the list of possible choices does not have to include all types of anomalies. For instance, we might want to generate anomalies following only Gaussian or exponential distributions with the same probability. This will be useful if we aim to test the behaviour of the agent with certain anomaly groups.

The possible distributions are shared also with the nominal histograms with the only difference that in the case of the nominal, it must follow one and only one distribution between the proposed. Table 1 below shows a resume of the hyperparameters of the synthetic dataset.

Name	Short description	Possible Values
Anomaly distribution type	Distribution followed by the anomalous histograms.	$x \in \{gaussian, uniform, exponential, random\}$
Nominal distribution type	Distribution followed by the nominal histograms.	$x \in \{gaussian, uniform, exponential\}$
Anomaly distribution choice	If “*random*” is selected as a value of “*anomaly distribution type*”, list of the possible distributions that an anomalous histogram will follow.	$x \in \{gaussian, uniform, exponential\}$
Histograms per dataset	Number of histograms that made up the dataset.	$x \in \mathrm{Z}^+$
Bins per histogram	Number of bins that made up one histogram.	$x \in \mathrm{Z}^+$
Percentage of nominal data	Percentage of nominal histograms in the dataset.	$x \in [0,100] \cap Z$

Table 1. Synthetic dataset hyperparameters resume.

4.2.3 Dataset samples

Once the hyperparameters of the dataset are defined and as well are set the parameters for each function, the dataset is generated. In an ideal case, we would like to cover all the possible relevant cases of anomaly and we wouldn’t like the agent to be trained on specific types of anomalies (however this is a possible scenario) thus for that reason, we want to include all possible types of anomalies.

Moreover, for each type of anomaly we want to generate distributions with different parameters to be sure they represent most of the possible events. For that reason, we define the upper and lower limit for each function parameter and let the algorithm choose a value between those with equal probability.

The only parameter that is statically defined and does not change inside the dataset is the mean and the variance of the Gaussian distribution from where are sampled the nominal histograms. Thus, the selected values cannot be used for creating anomalous histograms.

Figure 16 represents a sample of the dataset that will be used to train the agent. It is possible to observe one type of anomaly per distribution and the nominal histogram. Some of the histograms are easy to recognize at first sight, but especially the ones that are sampled from the same type of distribution that is used for the nominal ones (but with different mean and standard deviation) could be hard to classify also for the most trained shifters.

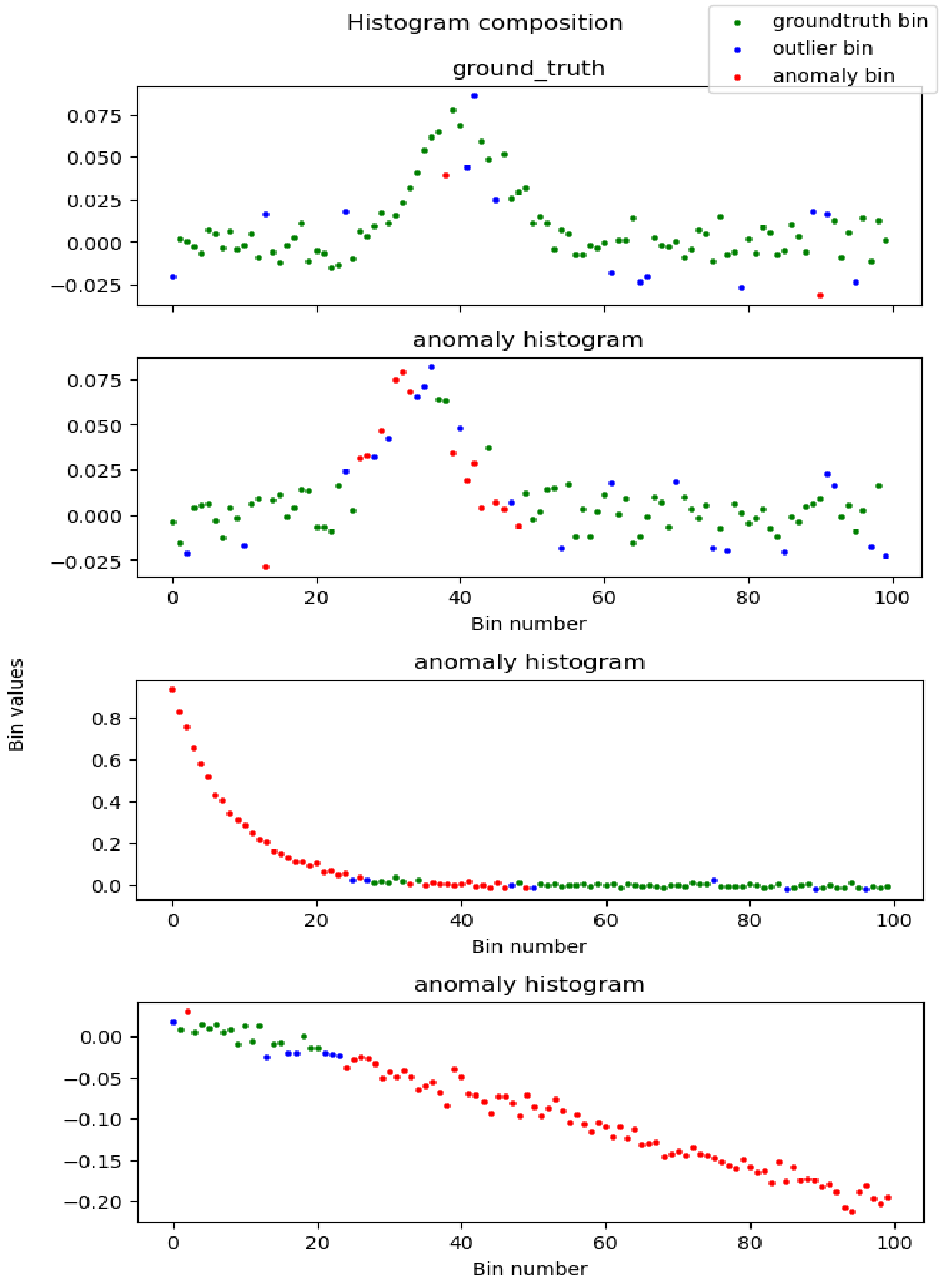


Figure 16. Synthetic dataset histograms samples

Chapter 5

Methodology and work conducted

In this chapter, we will describe the actual subject of the thesis and how the innovative project was carried out in the context of the LHCb experiment. Starting from how the environment was created to enable all the experiments, the trial and test steps and the exploration of the superhuman state**.**

5.1 Environment setup and agent workflow

In our environment we will have only one agent that is able to perform two discrete and deterministic actions:

1. Classify the histogram as anomalous.
2. Classify the histogram as nominal.

If the agents classify properly the histogram will receive 1 point, otherwise will be penalized with 1 point (reward of -1). In our scenario, since we have different types of anomalies, and we want to be able to classify all of them (not relevant if the algorithm can achieve the max reward only with one type) our best model will be the one with the higher mean reward and thus this will be the parameter that the algorithm will try to maximize. The mean reward is calculated as the mean of all the rewards achieved during one train iteration. In one train iteration, the agent will analyze all the histograms that are in the dataset, grouped for the batch size.

The observation space is a vector that is made up of all the bins that the histogram has, with the agent having access to all of them at the same time. Instead in the environment we will have one vector containing all the histograms of the dataset and another one that will contain the labels of the data, so at which class the histograms belongs, anomaly or nominal.

The final state of the training will be reached either if a specified mean reward is reached (mean reward equal to max reward) or all the programmed iterations have been done. To resume the workflow of the agent will be the following:

1. The environment will be initialized by creating the agent and loading the dataset, storing the histogram values in one vector and the corresponding labels in another one.
2. In every training episode the agent will have to classify the histogram that it will be provided and will get a reward depending on the correctness of its choice.

3. After seeing all the algorithms in the dataset, the reward mean will be calculated, and this will be the variable that the algorithm will have to maximize changing its behaviour.

5.2 RL agent's hyperparameters tuning.

The algorithm parameters described in Chapter 3 are typically tuned to optimize the performance of the PPO on specific tasks. The ideal values can vary depending on the specifics of the task and the environment. To perform this research Ray provides a tuning library called Tune.

Tune allows you to take a Python function, define a search space of possible hyperparameters, and it will then manage the execution of many instances of this function with different hyperparameters across your available resources (whether that's on a single machine or a large cluster) [41].

Tune is designed to be compatible with any machine learning framework, and it integrates with many popular optimization libraries, such as HyperOpt, Optuna, and scikit-optimize. It also supports advanced scheduling algorithms, such as HyperBand and Population Based Training, to efficiently allocate resources and select the best hyperparameters [41].

To speed up the tuning process for finding the best parameters we are using Tune, a type of hyperparameter optimization algorithm that is called “scheduling algorithm”. These Trial Schedulers can early terminate bad trials, pause trials, clone trials, and alter the hyperparameters of a running trial.

The selected scheduling algorithm is the Asynchronous Successive Halving Algorithm (ASHA), It is a variant of the Hyperband algorithm, which is a resource allocation algorithm that is used in hyperparameter tuning to find the best hyperparameters in less time [40].

ASHA is based on the principle of Successive Halving (SHA) but makes a key modification: instead of waiting for all configurations in the current rung to complete, it promotes configurations to the next rung as soon as they finish. This makes ASHA an asynchronous version of SHA and allows it to utilize resources more efficiently. [36]

The ASHA scheduler works by assigning a certain number of resources (e.g., CPU time) to each configuration and evaluating its performance. The configurations with the worst performance are pruned, and the remaining configurations are given more resources for further evaluation. This process is repeated until only one configuration remains, which is returned as the best configuration [40].

This approach allows ASHA to quickly identify promising configurations and allocate more resources to them, while discarding poor configurations early on, leading to more efficient use of resources. To determine the optimal setup for our environment, we have tested all the possible combinations of the parameters listed in the Table 2.

Parameter name	Possible values
Learning rate	$[1e-3, 1e-4, 1e-5]$
Clip param	[0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]
Kl coefficient	[0.0, 0.1, 0.2, 0.3, 0.4, 0.5]
Gamma discount factor	[0.5, 0.6, 0.7, 0.8, 0.9]

Table 2. Hyperparameter values used in the tuning process.

Moreover, regarding the learning rate, two different approaches were tested. The first approach involved maintaining a static learning rate throughout training, with the starting value remaining equal to the finishing value. The second approach involved periodic decreases in the learning rate, starting from 1e-3 and finishing with 1e-5. These decreases were implemented when the iteration number corresponded to 25%, 50%, and 75% of the total number of iterations set.

For the tuning process, we used a dataset where 50% of the histograms were anomalous and all types of the anomaly were equally represented. The best result is the one that has the best mean reward achieved during the training phase, and in this case, a mean reward of 0.99 over 1 has been achieved using the following values shown in Table 3.

Parameter name	Best value
Learning rate	Periodically changed, starting at $1e-3$ and finishing at $1e-5$
Clip param	0.3
Kl coefficient	0.0
Gamma discount factor	0.9

Table 3. Best algorithm hyperparameters values found after the tuning process.

5.3 Training overview

The training of the agent is done using the algorithms hyperparameters found before with the following dataset parameters that are shown in Table 4, for 12 iterations. In each iteration, the agent will see all the histograms once in batches of 512.

Name	**Short description**	**Possible Values**
Anomaly distribution type	Distribution followed by the anomalous histograms.	$random$
Nominal distribution type	Distribution followed by the nominal histograms.	$gaussian$
Anomaly distribution choice	If "*random*" is selected as a value of "*anomaly distribution type*", list of the possible distributions that an anomalous histogram will follow.	$gaussian,$ $uniform,$ $exponential$
Histograms per dataset	Number of histograms that made up the dataset.	1000
Bins per histogram	Number of bins that made up one histogram.	100
Percentage of nominal data	Percentage of nominal histograms in the dataset.	50%

Table 4. Dataset parameters used in the training phase.

The test phase, where the performance of the agent will be measured against a different dataset is normally done at the end of the training step. However, in our context at the end of every episode, we proceed to generate a synthetic test dataset, and we measure the agent performance. In that way, we can have an overview of how good is going the training of the agent measuring the effective accuracy, step by step, see how this is improving or not, and be able to establish a trend.

It is possible to Execute the training of the agent in two different modes that are described below:

- Human mode: In this mode, after the application of each agent's action, a screen that resumes what the agent has done will be displayed. As you can see in Figure 17, shows the observation space of the agent, so in this case, the histogram that he is classifying and the correctness of the action that he applied. The human supervisor can decide to resume the training at any moment, and in future steps decide to interact with it.

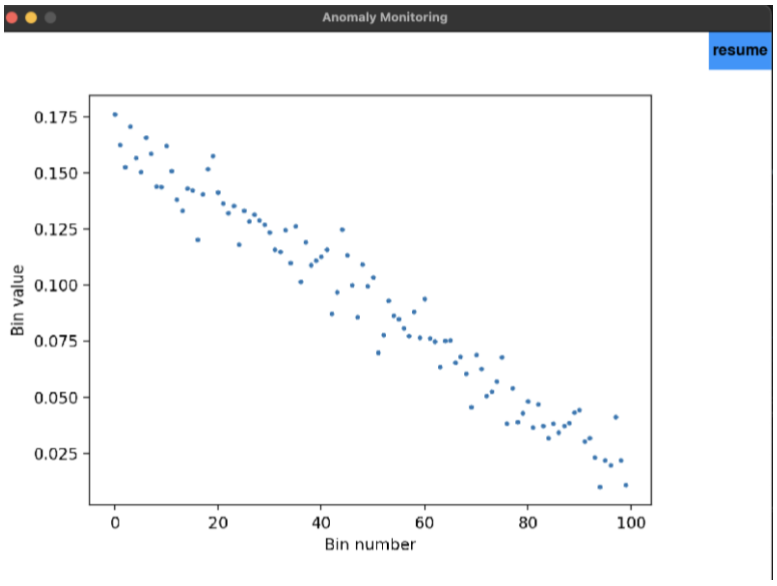


Figure 17. Human render mode during agent training.

- Machine mode: The training will be performed at the speed of the algorithm and cannot be paused. Feedback will be provided only if we had previously configured it, and it will show through the terminal.

After the first training phase, we can affirm that the algorithm is able to classify properly the anomalies and the nominal histograms. We reached approximately a mean reward in the training of 0.99 over 1 and a mean reward of 0.97 over 1 during the test phase (Approximately 97% of accuracy in the test) as you can observe from Figure 18.

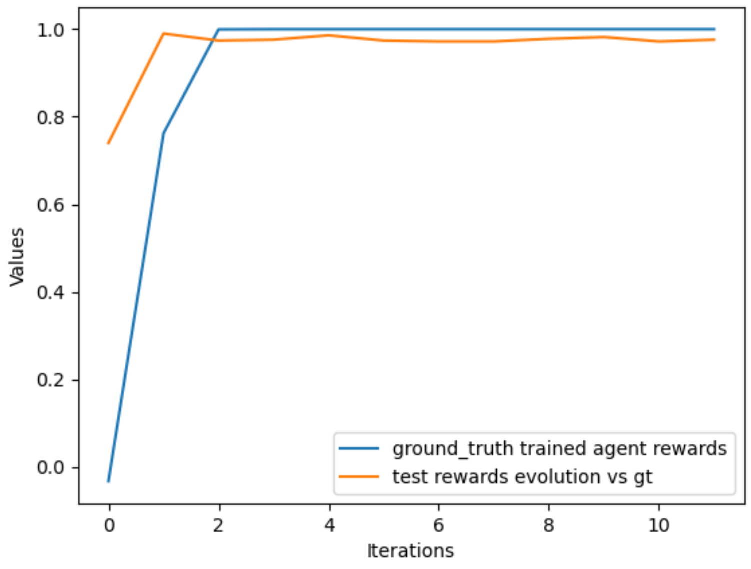


Figure 18. First agent training mean reward evolution in test and training.

In Figure 19 we can observe through the human mode of the algorithm, the results of the classification that the agent performs on the histograms, a red histogram means that the agent classified it as an anomaly, and a green one means that has been classified as nominal. In the bottom left corner can be observed the accuracy of the decision.

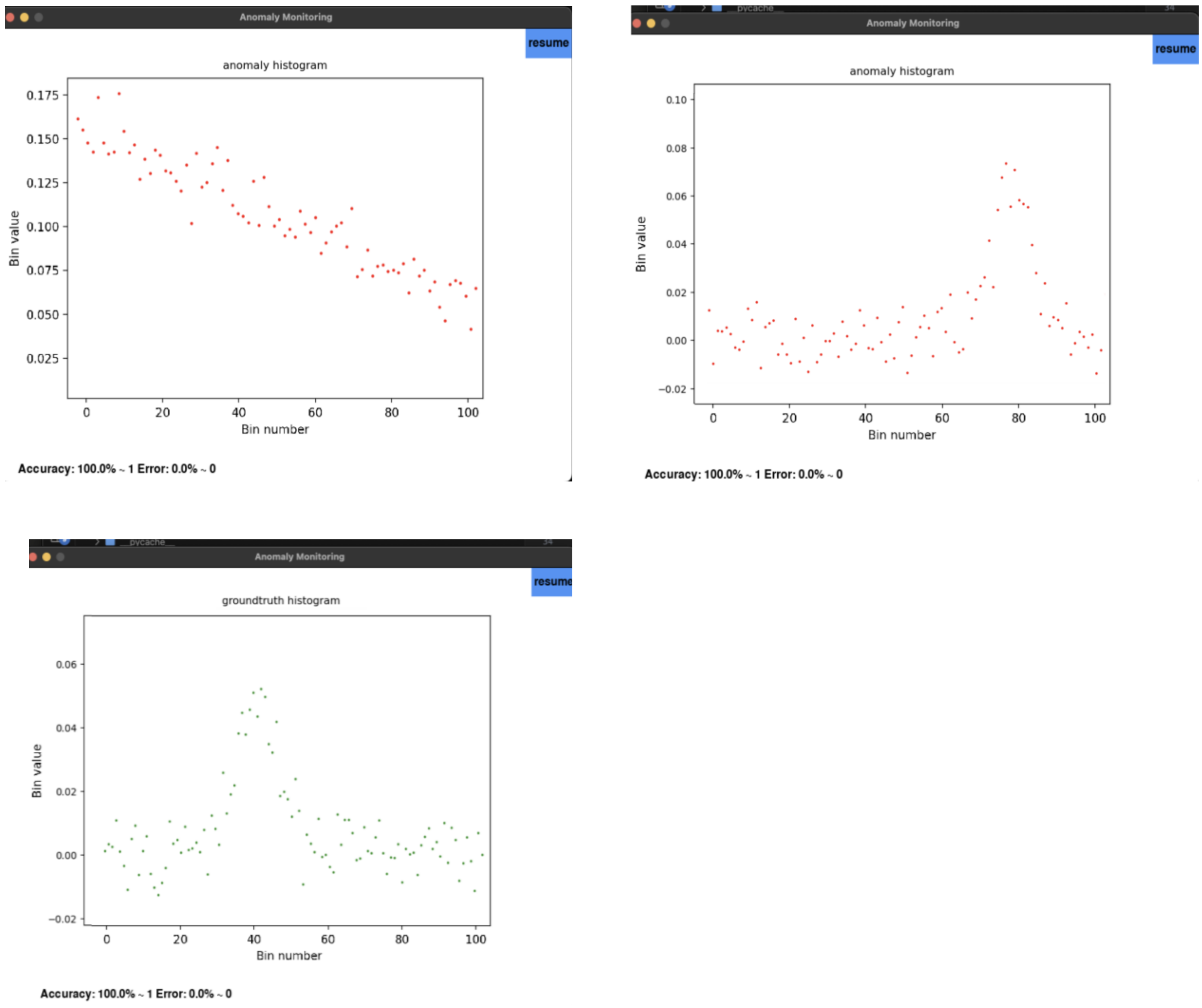


Figure 19. Anomaly classification evaluation through the human mode agent render.

However, in this first training simulation, we are training our agent with data that is 100% accurate, which means that is not biased by the possible human error factor and thus is not a faithful representation of the real-case scenario.

5.4 Human error simulation and machine-assisted human decision

As said in the previous chapters, since there is a human component involved in the training process, the process of how much humans make mistakes in training and how this affects the learning process of the agent has to be modelled. Furthermore, it is also needed to implement the workflow of how the human will believe in the algorithm and when he needs to start trusting the algorithm's decisions.

5.4.1 Human error implementation

Depending on several factors such as the experience of the shifter, the difficulty of the classification task, their level of attentiveness, and the shift schedule, shifters may misclassify the histograms. We can assume that the human error rate could be between 10% and 30% of the classifications.

To reproduce these conditions on the synthetic dataset we will create what we call “noisy labels”. A percentage of the labels, corresponding to the level of human error will switch class, which means that if we are considering that a human can classify wrongly 10% of the histograms, 10% of the labels will be changed, and a nominal histogram will be labelled as an anomaly and otherwise. However, the bin values will not be changed.

This will simulate the action of the human giving wrong feedback (indicating that the histogram is an anomaly or vice versa) to the agent, reproducing the human error behaviour. Three different human errors are tested, 10%, 20% and 30%. One training per human error level has been done, to observe how the agent performs and how much is affected by the different levels of human error and in the following chapter the results will be discussed. It is important to mention that the testing step is always realised by comparing the algorithm output with the ground truth labels and not the noisy ones.

5.4.2 Machine-assisted human decision implementation

The second behaviour that we want to model is the machine-assisted human decision, we aim to demonstrate that a training program where the human observes and trusts the agent's decision to a certain degree leads to convergence of the training process, resulting in performance levels beyond human capability achieving a super-human state.

The shifter always needs to give feedback to the algorithm, which is crucial for example if the shifter is instructed by the experts to change what is considered to be nominal at some point in time; however, on average the shifter can start "delegating" a fraction of the decisions to the algorithm, depending on the level of trust on it. How can we model this “level of trust”?

The main idea is to model that the human will start to believe in the algorithm when the mean reward obtained in previous training iterations is greater than an established threshold and will not trust it when the mean rearward is below that limit. This answer to the fact that until a certain point, the human will be more expert than the agent itself, so the shifter must continue to give feedback to the training and ignore the algorithm and when this inflexion point is reached, he has to start to apply the suggestions of the algorithm stopping the feedback. This strategy is resumed in the equation below.

$$f(x) = \begin{cases} believe\ in\ algorithm, & mean\ reward \geq threshold \\ do\ not\ believe\ in\ algorithm, & mean\ reward < \ threshold \end{cases}$$

To accomplish this, we introduce a variable that reflects the degree of distrust in the algorithm by the human. This value is correlated to the accuracy of the algorithm itself (defined over the noisy labels, as this is the only quantity that the shifter will be able to evaluate) and can range between 0 and 1 where 1 is equivalent to the probability of 100% distrusting the algorithm and 0 (0% probability of not trusting) which means the human will always follow any suggestion from the algorithm.

It should be noted that if the human was to trust the algorithm's decisions 100% of the time this would collapse the training process since the agent would receive maximal reward for any possible decision in all of the cases.

The probability of distrusting the algorithm will be updated after every training episode taking into account the mean reward obtained by the agent and the established threshold. To do that we have generated a smoothed Heaviside function $f(x)$ (Figure 20) where x is the mean reward achieved during each iteration analysing the 1000 histograms, and K is the threshold established for the training, which will determine the new value for the distrusting probability.

$$f(x) = \tanh\big((x - K) * 5\big) + 1)/2$$

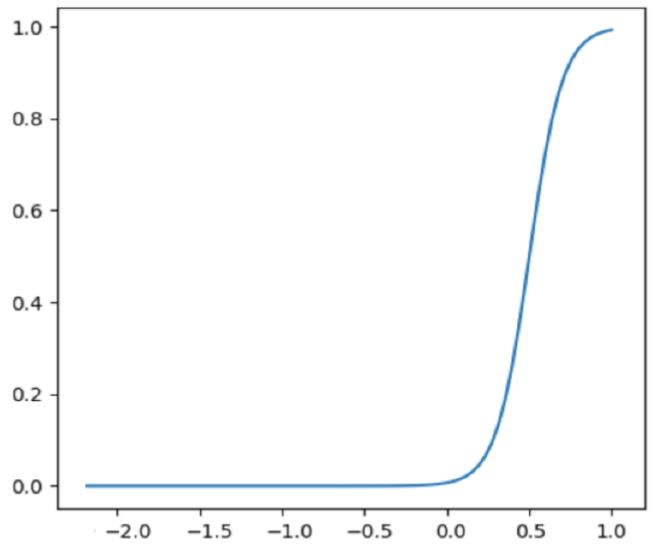


Figure 20. Smoothed Heaviside step function with threshold of 0.5 used to establish the distrusting probability of the human in the algorithm.

The smooth Heaviside-like function described above is a heuristic function we came up with to model different aspects of the human response. When the accuracy of the algorithm is very low, the human will most likely not trust the algorithm. Otherwise, when the accuracy reaches a certain threshold, the shifter will start to trust, and the fraction of cases in which the shifter delegates the decision to the agent will increase as the observed accuracy of the algorithm increases.

Since we are considering average behaviour (over many histograms and multiple shifters), we use smooth functions rather than linear ones. The selected function returns 0 when x is -1 when x is 1, and smoothly transitions between them with an inflection point at the given threshold.

Starting from the Heaviside concept, we created a second way to update the trustiness level that we called “Heaviside cooldown”. We wanted to replicate the behaviour of a person who starts to believe or change his opinion on a determinate event only when can recognize a trend or pattern on it. For example, if we see that the agent is continuously improving his accuracy, we will start to believe more in it. On the other hand, if we see that the agent is improving the accuracy score but between each increment has a little regression, we will have a more cautious approach and we will wait until the situation stabilizes.

To simulate that we are going to update the trust probability after n times that a trend is observed. For example, if $n = 3$ we will call the Heaviside function to update the value when we will observe a decreasing or increasing trend with at least 3 elements (The accuracy must have been increasing 3 times in the last 3 iterations). The n value chosen for the test is 2. Finally, the threshold levels chosen for the training with the Heaviside function have been 0.5, 0.75, and 0.85. In chapter 5 we will discuss and analyse the obtained results.

5.4.3 Workflow of the experiment.

The workflow for each experiment iteration that involves the exploration of the super-human performance, and the machine-assisted human decision will be the following:

1. The algorithm predicts all the histograms in the batch.
2. A fraction of the ground-truth perfect labels is flipped according to the fixed level of human noise, chosen for the experiment.
3. The previous labels are further modified in a second step, in which a randomly selected fraction of the labels is changed to be the same as the algorithm’s prediction. That fraction is obtained following the function in Figure 20, which takes as input the average reward measured in the previous training iteration.
4. The agent is trained using the reward corresponding to that last set of labels.
5. The mean reward obtained by the agent is computed, over the histograms in the batch, to be used in the next training iteration.
6. Before starting the new iteration, the performance of the agent and the human is evaluated in an independent (test) dataset, in terms of the reward achieved, concerning the perfect ground-truth labels.

At the end of the training, the vectors containing the mean rewards of the agent, the human and the human-assisted by the agent are plotted to evaluate the training session.

Chapter 6

Results analysis

In this chapter, we are going to discuss the results of the different experimentations where the agent has been trained with noisy labels and evaluate the performance of the human-assisted decision. The data shown in all the plots in this chapter refers to the test phase.

6.1 Experiments with different levels of noise

In the scenario of the noisy dataset, we have achieved the following results shown in Table 5 below. The relation between the accuracy, the mean rewards and the human error can be explained through the following equations where a is the accuracy, r is the reward and e stands for the error.

$$a = \frac{r+1}{2}\,; \qquad e = 1 - \frac{r+1}{2};$$

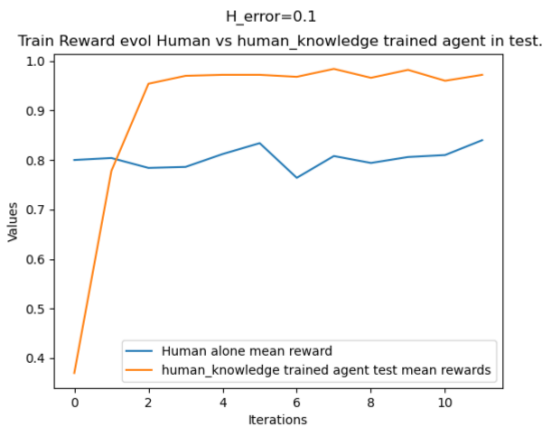
Error	**Human**		**Agent**	
	Mean reward	**Accuracy**	**Mean reward**	**Accuracy**
0%	1.00	100%	0.99	99%
10%	0.80	90%	0.95	97.5%
20%	0.60	80%	0.88	94%
30%	0.40	70%	0.76	88%

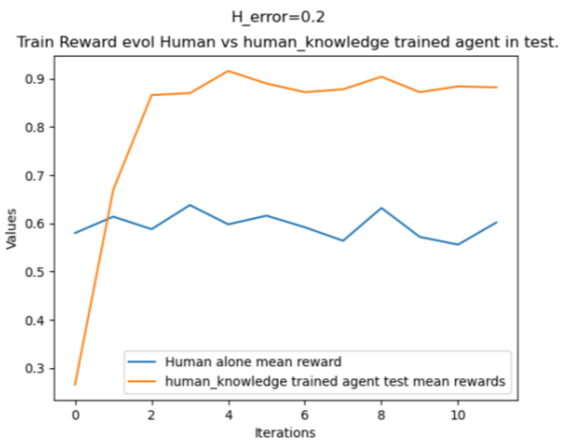
Table 5. Comparative between the Agent results and human results considering noisy labels.

Empirically, it is found that the algorithm learns beyond the level of introduced noise, indicating that it acquires superhuman performance. This aligns with recent literature showing that supervised learning can eliminate noise from imperfect labels if the noise in labelling is independent of specific local properties of the dataset, as is the case in our study.

For example, in the study called “A Spectral Perspective of DNN Robustness to Label Noise” by Oshrat Bar, Amnon Drory and Raja Giryes published in PLMR in 2022, the authors describe how networks have been shown to be robust to such errors [43]. Similar results are achieved in another paper called "How do neural networks overcome label noise?" by Amnon Drory, Shai Avidan and Raja Girye in 2018 where using an MNIST dataset the authors observe how DNNs are extremely resistant to noise when the corrupted labels are randomly spread in the training set [42].

In the plots of Figure 21 below we can observe the evolution during the training with the rewards obtained by the agent and by the human If this last one had to classify the histograms without the help of the algorithm.





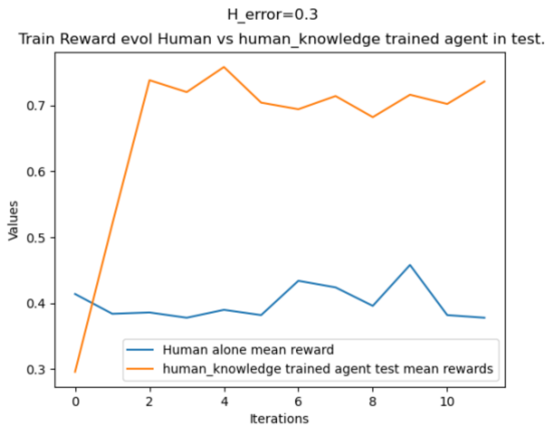


Figure 21. Evolution of mean rewards achieved by humans and agent with different levels of noisy labels. Y axe represents the mean rewards, the x axe the iterations.

6.2 Machine-assisted human decision

To achieve the main aim to demonstrate that a training program, where the human observes the agent's decisions and trusts it to a certain degree, leads to convergence we used heuristic models to represent the human's trust in the algorithm, which we model as a function of the average reward measured in recent training sessions, as well as how the human's decision changes on average when the human observes the algorithm's decision and trusts it.

Depending on the chosen threshold, the human error level and the technique used for updating the trustiness of the human in the algorithm we have achieved different results. However, we can observe two main scenarios.

The former is where the human starts too early to trust in the algorithm since the threshold on the mean reward is set too low with respect to the accuracy achieved at the end of every iteration in the training phase. In this case, the algorithm saturates the learning process too soon and the final reward is lower than the one that the algorithm could achieve if the human hadn't seen the predictions.

In Figure 22, the threshold is set too low to respect the optimal one, since we can observe instability during the training and the agent does have not enough iteration to learn from the human feedback, so the result is limited by human knowledge.

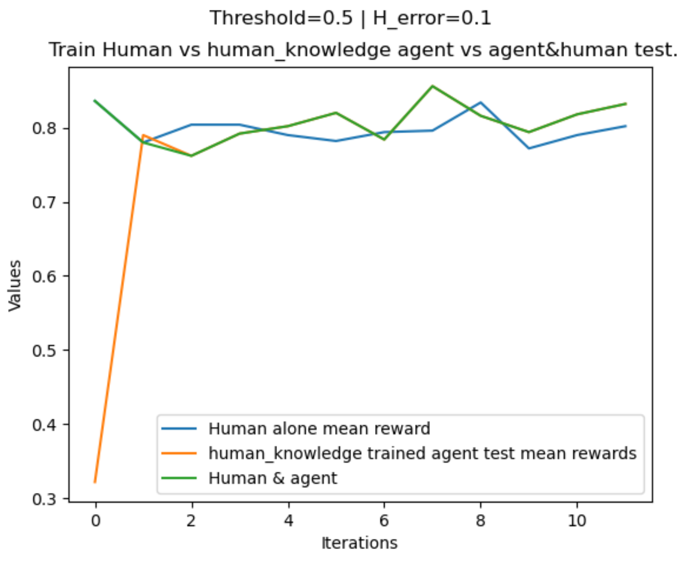


Figure 22. Example of wrong human assistance due to a low threshold. (y axe represents the mean reward values, x the iterations)

In fact, if we maintain the same level of human error and increase the threshold to the next as shown in Figure 23, we reach a better performance due to the fact that the human did not trust the agent immediately and through his feedback allowed it to learn properly from his knowledge.

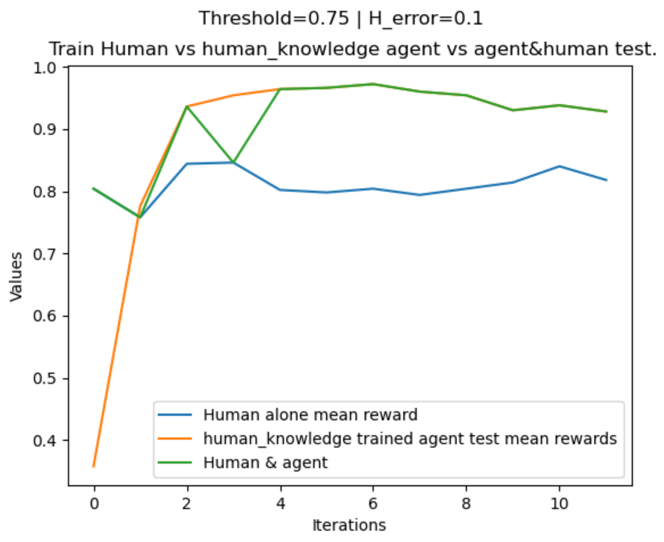


Figure 23. Example of good human assistance due to a better threshold. (y axe represents the mean reward values, x the iterations)

The latter presents the opposite situation, the human starts to trust the algorithm too late or never trusts the algorithm since the threshold is set to high respect for the mean rewards achieved and so, the agent is not able to influence the human and improve the results

As we can see in Figure 24, the threshold is not appropriate for the human error that we are observing, so the performance of the human-assisted agent (green line) is equal to the ones obtained by the human without assistance (blue line) and the agent improvement is not used (orange line).

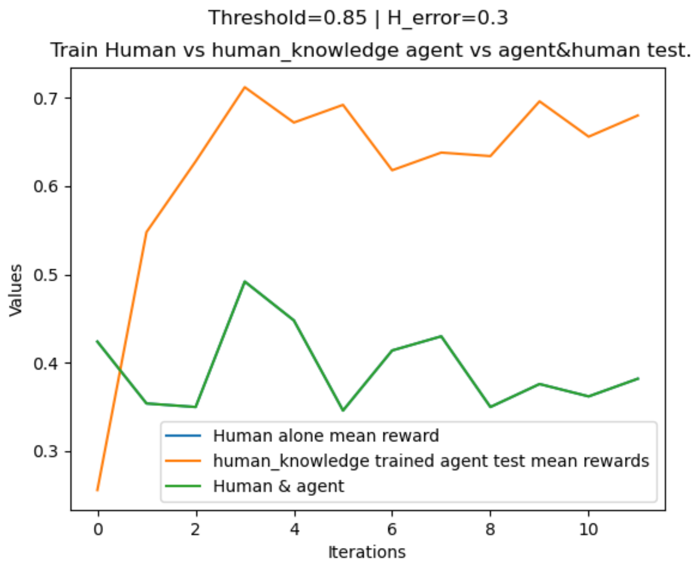


Figure 24. Example of wrong human assistance due to a high threshold. (y axe represents the mean reward values, x the iterations)

In Figure 25, we can have an overview of the scenario where the human has an error of 20% and immediately we can differentiate the case where the threshold is set properly showing a great improvement in the mean reward obtained.

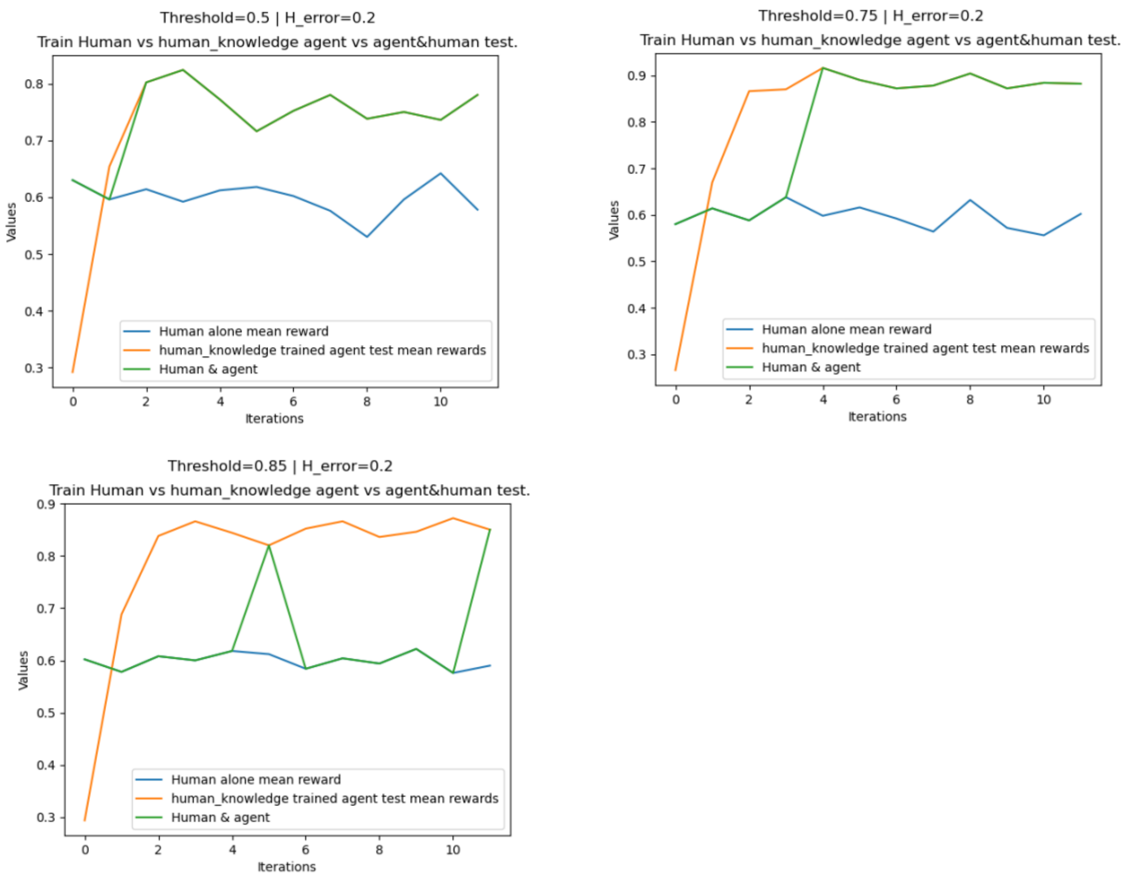


Figure 25. Human assistance training overview with human error of 20%. (y axe represents the mean reward values, x the iterations)

In Table 6 we can observe the optimal threshold (found empirically from these experiments) between the one proposed depending on the human error quantified for the shifter, and observing the results we can conclude that the optimal threshold is approximately inversely proportional to the human error.

Human error	Optimal threshold
0%	1.0
10%	0.85
20%	0.5
30%	0.5

Table 6. Optimal thresholds depending on human error.

In general, although we observe a certain dependence of the performance on the specific threshold chosen, the algorithm always achieves superhuman performance. The shifter also achieves superhuman performances unless the threshold is too high. In such a case, one could consider a reasonable modification of the trust function in which the human also starts to trust the algorithm if its performance is observed to saturate. This would then lead to the shifter achieving superhuman performance in all the cases, demonstrating the power of the chosen approach, independently of the threshold parameter.

Chapter 7

Conclusions and future steps

The project of this thesis constitutes a novel approach for anomaly detection and data quality assessment at the LHCb experiment at CERN, based on machine learning techniques. The novelty lies in the usage of reinforcement learning with human feedback for this task which, to the best of our knowledge, has never been done in the field of particle physics before.

The approach is motivated by the need to identify anomalies in a dynamical regime, during the period of the commissioning of new detectors, where not only the type of anomalies but also the definition of what is considered to be nominal/acceptable is changing with time. The eventual goal would be to assist the human shifters at the LHCb experiment in their decision regarding whether fractions of the data collected in the particle collisions are usable for research or not.

The thesis focuses on the first stages of such an ambitious project, regarding the modelling of a simplified, simulated framework in which the algorithm can be developed and tested, the development and optimisation of the algorithm, and several studies to assess the improvements brought by the algorithm to the data classification process.

The main conclusions from the studies are reported below. Additionally, the future steps, regarding the training of the algorithm with real data and its potential integration in the data-taking framework for LHCb are discussed,

7.1 Main conclusions

After analysing the results obtained from the experiments developed, we can affirm that:

1. The agent can accurately classify histograms when using input data that is 100% correct.

2. The agent has achieved successful classification despite the noise in the dataset, learning beyond the noise level and achieving the superhuman performance that we were looking for. This capacity of machine learning algorithms for de-noisifying a dataset is consistent with what has been reported in the recent literature on other systems, as discussed in the previous section.

3. We have designed a heuristic mathematical model of the human-machine interaction, including an expected change in the average accuracy achieved by an imperfect human shifter when they are informed of the predicted outcome from the algorithm for a given dataset. The model changes the decision of the shifter to be identical to that of the algorithm in a fraction of the cases which increases with the measured algorithm's accuracy over the previous data batch.

4. We have demonstrated, in the simulated setup, that this setup leads to a situation in which both the human and the algorithm learn from each other in a converging process, achieving super-human performance (concerning the average accuracy of the human decisions if the algorithm was not present).

One of the main limitations observed in the approach adopted is that the training phase is not as stable as desired. It may be necessary to explore alternative configurations to improve its stability. It should also be noted that the conclusions in this study are limited to the assumptions used in the modelling. Among others, the human noise introduced is uniform (independent of the features of the dataset) and we have chosen to parameterise the change in the decision of the human when assisted by the algorithm in a very specific way. While more variations can be considered in future studies, the conclusions of these first studies show great potential for the approach.

7.2 Future steps

As a starting point, this study can be considered for future research. The obtained results have provided a useful foundation for further investigation. Firstly, the algorithm needs to be tested with the data collected from Run 2 to establish its performance in real conditions.

In the next stage, the algorithm needs to be adapted to run in DM mode or in DQ mode. As anticipated in previous sections, running it in DM mode will require checking the training time and speeding it up, if necessary, to obey the conditions of the real-time data taking. To run the algorithm in DQ mode, a training scheme suitable to small datasets (histograms only per full run instead of every 10 min) needs to be developed and tested.

Finally, once these aspects have been implemented, a suitable interface must be created for human supervision and feedback during agent training. Finally, testing in a real-world context can proceed.

References

1. CERN. (2023, June 22). *What does "CERN" stand for?* Retrieved from CERN HOME: https://home.cern/about.

2. CERN. (2023, June 10). *About CERN*. Retrieved from Home CERN: https://home.cern/node/5011

3. CERN. (2023, June 11). *Bith web*. Retrieved from Home CERN: https://home.cern/science/computing/birth-web

4. Spectrum. (2023, June 12). *Analyzing the LHC Magnet Quenches*. Retrieved from Spectrum: https://spectrum.ieee.org/analyzing-the-lhc-magnet-quenches

5. CERN. (2023, June 14). *CERN Annual Personnel Statistics*. Retrieved from Home CERN: https://home.cern

6. CERN. (2023, June 14). *Our mission*. Retrieved from Home CERN: https://home.cern/about/who-we-are/our-mission.

7. CERN. (2023, June 14). *What is the universe made of? How did it start? Physicists at CERN are seeking answers, using some of the world's most powerful particle accelerators*. Retrieved from Home CERN: https://home.cern/node/5011

8. CERN. (2023, June 16). *Facts and figures about the LHC*. Retrieved from Home CERN: https://home.cern/resources/faqs/facts-and-figures-about-lhc

9. CERN. (2023, June 16). *Linear accelerator 4*. Retrieved from Home CERN: https://home.cern/science/accelerators/linear-accelerator-4

10. CERN. (2023, June 18). *Proton synchrotron booster*. Retrieved from Home CERN: https://home.cern/science/accelerators/proton-synchrotron-booster

11. CERN. (2023, June 18). *Proton synchrotron*. Retrieved from Home CERN: https://home.cern/science/accelerators/proton-synchrotron

12. CERN. (2023, June 20). *Super proton synchrotron*. Retrieved from Home CERN: https://home.cern/science/accelerators/super-proton-synchrotron

13. *Nobel Prize in Physics 1979*. (2023, June 20). Retrieved from Nobel e-Museum: https://web.archive.org/web/20040803075503/http://www.nobel.se/physics/laureates/1979/

14. CERN. (2023, June 20). *large hadron collider*. Retrieved from Home CERN: https://home.cern/science/accelerators/large-hadron-collider.

15. CERN. (2023, June 22). *LHCb experiment*. Retrieved from Home CERN: https://home.cern/science/experiments/lhcb

16. Perkins, C. (2023, June 22). *CERN: Everything you need to know*. Retrieved from Science focus: https://www.sciencefocus.com/science/cern/

17. May, A. (2023, June 24). *CERN: Organization, experiments and facts*. Retrieved from livescience: https://www.livescience.com/cern

18. CERN. (2023, June 24). *LHCb*. Retrieved from Home CERN: https://home.cern/science/experiments/lhcb

19. Federico, A. (2023, July 28). *Central shift roles and duties: Shift Leader and Data Manager.* CERN, the commissioning/running team of LHCb. Geneve: CERN.

20. Hull, A. (2023, 08 01). *What is Reinforcement Learning from Human Feedback?* Retrieved from Invisible: https://www.invisible.co/post/what-is-rlhf

21. Jacquet, F. F. (2014). Implementation and experience with luminosity levelling with offset beam. *CERN Yellow Report*, 183–187.

22. Günthe, P. A. (2023, August 10). *LHC_Sketch*. Retrieved from hassec: https://github.com/hassec/LHC_Sketch

23. CERN. (2023, August 11). *The Worldwide LHC Computing Grid (WLCG)*. Retrieved from Home.cern: https://home.cern/science/computing/grid

24. *Facts and figures about lhc*. (2023, August 12). Retrieved from Home cern: https://home.cern/resources/faqs/facts-and-figures-about-lhc

25. CERN. (2023, August 14). *LHCb's momentous metamorphosis*. Retrieved from CERNCOURIER: https://cerncourier.com/a/lhcbs-momentous-metamorphosis/

26. Adinolfi, M. (2023, August 14). LHCb data quality monitoring. *J. Phys.: Conf. Ser. 898 092027*, 8. Retrieved from Journal of Physics: Conference Series: https://iopscience.iop.org/article/10.1088/1742-6596/898/9/092027/pdf

27. Goecks, V. G. (2023, August 14). *Human-in-the-Loop Methods for Data-Driven and Reinforcement Learning Systems.* Retrieved from arxiv.org: https://arxiv.org/abs/2008.13221

28. *Human-in-the-loop machine learning: a state of the art*. (2023, August 16). Retrieved from SpringerLink: https://link.springer.com/article/10.1007/s10462-022-10246-w

29. Biao Luo, Z. W.-C. (2023, August 15). *Human-in-the-Loop Reinforcement Learning in Continuous-Action Space.* Retrieved from pubmed.ncbi.nlm.nih.gov: https://pubmed.ncbi.nlm.nih.gov/37418406/

30. Stone, W. B. (2023, August 16). *TAMER: Training an Agent Manually via Evaluative Reinforcement.* Retrieved from cs.utexas.edu: https://www.cs.utexas.edu/~bradknox/papers/icdl08-knox.pdf

31. Jingda Wu, Z. H. (2022, August 16). *Human-in-the-Loop Deep Reinforcement Learning with Application to Autonomous Driving.* Retrieved from Arxiv: https://arxiv.org/abs/2104.07246

32. Riku Arakawa, S. K.-i. (2022, August 17). *DQN-TAMER: Human-in-the-Loop Reinforcement Learning with Intractable Feedback.* Retrieved from Arxiv: https://arxiv.org/abs/1810.11748

33. Guansong Pang, A. v. (2022, August 17). *oward Deep Supervised Anomaly Detection: Reinforcement Learning from Partially Labeled Anomaly Data.* Retrieved from Toward Deep Supervised Anomaly Detection: Reinforcement Learning from Partially Labeled Anomaly Data: https://arxiv.org/abs/2009.06847

34. CMS collaboration. (2023, August 17). *Machine Learning applications for Data Quality Monitoring and Data Certification within CMS.* Retrieved from iopscience: https://iopscience.iop.org/article/10.1088/1742-6596/2438/1/012098/pdf

35. Shah, D. (2023, August 18). *RLHF (Reinforcement Learning From Human Feedback): Overview + Tutorial*. Retrieved from v7labs: https://www.v7labs.com/blog/rlhf-reinforcement-learning-from-human-feedback

36. Ray. (2023, 08 24). *RLlib: Industry-Grade Reinforcement Learning*. Retrieved from docs.ray.io: https://docs.ray.io/en/latest/rllib/index.html

37. OpenAI. (2023, August 25). *Proximal Policy Optimization*. Retrieved from openai: https://openai.com/research/openai-baselines-ppo

38. Moens, V. (2023, August 26). *Reinforcement learning PPO* . Retrieved from pytorch.org: https://pytorch.org/tutorials/intermediate/reinforcement_ppo.html

39. spinningup openai. (2023, August 28). *Proximal Policy Optimization*. Retrieved from spinningup.openai.com: https://spinningup.openai.com/en/latest/algorithms/ppo.html

40. *Tune Trial Schedulers*. (2023, August 29). Retrieved from docs ray: https://docs.ray.io/en/latest/tune/api/schedulers.html

41. *Ray Tune: Hyperparameter Tuning*. (2023, August 30). Retrieved from Ray API docs: https://docs.ray.io/en/latest/tune/index.html

42. Amnon Drory, S. A. (2018). *HOW DO NEURAL NETWORKS OVERCOME LABEL NOISE?* Tel-Aviv University: School Of Electrical Engineering.

43. Giryes, O. B. (2022). A Spectral Perspective of DNN Robustness to Label Noise. *PMLR*, 22.

Acknowledgements

How can I start? Honestly during these months of meetings, coding, meetings, writing, meetings, and I already said meetings? I never imaged reaching this point…

Thanks to my university supervisors Víctor and Juan, for making it possible for a walking disaster like me who is studying and working abroad to do everything on time and by email. You have been always available and patient with me despite the rushing and panic emails just before the deadline.

Muchísimas gracias, ya no sé ni como decirlo a Julián, por ser la definición exacta de un amor de persona. Incansable, siempre disponible en todo momento, no importa que se encuentre en un crucero, tren, avión, kayak , con un entusiasmo contagioso por cada pequeño avance que hacíamos en el proyecto.

Muchas horas pasadas delante esa pizarra blanca en tu despacho, explicándome la física como a un niño y contestándome a cada pregunta como si fuera la primera vez en vez de la decimoquinta, voy a echar de menos tu “hey que tal va la cosa” o maybe no, puede que lo siga escuchando quien sabe. Muchas gracias

Thanks to Suzanne, for being always available and kind to me, bringing me around the LHCb facilities like a child. We will meet in Amsterdam again for sure!

Thanks to jde, for being super supportive all the time either at work or outside it. To understand the pain, I mean the joy of the theses letting me participate in this awesome project despite all the possible complications with the Drupal work. Thanks for being always there and being that kind of supervisor that is always taking care of the people around you.

The supervisors are finished, yes, they are a lot because I am difficult to handle… Now it is time to say thanks to my partner in crime, I am speaking about you, Olivia. Buff, si tengo que contar todo lo que hemos vivido creo que tengo que escribir otra tesis. Mejor empezados desde el principio, gracias por ese cumpleaños donde me presentaste a Julián y desde el cual nació todo, gracias por pasar esos findes interminables conmigo, ya sin saber si los locos éramos nosotros o Ray. Gracias por ser un sol y contagiar esas risas tan tuyas, gracias por los stickers, los ánimos, las discusiones sobre quien de los dos tenia razón (yo siempre) sobre una u otra teoría (con Julián agarrando un paquete de palomitas para disfrutar del show). Gracias por enseñarme que no se pueden comparar plots con diferentes escalas… Gracias de todo corazon, por ser tu y estar a mi lado en esta aventura, te debo mucho. Ojalá empezar otra…

Thanks to my flatmate Cedric, for doing the best thing that a friend can do when you are writing your thesis. Giving you Mars ice creams!

Time to say thanks to the amazing Gang out-group, starting from the older ehm, wiser… Thanks to my wise homonym, for being the reference for… well everything, from the amazing DJ sessions at LHCb to the 1 thousand suggestions of places for drinks and dinner, and of course physics fun facts… Your help with this thesis at the swimming pool in Greece has been appreciated a LOT.

Thanks to the Romagnolo guy, Giovanni, for continuing to mismatch my region on purpose and for being such an amazing guide partner, neighbour and most importantly friend. (I am still waiting that kg of tortellini maledetto). Speaking about guides, the team is not complete without you Giulia, Thanks for sharing with me all the hilarious moments trying to decrypt the visit instruction of CMS, for being such a cinnamon roll (I loved this word that you taught me) and a master in elegance. Thanks for letting me understand the pain of a car school driver. Despite that, I am really happy to see you around for another 2 years.

Thanks to baby Milto, Mr. I wear sunglasses even in the darkest room, for being such an amazing shoulder to count on, for being that mate that you need when you want a coffee, for organizing amazing trips and treating his guests better than his family, for have a heart bigger than the engine of his alfa.

Thanks to my Chotos teammate Leonardo for sharing the joy and the sadness of the matches and coming to visit me (ofc not for the shifts) at Cern. Thanks for being the reassuring Venice captain that we all deserve and pushing me in the last steps of the writing.

Thanks to my Sansepolcro neighbour Noé and very far neighbour Alessandra, for being that inseparable couple that you could rely on every time. Thanks for all the nights at your place, 5/5 on TripAdvisor. Thanks for all the laughs and moments that we share together, starting from being my climbing partner to all the alcoholic-night ideas that you came up with. Thanks also for the anime suggestions!

Alessandra…. Thanks to your awesome Cocc… ehm your amazing cuisine. And all the smiles that you have caused. Thanks for trusting me to be your ski instructor.

Speaking about snow, thanks to Francesca for being such a lovely person and sharing the ski moments and the amazing views that you can see only when you are there.

Thanks to Matteo for letting us show the true magic of panigiri (Milto scansati proprio). Thanks to Max for letting me know in advance all the cursed stories about my flat. Thanks to Betto for being the spirit of the LHCb parties. Thanks to Carlo for the memes and the physics FACTS.

I am glad to have met you all, and I learned something from all of you (Giovanni no tu no), so I will bring with me always a piece of these moments.

Last but not least, thanks to my family and Mara, for believing in me, for pushing me in the difficult moments and for being that reference that is always present. Definitely, without your support I would not be there, in this amazing environment, knowing a lot of people every day, travelling a lot, and of course, learning something.