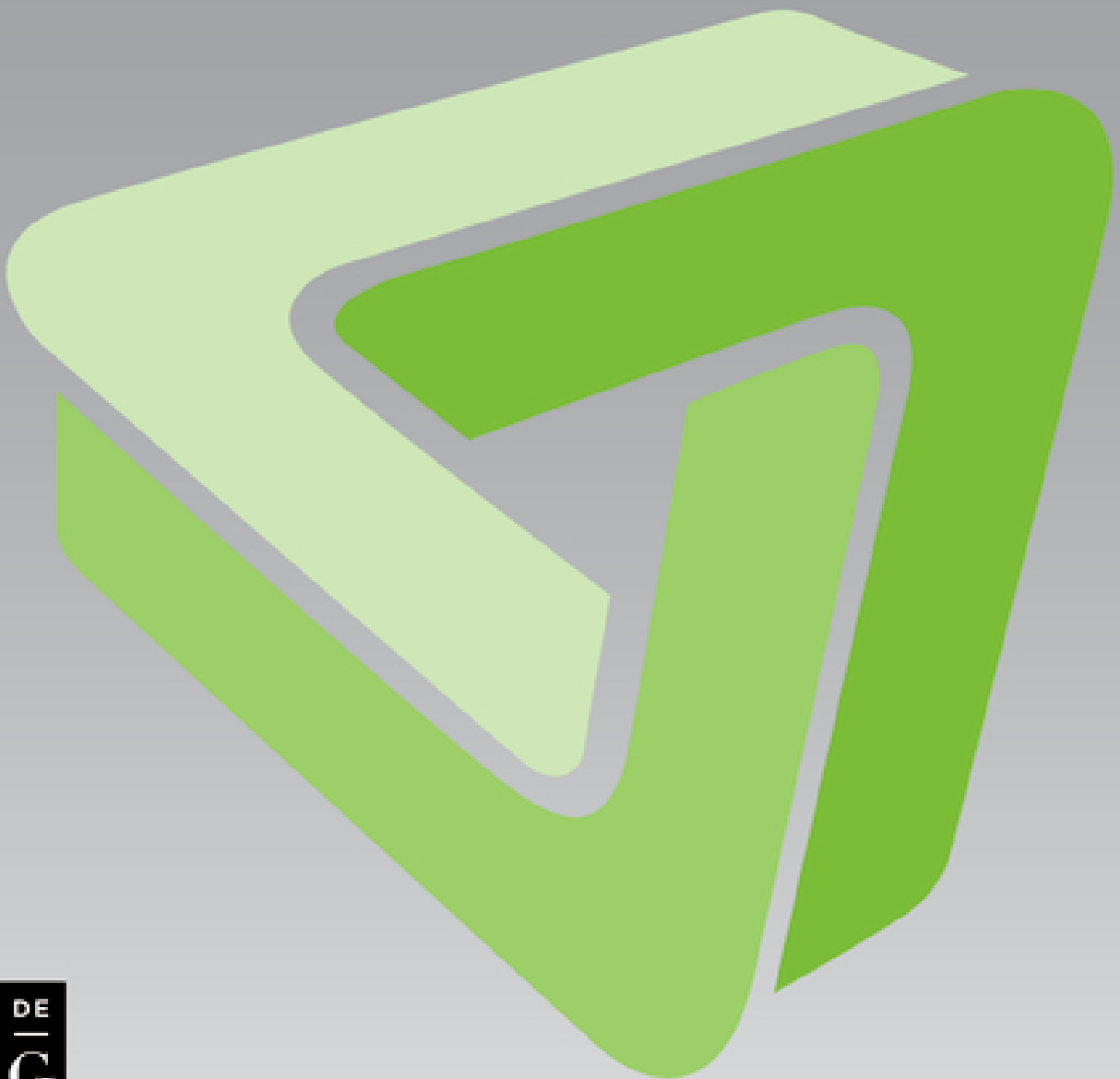
DE GRUYTER STEM

MACHINE LEARNING UNDER RESOURCE CONSTRAINTS

DISCOVERY IN PHYSICS

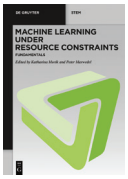
Edited by Katharina Morik and Wolfgang Rhode



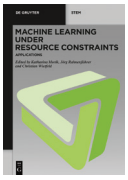
DE G

Katharina Morik, Wolfgang Rhode (Eds.)
Machine Learning under Resource Constraints · Discovery in Physics

Also of interest



Volume 1
Machine Learning under Resource Constraints. Fundamentals
Morik, Marwedel (Eds.), 2023
ISBN 978-3-11-078593-7, e-ISBN 978-3-11-078594-4



Volume 3
Machine Learning under Resource Constraints. Applications
Morik, Rahnenführer, Wietfeld (Eds.), 2023
ISBN 978-3-11-078597-5, e-ISBN 978-3-11-078598-2

Machine Learning under Resource Constraints

Final Report of CRC 876

Editor in Chief
Katharina Morik

Volume 2/3

DE GRUYTER

Machine Learning under Resource Constraints

Discovery in Physics

Edited by
Katharina Morik and Wolfgang Rhode

DE GRUYTER

Editors
Prof. Dr. Katharina Morik
TU Dortmund University
Department of Computer Sciences
Chair for Artificial Intelligence
Computer Science 8
Otto-Hahn-Str. 12
44221 Dortmund
Germany

Prof. Dr. Dr. Wolfgang Rhode
TU Dortmund University
Department of Physics
Chair for Experimental Physics 5b
Otto-Hahn-Str. 4b
44221 Dortmund
Germany

ISBN 978-3-11-078595-1
e-ISBN (PDF) 978-3-11-078596-8
e-ISBN (EPUB) 978-3-11-078613-2
DOI https://doi.org/10.1515/9783110785968

Library of Congress Control Number: 2022949268

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available on the Internet at http://dnb.dnb.de.

Cover image: Collaborative Research Center 876
Printing and binding: CPI books GmbH, Leck

www.degruyter.com

Dedication: We dedicate this book to the memory of our friend and colleague Bernhard Spaan, who walked this path between physics and computer science with us for many years until he sadly left us during the work on this book.

Contents

1 Introduction

Wolfgang Rhode
Katharina Morik

Abstract: The question of how to arrive at scientifically secured knowledge has accompanied research from the very beginning. Depending on the scientific context and the historical epoch, answers have been given, essentially depending on the demanded degree of truth and the scientific methodology. Recently, a new scientific methodology has emerged, which is best characterized as "probabilistic rationalism". This methodology is the subject of this book: Over the past decades, the fields of computer science and physics have collaborated to create AI- machine learning-based methods for analyzing massive amounts of data collected in modern experiments in view of their probabilistic properties. Artificial intelligence or machine learning is the methodology of the day. The considerations presented here are based on understanding the probabilistic character of scientific statements. In Dortmund, we have investigated not only isolated aspects of these analyses but the entire evolutionary process of knowledge expansion. In this first chapter, we combine interdisciplinary aspects of epistemology from the perspectives of physics, artificial intelligence, and philosophy to form an up-to-date and consistent model of knowledge acquisition. With this model, some known problems of existing epistemological approaches, e.g., the problem of inconsistent experimental results, can be overcome. The refutability of theories is based on testing, as in Popper's paradigm. The conclusion, however, is no longer a binary decision but a probability.

1.1 Basics, Questions, and Motivation

This book describes and discusses new data science methods developed as part of the Collaborative Research Center (CRC) "Data Analysis under Resource Constraints" to solve problems in the field of astro/particle physics. These methods contribute to the solution of the fundamental epistemological question of how conclusions to explanatory theories can be drawn from observations of nature or from measurements in experiments. Although methodologically a multitude of very different sub-problems had to be solved here, the methodological meshing of the algorithms, i.e., the entire analysis cycle, is at the center of our interest. In this first chapter, we sketch the sequence of the solution we have worked out and embed it in its epistemological context before we later turn to specific problems and exemplary approaches.

Classically, the problem to be solved first appears in the form of Plato's Allegory of the Cave, in which the researchers are fixed on a bench behind a wall so that they

https://doi.org/10.1515/9783110785968-001

only can see patterns on the cave wall opposite them. They cannot look back to see that these dancing shapes in front of them are the work of puppeteers moving figures in front of a fire. In this picture, the ray optics needed to understand represents the theory. And quite obviously, neither by logical nor by mathematical operations can we infer the theory directly from the shadows and prove its truth. However, as we will see, the problem can be solved by overcoming the linear view of the inference chain in favor of a cyclic-networked interpretation and the demand for truth by calculating probability intervals. The underlying fundamental epistemological problem concerns the question of whether there is a way to *of concluding backward* from the effect (in the picture: "from the shadow") to its cause.[1]

Newton demanded as a solution to this question that similar impacts must be caused by the same causes.[2] In this, he was fortunate that this postulate applied to the gravitational effects he considered in the sublunar world of the Earth's surface and the translunar world of the solar system and that he was thus able to unify the phenomena of both physical worlds. The subsequently rapid and successful development of classical mechanics suggested that the world of nature could be mapped onto a world of clearly solvable mathematical problems and theories. Also, the new path from the phenomena of electro- and magnetostatics to Maxwell's equations, though leading to a more complex descriptive theory structure, did not change anything about the epistemological classification of this form of explaining the world. In thermodynamics, statistical statements and correlations forced their way into the world explanation based on the ignorance interpretation of probability[3]. Later, in quantum mechanics, the more far-reaching demand for an identity of probabilistic structure and physical law turned against the realm of what could be said with certainty. At first, however, these developments seemed to be only disturbing artifacts within a deterministic explanation of the world.

The practical path from experiment to theory always seemed to include the necessity to abstract a few (preferably) factual and clear statements from the experimental measurement. These propositions, in which an observed phenomenon (e.g., the spiral track of a particle in a cloud chamber at a known magnetic field) is interpreted as something ("the particle is an electron with an energy in the interval between E_1 and E_2"), are the (basic) sentences, which are set up and checked by Popper according to the rules of logic, and that decide in critical rationalism via falsification or confirmation

1 "Cause" here and in the following in the sense of reason.

2 Rules of Reasoning: (I) No more causes of natural things should be admitted than are both, true and sufficient, to explain their phenomena. (II) Therefore, the causes assigned to natural effects of the same kind must be, so far as possible, the same [288].

3 From a deterministic starting point, one concludes: "In principle, all processes in physics are exactly calculable. Because we cannot do this at the moment because of our personal mathematical incapacity or because it would be disproportionate to the desired result in terms of time, we are content with a merely statistical description of the phenomenon"

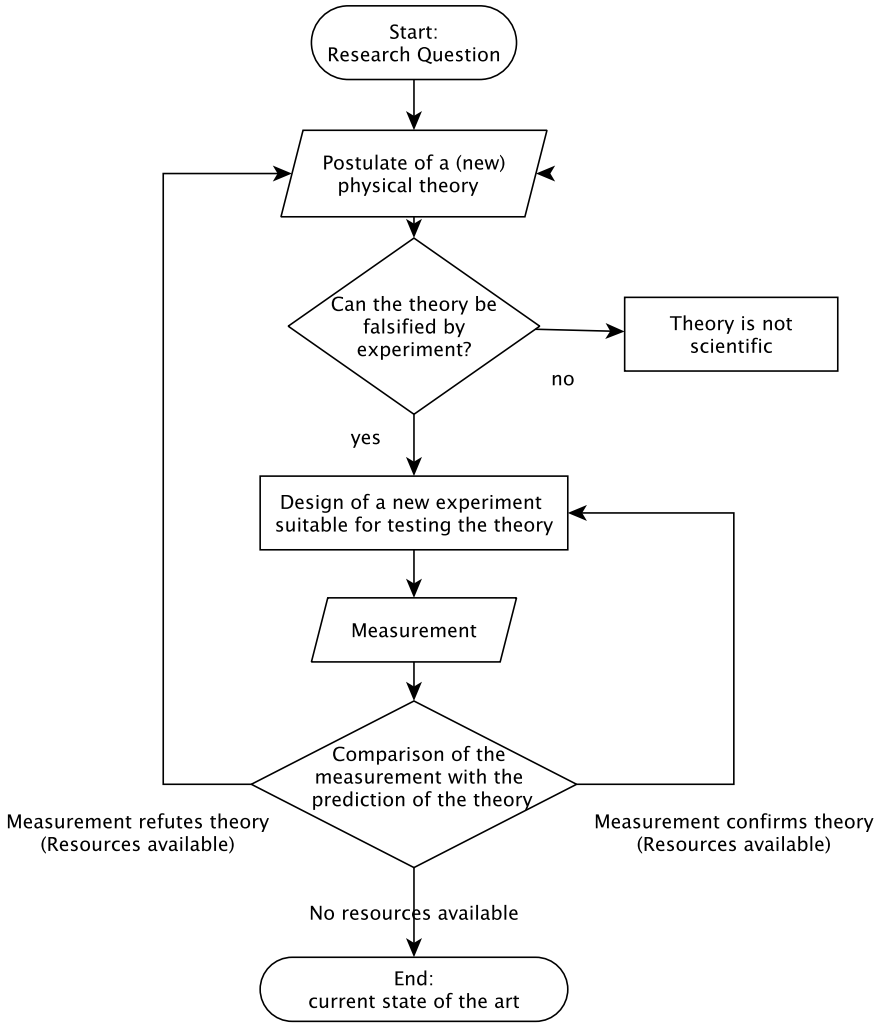


Fig. 1.1: Connection between a physical theory and measurements in critical rationalism. The process will continue as long as the necessary scientific resources are available. Problems arise because no clear, logical answers can be given at the decision points in many real-world cases.

about the fate of preferably structurally simple theories with very few free parameters. As all measurements are a record of a finite number of discrete information (data points), there is no logical conclusion to the generalizing mathematical function of the describing theory. In the natural sciences, falsificationism, the basic concept of critical rationalism, has therefore largely prevailed. In this context, from rationally conceived theories, functional connections between experimentally measurable quantities

are predicted, which are tested in experiments, attempting to falsify the postulated connection and with it the predicting theory (see Figure 1.1).

If falsification is successful, a new, better theory must be developed. If the experiment confirms the theory, the experimental tests should be continued under more stringent conditions. This epistemological model presented in 1933 by Karl Popper in his book *The Logic of Scientific Discovery* [315] is based on clearly defined logical conditions. However, as stated in various discussions between Karl Popper and his critics, critical rationalism is neither intended nor suitable for describing physics research's real temporal course. Popper was criticised because of this logical structure when trying to clarify under which conditions a theory can or must be regarded as refuted in the world of modern, very complex experiments and theories. How are experimental results to be classified if experiments yield inconsistent results or different experiments contradict each other? Which practical requirements for experimental statements must be met to refute a theory? How exotic may the requirements for experiments be so that a theory can still be considered disprovable by experiments and thus be called scientific?

1.2 Machine Learning as Model of the Scientific Process

Artificial Intelligence (AI) began as a model of cognitive processes. This applies to several levels: the neurons at the subsymbolic level, models of learning and decision-making at the individual level, and the process of scientific discovery at the epistemological level, which is our concern here.

1.2.1 Early Approaches to Scientific Discovery

As a model of the scientific process, machine learning started early on to discover regularities in data, generate hypotheses, and test them. Statistics is employed for hypothesis testing, but machine learning has a much broader view of the overall process of modeling, including hypothesis generation, self-supervision of the learning algorithm, creating representations, and the generation of new data by conducting experiments automatically [320]. Approaches to enhance given or learned models, called "explanation-based learning" were also discovered as part of the field [140].

Learning of equations from data by the ABACUS system aimed at modeling scientific discovery [158]. The ABACUS system learns equations from observations and derives their applicability conditions. Variables in the data are *qualitative proportional* if they rise (or decrease) together, or one rises if the other decreases, while other variables remain constant. These hypotheses are tested allowing a user-specified proportion of exceptions or a split of the data into relevant subgroups that can be characterized. For equation formation, sum, product, square, and their inverse operators generate hypotheses. The space of all possible hypotheses is restricted by redundancy and

tautology constraints. Learning is formalized as a search in the hypothesis space. The result is a set of rules, each with a logical description of the applicability. An example is detecting the ideal gas law in a data set of 6 variables $PV/nT = 8.314\ \mathrm{J\ K^{-1}\ mol^{-1}}$ with P/hPa being pressure , V/m^3 the volume, T/K the temperature, and n the number of moles. Another example is the rule set for the experiment with balls of different sizes falling through different fluids:

If substance = vacuum, then $v = 9.8175\ \mathrm{ms^{-2}} \times t$
If substance = glycerol, then $v \times r = 0.9556\ \mathrm{ms^{-2}} \times m$
If substance = CastorOil, then $v \times r = 0.7336\ \mathrm{ms^{-2}} \times m$

with v/ms^{-1} being velocity, r/m the radius of the falling ball, m/kg the mass of the body. This example shows the learned split into the different rules for different substances.

Volume III of the classic work *Machine Learning – An Artificial Intelligence Approach III* (1990) gives an overview of statistical learning, the integration of domain theories into learning from examples, learning by experimentation, enhancing imperfect domain theories, causal models, subsymbolic learning, and computational learning theory. The variety of topics covers much more than just the statistical analysis. The view of scientific discovery is broad and multi-faceted.

This short summary of early machine learning approaches sounds as if this would be everything we need for a model of scientific investigation. However, this is not the case. All these topics are still research subjects today. In order to support human scientific discoveries, machine learning moved beyond the early works in three ways, as we explain below.

1.2.2 Knowledge Representation – What is a Theory?

First, one law or decision function is not sufficient to represent a theory. A theory consists of many integrated rules. If the machine learning process aims at modeling the scientific process of how to make sense out of observations and integrate new concepts into what has been learned already, it needs to handle more than just one isolated concept. It needs a sound way of expressing interrelated concepts and their interpretation. A representation without a formal inference is just ink on paper requiring the human interpretation.

Therefore, artificial intelligence has developed formalisms for knowledge representation that expresses and interprets concepts and their relationships. One of them is description logic, which represents a theory in terms of concept definitions together with the two principle relationships between them: concepts used to define other concepts and concepts subsuming other concepts in a generalization hierarchy [73]. Description logic carefully restricts mathematical logic as little as possible to keep reasoning tractable. The theory is consistent, i.e., contradictions are efficiently recognized and prohibited. The theory not only performs inference in the conceptual space but also

decides whether an observation or measurement is covered by a concept. To this deductive reasoning, machine learning has added induction, which learns concepts from observations and places them into concept hierarchies. Following Popper's paradigm, machine learning exploited contradictions to learn a refined model. Inspecting contradictions has even been used to introduce new concepts into the representation so that the theory covers more observations coherently [279].

Another family of knowledge representations is Horn logic, which represents a theory by a set of rules that together form a program. The programming language Prolog [126] is itself a kind of Horn logic and has been used to program large knowledge bases with a deductive inference, that concludes a theoretical statement or predicts an observation. The other way around, inferring sets of rules from observations, is known as inductive inference. Of course, it would be pleasant if we could automatically find the smallest possible model that covers all observations. This sharp theory formation would be a strict axiomatization. However, inductive learning is a process that, at some point, may lack predicates or functions that are needed to compress the model, so that the smallest model cannot be found.[4] The formulation of inductive logic programming covers a soft minimality condition. It does not require the model to be the smallest, but only that none of its rules can be deleted without loss of coverage. The conditions on an induced set of rules $\mathcal{H}$, a theory, are

1. the minimal logic model of $\mathcal{H}$ is true in all minimal models of the examples $\mathcal{E}$ (possibly together with those of background knowledge);
2. there is no $h \in \mathcal{H}$ such that all examples could be deduced from $\mathcal{H}$ (possibly together with the background knowledge) without h;
3. all rules that fulfill 1) and 2) can be deduced from $\mathcal{H}$;
4. $\mathcal{H}$ is minimal.

This logic-based definition of Helft [195] for learning sets of rules from examples could also be turned into a description of classification learning [283]. In this paradigm, inducing a set of rules from observations resembles the scientific process since the rules are used to deduce or predict observations. Learning, the inductive step, concludes from the observations to the causes. Moreover, it includes the already acquired knowledge as background knowledge. The learned rule set is used by deductive inference to conclude the outcome from the cause. The key to inductive inference here is *generalization* in the mathematical framework of predicate or even higher-order logic.

4 Kleene stated that a first-order language is axiomatizable if finitely many additional predicates are added [228]. Stahl worked on predicate invention for better axiomatization of finite languages [357]. See also [290].

1.2.3 Towards Probabilities

The second reason why the early approaches are not sufficient is their two-valued logic. We may say that a disagreement with the strict Newton view already occurred within logical inference. In contrast to Newton, who offers just one true model, alternative explanations of phenomena have to be expressed and stored for further theory development [151]. Since we do not want to express arbitrary concepts, the notion of truth and consistency needs to be kept. The pure two-valued logic, however, is not sufficient for non-monotonic reasoning and for revising propositions of logic programs due to new observations. The model of knowledge acquisition here substantially differs from Popper's critical rationalism (to which it is otherwise very much indebted) by replacing purely logical decisions with a cycle of probabilistic evolutionary evaluations. As with Popper, completely wrong models of explanation can easily be sorted out. Which probabilities one has to drop as a criterion for specific theoretical approaches depends on the significance of the physical question under consideration, the customs of the physical discipline concerned, and the available resources. Methodologically, decisions based on only moderate differences in the probabilities of the explanatory approaches are not necessary. Due to the very good quality of the physical description of nature, many long-used models and theories in practice have a very high probability. In this way, the classical approach to knowledge can be understood as a borderline case. And, as always, when the accustomed result is understood as a borderline case of a more general consideration, we become observers and promoters of a new phase of science.

The novel paradigm of modeling evidence in favor of a hypothesis and evidence against a hypothesis allows us to characterize propositions as unknown, known, or predicted, or as its negation or as an excluding state—all to a certain degree. Emde's more probabilistic inference engine allowed for the revision and representation learning in the course of modeling [150]. New concepts could be introduced into the represented theory, but this time without sacrificing alternatives [390]. Instead, possible revisions and novel concepts that would help to solve a contradiction are maintained as alternatives, together with their derivation paths. The overall modeling process has been investigated based on a sound inference engine in restricted first-order logic. In addition, the tools that support all steps within it were integrated such that humans and machines could interactively build a complex knowledge base [281].

This view of epistemology fits well the with physics view, where there is no absolute space, no absolute time, and no absolute truth. Due to thermodynamical and quantum effects necessarily coupled to the electromagnetic interactions in the detection processes, even a fixed combination of theory parameters in top-down calculations cannot lead to any desired degree of precision and unambiguity in the measurement results. We improve our knowledge relative to the existing knowledge, which is itself relative to its context in the same sense.

The insight that uncertainty needs to be made explicit to allow for an ongoing process of modeling points to the probabilistic view of science, as put forward by Dempster

[142], Shafer, and Pearl [347]. The conference *Uncertainty in Artificial Intelligence* has been held every year since 1985. From a philosophical point of view, some machine learning approaches adopted a probabilistic view early on, e.g., [280].

This brings us to probabilistic representations of theories as Bayesian networks and graphical models. For our discussion, the most important characteristics of the Bayesian methods are i) that they combine knowledge in the form of a priori probabilities with learning from data and ii) that several hypotheses can be combined for the output of a prediction that is expressed as a probability of the outcome. Bayesian reasoning and learning both build on Bayes' law:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} \tag{1.1}$$

The hypothesis h that has the highest probability given the data, i.e., the Maximum A Posteriori (MAP) hypothesis is the resulting hypothesis. For a fixed data set, this is

$$\arg\max_{h \in H} P(D|h)P(h) \tag{1.2}$$

If we regard only the probability of the hypothesis conditioned on the data and we assume equal probability for all hypotheses, $P(h)$ is omitted because it is the maximum likelihood. For all hypotheses in the space of possible hypotheses, e.g., all classes that are to be predicted, the probabilities are calculated, and the one with the highest is chosen. The network structure of Bayesian networks and graphical models in general is determined by the conditional independence of two variables (or network nodes) given the third one. This, together with prior knowledge, reflects the scientific knowledge.

Bayesian models may characterize other learning models and, hence, give them a sound basis. Actually, the output of many machine learning algorithms is the MAP hypothesis. It is a very general description of learning. Although the minimization of the least square error—computed in a statistical manner—is usually not performed according to the Bayes law, it delivers the MAP if the prior probabilities are the same for all classes, which is then the maximum likelihood. Understanding neural networks, giving them a sound theoretical basis, also employs Bayesian statistics. One instance is to show that a neural network's inference with dropout approximates a deep Gaussian process in a Bayesian sense [174].

Causal reasoning is once again a hot topic in machine learning [301] and has also been investigated in a Bayesian manner [222]. Current approaches may well be described as probabilistic rationalism.

1.2.4 The Big Data Move

The third reason why the early methods are not sufficient anymore comes from a fundamentally different situation in today's knowledge acquisition in physics, where the data volume and granularity demand novel methods, as stated in the following.

Volume We are no longer dealing with a few individual observations or measurements, but with the collection of data volumes so large and complex that the compilation of a few individual statements classifiable by humans is no longer possible. The analysis of a massive amount of data makes such different qualitative demands on its epistemological treatment that the whole can no longer be described effectively as a sum of its purely logically interpretable single statements.

Granularity What is electronically recorded or measured in such large numbers is simple. It consists of charges, times, and positions—and nothing else. The measurements are only indirectly related to the dependencies (e.g., energy spectra, angular distributions, mass distributions, or regularities in their temporal fluctuations), which are to be investigated in the language of physical theories. Usually, most of the measured data is not only entirely irrelevant for the investigated phenomenon; it can also be disturbing. In spite of the large amount of data recorded, the measurement space can be very sparsely occupied. Therefore, in addition to mapping the data onto the theory, the selection of the subset of the data relevant to the physical problem to be analyzed has to be investigated first. In this view, phenomenon and measurement are linked by probability distributions, and may be suspended at a few measuring points (like e. g. in radio astronomy).

Background knowledge and model checking We already have a very sophisticated and successful model conception of the world fixed in mathematical theories. So the task is not to conclude from "something observed" to an explanatory relationship for the first time, but it consists in checking whether a model[5] is compatible with the measurement in the light of all known contexts. Given the granularity of the experimental data, model checking is no longer logical inference relating proposition to data. Moreover, the theory is not stated by a coherent set of logical propositions that could be used as background knowledge. Instead, the theory is expressed in the form of Monte Carlo simulations that map the physical knowledge about the investigated process with high precision and under consideration of all probabilistic parts of the problem description onto the charges, times, and positions to be registered in the measurement. Then data analysis classifies the difference between the measured values and the values expected from the theory and this is translated into physically relevant statements.

Modern scalable machine learning methods have been developed as a response to the situation of big data. The research question of how the levels of cognition can be integrated is still open. Neural networks model the low-level processes of recognition and action, which are not conscious, and have been tremendously successful in image

5 This model can, for example, contain the standard models of particle physics and cosmology and other theories confirmed so far. It is used for the explanation of the cosmos, which is supplemented by new effects of, say, dark matter, dark energy, sterile neutrinos or other not yet excluded phenomena of particle physics.

or vision understanding. Explaining the recognition patterns and allowing for a critical inspection of the trained neural network is not only attractive for the human supervision of computational learning; it also aims at a model of the interaction of neural and semantic cognition [358]. Again, the exploitation of contradictions plays a crucial role.

In order to apply these general insights to the relationship between experimental data recording and physical theory, we will first introduce the generic cycle of knowledge acquisition based on big data in physics. The key issues in data analysis will first be outlined per se in this framework before presenting recent methodological developments in the remainder of the book.

1.3 From the Physical Theory to the Physical Observable

The core of the statistical methods used in modern physics, together with the machine learning methods developed in computer science, consists of determining the probability densities needed to establish connections between the world of registered values and the world of formulas in theoretical physics. The principle of this form of knowledge acquisition is *top down* (in the forward direction); the aim is to derive phenomena from causes in simulations based on all existing physical knowledge by creating a complete virtual reality, including all known fluctuations and oscillations. On the one hand, the predictions of the measurement results calculated this way can be directly compared with experimental measurements.[6] On the other, this virtual reality, in which the (virtual) truth and the (virtual) measurement results are always known at the same time, can be used to check the validity, efficiency, and robustness of the applied inference methods of machine learning to be used in the backward direction, or *bottom up*, inferring elements of the physical theory from the data. Since our interest in this chapter is in epistemological understanding, we will refrain from describing the individual methods and their linkages, though these will be discussed in detail later in the book.

The paradigm is outlined in Figure 1.2. If an initial situation is given in the theory (the initial position of the blue sphere), its fall through a generalized "Galton's nail board" can be calculated *top down*. At each black nail (according to Galton) or red nail (in generalization), the sphere is deflected to the right or left until it lands in one of the square detectors or—as happens in real life, when the detector does not recognize a signal—gets lost in between.

The probability distribution of signatures for a given theory value can thus be calculated with presupposed theoretical knowledge. However, only the experimental

6 This comparison could show that measurement and theory do not fit each other, which would then be a starting point for a technical search for the reasons. Are there numerical problems? Is the experimental technique adequately described? However, this direct comparison can justify no deeper *bottom-up* inferences from experiment to theory. See Section 4.3 for solutions of the inverse problems.

signature (sphere in the blue square) is given after a measurement. Conversely, the epistemological problem requires determining the probability distribution for an initial value formulated in the nomenclature of theory *bottom up* based on the measured signature. Such a calculation is possible if certain methodical boundary conditions are met. Its epistemological meaning and the conditions are explained in this chapter.

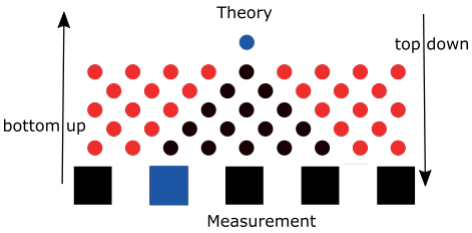


Fig. 1.2: Sketch of the relationship between theory and measurement in a simple model.

To understand the process, we first consider the *top-down* calculation of an experimental effect resulting from postulating a function $f(x)$ to be investigated in the frame of a physical model. Subsequently, the specific probability distributions of the data to be detected and registered by a suitable detector are to be calculated.

A Generic Detector We collect the experimental data electronically and hence free of subjective influences such as human reading errors. This fact also means that everything we can measure must ultimately always result from electromagnetic interaction, i.e., from the existence of charges or their change. The interaction of charges with the detector is read out electronically and (if necessary) stored. Such a measurement in the form of electronic detection of charge in a specific part of the detector will always take place at a particular point in time, e.g., at the end of a defined time interval in which the charge was previously deposited in that sensitive volume. This way, it can be compared, for example, whether a resulting voltage to be determined lies within an interval of reference voltages. The width of the reference voltage intervals correlates with the measurement accuracy.

Using electrical oscillators, high-precision clocks can be built. If necessary, these clocks can be used so that the sketched read-out process takes place in very short time intervals.[7] Thus the chronological sequence of charge measurements in the intervals represents the *bin-wise*, *digitized* measurement of time dependencies.

7 In practice, this can take place today with high frequencies of more than 10^9Hz, i.e., more often than one billion times per second. For the argument, the size of the frequency, resp. the time interval of the measurement is irrelevant.

A detector usually consists of a large number of sub-detectors (pixels). In each of these sub-detectors, the described processes take place. We know the position, form, and size of these electrically sensitive components. Thus a spatial image of the observed phenomenon can be created for each time interval. Because of the sub-detectors discrete nature, this image is also spatially divided into intervals, or *pixelated*. Neither the temporal nor the spatial intervals in which the detector is sensitive, have to fit together without gaps. A loss of signals is always possible. The theoretically possible signals will be registered only with a certain probability.

Therefore, the result of a measurement is always a spatially and temporally discrete sequence of numbers (integers). Its relative accuracy depends on the design of the experiment and the data acquisition electronics, but it is limited in any case. The exact amount of charge recorded electronically will always depend on thermal, problem-related numerical, or quantum mechanical fluctuations, which create differences in the digitized results. Finally, there will always be several (maybe unwanted) physical possible causes that change the detector's charge state. In any case, and beyond other causes, the desired signals are also influenced by electronic noise.

Everything on which our measurement of the world is based, everything that we can have as an unquestionable result of a single measurement, is therefore a multi-dimensional, spatially and temporally discrete frequency or probability distribution of the measured charges, positions, and times. To make possible statements based on such measurements, the measurements must be repeated until the shapes of the probability distributions are known with sufficient accuracy. In an idealized form, we can write a measured probability distribution as a continuous function $g(y)$, taking into account that the actual measurement result consists of the discrete elements of a vector $\vec{g}$, which contains in each element i the normalized frequency distribution, i.e. the normalized sum of the entries between the minimum and maximum y_i.

The Connection between Theory and Measurement The connection between a prediction of theoretical physics[8] $f(x)$ and the experimental result $g(y)$ is given by Equation 1.3. For simplicity's sake, the formula is only noted one-dimensional and continuous. The equation is nothing more than an expression of Plato's cave allegory

8 This can be a single value, a prediction in an interval, or an unlimited function.

we discussed above:[9,10]

$$g(y) = \int_a^b A(y, x) \cdot f(x)\,\mathrm{d}x + b(y) \tag{1.3}$$

In this equation, x is the variable of a physical theory whose connection with the measurement must be examined. For example, x could be an elementary particle's energy whose existence and properties must be investigated in a detector. Measured are the integer numbers y (charges in time intervals at specific positions). After a sufficiently large number of measurement repetitions, a frequency distribution of $g(y)$ containing the (many) single measurements of y can be specified. Also, the occurrence of an x is a discrete act, either because it is discrete (decay of an atomic nucleus leading to the emission of the elementary particle under investigation) or because the intensity of x is averaged over the time measurement interval.

An occurrence of x thus leads to a measurement of a state y. Between both worlds of theory and measurement results, there can be many levels of theoretical description ("Galton's nail boards in all imaginable forms and irregularities"). For example, we consider a telescope to detect high-energy neutrinos from various sources: If the first particle here is a muon-neutrino, it could, in a weak interaction, generate a muon which, as it propagates, reaches the detector and, via various forms of electromagnetic interaction, deposits energy there, part of which is emitted as Cherenkov light. These photons could pass through the detector if transparently designed. They may be scattered or absorbed until they cause the photoelectric effect in the detection instrument, which in turn may trigger electrons that—after amplification—can be detected by the measuring electronics.

All of the above processes must be understood in theory with high precision. This fact presupposes that this theoretical understanding is based on well-confirmed background knowledge established by other experiments. Since stochastic elements of thermodynamics, quantum physics, or quantum field theories sooner or later appear in the ladder of all descriptive theories (as in the example), even a fixed x will always pro-

9 In terms of Plato's cave, the $g(y)$ corresponds to the shadows on the wall, which result from the unknown reality $f(x)$ transformed by an also unknown imaging procedure $A(y, x)$ and that are disturbed by an unknown number of unwanted artifacts $b(y)$.

10 How difficult it is to discuss topics in an interdisciplinary way becomes clear also from the nomenclature in this book. In this introductory chapter, we follow the notation common in physics, mathematics, and rationalist epistemology, in which there is a dependence of physical quantities x determined by a theory, from which the randomly distributed observables $y = f(x)$ are calculated. In computer science, the view of dependence is reversed. There, the observables, known as features, are denoted by x, and the classifications derived from them are denoted by y, as would correspond to an empiricist-constructivist interpretation. In this book, the notation of computer science is retained in Chapter 10. Epistemologically, however, Chapter 10 is fully consistent with the considerations of probabilistic rationalism as we present it here.

duce a different y with a frequency distribution $g(y)$ characteristic for the measurement process, if the measurement is repeated several times.

As already discussed in consideration of the generic detector, we can assume that each measurement is repeated often, so that ensembles of measurement results $g(y)$ always have to be considered. The single measurements central to the classical epistemological discussions in the philosophy of science, or the significance of single measurements' analysis, would have to be treated as special cases. The components of the Equation 1.3 can thus be explained as follows:

- $\mathbf{g}(\mathbf{y})$ is a frequency or probability distribution **as obtained or derived from very many repetitions of the measurement**. This distribution represents the result of the measurement. Since the data is recorded electronically, it can only be based on distributions of positions, charges, and times or quantities calculated from them. This frequency distribution of the measurement results is additively composed of two components of different meanings:
- $\mathbf{b}(\mathbf{y})$ is the contribution of **unintentionally measured components**. This fact refers to signals recorded only for technical reasons because it is *a priori* unknown whether, say, a change in the detector's charge state is caused by the phenomenon under investigation or by another reason unrelated to the investigated question. The cause may be some form of electronic noise, or it may consist of various signals that can be physically analyzed and are important for answering other questions that do not play any role concerning the current question. This contribution, potentially consisting of different components, is called *background*. This background can be derived from theory or measured directly with the detector (e.g., with the signal switched off, if possible). That $b(y)$ must be a frequency, or probability distribution is given by the aforementioned technical reasons.
- $\mathbf{f}(\mathbf{x})$ is the **frequency distribution of the variable x that is to be inferred in the experiment** and that causes the measurement results. The variable x is written as part of the world of physical theories in physical units. It must therefore be translated into the world of electronically recorded positions, charges, and times or the quantities calculated from them, in which $g(y)$ is given. This translation depicts the understanding of the measurement process. Fixed values for x are conceivable but are practically unrealizable. In experiments, it is almost always the case that only a very narrow probability density around the desired value can be realized.
- The **translation of a single value x into a contribution to the measured frequency distribution** $g(y)$ is performed by the so-called design function $\mathbf{A}(\mathbf{y}, \mathbf{x})$. This function depicts all physical properties of the experiment, including the subdetectors. It summarizes the sequence of all interactions and their stochastically generated results, in our example, this corresponds to all that intermediates between the energy of the primary neutrino x and the frequency distribution $g(y)$ registered by the electronics. Only in rare and simple cases can $A(y, x)$ be given analytically as a solution of nested integrals. How this function can be numerically determined is explained below in the section on Monte Carlo calculations.

– By the integral $\int \mathbf{A}(\mathbf{y}, \mathbf{x}) \cdot \mathbf{f}(\mathbf{x})\, \mathbf{dx}$ all contributions containing the **relevant information** for the examined question are summed up to the measured distribution. To calculate the contribution, the causal theoretical function $f(x)$ is folded with the design function $A(y, x)$, which translates the values from $f(x)$ into the measured units of $g(y)$. The integration must be done via the probability (resp. frequency) $f(x)$ of all possibly contributing x (via the energy spectrum of the neutrinos to be detected, say) to obtain the sum of all contributions to the signature $g(y)$ to be measured.

If the physical function $f(x)$ is known, the expected distribution of the measured values $g(y)$ can be derived if the design function $A(y, x)$ and the background $b(y)$ are also known. Interpreting Equation 1.3 as a modern notation of Plato's cave allegory, the mapping of theory onto experiment is described from an omniscient perspective, in which the experimental result can only be assigned to the theory because all involved functions are known. If the blue sphere's position above the Galton's board is known, the frequency with which it falls in this or that pot detector can be calculated. This procedure describes the inference from cause to effect.

Monte Carlo Simulations As evident from the example of the neutrino detection, the previous discussion shows that the central problem is the calculation of the design function $A(y, x)$. For practical purposes, it cannot be expected that this function can be given by analytically solving the large number of nested integrals that describe the different stages of the interactions. In the search for a numerical representation of the design function, however, the *Monte Carlo* method to solve integrals by stochastic evaluation is used.

To illustrate this method, imagine that an unknown area and known area enveloping the first one are evenly covered with randomly distributed points. The number of points on both surfaces can be counted, and their ratio can be calculated to determine the unknown surface.[11,12] The Monte Carlo method is called *Monte Carlo* because of the random scattering of the points. By a high number of randomly distributed points, arbitrarily exact results can also be obtained on sub surfaces. This is relevant for the mentioned nesting of processes. What at first seems to be only a halfway smart numerical trick to the solution of nested integrals, opens up on closer inspection a multitude of epistemologically relevant possibilities. Let us now inspect the world of Monte Carlo simulations in more detail.

11 There are more elegant numerical ways to perform the method in practice.

12 A circular surface can be calculated (as a simple example) so that a square with an inscribed circle is evenly covered with equally distributed random points. To determine the circular area, one counts the points in the square and the circle and finds the ratio of the number of points to the area.

For this purpose, again, as an example, the propagation of the elementary particle muon is discussed: Muons are "heavy electrons" whose interaction probability is so low that with the appropriate starting energy, they can penetrate kilometers into materials such as stone or water. Various processes (the most important being ionization, pair formation, bremsstrahlung, and nuclear interaction) may occur. Each of these processes takes place with a specific probability (depending on the energy and material). It has specific properties regarding the amount of energy loss suffered by the muon and the muon track's deflection during an interaction. The corresponding probabilities have been calculated in quantum field theory. On their way through matter, muons, like spheres on Galton's board, undergo a multitude of corresponding interactions [148, 229].

Because of the originally stochastic character of the (quantum field) interactions, a muon's path cannot be determined deterministically. However, some of its properties can be calculated by suitable averaging and integration over all the probability distributions involved. These include the mean position, the mean velocity, the mean deviation of the muon from a straight path after crossing a certain thickness of the material, and other quantities. This form of calculation corresponds to classical approaches and the attempt to produce testable predictions [155]. Apart from the effort involved in this procedure, the method has the disadvantage that it can provide only the centers and perhaps the moments of the resulting probability distributions for the investigated properties ($g(y)$).

However, as discussed above, we are interested in the complete probability distributions of specific signatures. This desired result is obtained by randomly generating individual muon trajectories using the Monte Carlo method. In our example, it is possible to determine, based on the quantum field theory, the interaction probabilities and to derive where the next interaction will occur. Then, we can find which of the partial cross-sections mentioned above (pair production, bremsstrahlung, ...) will be employed, how much energy will be released in the material, and then at what angle the muon continues its path. In this way, the possible muon trajectory can be calculated step by step until the muon stops or decays.

With the large numbers of possible muon trajectories that result, the initial condition (muons start at the position $\vec{p}$ in direction $\vec{a}$ with energy E) can be correlated with, say, the distribution of the charges, times, or positions y_i at which (some of the) muons hit the detector, or simultaneously the number of hit sub-detectors y_j. This is always and independently of all the individual properties of a specific path possible. Geometrical peculiarities (different materials at different positions, geometrical boundary conditions, etc.) can easily be integrated into the process so that, finally, very realistic simulation results can be generated. A distribution of the resulting y is always assigned to a primary x. The design function $A(y, x)$ is thus obtained from determining the distribution of the final y assigned to a primary x.

From Integrals to Matrices A further step is relevant for understanding the cognitive process because it prepares the inversion of the direction of cognition (*top down* → *bottom up*). So far, we have written the cave allegory as a convolutional **integral**, taking into account the continuous character of theoretically calculated probability distributions. However, due to the necessarily discrete nature of the measurement results, Equation 1.3 can also be written as Equation 1.4 in the language of linear algebra:

$$\vec{g} = A\vec{x} + \vec{b} \tag{1.4}$$

The vectors $\vec{g}$, $\vec{x}$ and $\vec{b}$ contain in each element the number of entries counted in the respective specific interval as binned frequency distributions. The size of the intervals is arbitrary and can be chosen depending on the problem. In $\vec{b}$ the contributions of the background are counted, in $\vec{g}$ all contributions to be measured, and in $\vec{x}$ the initially primarily continuous function $f(x)$ is integrated bin-wise from the lower to the upper interval boundary. The translation of numbers with physical units into numbers in detector units is done here using the design matrix A, in which (as described) all Monte Carlo results are assigned and counted. In this picture, the properties of the matrix A can be understood as follows.

- An ideal detector would register every particle crossing its geometrical volume and measure observables for all its properties with absolute accuracy. The determination of the particle energy in such a detector could work via the number of cells hit by the particle and, per definition, perfectly correlate to the energy. The matrix A is then a diagonal matrix where the probabilities contained on the main diagonal (besides normalization factors) would be equal to 1. All other entries are equal to 0.
- A detector may be crossed by some particles that hit its mechanical structure but not its sensitive volume. If the existing measurements were absolutely accurate, the matrix A would also be represented by a diagonal matrix. Because of the lower reconstruction probability, the main diagonal's probabilities are less than or equal to one. All other entries are zero here as well.
- If, in addition, the measuring accuracy is not perfect, i.e., if several possible hit numbers are assigned to **one** energy (perhaps because the track lengths of the particles with the same energy in the detector and so the number of hit cells vary *also* by geometrical reasons), the elements next to the main diagonal will begin to populate with entries unequal to 0. This range becomes the wider, the probabilities outside the main diagonal become the larger, the worse the resolution (the higher the probability for misallocations).

The matrix A is thus determined as the result of the Monte Carlo simulation by counting the combinations of the physical quantity x (in our example, this corresponds to energy) and the detector quantity y (in our example, this corresponds to the number of hits in a sub-detector) in each element of the matrix. A high number of simulated entries in one element means that this number was determined with a low statistical error. Few

entries mean that this combination occurs very rarely and is therefore only known with a relatively sizeable statistical error. Normalization to the number of generated events produces probabilities in the matrix.

In summary, it can be stated that single measurements produce singular entries to the concerned probability distributions. A large number of measurements determines the shape of these distributions. With numerically highly accurate Monte Carlo simulations, based on all known physical knowledge, the expected measurement result can be calculated (deductively concluded) for a known function $f(x)$. The whole complicated procedure can be summarized by a simple linear algebra relation.

1.4 From the Measured Variable to the Measuring Point

With the goal in mind of concluding from measurements to theories, one may read Equation 1.3 in the integral form in the backward direction: $g(y)$ is measured, A and $b(y)$ are then calculated or measured, and $f(x)$ is unknown and to be determined. With this status of knowledge, the unknown physical function $f(x)$ can be inferred from the measured numbers. In this reading, Equation 1.3 is a Fredholm's integral equation of the second kind. Because the described process in the forward direction is a convolution integral, the solution of this inverse problem is also called deconvolution.

The equation in its algebraic form (1.4) symbolically states the steps necessary for a solution of the problem. Therefore these mathematical steps shall be separately discussed before they are put into the final context of the multi-dimensional analysis of large amounts of data with methods of machine learning:

1. **Monte Carlo calculation of the design matrix A:** As explained above, to determine the matrix A, the simulation problem in the forward direction first has to be solved. For this purpose, it is necessary to write the problem so that the simulation only includes correlations that can be assumed to be known for the present analysis.[13] To determine A, a function $\hat{f}(x)$ (or a resulting vector $\vec{x}(\hat{f}(x))$) is assumed, which serves only to determine the correlation between $\vec{f}$ and $\vec{g}$ using the Monte Carlo method. The function $\hat{f}(x)$ chosen for this purpose is, in principle, arbitrary and only for computational (resource-saving) reasons should $\hat{f}(x)$ be chosen such that A can be calculated with minimal computational costs. We can consider the simulation problem solved with the previous explanations.

13 Of course, the input for every simulation input is always subject to uncertainties from different sources. How these uncertainties affect the results is examined and considered in special analyses but should not distract our thinking. As a result of such systematic investigations, one finds that in addition to the *statistical uncertainty* from the number of measurements comes a further *systematic uncertainty* from ignorance of the exact conditions and correlations. The extent of this ignorance, however, is assumed to be estimated based on previous experiments.

From an epistemological point of view, it is essential to notice that the reversal of the inference direction also changes the direction in which the matrix A is read. For inferences from cause to effect, A determines which probability distribution of experimental signatures $\vec{g}$ is assigned to a dedicated theory value x. Here, for the inferences from effect to cause, A assigns to a dedicated measured signature g the probability distribution of possible theoretical causes $\vec{x}$. This also applies to the probability that a measured signature is triggered by a signal event, as by a background event. Furthermore, this is also valid for the probability with which the size of a property from the world of physical theory (the energy) can be assigned to a measured signature (hit count).

2. **Background subtraction**: To subtract the background $b(y)$, it must either have been simulated with Monte Carlo methods or alternatively have been measured with the detector under special conditions. Then it can be (symbolically) subtracted from $g(y)$ in Equation (1.4):

$$\vec{g} - \vec{b} = A\vec{x} \tag{1.5}$$

The distinction between signal and background in real measurement data will be discussed below.

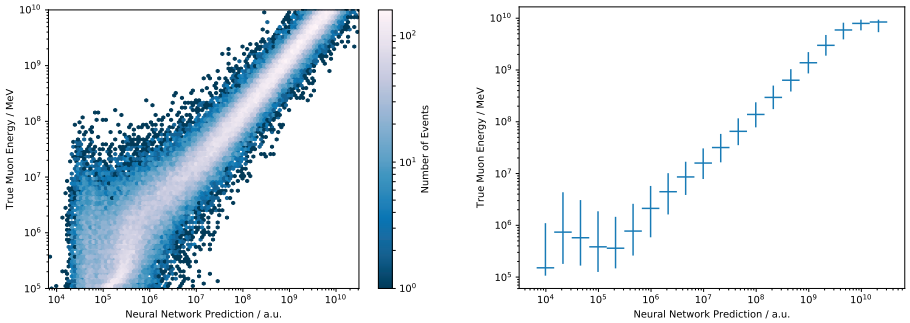
3. **Solution of the inverse problem**: The goal of calculating the vector $\vec{x}$ from the measurement of $\vec{g}$ would be fulfilled by multiplying Equation 1.5 with the inverted matrix A from the left:

$$\mathbf{A^{-1}}\left(\vec{g} - \vec{b}\right) = \vec{x} \tag{1.6}$$

Apart from the mathematical difficulties that can occur with matrix inversions, one central problem remains to be discussed: When calculating the matrix A, we have seen that its elements are based on the simulation of large numbers of entries. Near the main diagonal of A, there are numbers close to probabilities of 1 with small statistical uncertainties derived from many entries. Far from the main diagonal, elements with only a few entries generate probabilities close to 0 based on a small entry number and correspondingly large statistical errors. An example of correlations for the construction of a realistic matrix A is given in Figure 1.3, where the estimation of the energy of muons is presented as it might appear in a generic neutrino telescope.

"Measured" in the simulated detector of Figure 1.3 are times and charges at many positions for 100 000 muons with different energies. Then, using a trained neural net[14], this multi-dimensional information is projected to a one-dimensional

14 Neural nets as machine learning algorithms will be discussed below. For the moment, we understand the "trained neural net" as a somehow invented function to calculate numbers more or less correlated to energy.



(a) Color-coded correlation of combinations between a calculated neural net output and a physical quantity (muon energy).

(b) Binned mean values of the left frequency distribution. The error bars indicate the width of the distribution.

Fig. 1.3: Example: Estimation of the muon energy as it might appear in a generic neutrino telescope. The figure shows the inference from a neural net output (x-axis) to the true energy (y-axis). The difference between a simple calibration function (right) and a frequency distribution (left) for the same data is obvious. The Monte Carlo simulations were executed with the program PROPOSAL [148, 229, 355].

quantity in arbitrary units correlated to the energy. The figure shows the typical correlation between the output of the neural net on the (x-axis) and the logarithm of the true neutrino energy in the simulation (y-axis). A given muon energy can lead to different neural net outputs; vice versa, a certain neural net output can have been activated by different muon energies. This is a necessary consequence of the thermodynamic and quantum physical processes involved. The correlation with its statistical consequences via matrix A is considered in the procedure discussed here.

On the left of Figure 1.3, the frequency of the occurring combinations is color-coded. Rare single entries are visible as single black points. Below it is shown that they cause the ill-posedness of the problem. In classical analytical approaches, one would have tried to identify a unique calibration function fitted to the mean-values and width of the distribution of the correlations, as depicted on the right in Figure 1.3 for the same data. Such a calibration function, however, leads to false ideas about the unambiguousness of the assignment between the derived quantity (neural net output) and the searched-for physical quantity.

When matrix A is inverted, large numbers become small and vice versa. Nevertheless, large uncertainties remain large and small ones remain small. Thus, single, random entries in A, irrelevant in the forward direction, might become large, noisy terms (large numbers with considerable uncertainty) after inversion, disturbing a clean determination of a result. Therefore the class of these inverse problems belongs to the "ill-posed problems", where small numerical differences in the ar-

gument (single random entry in A) can produce large differences in the result. This unpleasant situation can be cured by suppressing the distracting terms in A. This so-called regularization is based on additional assumptions about the properties of the solution chosen to eliminate the fluctuations. Thus, a similar but not ill-posed problem is created. That way, however, necessarily "prejudices" about the properties of the solution enter the mathematical problem. Therefore, the impact of the regularization on the solution of the problem has to be investigated. In no case it is permissible to change the physical interpretation of the unfolding result. We consider the measurement of the function $f(x) = N \cdot x \cdot e^{-ax}$ in 20 measuring points (bins), as an example. Instead of using simulations to determine the detector properties, we assume a simple illustrative Matrix A: our measurement is performed with a detector that does not lose any signals but assigns them with a probability of ϵ (in a friendly manner symmetrically) to correlated matrix elements. If the probability for mismatches ϵ would be = 0, the matrix would be diagonal and the detector ideal. The matrix is shown in Equation 1.7.

$$A = \begin{pmatrix} 1-\epsilon & \epsilon & 0 & 0 & \dots & 0 \\ \epsilon & 1-2\epsilon & \epsilon & 0 & \dots & 0 \\ 0 & \epsilon & 1-2\epsilon & \epsilon & \dots & 0 \\ \vdots & & & \ddots & & \vdots \\ 0 & \dots & 0 & \epsilon & 1-2\epsilon & \epsilon \\ 0 & \dots & 0 & 0 & \epsilon & 1-\epsilon \end{pmatrix} \tag{1.7}$$

In Figure 1.4 the function $f(x)$ to be measured is shown as a solid line. If A were inverted and the Equation 1.6 applied, the strongly oscillating blue measuring points might be obtained because this is an ill-posed problem. If the relevant uncertain terms are moderately suppressed by regularization, the green measuring points are derived, fitting very well to the initially assumed function. However, regularization that is too strong forces the (now violet) measuring points more and more to a straight horizontal line.

Epistemologically, it is notable that inferences from the effect on the cause lead to inverse problems. Inverse problems are by their nature ill-posed and can produce fluctuating solutions. The fluctuations can be suppressed by regularization, i.e., creating a suitable variation of the mathematical function to be solved. Inevitably, however, it is then necessary to weigh physically possible fluctuations of the measurement result against the bias strength of regularization as a smoothing condition of the solution.

Multidimensionality Before discussing the solution of the problem with means of computer science, the limitation of the observed quantity to one dimension shall be removed. With each measuring act in an actual detector, many measured variables are electronically read out. Further quantities are calculated from these original num-

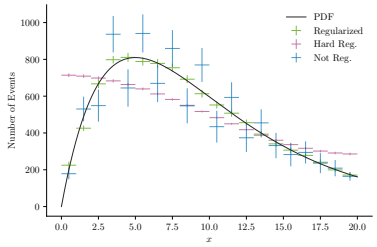


Fig. 1.4: Example solution of the inverse problem without regularization (blue measuring points), with appropriate regularization (green measuring points) and with too hard regularization (violet measuring points).

bers. Analogous to the above discussion, the central data analysis question is how probability densities for the physical properties of the investigated events can be calculated from (highly) multi-dimensional measured signatures. Not only the measured data but also the properties to be reconstructed in the world of physics will usually be multi-dimensional (although in fewer dimensions).

Signal-Background Separation In the first step of data analysis, the signal events of interest must be filtered out of the entire data set (subtraction of the background b, see above). Since the unwanted but technically unavoidable background can often be many orders of magnitude larger than the searched-for signal, this indicates the search for a tiny needle in a gigantic haystack. In the exemplary astroparticle physics data analyses, the background may exceed the signal by up to 10 orders of magnitude. In our example, a wrong decision may only be made less than once in a trillion events. Also, for less ambitious problems, fast, highly precise, and reproducible decision procedures are required in which the error rate is also calculable. Such tasks can no longer be solved with the necessary precision by human decisions. Instead, they require the use of high-precision robust rating algorithms developed in the framework of machine learning.

These **machine learning** rating algorithms are optimized according to to-be-investigated quality criteria. First, the signal-background separation task is defined in the world of simulations. In this world, it is known which multi-dimensional signatures occur with which frequency and which of these signatures are assigned to the signal or the background class, respectively. For this purpose, many examples of the sought signal events and all possible variants of background events are generated with Monte Carlo methods.

The goal of machine learning is to identify mathematical operations to be applied to the many dimensions of generated and, in principle, measurable variables (number of hits in the detector, their spatial distribution and temporal sequence, size of the

deposited charge, etc.) and then to assign a value to each event on an at least one-dimensional finite scale. (In advanced applications, as discussed in Chapter 9, multi-dimensional output assignments are also possible.) The variables, the mathematical operation, and their internal structure are optimized to maximize the distance between signal and background events on that scale. For this optimization, simulated signal and background events are successively evaluated by the algorithm. The larger the number of these examples is, the better the algorithm can be optimized. Anthropomorphically, the process of entering an example is called "showing", and the process of optimization is called "learning".

Thus, the algorithms are optimized for separation tasks. Since the necessary decisions are statistical processes, the accuracy with which a decision is made must be specified. Furthermore, it must be tested whether the algorithm is sufficiently tolerant to unavoidable imperfections of the simulation compared with the experimentally determined data. If one is finally sure about the procedure, one knows how many simulated signal and background events are expected in the data set after applying the separation algorithm, including the uncertainties of these numbers. Applying the so-determined algorithm to the world of measured data, one can transfer the selection rates determined in the simulation to the world of measurements.

In machine learning research, many algorithms have been developed in which representations are structured with entirely different approaches to map multi-dimensional measurement results to one-dimensional order parameters. Their regularization and robustness to small changes are investigated by machine learning theory, see Chapter 3.

The performance of the algorithm in recognizing structures and its dependence on the precision and the number of Monte Carlo events depend on how many free parameters have to be calculated to solve the task. Only a few free parameters are used in *discriminant analyses*, in which for the separation task **flat**, n – 1-dimensional hyperplanes are optimally placed in the n-dimensional space of the measured signal and background distributions. Distributions that lie curved and nestled into each other are obviously difficult to separate with this method. Hence, kernel functions such as the Radial Basis Function have been introduced to map the measurements to a space that allows correct separation by hyperplanes, here, correctly separating signals from the background [380]. The "kernel trick" [343] is indirectly present in neural networks.

In extreme cases, the optimization of algorithms that are capable of learning non-linear functions may lead to a separation in the simulation world that is too good, in the sense that the positions *of all* generated events are parameterized, or "learned". This effect is called *overtraining* or overfitting. If applied to real data or even to another simulation data set, such a learned model would fail because it does not find signatures to be *identical* to those learned with the simulation.

Which machine learning approach is chosen and how it is validated that it delivers adequate and expected results has to be analyzed separately for each separation task.

The first examples of such algorithms that we want to consider are simple decision trees. A decision tree is a hierarchical set of conditions that defines that a dataset (event) is classified in one or another form if the measured parameters lie in specific logically nested intervals. Very simple classifications can be executed in a single decision tree with a small number of conditions. The more complicated the signal-background separation is, the more conditions would have to be nested and inserted into the tree. These conditions, however, are determined based on the simulated Monte Carlo events. If a sufficiently large number of conditions is selected, individual properties of the Monte Carlo events are mapped, and thus the classification of all simulated events could be stored. In this case, the algorithm would not be able to learn to generalize the topology. Such a generalization would be indicated by a robust parameterization of the placing of the searched-for signal-background-separation-plane. Without this generalization, the separation will fail in all practical applications.

Alternatively, what is known as a random forest can be applied. In a random forest, an ensemble of decision trees with moderate branching possibilities is defined instead of one widely ramified, deep tree. Not all trees of the forest are provided with the same information. The trees are further only given randomly selected information (some simulated observables plus the label showing the true class affiliation). Also, on which branching level which information is used in the decision process is determined randomly. After the numerical optimization of a random forest with simulated events (called "training"), all trees of the forest "vote" by calculating their scale value for an event, answering the question of whether an event belongs to the signal or background. This results in the above required one-dimensional scale on whose one side the trees ideally vote for "background" (scale value close to 0), and on whose other side the trees vote ideally for "signal" (scale value close to 1). A cut on this scale can then be used to determine the ratio of signal to background events, the purity of the finally selected signal sample, the efficiency of the selection (how many signal events survive the cut?), the robustness (is the forest as desired insensitive to, say, Monte Carlo fluctuations?), or in the agreement between Monte Carlo and measured data. Finally, the decision process of the specially trained and tested random forest is executed on the measured data. As desired, a sample is obtained in which the number of signal and background events and their statistical uncertainties are known.

In this way one finds a mapping from the high-dimensional space of the (real or simulated) measurement data to a one-dimensional order parameter. The measured data's multi-dimensional character is preserved and supplemented by the order parameter calculated by the random forest. Considering the measured variables as carriers of a multi-dimensional probability distribution, the signal-background separation can be understood as a transformation of the multi-dimensional probability distributions for the signal and background samples enabling a separation of both topologies by a simple linear cut. The strength of a selected signal can now be given as the probability that the signal is inconsistent with a random background fluctuation. This signal data set can then be selected and used for further analysis.

Solution of the Inverse Problem After the signal-background separation, a dataset may be assumed containing only a known small number of background events that can be considered as negligible for our purposes. In the example, one would now be sure to have a dataset consisting of signatures of the sought-for neutrinos as signal events. Now, these signal events must be examined for other properties, such as the energy spectrum of the neutrinos.

The paradigm of signal-background separation requires that signal and background are perfectly known in all properties except the relative number of events in the classes. Therefore, they can be described just as correctly in the simulation. If, however, a quantity such as the energy is to be measured, then (at least) the energy distribution of the singular events cannot be wholly known. Therefore the simulation cannot describe the distribution to be measured perfectly. For the deconvolution, a weaker paradigm is postulated, according to which only the experimental effect of the searched phenomenon has to be known. Only the simulation of the detector in all of its properties (e.g., the response to the energy) must agree in nature and in the Monte Carlo world.

Regardless, the unfolding analysis is based on the same data situation in measurement and simulation as the signal-background separation: we consider a multi-dimensional spatio-temporal pattern of positions, times, charges, and variables derived. The Monte Carlo dataset for each event also contains the true information about the quantity (energy) to be measured, which is unknown in the world of experimentally recorded data. Again, a mapping from the multi-dimensional space to a one-dimensional parameter is sought. And again, this is a task that can be formulated mathematically as an optimization task: the simulated examples are used or "shown" one after the other to a machine learning algorithm to adjust the inner parameters of the learning algorithm so that the sought-after quantity (in our example, the energy of neutrinos) can be determined with the best possible accuracy. In place of the bipolar finite scale for the separation, a continuous and unlimited magnitude (correlated to the energy, say) is calculated (cf. Figure 1.3 and the example given there). The primary considerations for machine learning, however, will not change. Suppose a relation between the many technical quantities (charge, time, position, and derived from that) and the quantity (energy) to be measured physically is established. In that case, the inverse problem (Equation 1.6) is solved under a controlled addition of assumptions to suppress unwanted properties of the solution (regularization). The analysis optimized in the Monte Carlo world is then equally applied to the measured data. If necessary, the signal to be measured is corrected for signatures that could not leave a trace in the detector. Also, a normalization of the event numbers considered so far to the physical theory quantities, including their unit, is performed by what is known as acceptance correction.

Thus, the deconvolution leads to a projection of the multi-dimensional measurement data into the one- or few-dimensional space of the quantities predicted in theoretical physics. Since the statistical character of the simulated or measured data was at no point limited or cut off, the expectation values and the corresponding confidence

intervals can be specified for each measuring point in physical units. The result of the analysis consists of measuring points with their statistical errors. If a definite prediction for a measured value was calculated in theoretical physics before the measurement, it is now possible to specify the probability with which this prediction and the measurement match. If the theory contains parameters that cannot be predicted, they can be adjusted according to the physical measurement points (fitted). These parameters can then be introduced into further calculations and tested there again.

1.5 The Epistemology of Physics in Concert with Machine Learning

As **epistemological lesson**, in a generalization of Popper's approach of falsification, we see that—even if some results are obvious—only probabilistic evaluations of measurements and theories are appropriate for the further discussion of the results and the consequences.

Since the basic ideas of Popper's *Logic of Scientific Discovery* were formulated, thinking about theoretical physics has changed. Originally, the evolution from Ptolemy to Copernicus to Kepler to Newton and finally to Einstein could be seen as a development that suggested the goal of elegantly developing theories using new approaches and mathematical methods and adding only a few measurable new parameters. Thus, this seemed to be a paradigm for the further development of theoretical physics. Today, whole classes of theories are proposed and calculated simultaneously. These classes of theories may contain many free parameters with possibly substantial uncertainties, so a complete refutation of such a class of theories is hardly possible. Modern epicycles, if you will, are built into the structure of the theory from the outset in such a way that they need not stand out as additional constructions. Nevertheless, there are, of course, parameter ranges within the classes of these proposed theories, which can be meaningfully investigated by experiment.

This does not prevent understanding theories as products of rational considerations, and the goal of these considerations may still be to develop a mathematical description of the world that is as complete and uniform as possible. However, since a class of theories can rarely be fully tested, it would be a very hard demand[15] to link the scientific validity of a theory to its complete refutability. However, an experimental test of parts of the parameter range may be possible.

Which parameter range is experimentally testable in which combination can often only be determined by means of Monte Carlo methods because of the large number of free parameters in the theories and the possible complexity of its dependencies.

15 With this demand one would remove many extensions of the Standard Model of particle physics from the field of scientific research because they contain parameters that are not measurable in this world.

In these approaches, numerical values for the free parameters are randomly drawn, based on which experimentally verifiable values are identifiable for, say, the resulting mass of a potentially existing dark matter particle or the flux of particles (neutrinos). The Monte Carlo method is thus suitable for testing the range of potentially occurring measurement results for two reasons. First, the relationship between the parameters and the possible measurement results is not always obvious. Second, parameter regions excluded by other experiments can easily be considered.

This fits nicely with the machine learning view of modeling mentioned above [276]. There, under the heading of *sloppy modeling*, every current state of knowledge was considered preliminary. In the view presented here, the knowledge is expanded iteratively and evolutionarily: cyclically, the definition of the problem, the simulation of measurement results based on existing knowledge, the search for deviations from it in the experiment, the interpretation of the measured probabilities in terms of theoretical physics, and finally the revision of the problem follow each other (Figure 1.5). The acquisition of knowledge is no longer based on complete inferences from the effect to the cause but instead on playing through all theoretically conceivable possibilities and constantly comparing and optimizing these predictions using new measurement data. The concrete and solid empirical basis for this purpose consists of the measured positions, times, charges, and the quantities calculated directly from them.

Note that this picture enhances the well-known Cross-Industrial Standard Process (CRISP) of data analysis [117]. First, the data is not in the form of a readily given database. Instead, data is produced based on physical knowledge through simulation and based on a physical detector and a triggering process. Both data-generating processes may exploit machine learning methods. Adversarial training using generative models might produce helpful additional data. On the other hand, active sampling might select the most useful data for learning. In principle, also the optimization of the trigger is a learning task. The cycle continues with four learning tasks, where the first two differ considerably from the CRISP modeling in that they work on streams and under real-time conditions. The following two learning tasks use databases that can be stratified beforehand. The physical interpretation of the results may lead to novel questions or inspire novel analysis approaches.

The preceding considerations show that the first solution to a problem is not possible without human intervention. Once solved, selection and reconstruction problems can be solved automatically over and over again. Based on this, the researchers involved in knowledge acquisition must now perform dedicated tasks. Theoretical physicists will develop new concepts for interpreting observed phenomena, constraining the parameter range. Experimental physicists will invent new techniques to enable qualitatively new measurements and further optimize existing techniques. Computer scientists will develop new algorithms and methods that will allow the ever-increasing amount of data to be analyzed more accurately with fewer and fewer resources.

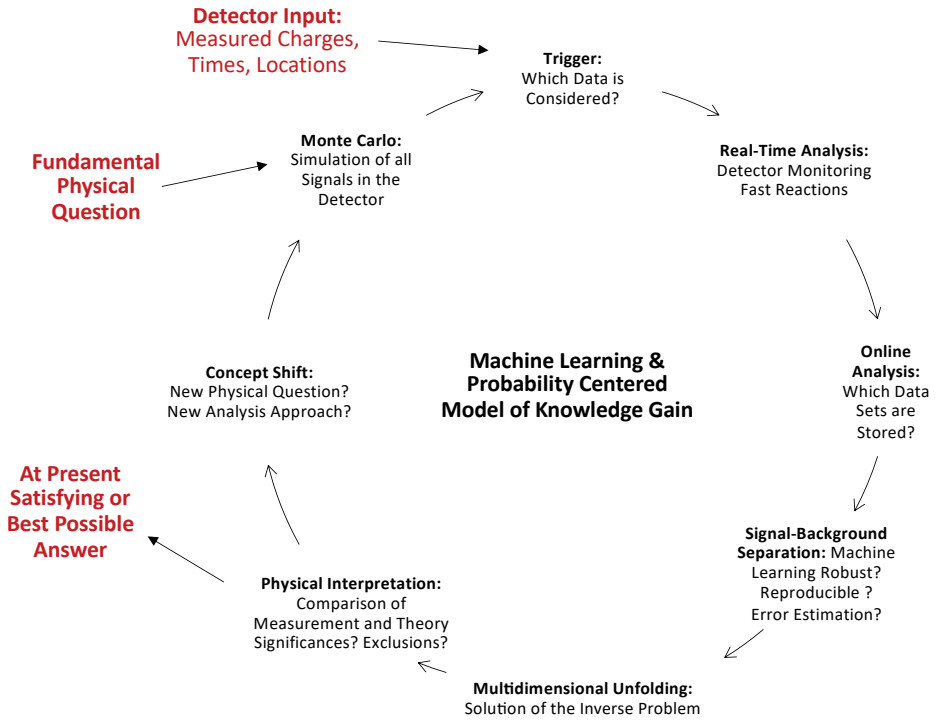


Fig. 1.5: Cycle of methodical steps for knowledge expansion. Indicated are the most important aspects under which the multidimensional probability distribution based on the measurements is investigated and transformed for quality assurance and knowledge gain. The figure symbolizes a process whose details depend on the problem being studied. In particular, cycles are conceivable that can skip certain steps or require additional steps (e.g. for data reduction or estimation of properties).

1.5.1 Further Reading in this Book

The process we have discussed here from an epistemological point of view is detailed by more technical contributions throughout this book. These contributions are presented in an order following the methodological steps shown in Figure 1.5. From data acquisition, including the preparing and calibrating of the detectors, through sampling and preprocessing data to classifier learning and unfolding, the chapters offer methods of how probabilistic rationalism can be performed in practice.

The methods presented and discussed in this book are suitable for application to problems from the environment of astroparticle and particle physics. Therefore, the underlying physical issues are presented and explained in Chapter 2.

Chapter 3 gives an overview of the key concepts from machine learning that will be used further on.

The first group of questions to be solved for each physical experiment leads from the acquisition of data through the structure of this data to the first trigger decision through which the experimental data are fed into the analysis cycle. Newly developed concepts in this regard are presented and discussed in Chapter 4.

Physical knowledge has advanced to such an extent that all measurable signals can, in principle, also be simulated. However, the calculations are so resource-intensive that machine learning methods can also be used here to reduce the effort. The correspondence between data and the Monte Carlo world must also be demonstrated technically and physically so that machine learners can be trained on the Monte Carlo data. Developments in this area are presented in Chapter 5.

How to access the research data on different time scales? Problems are posed by high-speed access to enormous amounts of data during the data analysis, long-term access to all research data for ten years after the end of the experiment, and permanent access to all physical results. Example approaches to this problem are discussed in Chapter 6.

The question of representation learning remains the key to scientific processes. It can be modeled explicitly by automated feature extraction from signals [270]. Feature extraction for physical experiments is presented in Chapter 7.

Since the recorded background often exceeds the signal under investigation by orders of magnitude, new efficient and robust methods for signal background separation become crucial. These are presented in Chapter 8.

Because of their great importance in applications, several new deep learning applications are explained in Chapter 9. Representations are learned implicitly by neural network-based learning because we might interpret higher weighted internal nodes at hidden layers as representations. References to representation learning in unsupervised learning can be found in [205]. Boltzmann machines are made for representation learning [307, 335]. Graph neural networks combine the advantages of probabilistic graphs with those of neural networks and successfully learn representations, because they introduce structure into deep learning [162, 395]. For an overview and current research, see Section 4.3 in Volume 1. In this book, the implicit approach is detailed for gamma-ray astronomy in Section 9.4.

The issue of inverse problems and deconvolutions, the way from the frequency distribution to the physical statement, is investigated in Chapter 10. There, the representation is explicitly designed for learning as an extra step in an overall learning process on streams.

2 Challenges in Particle and Astroparticle Physics

Bernhard Spaan
Wolfgang Rhode

Abstract: The elementary particles, the history, and the structure of our universe are studied in the physical disciplines of particle physics and astroparticle physics. Intending to study these fundamental processes and structures because of the expansion of the universe, both disciplines are closely connected. For this purpose, one can either study the high-energy interactions of particles in hot environments or analyze the information transmitted by messenger particles from distant events and sources in the universe. The former is studied in experiments at the Large Hadron Collider at CERN, the latter in various types of astroparticle detectors or telescopes. Both physical disciplines complement each other in the information accessible to them, are similar in the extreme demands placed on the resources required for data analysis and data quality, but differ due to the design and operation of the detectors in the generic properties of the collected data. Since the operation of the detectors is designed to answer specific fundamental questions, the guiding questions of particle and astroparticle physics are presented and explained in this chapter. Since many (if not all) of the later discussed exemplary applications of the computer science findings of the CRC 876 have been developed for the detectors LHCb, IceCube, MAGIC, FACT, and CTA, a special emphasis will be put on questions investigable with these detectors.

2.1 Physical Motivation, Problems, and Examples

In general, the epistemological considerations sketched in Chapter 1 do not only apply to the treatment of questions from a specific field of research; they claim general validity. However, this book is aimed at readers with different scientific backgrounds, so it seems appropriate to present a paradigmatic motivation for particle and astroparticle physics to which the examples given later may refer.

Our understanding of the physical world is essentially shaped by the two very successful theoretical standard models of cosmology and particle physics. The former focuses on the evolution of the universe in space and time, the latter on the interaction of its smallest particles. Both models are connected in many ways so that their statements complement but also control each other. Basically, both models are or were initially adapted to the cognitive preferences of man, symmetrical in theory—so symmetrical that for some time now, the justification of the asymmetry observed in the world has become a central research focus in the field of astrophysics and particle physics.

https://doi.org/10.1515/9783110785968-002

There are many fundamental questions that can be solved only by adding asymmetries to the standard models. Why do elementary particles have mass? Why is there more matter than antimatter? Why have no magnetic monopoles been detected (so far)? Why did the universe expand at different velocity scales? Why is matter distributed so differently in the universe? Why does dark matter exist? How (and where) can cosmic rays gain energy?

The physical disciplines of particle physics and astroparticle physics try to find ways to clarify these questions in different types of experiments. In particle physics, high-energy elementary particles are shot at each other in terrestrial accelerators under very controlled conditions so that the symmetry of interactions can be studied under the energetic conditions of the early universe. In astroparticle physics, astronomical objects are used to accelerate particles to the highest energies. And either these messenger particles themselves or their interaction products are detected and interpreted on Earth. The challenges in astroparticle physics and particle physics are complementary. In particle physics, the conditions under which the first interaction occurs in the experiment are very well controlled. The experiments consist of different functioning inhomogeneous subdetectors in which data with different physical interpretations are recorded. In astroparticle physics, the detectors and, thus, the recorded data are comparatively homogeneous. However, the experimental conditions, such as the particle type, the location of the interaction, and its energy, cannot be controlled. Moreover, the detectors can change their properties with, e.g., the weather conditions, the incidence path through the surrounding natural media, or the time during the measurement.

2.2 Astroparticle Physics

Astroparticle physics is an approximately three-decade-old field of research [157] that has emerged from the extension of astronomy by experimental methods and theoretical concepts from particle physics. Its understanding is impossible without the current theoretical standard models of particle physics and cosmology and their possible extensions. Thus, it is based on the classical field theories (for gravity) and quantum field theories (for particle interactions), which have not yet been unified. The essential components of astroparticle physics are shown in Figure 2.1.

In astrophysical sources (such as stars, supernova remnants, the nuclei of active galaxies, and many other objects), temperatures are so high that matter must be considered as plasma (atomic nuclei and electrons are separated and move independently). The charged particles cause irregular magnetic fields and are vice versa, accelerated by these magnetic fields towards very high energies. The electrons and nuclei from the plasma can interact with other particles or emit radiation so that the sources can finally be observed from the detection of the light of electromagnetic radiation between the radio and gamma frequency ranges. Also, the neutrinos produced in hadronic interactions in the sources can be detected. A general glow in the sky of astroparticle

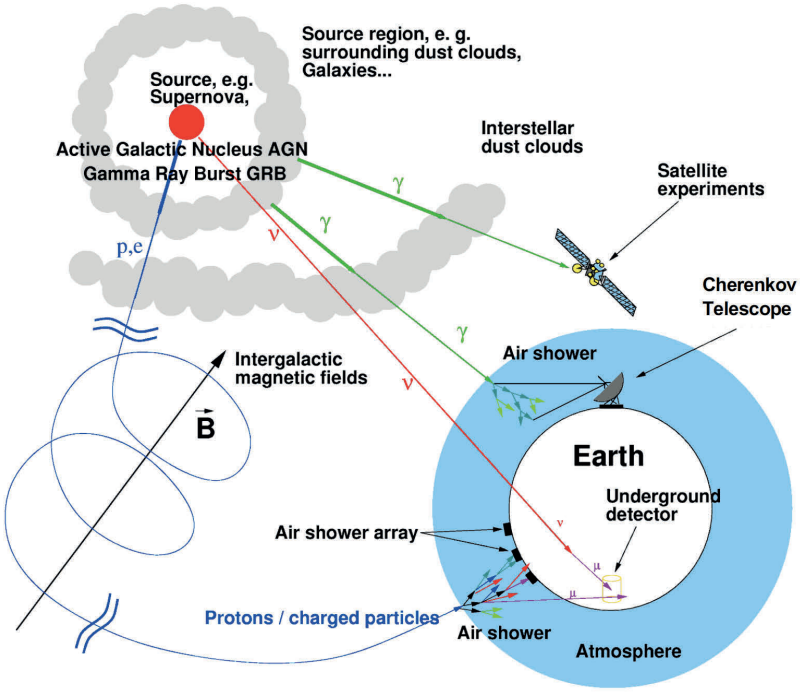


Fig. 2.1: Sketch of astroparticle physics. The figure shows the astronomical sources, the messenger particles, and the different detection scenarios (adapted from [385]).

physics is produced by the charged nuclei, which mostly lose their directional information when flying through the different magnetic fields on their way and are therefore of limited use for astronomical purposes. After repeated interactions, the follow-up products of astrophysical accelerations can be detected with different techniques using satellites, earthbound telescopes of different designs, and underground detectors. These individual and globally distributed detectors also use techniques developed in particle physics. To arrive at a complete understanding of astrophysical sources, the information from all detectors must be combined. At the latest, after this combination, the different components of the cosmic rays must be separated and understood.[1]

Depending on which of the three components contributing to the measurements (source properties, radiation propagation between the source and Earth, or interactions of elementary particles) is considered unknown, insights can be gained in astroparticle astrophysics, cosmology, or particle physics. Because of the complexity of the required physical descriptions and though much is known with precision, the assumption of

1 For further discussion see [156].

completely controllable initial conditions cannot be made in this field. What is detected is the result of a concatenation of processes described by different theories and phenomenological approximations. In the end, these results may be precision measurements used, say, to close measurement gaps between experiments, test predicted signatures, or discover unexpected new phenomena that can only be explained by an extension of physical concepts. In the three decades of its existence, astroparticle physics has contributed to the revision of outdated theoretical ideas through many observations, despite or because of its complexity. And thus it might be well suited for the current discussion of epistemological boundary conditions of current physics using the science of big data.

In the historical development of physics one repeatedly finds dependencies between new physical insights and the new mathematical methods applied to them. The method of machine learning was developed in parallel to astroparticle physics. And it took some time until the early insight of computer science into the necessarily probabilistic nature of analytical results [280] was accepted in physics.

2.2.1 Experiments

From the field of astroparticle physics, two classes of experiments have given rise to and exemplified the development of machine learning methods in our work: neutrino telescopes and gamma telescopes, named after the messenger particle used to transmit information from the astrophysical source to Earth. Thus, both telescope types do not detect the astrophysical sources in the sky directly. They do not see the primary messenger particle but the Cherenkov light emitted from charged, high-energy secondary particles traversing a clear medium (water, ice). These secondary particles are produced in the form of an electromagnetic cascade in the atmosphere for primary gammas. In a charged current interaction of primary neutrinos, charged leptons (electrons, muon, taus) are produced. Also, a dominant part of the undesired background stems from the same source for both types of telescopes: cosmic rays in the form of atomic nuclei interact in the high atmosphere, also producing electromagnetic and hadronic cascades, including neutrinos. The background consists of the Cherenkov light induced by these particles and—depending on details of the data-taking—is many orders of magnitude stronger than the signal.

2.2.1.1 Neutrino Astronomy

Tim Ruhe

As stated above, neutrinos cannot be directly detected, and their detection thus relies on the detection of their leptonic partners, which are created in interactions with nuclei. Depending on the neutrino flavor, the interactions may happen either inside the detector or in the surrounding medium. The detection of neutrinos is nonetheless challenging due to their small interaction probability. This small interaction probability can, however, be accounted for by long exposure times, by utilizing large detection volumes or, ideally, by an optimized combination of both. As the detection of charged particles with velocities of approximately c (the speed of light in a vacuum) relies on the Cherenkov effect, and thus on the detection of Cherenkov photons, possible detection media also need to be as transparent as possible for blue light.

This requirement for transparent detection media with volumes of cubic kilometer-scale leads to basically two natural media that can be utilized for the construction of neutrino telescopes: water and ice. Both options have been realized in practice and come with their very own experimental challenges. While water has been used for the neutrino telescopes Baikal [72], Antares [39] and KM3NetT [37], ice was used for AMANDA and IceCube [21].

When utilizing ice, one of the main challenges is the remote location of the telescopes, which requires sophisticated logistics during construction and operation. Fortunately, the South Pole is a location that offers large volumes of ice as well as research infrastructure via the Amundsen Scott South Pole station. Additional experimental challenges arise from the fact that water bubbles frozen in the ice cause a significant amount of light scattering. The amount of air bubbles, however, decreases with depth, and this challenge can be mitigated—at least to a certain extent—by deploying sensors at depths below 1500 m. Sensor deployment is carried out via the use of a hot water drill, and the deployment of sensors in a single borehole takes approximately 48 hours. After deployment, the sensors are frozen into place and cannot be accessed for maintenance. This thus requires the use of extremely reliable sensors, both in a mechanical and an electronic sense. The sensors used in IceCube were found to operate in an extremely reliable fashion, and in fact, most sensor failures did occur during deployment.

With respect to deployment and maintenance, the use of water is somewhat less challenging, as sensors and sensor units can be deployed by ship and also recovered for maintenance if necessary. Research infrastructures required for the operation of the telescopes are generally located at the coast in the vicinity of the installed telescopes. Although the scattering of Cherenkov photons only plays a minor role in water, two other main experimental challenges arise. The first one is associated with the decay of ^{40}K, which, due to its natural abundance, gives rise to increased noise rates observed by the sensors. A second challenge arises from the presence of bioluminescence in

the ocean, which also contributes to the increasing background noise recorded by individual sensors.

Although neutrino telescopes in ice and water are equally important for neutrino astronomy as they complement each other with respect to the utilized techniques but also with respect to the observation of certain parts of the sky. The remainder of this section will briefly introduce the IceCube detector. This is due to the fact that all neutrino-related analyses in this book use data obtained with the IceCube neutrino telescope. A brief understanding of the detector itself is thus a prerequisite for a comprehension of the analyses.

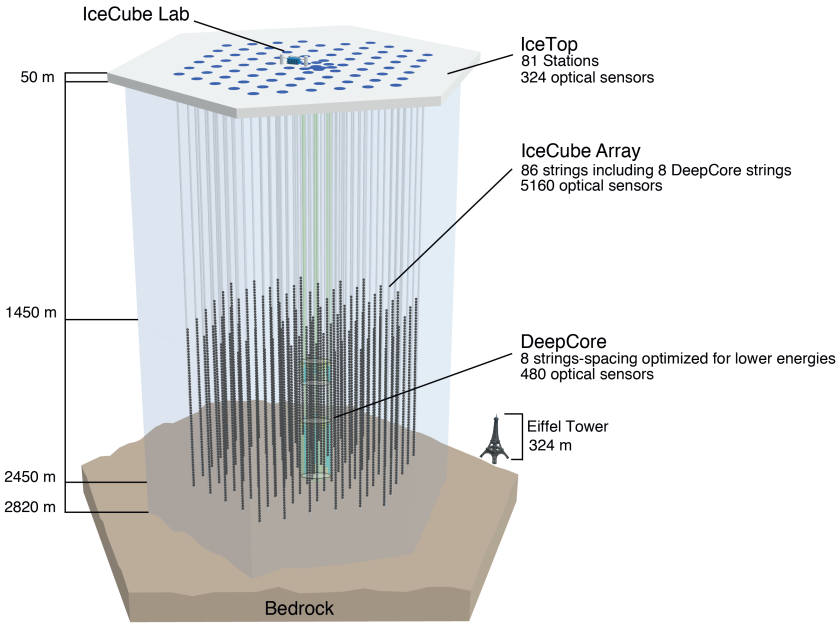


Fig. 2.2: Sketch of the IceCube neutrino observatory. Image courtesy of the IceCube collboration.

Figure 2.2 shows a schematic sketch of the IceCube neutrino observatory, located at the geographic South Pole. The detector consists of 86 strings arranged on a triangular grid with a string-to-string spacing of approximately 125 m. Strings in the center of the detector have a smaller spacing of roughly 70 m and form the DeepCore sub-detector. While the energy threshold of the entire in-ice array is $E_\nu \approx 100\,\text{GeV}$, DeepCore is optimized for neutrino energies as low as $E_\nu \approx 10\,\text{GeV}$.

IceCube strings are equipped with Digital Optical Modules (DOMs), which house a downward-facing 10" photomultiplier tube, as well as high voltage supply and readout electronics. The DOM-to-DOM distance on a string is 17 m, except for DeepCore strings, where the DOM spacing is only 7 m. In order to utilize the clear deep ice at the South

Pole, DOMs are deployed at a depth between 1450 m and 2450 m below the surface. The surface component IceTop, which serves as an air-shower array, completes the detector [21]. Although primarily designed for the detection of high-energy neutrinos from astrophysical sources, IceCube is a multi-purpose detector with a large scientific portfolio, as depicted in Figure 2.3.

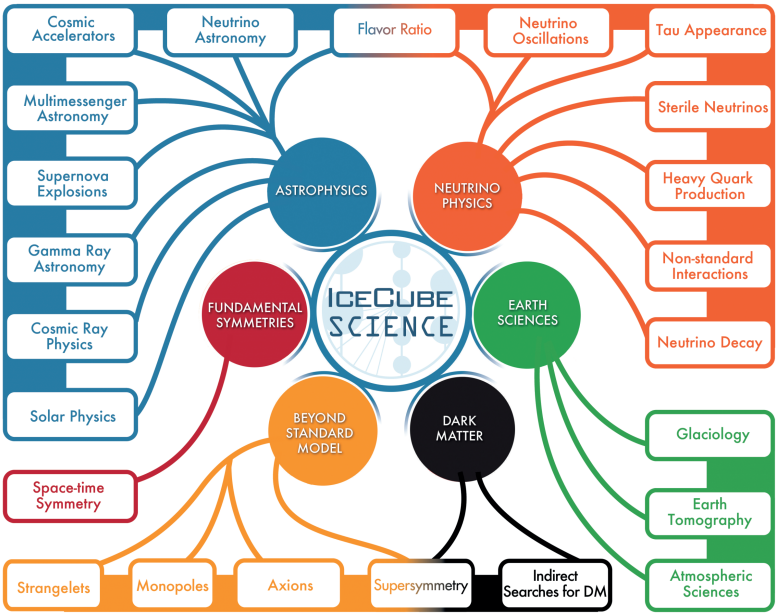


Fig. 2.3: The IceCube science portfolio. Graphic courtesy of the IceCube collaboration.

Although the instrumented volume of the detector is one cubic kilometer, the actual size of the detector can be larger or smaller, depending on the analyzed particle type and interaction. On the one hand, electron neutrinos and neutral current interactions might require the definition of a veto region. Often the outer string of DOM layers is used for such veto regions, which of course, lowers the volume available for the detection of particle interactions. Muons, on the other hand, can traverse large distances in ice and can consequently be produced far outside the instrumented volume. While this increases the volume available for particle interactions, the lack of knowledge on the interaction vertex at which the muon has been produced is a significant challenge for the reconstruction of neutrino energy spectra (see Section 10.6). Furthermore, the long-range of muons allows atmospheric muons produced in cosmic-ray interactions in the atmosphere to enter the detector in large numbers. Although they are interesting

in themselves, these atmospheric muons are the largest background component in basically all neutrino analyses.

Based on the event signature, atmospheric muons cannot be distinguished from neutrino-induced muons, but the simplest form of background rejection can be achieved via geometrical arguments: as the Earth is opaque for muons, upward-going neutrinos have to originate from neutrino-interactions. Discarding all downward-going muons is thus a promising background rejection strategy.

This strategy comes with a caveat, however: a small fraction of downward going muons will be falsely reconstructed as upward going. Due to the overwhelmingly large number of atmospheric muons relative to atmospheric muons, this *small fraction* still leads to a signal-to-background ratio of approximately 10^{-3}. This remaining background is significantly harder to discard but can be efficiently rejected by the use of machine learning-based analysis techniques. These approaches and, of course, the lessons learned from them, are the subject of this book.

A physics analysis is not complete once the background has been sufficiently rejected. In order to gain physics insight from the data sample, it needs to be analyzed, and physics parameters need to be extracted. A particularly challenging type of analysis in astroparticle physics is the reconstruction of energy spectra. The challenges arise from different sources. First, the underlying processes of particle production are governed by stochastical processes, which are mathematically covered by the Fredholm integral equation of the first kind. This, in turn, means that the sought-after spectrum can only be successfully recovered from distributions of measured observables via the use of deconvolution techniques. Further challenges arise from a secondary amount of smearing introduced by the detector itself. For neutrino telescopes, this additional smearing is mainly introduced by the fact that muons can be produced far outside the detector. While awaiting their detection and traveling towards the detector, these muons consequently lose a certain part of their energy. The energy of the incoming muon (neutrino induced or not) is thus somewhat smaller than the energy of the muon at its production vertex. Electronic noise and uncertainties in the description of impurities in the ice only add to this. It is then the task of the deconvolution algorithm to recover the neutrino energy spectrum from the inaccurately measured distribution of neutrino-induced muons. It becomes quite clear that it is a challenging task that requires a certain amount of algorithmic development. We have thus dedicated an entire chapter to deconvolution for an in-depth coverage of this topic (see Chapter 10).

The quality of spectral reconstruction does improve with the quality of the underlying energy reconstruction, but neutrino telescopes also suffer from their relatively poor angular resolution. IceCube, for example, has an angular resolution of $\approx 0.7^\circ$ for tracks and of $\approx 15^\circ$ for cascades. For comparison, the size of the Sun and the Moon in the sky is on of the order of 1°. A good angular resolution is, however, a prerequisite for the detection of astrophysical neutrino sources. Over the past decade, deep neural networks have been extremely successful in various task areas, e.g., image reconstruction.

Within this book, we also show how neural networks can be applied to reconstruction tasks in neutrino astronomy (see Section 9.2).

Although spread out over several thematic chapters, this book will guide the reader through an entire analysis in neutrino astronomy, ranging from data acquisition (Section 4.2.3), feature selection and track reconstruction (Section 7.2) to background rejection (Section 8.3.1) and finally to the deconvolution of neutrino energy spectra (Section 10.6).

2.2.1.2 Gamma-Ray Astronomy

Lena Linhoff
Alicia Fattorini

Just as neutrino astronomy, ground-based gamma astronomy exploits the Cherenkov effect to visualize particles that are not directly accessible. In this case, the Earth's atmosphere is used as a detector medium to observe gamma rays in an energy range from roughly 50 MeV to 30 TeV originating from distant galaxies or supernovae. By measuring and analyzing these particles, we gain knowledge about the gamma-ray sources in the universe, their composition, and the mechanisms that drive their behavior.

If a high-energetic particle, no matter if it is a gamma ray or a charged particle, hits Earth's atmosphere, it produces a cascade of charged particles. These so-called secondary particles inherit a fraction of the primary particle's energy and propagate through the air at high velocities, which might be faster than the speed of light in that medium. The particle cascade emits Cherenkov radiation, which is visible for a few nanoseconds as a large blue to the ultraviolet light cone. From the size, shape, and orientation of this Cherenkov cone, one can derive information about the energy, type, and origin of the primary particle.

Since Cherenkov light is extremely faint and the cone is visible only for a few nanoseconds, a big light collector and camera are needed that are capable of resolving these very short timescales. These requirements are technically realized as Imaging Air Cherenkov Telescopes (IACT). IACTs consist of large spherical mirrors with a diameter of several meters that reflect the Cherenkov light into a camera equipped with pixels that are capable of detecting detect single photons. In very high energy gamma-ray astronomy, the camera pixels typically consist of several hundred photomultiplier tubes (PMT) or silicon photomultipliers (SiPM).

When the camera is triggered by incoming light, a time series of roughly 100 nanoseconds is recorded for every pixel. Based on these time series, the number of photons hitting the pixel and their arrival time are calculated. Once these values are derived, the pixels containing the actual Cherenkov shower are selected from the whole camera image based on their photon charge and arrival time. The selected pixel groups are then parameterized, taking into account their shape, position in the camera, and

light distribution. Once the data is reduced this way, one ends up with a set of parameters belonging to a single event. In modern telescope setups, two or more Cherenkov telescopes are used to observe the same shower from slightly different positions at the same time. This way, the parameter sets of all telescopes are combined, and the position and energy reconstruction can be improved using the information of all observing telescopes. The resulting data set is then used to estimate the primary particle's type, energy, and origin via the data analysis techniques further explained in the following chapters. As soon as the energy and origin are reconstructed, the source's spectral energy distribution and flux variations as a function of time can be computed and studied further.

As already described for neutrino astronomy, gamma-ray astronomers also have to deal with a large background that has to be separated from the actual signal in one of the first steps. With a factor of roughly 10^5, the background is, in this case, dominated by hadronic particles that induce similar showers as gamma rays. In contrast to gamma ray-induced showers, these showers cannot be traced back to a specific source because charged particles are deflected by magnetic fields on their way through the universe. Therefore they do not carry any source-specific information and have to be separated from the gamma-ray events.

An IACT's observation capacity is limited mostly by sunlight and moonlight and its location on Earth, which restricts the sources that can be observed. Since the Cherenkov showers are very dim and the camera electronics are extremely sensitive, ground-based gamma-ray observations can take place only during nights with no or moderate moonlight and far away from man-made light sources in urban areas. Furthermore, the sky must be transparent for the Cherenkov light, which means that bad weather conditions and clouds heavily affect the observation quality. Therefore IACTs are mostly built at remote places, several hundred to thousands of meters above sea level, to ensure stable observation conditions.

Among other IACTs, the Major Atmospheric Gamma-Ray Imaging Cherenkov Telescopes (MAGIC [47, 48]) are the world's largest operating stereo system. MAGIC is located at the height of 2200 m at the Roque de los Muchachos, the main volcano of the Canary Island, La Palma, Spain. Each telescope has a mirror with a diameter of 17 m, focussing the light into the camera with 1039 PMTs. The telescopes are sensitive to gamma rays with energies between 30 GeV and 100 TeV, depending on observation and trigger settings. The construction of the first telescope was completed in 2004, and the second telescope started operations in 2009. With a series of upgrades in 2011 and 2012, MAGIC got a stereo trigger and readout system to perform measurements in stereo mode.

MAGIC has provided many scientific insights over the years. With the detection of very-high-energy gamma rays from the known blazar, TXS 0506+056 in 2017 [55], a breakthrough was made in the astrophysics community regarding the question of the origin of high-energy cosmic rays. The observed gamma rays were temporally and spatially correlated with a neutrino event reported by IceCube, which confirmed blazars

to be the most promising candidates for neutrino sources. With its lightweight structure based on carbon-fiber tubes, MAGIC is able to reposition with a speed of 7 °/s. After a gamma-ray burst (GRB) was detected with the Swift-BAT satellite in 2019, an alert was sent to the Gamma-ray Coordinates Network (GCN), and MAGIC managed to start observations only 50 seconds after the burst. With this effort, a gamma-ray burst was detected with an IACT for the first time [365, 383].

Next to MAGIC is a second telescope, the First G-APD Cherenkov Telescope (FACT) [53], which was originally designed for employing and testing novel camera technology. It is the first IACT with a camera made of SiPMs (1440 pixels) instead of PMTs. The FACT mirror has a diameter of 3.5 m. The telescope is sensitive to gamma rays with energies from several hundred GeV up to about 10 TeV.

Fig. 2.4: The LST-1 (back) and the FACT (front) at the Roque de los Muchachos Observatory in La Palma, Spain. Photo: Maximilian Linhoff/TU Dortmund University, 2021.

Meanwhile, the new generation of IACTs is under construction. The Cherenkov Telescope Array (CTA) [120] will consist of more than 100 telescopes in La Palma, Spain, and Paranal, Chile, providing a view of the northern and southern skies. Three types of telescopes are planned: Large-, Medium-, and Small-Sized Telescopes, called LSTs, MSTs, and SSTs. Each type has a specific size of the reflector surface and covers a certain energy range, resulting in a wide total energy range from 20 GeV to 300 TeV for the whole telescope array. CTA will be ten times more sensitive than the current generation of IACTs and will have a higher energy resolution. The construction of the first LST at La Palma was completed in 2018, and after the commissioning phase, three further

LSTs will be built at the site. With unprecedented sensitivity from IACT systems, CTA will open a new window on the universe and enable the discovery of new gamma-ray sources .

2.3 Particle Physics

Bernhard Spaan
Holger Stevens

The discovery of nuclear fission in 1939 is one of the primary reasons for establishing particle physics. Nowadays, the focus of interest is not on the atomic level but on subatomic elementary particles. These elementary particles are influenced by multiple forces. In the 1970's, a theory was developed that describes the particles and the corresponding forces, known as the standard model of particle physics. With the model, predictions of various physics processes are possible. The most straightforward prediction is the ratio of different decay modes from a so-called mother particle. To test such predictions, such decays must be observed. As the particles of interest are not stable and decay very fast, they needed to be produced in a detector. The particles are produced by accelerators, which collide high energetic particles. From the free energy in the collision, new particles can be formed. This transformation possibility is commonly known as Einstein's equation $E = mc^2$. It is important to note that the produced particles' type, energy, and flight direction are unknown a priori. Therefore one of the main challenges of particle physics detectors is to reconstruct and identify particles. Because they decay fast, this is done indirectly via the reconstruction and combination of their daughter particles. An essential tool is Monte Carlo simulations. They are used before and during the construction of a detector to simulate the expected performance. However, the simulation is also used for cross-checks and error estimations to analyze the actual data.

2.3.1 Experiments

The LHCb experiment at the Large Hadron Collider (LHC) at CERN has been designed to probe the standard model with high precision measurements. The main focus of the experiment is the analysis of particles consisting of at least one heavy quark, the b-quark or its antiparticle, the $\bar{b}$-quark—sometimes also called the beauty- or bottom quark. The so-called b-hadrons (B-mesons or b-Baryons) are relatively short-lived with a lifetime of about 1.5 ps, thus decaying after traveling on average approximately 1 cm to a few other particles that are long-lived and thus can be detected by the experiment. At the Large Hadron Collider, bunches of protons collide with other bunches of protons

at center-of-mass energies of up to 14 TeV. The collision of heavy ions with heavy ions or heavy ions with protons is also possible. LHCb can take data in all running scenarios. However, the experiment primarily collects proton data. In these collisions, pairs of b-quarks, namely a b and $\bar{b}$ (the anti-b-quark), are copiously produced. However, processes involving the production of lighter quarks or gluons are much more frequent by approximately a factor of 100. In addition, many other particles are produced in a single collision of 2 protons. For example, in a collision of 2 protons where a $\bar{b}$-pair is produced, only a few percent of all particles visible in the detector stem from the decay of the B-hadrons. b-hadrons have typically several hundred different decay modes where those of interest for the data analyses of LHCb occur only with a fraction of 10^{-3} or less, sometimes as low a $^{-}$ or even less than that.

Processes of particular interest involve neutral B-mesons ($B_s^0 : \bar{b}s$, $B_d^0 :: \bar{b}d$) decaying to final states that are sensitive to violation of the symmetry between particles and antiparticles. The analyses of these decays bear the potential to shed some more light on the origin of the CP-violation and thus on the mystery of the apparent asymmetry between matter and antimatter in the universe, which the standard model of particle physics cannot explain. The need for precision measurements of these types of decays is the driving factor in the design of the LHCb-experiment. Due to a very subtle mechanism, neutral B-mesons can oscillate between particle and antiparticle states during the short time between production and decay ($B\bar{B}$-oscillations). Since there are certain types of B-meson decays with common final states (CP-eigenstates) for B^0 and anti-B^0-mesons ($\bar{B}^0$), the interference between ($B\bar{B}$-oscillations and decays into such CP-eigenstate give rise to measurable CP-violation effects. The measurements of the magnitude of these CP-asymmetries are then used to probe the standard model of particle physics with precision. The measurement principle is to determine differences between decay rates of B- and $\bar{B}$-mesons at the time of production $t = 0$ as a function of time until they decay. This measurable asymmetry A_{CP} can be written as follows:

$$A_{CP} = \frac{\Gamma(B^0(\to f_{CP}) - \Gamma(\bar{B}^0(\to f_{CP}}{\Gamma(B^0(\to f_{CP}) + \Gamma(\bar{B}^0(\to f_{CP}}, \tag{2.1}$$

where f_{CP} denotes a CP-eigenstate such as $J/\psi K_S^0$ or D^+D^-. Obviously, it is impossible to tell from the decay final state whether the initially produced meson has been a B- or a $\bar{B}$ meson. Likewise, it is impossible to know whether the decaying meson was a B- or a $\bar{B}$ meson at the time of decay. However, it is essential to determine the "flavor" of the initially produced meson, i.e. B or $\bar{B}$. As pointed out, the design of the experiments reflects the need to perform these precision measurements. Compared with other large particle physics detectors at the LHC, such as ATLAS or CMS, LHCb has a different shape, as can be seen in Figure 2.5, where cross-sections of CMS and LHCb are shown. CMS is a so-called 4π-Detector, with a maximum coverage symmetrically around the interaction point where the protons collide. By contrast, the interaction point at LHCb is not the center of the detector but almost at one side. The detector topology resembles a forward spectrometer typically used in fixed-target experiments. It thus detects only particles

emitted from the interaction point within an angle between ≈ 15 and ≈ 300 mrad around the beam axis. Although only a tiny fraction of particles produced in a collision can be detected, it turns out that about $\frac{1}{4}$ of all $b\bar{b}$-quark pairs, and thus their decay products are directed into this region. This phenomenon is caused by the fact that protons are very complex objects, consisting of so-called partons, which consist of 3 valence quarks and a vast number of quark-antiquark pairs as well as gluons. When

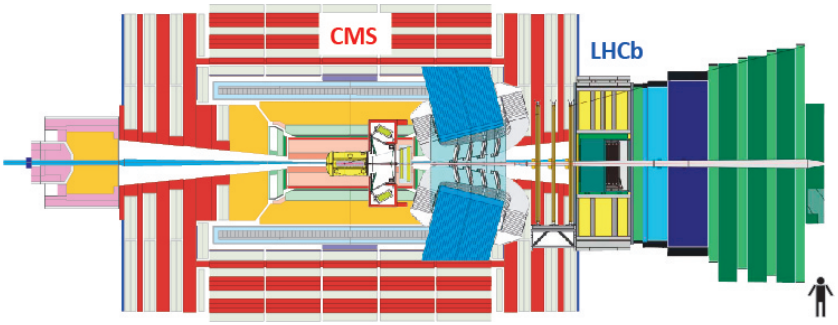


Fig. 2.5: Cross-section of the detectors: CMS (left) and LHCb (right). [244]

two protons collide, the primary interaction at the LHC is typically the interaction of 2 gluons, which may produce a $b\bar{b}$-quark pair, whereas the remainder of the proton is initially not involved in this process. The momentum of a proton is the sum of the momenta of all partons within the proton. Thus, the partons participating in the primary interaction may carry very different momenta, resulting in a boost of the $b\bar{b}$-quark pair along the beam axis. Another feature of the strong interactions is that free quarks cannot be observed. It turns out that the binding potential between a quark and an antiquark as a function of the distance between the quarks r can be described as

$$V_{q\bar{q}}(r) \propto -\frac{\alpha}{r} + \kappa r, \tag{2.2}$$

where α is the (running) coupling constant of the strong interaction and $\kappa \approx 1\,\mathrm{GeV/fm}$. Therefore, an energy of ≈ 1 GeV is needed to separate two quarks by just 10^{-15}m. Now consider a $b\bar{b}$ being created with high momenta in opposite directions in the rest frame of both colliding partons. An enormous field-energy will be created between the quarks separating from each other. According to Einstein's famous formula $E = mc^2$, it turns out that it is energetically preferred to create numerous $q\bar{q}$-pairs out of the field energy, which subsequently find partners to form hadrons such as mesons ($q\bar{q}$-pairs) and baryons (3 quarks) and antibaryons (3 antiquarks). These newly produced particles are then visible in the detector. This simplified picture should be used to understand what events at the LHC look like. Thus, several hundred particles can be produced in a high-energy proton-proton collision at the LHC within a range of mostly a few femtometer, thus appearing to stem from a single point, known as the primary vertex.

This brings us back to the $b\bar{b}$-pairs being produced with momenta, which points to the acceptance of the LHCb detector. They will form B-hadrons accompanied by numerous other particles. The lifetime of about $1.5 \cdot 10^{-12}$s makes these particles unique. Many other particles will decay almost immediately. Thus they will decay within the range mentioned above of few femtometer. B-hadrons will travel on average ≈ 1 cm before they decay. Only very few charged particles species have lifetimes that allow them to travel through the entire detector: electrons ($e^{\pm}$), muons ($\mu^{\pm}$), pions ($\pi\pm$), kaons ($K^{\pm}$), and protons (p, $\bar{p}$), and very light nuclei such as deuterons, which are produced only very rarely. As the decay of B-mesons such as B_d^0 or $B\bar{0}_s^0$ will follow an exponential distribution, the average time between production and decay is thus the lifetime. The actual decay time t can be determined by their flight distance ℓ:

$$\ell = \beta\gamma t, \tag{2.3}$$

where β is the velocity of the B-meson v divided by the speed of light c, and the Lorentz factor γ is:

$$\gamma = \frac{1}{\sqrt{1-\beta^2}}. \tag{2.4}$$

As B-mesons ultimately decay into long-lived particles, we can use the measured trajectories of the charged long-lived particles from B-meson decays to infer their location of decay known as the secondary vertex. Likewise, the primary vertex can be located similarly. Thus, we have derived a crucial design criterion for the LHCb detector, namely *to have the ability to reconstruct primary and secondary vertices with precision.* The next design criterion can be easily derived: B-mesons have to be identified, and their momenta have to be measured with precision to derive the decay time from their flight distance ℓ. The LHCb detector is shown in Figure 2.6. Several tracking detectors measure the trajectories of charged particles. At the interaction point, one can find the Vertex Locator (VELO), which is capable of measuring track hits very close to the beamline. The innermost sensors have a radial distance from the beam of just 5 millimeters, which is perilously close compared with other detectors. Using the vertex locator, the track parameters can be measured with high precision, enabling us to extrapolate them towards the beam to find primary and secondary decay vertices. As seen from the interaction point, there is one tracking detector (TT, tracker turicensis) just in front of the magnet, and another tacker just behind the magnet, consisting of the Inner Tracker (IT and the outer Tracker (OT). With the current upgrade of the experiments, all three trackers will be replaced by a new tracking detector. The IT/OT tracker will be replaced by the ScFi-Tracker (Scintillating Fibre), whereas Upstream Tracker (UT) has a very similar sensor technology to the TT. The Upgraded VELO will now contain a pixilated sensor, increasing the number of channels significantly. With tracking detectors in front of and after the magnet, the bending of the charged tracks in the magnetic field can be measured, which enables the determination of their momenta. However, several ingredients are still missing as we need to know which type of particles have been measured. This is highly relevant for the measurements on CP-violation or rare decays

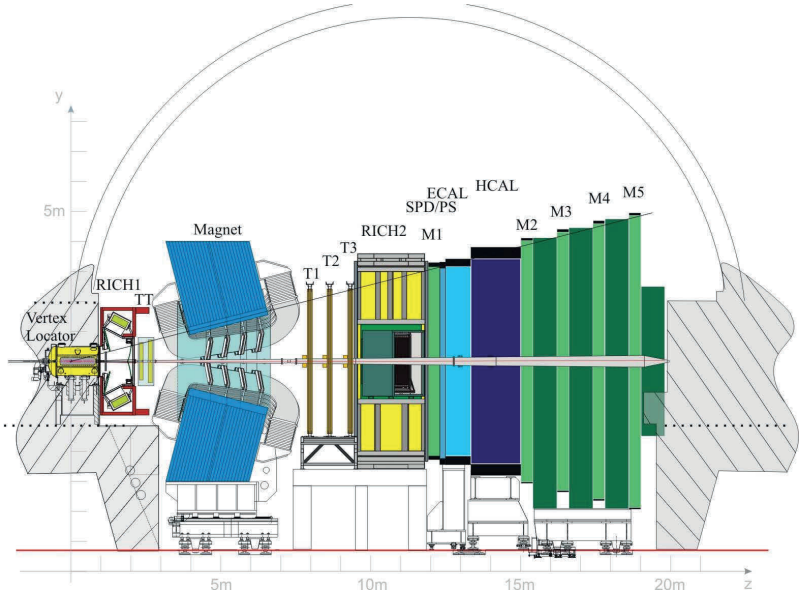


Fig. 2.6: The LHCb detector. [51]

of B-mesons. Therefore, the next design criterion is to have *the capability of identifying the different particles in a given event*.

Several sub-detectors are devoted to this task. The sub-detector furthest away from the interaction point is known as the Muon system, a dedicated muon identification system. This system identifies muons by particles traversing the detector because all other particle species will be absorbed by the calorimeters and the absorbers of the muon system with the highest probability. Typically, all detectors at particle colliders will have a muon system, as the presence of muon is an excellent indication of an exciting physics process in the event. The aforementioned calorimeters consist of an electromagnetic calorimeter followed by the hadron calorimeter. Contrary to all other charged particles with high momentum, the electron will be stopped in the calorimeter and thus deposit its total energy. The measurement of the deposited energy, together with the momentum, provides information about the presence of an electron. The other large detectors are equipped with calorimeters and muon systems.

Identifying pions, kaons, and protons is a bit more complex, as, unlike electrons and muons, they do not have specific properties that allow their identification. However, they have different masses, which results in particles with the same momentum having different velocities. Thus, a measurement of the velocity can be used to infer the mass of a particle when the momentum is known. Due to the relativistic nature of particles with high momenta, the velocity of the particles with different masses is very similar, making a direct measurement of the velocity difficult. However, using Ring-Imaging Cherenkov detectors, information on the velocity of high momentum particles can be

inferred. LHCb has two of those detector types, called RICH1 and RICH2. They make use of the fact that light is emitted when a charged particle traverses a medium with a velocity greater than the speed of light in this medium. The light is emitted under a velocity-dependent angle with respect to the trajectory. With sophisticated optics, this light is a projector to a sensor plane, where the light associated with the trajectory appears as a ring.

It might appear that the purpose of the calorimeters is mainly to identify electrons and absorb other particles. The electromagnetic calorimeter is responsible for measuring the energy of lighter particles, such as electrons and photons, while the hadron calorimeter samples the energy of hadrons, i.e., protons, neutrons, and other particles containing quarks. In principle, the detector components are now available to perform the measurements. However, the data from the sub-detectors must be transformed into physical quantities used in the analysis. For example, individual hits in an event need to be associated with a track, for which the momentum must be determined. This will be discussed later.

The measurement of A_{CP} shall now be described to illustrate the interplay of the various sub-detector components. The decay channel $B^0 \rightarrow J/\psi K_S^0$ is known as a golden mode, which has a large *CP*-asymmetry and very little theoretical uncertainties in the interpretation of the measurement. The J/ψ is a short-lived meson, which is reconstructed in the decay mode $J/\psi \rightarrow \mu^+\mu^-$. Thus, the excellent muon identification capabilities ensure the proper selection of the signal. The K_S^0-meson is reconstructed in the decay mode $K_S^0 \rightarrow \pi^+\pi^-$. The electrically K_S^0 has a lifetime of $\approx 10^{-10}$s which results in much larger decay flight distances of up to O(1 m) when compared with that of the B^0-meson. Therefore, the precision of determination of the secondary decay vertex of the B^0-meson relies mainly on the proper reconstruction of the J/ψ decay vertex. From the measured momenta $\vec{p}_{\mu\mu} = p_{\mu^+} + p_{\mu^-}$ and energies $E_{\mu\mu} = E_{\mu^+} + E_{\mu^-}$ of the muons, it is straight forward to determine the invariant mass $m_{\mu\mu}$ of the system:

$$m_{\mu\mu}c^2 = \sqrt{E_{\mu\mu}^2 - p_{\mu\mu}^2 c^2}.$$

The reconstructed invariant dimuon mass spectrum has a shape that resembles a Gaussian distribution on top of some background. A typical invariant mass spectrum indicates signal (J/ψ) and background. Figure 2.7 depicts the general shape of the spectrum and a certain background level. The presence of background is due to several sources. For example, combinatorial background arises if two muon candidates are combined where one or two of the muons candidate do not stem from the J/ψ decay. Their respective invariant math will cover a wide mass range, whereas in the signal case, the invariant mass shape follows mainly the mass resolution.

In addition, there is the possibility that a muon candidate is identified as a muon. This might happen with a relatively low probability should, say, a hadron not be absorbed. Although the individual survival probability is very low, the number of hadrons is significantly larger than the number of true muons in the events. The data analysis has a handle to change the background level. For example, the information on the

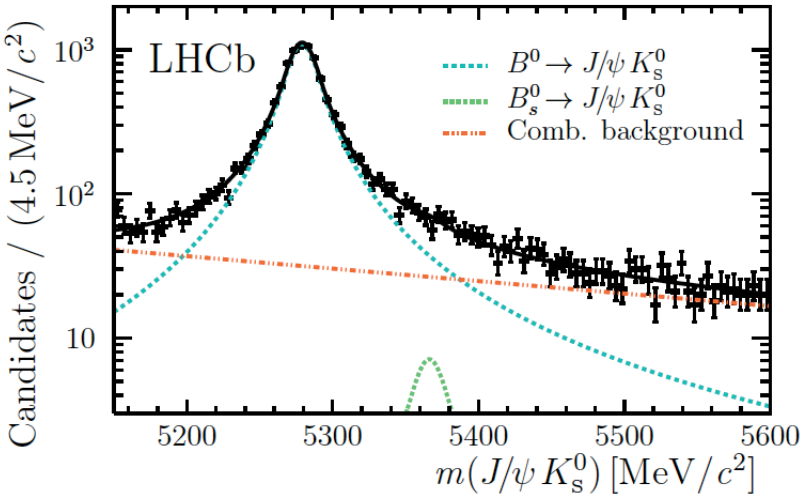


Fig. 2.7: Illustration of a typical invariant $\mu^+\mu^-$ mass distribution, depicting signal and background.

quality of the secondary vertex reconstruction can be used to reduce the probability of combining particles that do not originate from the same (secondary) vertex. Similarly, the K^0_S-mesons will be reconstructed from two oppositely charged pion candidates. There are some possible backgrounds, such as decays from other similarly long-lived particles decaying, for example, to a proton and a pion. Again, the misidentification of particles gives rise to background levels. As pointed out, particle identification using various sub-detectors comes with a certain misidentification probability. In order to minimize the misidentification probability, the information provided by all sub-detectors is combined to form a single variable called ProbNN. The combination turns out to be complex and is based on a neural network approach that gives variables their name.

In principle, reconstructing B^0-mesons candidates from J/ψ- and K^0_S-candidates follows the same procedure by determining the energy and momentum of the B^0 candidate by adding energies and momenta of the J/ψ- and K^0_S-candidates. However, for basically all analyses, it is of utmost importance to minimize the effects of backgrounds on the measurement of, e.g., CP-asymmetries. Accepting only J/ψ- and K^0_S-candidates with masses in the range defined by the resolution function is standard practice. Many other criteria can also be used to optimize the selection further. In most analyses, machine learning methods are used to achieve an optimal selection. Boosted decision trees or neural nets are typically used.

The selection of the signal channel to measure CP-asymmetries is just the first step. As pointed out, it is necessary to determine momentum and flight distance to derive the decay time of the particle. The momentum is known after the selection. To

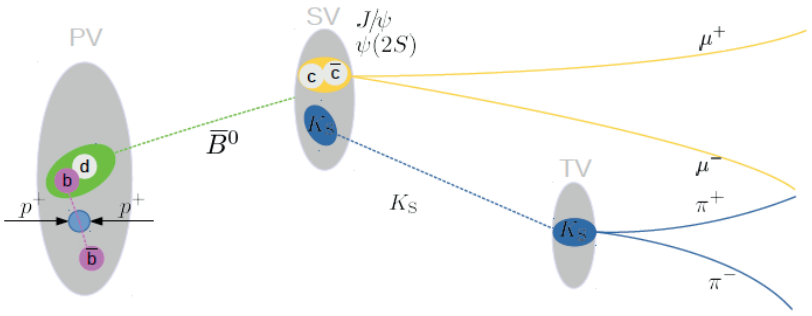


Fig. 2.8: Illustration of a typical B-decay topology. Charged tracks emerging from the primary vertex and other vertices are depicted as solid straight lines. The tracks are extrapolated using the track parameters determined by the tracking systems. The B^0-meson (dashed-dotted red line) travels on average ≈ 1 cm before it decays—in this case: $B^0 \rightarrow J/\psi K_S^0$. The J/ψ would decay almost instantaneously, where for the analysis, the decay into two leptons is relevant ($\mu^+\mu^-$, e^+e^-). The K_S^0-meson (dashed-dotted green line) has a considerably longer lifetime than B-mesons.

determine the flight distance, the charged tracks associated with the reconstructed B-candidate are used to reconstruct the secondary vertex of the B-meson. Therefore, the measured track parameters of the track are used to extrapolate the trajectory of the track towards the beamline. Due to the long lifetime of the K_S^0-mesons, the extrapolation of the track parameters of the pions will have considerably larger errors at the location of the B decay vertex. Thus, the precision in the measured vertex positions arises mainly from the muon track parameters. The location of the primary vertex can be readily determined from the numerous other tracks in the event. A sketch of the decay topology is shown in Figure 2.8.

The next ingredient of analysis is called tagging, the determination of the "flavor" of the B^0-meson under consideration at the production time. The determination of whether it was a B^0 or a $\bar{B}^0$ at $t = 0$ relies on the fact that B-quarks are produced in pairs. Since only a fraction of the produced b-quarks will end up in neutral B-mesons, information on the other B-hadron produced in the same reaction will provide an answer. There is more useable information in the event, say, from mesons produced close to the neutral B-meson under consideration. In order to retrieve the desired information, excellent particle identification capabilities are necessary—another strong constraint in the design criteria for the LHCb experiment. Despite the excellent detector, the probability of having a proper tag in a given event is very small.

What is known as tagging power ϵ_{eff} quantifies the effective statistical reduction of the data sample due to imperfections in the tagging. Thus, the statistical uncertainty σ on a time-dependent asymmetry measured on a sample of size N depends on ϵ_{eff} as

$\sigma \propto 1/\sqrt{N}$. The tagging power depends on the channel in which the CP-asymmetry is measured. Despite all efforts, its value is typically well below 10 %. Therefore, LHCb invests a continued large effort to improve the tagging power. A doubling of the tagging power corresponds to doubling the data sample with no further improvement, which amounts to several years of data taking. In order to obtain the highest possible tagging power, new tagging methods are being developed, based on deep neural networks.

Obviously, it is necessary to determine the efficiency of the detection and extraction of physical observables such as CP-asymmetries. In addition, the influence of backgrounds present in the data set needs to be known with high precision. For this type of task, extensive Monte Carlo simulations are typically used.

3 Key Concepts in Machine Learning and Data Analysis

Katharina Morik
Mirko Bunse

Abstract: Throughout this book, machine learning is employed in order to enhance the knowledge about the structure of our universe and the understanding of particle interactions through large data-producing physical experiments. This chapter gives an overview of the structure of machine learning as a scientific discipline. Since we cannot detail the methods and their foundations, we add pointers to relevant textbooks and survey papers. Our goal is to raise awareness about the theoretical basis of machine learning so that the software that machine learning generously offers is always used with the appropriate caution.

3.1 Overview of the Field of Machine Learning

Machine Learning (ML) is the field of Artificial Intelligence (AI) that builds or enhances a model of some phenomenon. We will give a short overview here, which shows the structure of this large field. The levels of machine learning and their related theories that we discuss are:

- The learning tasks that are most often given by objective functions to be optimized are also well-known in statistics, see Section 3.1.1.
- Algorithms and libraries follow paradigms of processing, see Section 3.1.2. They define pipelines of steps in the overall learning as described in Section 3.1.3. The particular step of feature selection is illustrated by the minimum redundancy and maximum relevance method in Section 3.1.4.
- Optimization methods, in particular Newton-Raphson and Stochastic Gradient Descent, are explained in Section 3.2.

The structural view is completed by pointing at the theoretical questions of machine learning (in Section 3.3). For the classes of learning methods, we selected to present tree models (including the ensemble of trees) in Section 3.4 and deep neural networks in Section 3.5 because these are the most common methods in the analysis of physical data.

https://doi.org/10.1515/9783110785968-003

3.1.1 Learning Tasks

Many approaches to machine learning can be described as finding a predictive function $h : \mathcal{X} \to \mathcal{Y}$ where the learning task is specified by $\mathcal{X}$, the domain of the input data, $\mathcal{Y}$, the domain of the output or label, and the risk measure that quantifies the quality of the prediction. Instantiating the general scheme of learning tasks gives us the definition of the most important classes of learning tasks.

Definition 1. ***Unsupervised Learning***
Given a set of observations $D_U = \{\vec{x}_i \in \mathcal{X} : 1 \leq i \leq N\}$
and a function $ext : \mathcal{C} \to 2^{\mathcal{X}}$ *returning all objects* $\vec{x} \in \mathcal{X}$ *that are mapped to* c,
find concepts $\{c_j \in \mathcal{C} : 1 \leq j \leq k\}$
such that each $\vec{x}_i \in \mathcal{X}$ *is mapped to a concept* $c_j \in \mathcal{C}$,
and such that some quality function is optimized through this mapping.

Definition 2. ***Supervised Learning***
Given a set of labeled observations $D_L = \{(\vec{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y} : 1 \leq i \leq N\}$,
find a mapping $h : X \to y$
such that a quality measure is optimized.

Where the scheme of learning tasks might look simple, if we elaborate on the three parts of it, we see how broad the space is that it covers.

- First, the **input** is characterized, ranging from formulas, database records, value series, sequences, images, and linguistic input, to graphs or even other models. Often, the input data are a sample of vectors D_U with $\mathcal{X}$ being a d-dimensional random variable. Each component is a feature or attribute of a certain domain. If all features are real numbers, $\vec{x} \in \mathcal{R}^d$. The vectors might be organized into some matrix A or into a series over an index $t_0, t_1, \ldots t_T$. For graphs, in addition to the vectors for nodes, there are also edges and possibly even edge features.
  Learning tasks with just these input data are called unsupervised learning. Cluster analysis and matrix factorization are popular methods of unsupervised learning. A clustering delivers a set of concepts $c_1, \ldots, c_k$ where each concept covers a set of data points.
  A method for selecting the appropriate set of features will be described in Section 3.1.4.
- Second, the **target** is specified. If a target value or label y_i is given for every $\vec{x_i}$, we call the task supervised learning. Classification assigns a class to an example, $h : \mathcal{X} \to y, y \in \{+1, -1\}$. Regression assigns a real number to an instance, i.e., $y \in \mathcal{R}$. For time series data, the task is either a classification, i.e. the time series is an instance of a class y, or a forecast, i.e. given a series $\vec{x}_{t_0}, \vec{x}_{t_1}, \ldots \vec{x}_{t_T}$ find $\vec{x}_{T+s}$ for a time span s.

It is possible to also have more complex output [370]. In speech recognition, for instance, a value series of audio input has to be mapped to a series of words, $h : \mathcal{X} \to \mathcal{Y}$.
Since the acquisition of the labels y_i for all given data D_U might be costly, semi-supervised learning uses a small set of observations $x_1, \ldots, x_m$ together with their labels $y_1, \ldots, y_m$ in addition to a huge set of unlabeled observations $x_{m+1}, \ldots, x_N$. It was introduced in the form of directing clustering through expert given advice [127] and then became generalized for text data [340] and graphs [266].
Methods that actively determine which labeled data have the best utility for learning are called active learning. They optimize the utility of labeling for learning regarding its cost. Self-supervised learning goes even one step further in that no human annotation is needed. Instead, the learner forms learning tasks on the basis of the given data. A component of example vectors or some part of a value series is used as the label that is to be predicted. The model is tested on examples with a mask hiding the label part. A more sophisticated approach is to distract the examples, train models on the changed and the true data, and learn how to distinguish between models [75].

– Third, a **loss function, quality measure** , or **risk function** is indicated, i.e., a condition that must be valid for the learned model. $R_D : \mathbb{R}^d \to \mathbb{R}$. It is evaluated on labeled or unlabeled data D_L or D_U. This opens the floor for optimization techniques [95, 293]. The broad field of optimization ranges from evolutionary algorithms [74] to (stochastic) gradient descent [92].
In addition to minimizing the loss or maximizing the likelihood, the goal is to minimize the model complexity and prevent it from overfitting the training data. In general, regularization adds a penalty term to the loss function. This can also be coined as multi-objective optimization [269]. Quality measures and optimization will be described in Section 3.2.

Let us illustrate the three parts of a learning task by regression to see their interaction more clearly. Learning a linear regression function is a simple but widespread task. Regression functions appear in three notations: as a sum, as a matrix product, and as a scalar product:

$$\hat{y} = h_{\vec{\beta}}(\vec{x}) = \sum_{i=1}^{d} \beta_i x_i = \vec{x}^T \vec{\beta} = \vec{x} \cdot \vec{\beta}$$

We see that the parameter vector $\vec{\beta}$ affects the predictions by weighting the features x_i in a weighted sum.

To learn a model from the data set D_L, we must decide on a risk measure. For linear regression, a popular choice is the mean squared error between the predictions and

the ground-truth values.

$$\mathrm{MSE}(\vec{\beta}; D_L) = \frac{1}{N}\sum_{i=1}^{N}(y_i - h_{\vec{\beta}}(\vec{x}_i))^2$$

Due to the simplicity of the linear regression model, it is possible to find a parameter vector $\vec{\beta}^\star = (X^\top X)^{-1}X^\top Y$ that is optimal in the sense of the MSE. In general, however, finding a solution is not that easy. More complex models and loss functions require numerical optimization techniques to select an optimal parameter vector.

The risk measure is typically defined as the average value over example-wise loss values. These loss values are specified through a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, which assigns a score to each individual prediction $\hat{y}$. Namely, the (empirical) risk R_D, which is to be minimized over $\vec{\beta}$, is defined as

$$R_D(\vec{\beta}) = \frac{1}{N}\sum_{i=1}^{N}\ell(h_{\vec{\beta}}(x_i), y_i) \tag{3.1}$$

For instance, the mean squared error for linear regression is based on the loss function $\ell_{\mathrm{MSE}}(\hat{y}, y) = (\hat{y} - y)^2$. A different choice is the zero-one loss with $\ell_{01}(\hat{y}, y) = 0$ if $\hat{y} = y$ and $\ell_{01}(\hat{y}, y) = 1$ otherwise. This latter choice results in R_D estimating the probability of misclassifications and is therefore a suitable measure for classification tasks. Other choices of ℓ include the hinge loss, the Huber loss, and many others [327].

Independent of which particular loss function is employed, we want the resulting risk to be as small as possible, so that the predictions $\hat{y}$ are as close to the true outcomes y as possible. However, we must keep in mind that the empirical risk R_D is only a substitute for a greater goal: a minimum *expected* risk R, which is valid for the entire data distribution.

$$R = \int_{\mathcal{X}\times\mathcal{Y}} \mathbb{P}(x, y) \cdot \ell(f_\beta(x), y)\,\mathrm{d}x\,\mathrm{d}y \tag{3.2}$$

This greater goal stems from the fact that we want to predict future data that is unlabeled. In other words: we intend to learn prediction functions that generalize from the observed data D_L to any data set from the same distribution. The difference between R and R_D becomes apparent when we split the data into training and validation sets: if the model class $\mathcal{H}$ is powerful enough to memorize the training set, so that $R_D = 0$, we typically observe a validation error that is greater than zero.

One lesson from observing the difference between R and R_D is that we typically want to prevent our models from a mere memorization of the training set D. To this end, we can employ regularization techniques that impose additional constraints on the model structure. The objective of a regularized optimization task, with a regularization function $r : \mathbb{R}^d \to \mathbb{R}$ and a regularization strength $\lambda \in \mathbb{R}$, is:

$$\vec{\beta}^\star = \arg\min_{\vec{\beta}} R_D(\vec{\beta}) + \lambda r(\vec{\beta}) \tag{3.3}$$

The function r typically does not depend on the training data D but on structural properties such as sparseness, that are desired in an anticipated solution. We substantiate this discussion with two exemplary regularizers, namely the L1 and the L2 norms. The L2 norm, also known as the Euclidean norm, is a popular penalty term that imposes small parameter values. It has the additional benefit of facilitating optimization by being strongly convex. In neural networks, the L2-norm regularization is also called "weight decay".

$$r_{\mathrm{L2}}(\vec{\beta}) = \| \vec{\beta} \|_2 = \sqrt{\sum_{i=1}^{d} \beta_i^2} = \sqrt{\vec{\beta}^T \vec{\beta}} = \sqrt{\vec{\beta} \cdot \vec{\beta}}$$

Picking up the linear regression model, the L2-norm regularization leads to the popular ridge regression technique:

$$\min_{\vec{\beta}} \sum_{i=1}^{N} (y_i - \sum_{j=1}^{d} x_{ij} \beta_j)^2 + \lambda \sum_{j=1}^{d} \beta_j^2$$

The L1 norm, also known as the Manhattan distance, promotes a different structure for the solution. Instead of imposing small parameter values, it promotes some parameters to be close to zero while leaving other parameters intact. In a linear model, L1-norm regularization can thereby perform a selection of features simultaneous to learning the prediction model.

$$r_{\mathrm{L1}}(\vec{\beta}) = \| \vec{\beta} \|_1 = \sum_{i=1}^{d} \beta_i = \vec{\beta}^T \vec{1}$$

Regarding linear regression models, L1-norm regularization leads to the popular LASSO regression technique. For linear dynamical systems (LDS), L1 regularization has been enhanced by a reparameterization approach based on an estimation of time-variant dynamics [310].

Bayesian statistics realizes regularization through a given prior probability distribution that decreases the complexity of the model. The SVM even selects the model that minimizes the error and the model complexity at the same time [378, 381], as measured by the VC dimension (see Section 3.3.1) [379]. Viewing machine learning as data compression under minimum description length [325, 326] has further led to frequent set mining in very large data volumes [384].

3.1.2 Processing Paradigms of Machine Learning

Data may be given as a large data set or may come in as a stream of data. The very large data sets that do not fit into the memory of a single machine require distributed processing. Computing on large distributed compute clusters has led to the programming paradigm of **map & reduce**. A small example illustrates the idea. In the map step, a function is applied to each element of a list, e.g., $map(+1)[1, 2, 3]$ delivers $[2, 3, 4]$. In

the reduce step, a function is applied to the overall list, e.g., $reduce(+)$ then delivers $[2 + 3 + 4]$ [135].

Computing on streams does not allow the algorithm to look at a data point more than once, in the extreme case. The potentially infinite stream moves through the algorithm, which processes a data point and lets it go. A data structure to handle these streams, however, needs to be finite like any other data structure. Therefore, it is often necessary in practice to employ algorithms that only approximate the true properties of the data stream, for instance, in approximate counting [259]. Novel algorithms for learning from streaming data are proposed in Chapter 3 in Volume 1. An open-source library of learning algorithms for massive data streams is MOA [83].

Astrophysical data are *big data*. The notion of big data indicates a very large *volume* of data arriving with high *velocity* and in a large *variety* of types, which need to be handled. Open-source software such as Apache Hadoop "allows for the distributed processing of large data sets across clusters of computers using simple programming models".[1] The Hadoop Distributed File System (HDFS) offers a high throughput of data via parallel and distributed data management. More generally, computer science has developed architectures for big data. The lambda architecture combines the storage of large data sets in a batch layer with a real-time component, the speed layer [261]. The kappa architecture stores the historical data in a way that processes them in a data streaming manner [161]. Chapter 6 presents ways of storing large astroparticle data in more detail. For a framework for learning from data streams see [87]. It has been exploited for astroparticle analyses [88]. A study of big data management and processing for the Cherenkov telescope FACT shows the overall pipeline of streaming data analysis [278].

3.1.3 Machine Learning Pipelines

In general, a data science pipeline starts with the most demanding step, the mapping from a scientific question to a learning task. Most often, the scientific question is split into several ones, each with its own learning task. Chapter 1 describes the interplay of theory development and data analysis in terms of epistemology.

Sampling from all observations the data to be analyzed follows the scientific concern by, say, structuring the observations according to certain concepts. In neutrino detection, for instance, we might be interested in muon, electron, or tau neutrinos and thus form separate learning processes for these concepts. This is also true for simulated data. If one class is dominating, we might change the given distribution. Section 5.2.2 shows how active class selection samples disproportionately from a skewed distribution in order to achieve a sound classification.

1 http://hadoop.apache.org.

The given data format often needs to be transformed for learning. For instance, standard representations of time-stamped data can be transformed in several ways to allow for a successful training of specific learning tasks [277]. The overall process of learning needs to be documented with all meta-parameters. Machine learning frameworks such as RapidMiner[2] offer reproducible, adaptable, and easy-to-understand processes. A complete learning pipeline for the successful learning of neutrino recognition in the IceCube experiment has been developed with RapidMiner [331].

What is most important for successful machine learning is the features of the observed items. Selecting the features that ease learning is a first step. There are three types of feature selection. First, *filter approaches* like the t-test [168] or SAM-statistics [372] compute a scoring function on features, disregarding feature interplay. Second, *wrapper approaches* [230] train a learner with possible feature sets and evaluate each feature set by the accuracy of the embedding learning. Each feature set evaluation demands a cross-validated training of the used learning algorithm. Third, some learning algorithms provide the user with an *implicit feature ranking* that can easily be exploited for feature selection. Such embedded approaches use the weight vector of a linear SVM [381], or the frequency of feature use of a Random Forest (RF) [97]. They are aware of feature interplay and are faster than wrappers but depend on the learning algorithm used. In Section 3.1.4, we describe a general and efficient method of feature selection.

Processes of extracting features from the raw data are often tailored to particular scientific questions. Chapters 7 and Section 8.4.1 present this for particular astrophysical data.

Unsupervised learning may deliver features for a succeeding supervised learning. Unsupervised learning delivers pseudo-labels that are used to optimize an overall cost function as in supervised learning. This approach has also been put forward for neural networks [114].

Such a two-step procedure has been taken to the extreme of one-shot learning [249]. *One-shot learning* adapts given knowledge, which may have been learned before, to a small set of examples. A Bayesian approach works especially well for image data since there are regions of interest that could be interpreted as shapes and characteristics of appearance, e.g., a set of textures. First, pictures of a set of categories are presented to train a prior probability density function. The probabilities of shape and appearance for categories are the given knowledge that forms the space of features. Then, only one labeled example is needed to correctly classify all the instances that are close to it in the learned feature space. This is one type of one-shot learning.

Extracting the features that allow classifier learning with high accuracy can be automatized as a process, where the outer loop of learning and its evaluation with respect to a quality measure embeds an inner loop that creates novel features on the basis of given data. Autonomous feature extraction relies on a well-structured space

2 https://rapidminer.com.

of possible features. Shape and texture have been structured image classification. For value series, a structure over base transformations and shapes of curves has been developed and used for autonomous feature extraction via an evolutionary algorithm [270]. There, features are represented as trees of methods. The evolutionary algorithm creates new features by adding methods to a tree or combining trees. It optimizes over populations of method trees, and in that way performs feature selection.

Some learning algorithms provide the user with an implicit feature ranking that can easily be exploited for feature selection. Such embedded approaches use the weight vector of a linear Support Vector Machine (SVM) [381] or the frequency of feature use of a Random Forest (RF) [97]. As has already been underlined by Tom Mitchell, the hidden layers in a neural network discover useful intermediate representations [274]. This corresponds to autonomous feature extraction with evolutionary algorithms in that the nested loop is the neural network's backpropagation, the outer learning is the last layer, and the hidden layers create possible representations. The supervised training of feed-forward networks is considered representation learning because they extract patterns from the data in the hidden layers and optimize them such that the learned weights strengthen the relevant local patterns [183].

Finally, we might want to optimize the learned model itself. In contrast to the terminology in physics, the term quantization in machine learning refers to compressing a model by turning real-valued numbers into binary or integer ones. Binarized Neural Networks (BNN) quantize the weights now to binary values [207]. A quantization scheme for Tensor Flow maps real numbers to a binary representation and uses integer-only arithmetic during inference and floating-point arithmetic during training, again for saving resources [214]. An approach that fully trains Markov Random Fields (MRFs) using only integer values has been developed in order to save energy [308]. It allows learning and inference on ultra-low power devices that use integer-only arithmetic.

3.1.4 Minimum Redundancy Maximum Relevance (MRMR)

The features span the space in which concepts can be learned. Too many features bring with them the curse of high dimensionality. Having many features might slow down learning. Hence, the goal of feature selection is to find a subset of features that allows predicting the target concept well and has minimal redundancy. Correlation-based feature selection (CFS) [190] iteratively adds the feature which has the best ratio between predictive relevance of the feature and its correlation with the already selected features. Both, predictiveness and correlation, are measured by the entropy-based symmetrical uncertainty:

$$SU(f_i, f_j) = \frac{2 \cdot IG(f_i|f_j)}{H(f_i) + H(f_j)} \tag{3.4}$$

where the information gain IG of feature f_i with regard to feature f_j is divided by the sum of the entropies H of f_i and f_j. Ding and Peng [144] generalized CFS with the capability for handling numerical variables calling it *Minimum Redundancy Maximum Relevance FS* (MRMR). For numerical features, the F-test is used. It reflects the ratio of the variance between classes and the average variance inside these classes. For a continuous feature X and a nominal class variable $y \in Y$ with C classes, both from a data set with n examples, it is defined as

$$F(X, Y) = \frac{(n - C) \sum_c n_c (\bar{X}_c - \bar{X})^2}{(C - 1) \sum_c (n_c - 1)\sigma_c^2} \tag{3.5}$$

with per-class-variance σ_c^2 and n_c the number of examples in class $c \in \{1, .., C\}$. The redundancy of a numerical feature set is measured by the absolute value of Pearson's correlation coefficient

$$R(X, Y) = \frac{Cov(X, Y)}{\sqrt{Var(X) \cdot Var(Y)}} \tag{3.6}$$

or its estimate

$$r(X, Y) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}. \tag{3.7}$$

In order to stabilize the feature selection method, its variance is reduced by bagging, i.e., the feature selection algorithm is executed in parallel on different subsamples of the data, thus delivering an ensemble of feature sets [96]. A fast implementation of an ensemble of MRMR feature selection, which is well-suited for high-dimensional data, has been created [344]. Features that are selected earlier and by many sets in the ensemble are combined to become the one selected feature set.

3.2 Optimization

By now, we have expressed the task of learning a prediction model from data as the task of minimizing an empirical risk function. In the following, we exemplify numerical optimization by the two most popular methods, stochastic gradient descent and the Newton-Raphson method. Both methods search for an optimal parameter vector $\vec{\beta}^\star$ by updating an intermediate estimate $\vec{\beta}^{(k)}$ over multiple iterations, as described in Algorithm 3.1. The two algorithms differ in their choice of search directions $\vec{p}^{\,(k)}$, see line 3 in this algorithm.

Algorithm 3.1: A canonical algorithm for numerical optimization.

Input: an initial guess $\vec{\beta}^{(0)} \in \mathbb{R}^d$ and a risk function $R_D : \mathbb{R}^d \to \mathbb{R}$
Output: a minimizer $\vec{\beta} \in \mathbb{R}^d$ of R_D

1 $k \leftarrow 1$ **while** *a stopping criterion is not met* **do**
2 Choose a search direction $\vec{p}^{(k)} \in \mathbb{R}^d$, using $\vec{\beta}^{(k-1)}$ and R_D
3 Choose a step size $\alpha^{(k)} \in \mathbb{R}$, using $\vec{p}^{(k)}$, $\vec{\beta}^{(k-1)}$ and R_D
4 $\vec{\beta}^{(k)} \leftarrow \vec{\beta}^{(k-1)} + \alpha^{(k)} \cdot \vec{p}^{(k)}$
5 $k \leftarrow k + 1$
6 **end**
7 **return** $\vec{\beta}^{(k-1)}$

3.2.1 Stochastic Gradient Descent

Machine learning tasks are typically characterized by large numbers of training examples that need to be handled with limited resources. This profile needs to be addressed by the numerical optimization method, which takes out the learning according to Equation 3.3. Stochastic Gradient Descent (SGD) is a family of optimization methods that is particularly suitable if N, the number of training examples, is large.
We consider the popular mini-batch variant of SGD, which chooses each search direction $\vec{p}^{(k)}$ based on a mini-batch of $b \ll N$ examples with random indices. Since each of these mini-batches $I_k = \{1 \le i_j \le N : 1 \le j \le b\}$ is a random draw from the complete data set D, it follows that the estimates $\{\vec{\beta}^{(k)}\}$ of the SGD algorithm are a stochastic (Markov) process. This behavior is in contrast to complete-batch algorithms, which access all instances of D in each iteration and therefore produce a deterministic sequence of estimates. Another characteristic of SGD is that only the gradient ∇R_D of the objective function is accessed, but not any higher-order derivative of R_D. Therefore, the per-iteration cost of the mini-batch SGD is very cheap, i.e., it depends linearly on b but does not depend on N. The search direction of this variant is given by the following equation, where all gradients are taken with respect to the parameter vector $\vec{\beta}$:

$$\vec{p}^{\,(k)}_{\text{SGD}} = -\nabla R_{I_k}(\vec{\beta}^{(k)}) = -\frac{1}{N}\sum_{j=1}^{b} \nabla \ell(f_{\vec{\beta}^{(k)}}(x_{i_j}), y_{i_j}) \tag{3.8}$$

Accessing only b examples in each iteration causes a considerable amount of noise in the parameter updates. This issue can be handled by appropriate choices for the step sizes, e.g. by decaying values $\alpha^{(k)} < \alpha^{(k-1)}$. However, since small step sizes can slow down the learning process, there are several other techniques that address gradient noise more directly.

For instance, momentum-based SGD variants alter the parameter update rule of the canonical optimization algorithm presented above. Namely, these variants alter line 5 of Algorithm 3.1 to the following assignment, introducing an additional step size

$\gamma^{(k)} \in \mathbb{R}$:

$$\vec{\beta}^{(k)}_{\text{momentum}} \leftarrow \vec{\beta}^{(k-1)} + \alpha^{(k)} \cdot \vec{p}^{(k)} + \gamma^{(k)} \cdot (\vec{\beta}^{(k-1)} - \vec{\beta}^{(k-2)}) \tag{3.9}$$

The additional step size $\gamma^{(k)}$ weights the contribution of another direction, which is the difference between previous updates. Thereby, the parameter updates are "stabilized" in the sense that they maintain earlier search directions to the degree that is configured through the scalar sequence of step sizes $\{\gamma^{(k)}\}$. Momentum-based techniques are widely applied in practice. For instance, the famous optimization method Adam [227], popular for learning deep neural networks, is based on two orders of momentum to achieve even more stability.

The suitability of SGD for large N stems from the fact that the number of iterations and the per-iteration cost do not depend on N [93]. This property is in contrast to *full-batch* algorithms, where the per-iteration cost is indeed proportional to N. Since the per-iteration cost of SGD is comparably cheap, it scales well with large data sets.

Beyond its desirable scaling behavior, SGD has two more advantages over full-batch algorithms: first, it optimizes not only the empirical risk R_D but also the expected risk R directly. In this regard, SGD is better aligned with the actual goal of learning a generalized prediction model. Second, SGD can optimize functions that are non-convex if these functions are well-behaved in less strict terms [93].

3.2.2 Newton-Raphson Optimization

The Newton-Raphson method [293] differs from SGD in two central aspects. First, it uses not only gradient information but also the second derivative of the objective function R_D. Thereby, it is able to assess the curvature of the search space in exchange for a cost that is quadratic in the number of parameters. Second, is the batch variant of the Newton-Raphson algorithm, i.e., each iteration computes the full derivatives of R_D using all examples in D. Due to these differences, the per-iteration cost of Newton-Raphson is considerably higher than the per-iteration cost of SGD. However, since full derivatives capture considerably more information than an SGD update, fewer iterations are typically needed. Newton-Raphson can therefore outperform SGD in terms of the computational resources that it needs to find an accurate solution if the number of training examples N and the number of parameters d are both sufficiently small. Optimization tasks with a small N and a small d are not very typical in the empirical risk minimization framework of Equation 3.1, but such tasks do have their relevance in other aspects of data analysis such as in the deconvolution problem put forward in Chapter 10.

Complying with the canonical optimization algorithm from Algorithm 3.1, the Newton-Raphson method takes out multiple iterations, starting from some initial guess $\vec{\beta}^{(0)}$. To find a search direction $\vec{p}^{(k)}$, the method evaluates a local second-order Taylor

approximation $\widehat{R}_D{}^{(k)}$ of the actual objective function R_D:

$$\widehat{R}_D^{(k)}(\vec{\beta}) \;=\; \frac{1}{2}\,\vec{\beta}^{\top} H\,\vec{\beta} \;-\; \vec{\beta}^{\top}(H\,\vec{\beta}^{(k)} - \vec{h}), \tag{3.10}$$

where $\vec{h} = \nabla R_D\left(\vec{\beta}^{(k)}\right)$ is the full gradient and $H = \nabla^2 R_D\left(\vec{\beta}^{(k)}\right)$ is the Hessian of the actual objective R_D at the latest estimate $\vec{\beta}^{(k)}$. The minimum of this local approximation can be computed analytically. It defines the search direction of the Newton-Raphson method:

$$\vec{p}\,{}^{(k)}_{\text{Newton-Raphson}} \;=\; -H^{-1}\vec{h} \tag{3.11}$$

SGD and Newton-Raphson are examples of a broad research field that covers numerical optimization algorithms. The interested reader can find additional information on batch algorithms in reference [293]. Stochastic algorithms are covered in the survey by Bottou and colleagues [93].

3.3 Theories of Machine Learning

Machine learning research aims at answering the following questions:

- Which guarantees can be given regarding the error? (Error bounds)
- Which model class is best suited for the problem? (Model selection)
- How many examples are needed? (Sample complexity)
- How does a model scale in terms of runtime, memory, and energy, if the number of examples and the number of dimensions increase? (Resource bounds)

At the most abstract level, machine learning works on *formulas* like that of regularized error minimization of a regression function from Equations 3.1 and 3.3 above. In terms of these functions, the learning tasks are specified. Proving the error bounds of learning models is the shared subject of machine learning and statistics. As has already been stated, also the field of optimization also plays a role. Almost every paper at a machine learning conference such as ICML or ECML PKDD presents or at least substantially bases its results on a proof of error bounds. For an introduction, we recommend the books on statistical and probabilistic approaches of machine learning [192, 284].[3] Another school of theory at this abstract level is the computational learning theory which investigates learnability on the basis of the representations of the feature and the hypothesis space. In Section 3.3.1 the main idea is shown with some hints for further reading.

Given a physical problem and experimental data, no class of learning theories is a priori well suited. As the *no free lunch theorem* points out, every selection of a learning method comes along with its requirements on the one hand and its theoretical guarantees on the other [388, 389].

3 For a series of video lectures on foundations of machine learning, taught by Ulrike von Luxburg, see: https://www.youtube.com/playlist?list=PL05umP7R6ij2XCvrRzLokX6EoHWaGA2cC.

Within a selected general class, there are criteria that help to select a particular method. The most important characteristic of an algorithm is the distinction between batch and online, or distributed and centralized processing. For the development of an application, efficient *algorithms* are to be studied. Worst-case complexity and proven tight accuracy bounds are known for diverse algorithms of the same model class. The competitions of implementing frequent set mining, a task of finding all frequent co-occurrences of database items [163], are well known. 12 varying levels of implementations were reported with their different resource consumption, e.g., memory usage, runtime, and the compression of the resulting model.

Today, we go even further to the level of *implementation on particular hardware*. In earlier days, learning algorithms were tailored from CPU to GPU, as discussed in [309]. Hardware is now particularly designed for the low-latency and high-throughput computational demands of machine learning. The non-von Neumann architectures In-Memory Computing (IMC)/ Processing-In-Memory (PIM) are being extensively researched [78]. Currently, FPGAs are frequently used machine learning accelerators. Different implementations of an algorithm for a von-Neumann architecture and for FPGAs have been carefully explored [107]. Also, the optimization for a fast execution on a particular architecture received interest [231], especially computing on multicore architectures (see Section 6.4 in Volume 1). For convolutional neural networks in particular, their abstract description in terms of the Open Neural Network Exchange Format (ONNX) allows them to synthesize an implementation on the high-level interface of FPGAs and optimize it [176].

Recently, the connection of machine learning algorithms with hardware architectures has become even closer in that a given learning algorithm is not only adjusted to a given architecture; the learning algorithm itself takes care of possible hardware failures within its training procedure. This is particularly relevant for hardware that trades in accuracy for energy savings. In-memory computation, for instance, saves energy but may deliver wrong results. A novel max-margin optimization binarized neural networks succeeded in optimizing the bit-error tolerance [110].

The level of model classes with its ties to statistical learning theory is important, but machine learning investigates more levels of abstraction in collaboration with other fields of computer science. Algorithms for a model class for distributed or streaming learning are based on theoretical computer science work as in [129, 130]. The level of implementations on a specific hardware links machine learning with computational architectures [260]. Particular hardware has even been designed especially for machine learning [217]. Machine learning investigates and contributes to all the levels: from the model class over the algorithms to implementations and even computer architectures. Learning is orthogonal to the hierarchical levels of computer science.

Tab. 3.1: An example of the concept class that consists of conjunctions of Boolean literals.

c_1:	*rainy*	*AND*	*not play golf;*
c_2:	*rainy*	*AND*	*play golf;*
c_3:	*not rainy*	*AND*	*not play golf;*
c_4:	*not rainy*	*AND*	*play golf;*
c_5:	*sunny*	*AND*	*not play golf;*
c_6:	*sunny*	*AND*	*play golf;*
c_7:	*not sunny*	*AND*	*not play golf;*
c_8:	*not sunny*	*AND*	*play golf;*
c_9:	*rainy*	*AND*	*not sunny;*
c_{10}:	*rainy*	*AND*	*sunny;*
c_{11}:	*not rainy*	*AND*	*not sunny;*
c_{12}:	*not rainy*	*AND*	*sunny;*

3.3.1 Computational Learning Theory

In addition to the statistical theory of machine learning, a distribution-independent theory known as the Probably Approximately Correct (PAC) learning, offers bounds of learnability [375]. It is based on the idea of hypothesis spaces. Which hypotheses can be expressed in a particular formal system of representation? An easy concept class is the conjunctions of Boolean literals. Literals include *rainy*, *sunny*, and *play golf*. The hypothesis space $\mathcal{C}$ would then be the set of concepts c_1 to c_{12}.

Learning identifies within the hypothesis space those concepts that are consistent with the examples, i.e., there is no logical contradiction between the true hypothesis and all the examples. In other words, a concept is a hypothesis that is determined by the set of instances that it covers. If there is an example in contradiction with the hypothesis, it is counted as an error. For all domains, we shall have different literals that define the concepts. This is not what the learning theory cares about. The theory is to state whether the class of all concepts that can be expressed in this representation is learnable from examples. Learnability is defined with respect to the number of computing steps that are necessary to identify the target concept in the worst case.

Definition 3. *A concept class $\mathcal{C}$ is* PAC-learnable *by a learning algorithm $\mathcal{A}$ using hypothesis space $\mathcal{H}$, if for all $c \in \mathcal{C}$, distributions $\mathcal{D}$ over the instance space X, ϵ such that $0 < \epsilon < 1/2$, δ such that $0 < \delta < 1/2$, $\mathcal{A}$ will with probability at least $(1 - \delta)$ output a hypothesis $h \in \mathcal{H}$ such that the $error_{\mathcal{D}}(h) \leq \epsilon$, in time that is polynomial in $1/\epsilon$, $1/\delta$ and in the size of the instance space and in the size of the concepts space complexity.*

PAC learning polynomially bounds the number of computation steps needed in the worst case to learn a classifier for a class of concepts. A proof of PAC learnability usually first shows that each target concept in $\mathcal{C}$ can be learned from a number of examples, which is polynomially bounded by ϵ and δ. This lower polynomial bound of the number

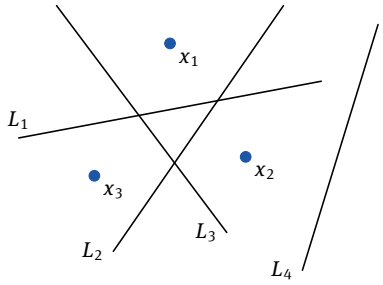


Fig. 3.1: There exists a set of three points $\{x_1, x_2, x_3\}$ for which any binary labeling can be separated by a single, straight line $L \in \{L_1, L_2, L_3, L_4\}$. Therefore, the VC dimension of straight lines is at least three.

of examples is called the sample complexity of the learning algorithm [191, 349]. It shows that the time for processing one example is also polynomially bounded. The overall idea is described more elaborately in [273] and many proofs can be found in [286].

PAC learning often measures the model complexity in terms of the *Vapnik Chervonenkis (VC)* dimension [379]. A PAC learning algorithm is then required to learn a concept class in time that is polynomial in the VC dimension of the hypothesis space.

Definition 4. *A set $\mathcal{H}$ of hypotheses shatters a set D of examples if each subset of $\mathcal{H}$ could be separated by a $h \in \mathcal{H}$. The* VC-dimension *of a set of hypotheses $\mathcal{H}$ equals the maximum number d of examples in D that could be shattered by $\mathcal{H}$.*

We illustrate this by 2-dimensional data and the hypotheses in the form of separating planes as shown in Figure 3.1. *One* set of three points could be shattered by straight lines, but there is *no set* of four points that could be shattered by straight lines. Hence, for the straight line hypothesis space, the VC-dimension is 3.

For the proof of the VC-dimension d the following has to be shown:

- There exists one set D with d points that could be shattered by $\mathcal{H}$.
  $VCdim(\mathcal{H}) \geq d$
- There does not exist a set D' with $d + 1$ points that could be shattered by $\mathcal{H}$.
  $VCdim(\mathcal{H}) \leq d$

The VC-dimension denotes the model complexity and hence allows us to select the least complex model that still learns the target concept. It also gives a hint to the confidence we can have in a learning result. A large VC-dimension indicates a large confidence. There is even a learning method that exploits the VC-dimension by regularizing its internal optimization such that it guarantees a unique and optimal learning result. This method is the support vector machine [380].

The VC-dimension allows us to write a lower bound on the sample complexity. The theorem has been proven for all learners and concept classes [149]. It is given below according to Mitchell [274].

Theorem 1. Lower bound on sample complexity. *Consider any concept class C such that $VCdim(C) \geq 2$, any learner $\mathcal{A}$, and any $0 < \epsilon < \frac{1}{8}$, and $0 < \delta < \frac{1}{100}$. Then, there exists a distribution $\mathcal{D}$ and target concept in $\mathcal{C}$ such that if $\mathcal{A}$ observes* fewer *examples than*

$$max \left[\frac{1}{\epsilon} log(1/\delta), \quad \frac{VCdim(\mathcal{C})-1}{32\epsilon} \right] \tag{3.12}$$

then $\mathcal{A}$ outputs a hypothesis h having error $error_{\mathcal{D}}(h) > \epsilon$ with probability at least δ.

The theorem states that with fewer examples, no learner can PAC-learn every target concept in $\mathcal{C}$. This very general bound for all learners is turned into more specific bounds, when the VCdim is known for the model class.

PAC learning has investigated the learning of neural networks from its very beginning [256] and succeeded in showing that neural networks are capable of approximating arbitrary functions [224]. As is sketched in Section 3.5, the theoretical analysis that gives us tight bounds and a deep understanding of deep learning is still an active research area.

3.4 Tree Models

Decision trees are one of the most successful learning methods: often applied, based on probabilistic theory, and easy to implement. They recursively divide the feature space into smaller and smaller hyper-rectangles until there are only members of one class in the hyper-rectangle, which makes it a leaf node, or until some other stopping criterion is met [319]. While usually used for classification in supervised learning, decision trees may also model regression tasks. Training a tree, where each node covers a set of examples, is performed by selecting the best feature for splitting the current node, creating sub-nodes for each feature value, and passing the examples to those fitting their feature value until a node covers only examples of the same class or has a clear majority of one class. The learned model classifies a previously unseen example by passing it according to its feature values to its leaf.

Selecting the splitting feature X_j often follows the information gain criterium. The probability p_+ that an example belongs to class + is the entropy I:

$$I(p_+, p_-) = (-p_+ \log p_+) + (-p_- \log p_-)$$

A feature X_j with k values divides a set of examples $\mathbf{X}$ into k subsets $\mathbf{X}_1, \ldots, \mathbf{X}_k$. For real-valued features, the numerical values are partitioned into some intervals, so that these intervals are handled along with the discrete features. Binary decision trees always

split into a left and a right branch. For numerical values, a threshold t is used $x_j \leq t$. The best feature X_j or the best threshold t is selected based on a quality criterion such asinformation gain:

$$Information(X_j, \mathbf{X}) := -\sum_{i=1}^{k} \frac{|\mathbf{X}_i|}{|\mathbf{X}|} I(p_+, p_-)$$

The information gain is the difference between the entropy of the examples with and without the segmentation by X_j. It is calculated with respect to the particular set of examples at each sub-tree.

The higher the information gain, the closer a feature is to the root. In this sense, the place in the tree seems to indicate feature importance. However, the order of selected features is *not* a precise feature weighting algorithm. In particular, correlated features along a path in the tree do not share the importance but add it. Hence, they violate the condition that the sum of weights of alternative features must be constant independently of the actual number of alternative features used. This condition is important since it guarantees that a set of alternative features is not more important than a single feature [271].

Decision tree learners are not robust with respect to the order in which examples arrive. Hence, in their original form, they are not applicable to data streams. For a data stream setting, the VeryFastDecisionTree (VFDT) approach has been introduced [146]. They are efficient in that the runtime is proportional to the number of features; moreover, the examples of the stream are not stored but processed just once. In the beginning, a sample of observations at a node is kept. For these, the split criterion is evaluated. The difference between the top features is required to be larger than some ϵ. Reading in additional examples and evaluating the split criterion is continued until the Hoeffding bound is reached. From then on, only aggregated statistics are stored at a node. The Hoeffding bound guarantees that a sum or difference of independent random variables most likely will not deviate more than a constant from the expectation value. Here, it states that with probability $1 - \delta$, after seeing n examples, the feature which receives an evaluation (e.g., information gain) that is ϵ larger than that of the next best feature is indeed the best split criterion also for future examples. ϵ can be calculated:

$$\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}}$$

with R being the range of feature values, e.g., the log of the number of classes.

3.4.1 Ensemble Methods

The idea of training many decision trees in parallel and letting each tree vote for a class, hence delivering the majority vote as result, is known famously as a *Random Forest* (RF) [97]. The ensemble of many simple models achieves a higher robustness

than a complex model that covers the same observations. Since ensembles are trained in parallel, their runtime is also an advantage. The algorithm for training the random forest is shown in Algorithm 3.2. The application of the random forest delivers the

Algorithm 3.2: The Random Forest algorithm.

Input: number of decision trees l and n examples and desired number of features k
Output: $h_1 \ldots h_l$ mapping X to Y

1 **forall** *the* $i = 1 \ldots l$ *decision trees in the forest* **do**
2 Sample n examples without removal
3 choose $k \ll K$ features from the K given ones randomly,
4 split the set of examples at the node according to the best split, given by, e.g., the information gain
5 **if** *the resulting nodes have enough examples of the same class* **then**
6 stop and output h_i
7 **else**
8 go to line 3
9 **end**
10 **end**

classification

$$h(x) = sign\left(\frac{1}{l}\sum_{i=1}^{l} h_i(x)\right)$$

The procedure is a kind of a bootstrap aggregation—bagging for short. The statistical bootstrap method draws samples and fits models to each of them. The output of bagging is the averaged output of all the models or the majority vote. In any case, the variance of the prediction over the data is decreased. Among the many successful applications of Random Forests are those in astrophysics, namely the IceCube experiment [16, 18, 331] and several Imaging Air Cherenkov Telescopes [48, 258, 294].

Another ensemble method is boosting, first introduced as AdaBoost [169]. Like the Random Forest, it also consists of several learners and decreases the variance of the learning result. In contrast to bagging, boosting is an incremental method that directs the training to areas of the example space that are difficult to learn. It is based on PAC learning, where it has been shown that it is sufficient to have hypotheses that are only slightly better than pure random on the training data because these can be boosted to become arbitrarily correct. The algorithm of AdaBoost is given in Algorithm 3.3. On the one hand, the weak classifiers are weighted by α. On the other, the weights of the examples w_m are updated such that the ones misclassified by $h_m(x)$ receive more impact by $exp(\alpha_m)$. Hence, the next weak classifier $h_{m+1}(x)$ will fit the previously not

Algorithm 3.3: The AdaBoost algorithm.

Input: N examples and the desired number of decision trees M

Output: a prediction rule $h : \mathcal{X} \to \mathcal{Y}$

initialize example weights: $w_{1,i} \leftarrow 1/N \;\; \forall \; i = 1, 2, \ldots N$

forall $m = 1, 2, \ldots M$ **do**

  train a weak classifier $h_m(x)$ using the weighted examples

  compute the error $error_m$ on all data

  $\alpha_m \leftarrow log((1 - error_m)/error_m)$

  $w_{m+1,i} \leftarrow w_{m,i}\, exp(-\alpha_m y_i h_m(x_i)) \;\; \forall \; i = 1, 2, \ldots N$

end

return $h(x) = sign\left(\sum_{m=1}^{M} \alpha_m h_m(x)\right)$

well-covered areas of the example space. There are several applications of boosted decision trees in physics. See Chapter 8.

3.4.2 Implementations and Hardware Considerations

Decision trees and their ensembles have a statistical or PAC learning description as described above. This abstract level is complemented by algorithmic challenges and their implementation on diverse hardware. The highest performance has been achieved by *gradient-boosted trees*. The implementation *XGBoost* is a truly scalable algorithm using external memory and processing the training in a parallel manner on GPUs, exploiting gradient descent optimization (cf. Section 3.2.1) [119].

Other approaches optimize the application or evaluation of the learned model. This is particularly important if the training may be run on a large computing center, but the learned model is to be applied in the wild of a physical experiment like the Cherenkov Telescope Array (CTA) or IceCube. The execution of the model is then restricted in runtime as well as in memory. Traversing a large ensemble of decision trees has been developed for fast inference [242]. A probabilistic view of executing decision trees has been developed in order to optimize the data layout and enhance the cache usage [107]. The improvement is based on the systematic use of tree usage statistics. Two different implementations of decision trees are investigated, namely, an optimized if-then-else tree and an optimized path layout. These implementations exploit computing architectures better and should be considered for real-time applications under resource constraints. Section 7.3.3 in Volume 1) provides more information and explains how to generate optimized code for specific computing architectures.

Machine learning models are often compressed or quantized to use fewer resources. For decision trees, the pruning of sub-trees was put forward from the beginning on [319]. Also, the selection of ensemble members has received attention, e.g. [371], but

this is an active research field. Before deploying a tree model, it is worth determining the appropriate pruning method.

3.5 Neural Networks

Neural networks have attracted attention from their inception. Here, we present the structure of the field and indicate selected literature for further studies.

Neural networks are—most often acyclic—directed graphs with the nodes being organized in layers. The input layer consists of nodes x_i for the input features. The output layer gives the result of the network y or has several so-called *heads* each being a target in a multinomial neural network. Layers between input and output are called *hidden layers*. A neural network with hidden layers is called a Deep Neural Network (DNN) . Figure 3.2 shows a neuron with the weights of the incoming nodes that are summed up and the non-linear activation function, here, the Rectified Linear Unit, which computes $ReLU(z) = max\{0, z\}$. Other activation functions are sigmoid, tanh, and softmax. Just one such neuron is also called a perceptron. Having layers of several such perceptrons is then also called a Multilayer Perceptron (MLP). The connections between the nodes in the following layers may be such that every node of the preceding layer is connected with every node of the next layer, yielding fully connected layers. There are also types of networks with fewer connections between layers. Since the inference feeds the computed values from the incoming signals to the next hidden layer(s) until the output layer is reached, it is a feedforward neural network. If computation also includes feedback connections, the neural network is called a recurrent network.

The term "neural network" originates from the idea that the combination of a weighted sum and a non-linear activation function is reminiscent of a biological neuron cell. In fact, the neuron cells of the human brain become activated when they receive a sufficiently strong signal from their input neurons. However, a neuron cell is much more complex than the simple mathematical function that we call "neuron" here. Also, a brain is much more complex than a simple concatenation of neuron layers. Therefore, a neural network should not be misunderstood as an appropriate model of the biological brain.

Training a neural network is performed by optimizing the weights between the nodes of succeeding layers. These parameters are to be determined such that for all possible inputs, the respective true output value is returned. An output that does not fit the true label starts a *backpropagation* of the error from the last to the first layer. We consider the DNN a chain of functions. Hence, we can propagate the gradients of the loss function using the chain rule. The derivative of $f(g(x))$ is

$$\frac{\partial f(g(x))}{\partial x} = \frac{f}{\partial g}\frac{\partial g}{\partial x}$$

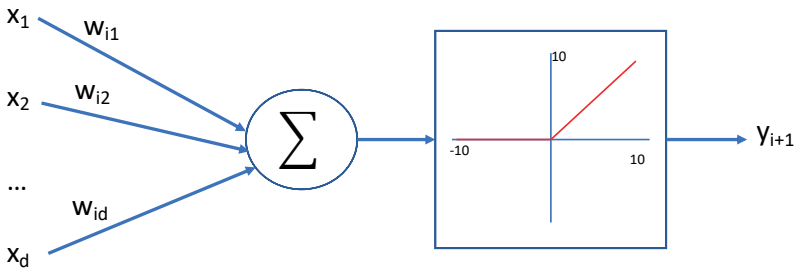


Fig. 3.2: General picture of a neuron with incoming units x_i, the activation function with the summation of the weighted incoming values, and the non-linear ReLU function together computing the resulting unit y_{i+1}.

The backpropagation algorithm stores the derivatives of f with respect to all variables $x = f(w)$, $y = f(x)$, $z = f(y)$. For x, it is

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}$$

For w it is

$$\frac{\partial z}{\partial w} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}\frac{\partial x}{\partial w}$$

As is seen, for the three multipliers, only one needs to be calculated in addition, the others are already stored. This makes backpropagation very efficient.

Once the derivatives are calculated, the optimization uses them. Iteratively, the parameter matrix is updated until a sufficiently good matrix is found. Most often, stochastic gradient descent is used as the optimization method (cf. Section 3.2.1).

For the abstract description of neural network training with backpropagation of errors and weight updates to optimize the network, see [192]. However, this is only a small part of what DNNs are about and what makes them successful. It is the design and development of algorithms and their implementation on diverse computing platforms that produces their excellent performance. For a comprehensive description, it is very much recommended to read the book *Deep Learning* by Goodfellow, Bengio, and Courville [183].

3.5.1 Architectures of DNNs

A DNN consists of a series of layers, each of which can carry out different computations. When we speak of an *architecture*, we mean a particular series of layers, leaving aside the optimization algorithm or the data with which the architecture is trained.

The most fundamental type of layer is called the *dense layer*; it multiplies a weight matrix W to the full vector of incoming values $\vec{h}$ and possibly adds a bias term $b \in \mathbb{R}$. The weighted sum is then fed into a non-linear activation function u, such that the output of the dense layer is $u(W^\top h + b)$, as shown in Figure 3.2. The weight matrix is then optimized during the learning process, and the non-linear function enables the model to learn non-linear dependencies within the data. However, the dense weight matrix W introduces many parameters into the model, the training of which can be ineffective in terms of resource consumption.

Convolutional layers circumvent this problem by sharing parameters among pairs of inputs and outputs. Namely, a kernel of parameters that is much smaller than the input dimension, is moved over the input. The name of this type of layers stems from the *convolution* of two functions, where an input function $x(a)$ and a kernel function $w(a)$ of measurement a deliver a feature map

$$s(t) = \int x(a)w(a-t)\,\mathrm{d}a. \tag{3.13}$$

The discrete convolution over integer values t is

$$s(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a). \tag{3.14}$$

This re-use of parameters leads to fewer connections between the nodes of the preceding and succeeding layers. Moreover, the computation is straightforward to parallelize. Another important property of convolutional layers is their translational invariance. For object recognition in an image, it is not important where exactly the object is. Similarly, patterns in audio input may occur at different widths and heights but should be recognized anyhow. A DNN architecture that uses convolutional layers is called a *convolutional neural network* (CNN).

A more drastic decrease in the number of connections is achieved through the *dropout* layer. This layer randomly ignores, at the succeeding layer, a certain percentage of incoming values with their weights. Dropout can be seen as a regularization of the model.

For image processing, the *pooling* layer is widely used. It summarizes the values in a rectangular neighborhood of nodes by, say, the maximum value or by the average.

The *batch normalization* layer takes the incoming values $z_1, \ldots, z_m$ and calculates mean and variance.

$$\mu = \frac{1}{m}\sum_i = 1^m z_i$$

$$\sigma = \sqrt{\frac{1}{m}\sum_i^m = 1(z_i - \mu)^2}$$

How to combine these building blocks or define new ones is a matter of active research. Several network architectures have been proposed. Starting from AlexNet [233] with

its tremendous success on ImageNet, which offers images labeled into 1000 object categories, such as keyboard, mouse, pencil, and many animals [143], the Oxford Visual Geometry Group proposed a CNN of 19 layers named VGG-19 that is well suited for the recognition of objects in images [350]. *Residual networks* structure a CNN into repeating blocks of convolutional and batch normalization, where each block adds a shortcut from the first to the last layer of the block. This allows enhancing the depth of the network without difficulties in optimization [193].

EfficientNets scales, at the same time, a learned base model in width, depth, and (image) resolution [363]. Experiments show good results for scaling two very different network architectures, namely the deep ResNet and the energy-efficient MobileNet. Bello et al. disentangle architecture and resolution again [81]. They scale the depth and the width depending on the amount of overfitting and apply a slower resolution increase than in EfficientNets. The race for better speed and accuracy in training neural networks thus continue.

3.5.2 Robustness of DNNs

Neural networks are fragile in many respects. On the one hand, little changes in the architecture may have large changes in predictive performance as a consequence. On the other hand, little changes in the data may change the classification of the neural network tremendously. Famous examples are the images that are imperceptibly changed but output a completely different class [362]. Optimizing the least changes leading to the worst accuracy has been called *adversarial attack*. Many types of perturbations have been studied, not only on the data but also on the physical objects that are perceived. In [154], some stickers were attached to a stop sign, and it was classified as "speed limit 45". Defense mechanisms were invented in order to make neural networks more robust. Recently, guarantees have been developed that prove the robustness against particular perturbations [132]. Finally, robustness might also refer to changes in the data distributions [124].

3.5.3 Deep Learning Theory

Neural networks with a single hidden layer of n nodes have been proven to be universal approximators of any measurable function, assuming activation functions $\Psi : \mathcal{R} \to [0, 1]$ that have countably many discontinuities [203]. The number of hidden nodes is not known in general but depends on the function to be approximated. In the worst case, one hidden node is needed for each configuration of the input, i.e., the number of hidden nodes is exponential if we have just one hidden layer. Applying the VCdim (see Section 3.3.1), bounds of the sample complexity of DNN were proven that depend both on the depth and the width of the network. Deeper models require less hidden

nodes in each layer. This is the advantage of deeper networks [364]. The complexity of a DNN can be shown without referring to the width of the layers, but usually depends on its depth. For $||\vec{x}|| \leq B$, d layers, and the matrices $W_1, \ldots, W_d$, the complexity has been shown with regard to the Frobenius norm at most $M_F(j)$ of W_j and m as seen in examples from [181]:

$$\mathcal{O}\left(\frac{B\sqrt{d}\prod_{j=1}^{d} M_{F(j)}}{\sqrt{m}}\right) \tag{3.15}$$

The authors then convert depth-dependent bounds into depth-independent bounds, which are based on some control over the norm of the parameter matrices.

Learning algorithms with a large capacity are capable of fitting randomly labelled data. With increasing depth, neural networks have an increasing capacity of representation, i.e. they approximate increasingly complex functions. Hence, they are capable of fitting pure noise. Does that contradict the statement that DNNs generalize? In general, machine learning should not overfit the training data, or even memorize them, but perform well on previously unseen data! Inspecting the optimization, it was found that true patterns are learned before the overfitting occurs and that dropout and other regularizations prevent the optimization from memorizing [58]. Other approaches to explaining the generalization of DNNs are the coherent gradients of similar examples pointing in the same direction [118] and the stiffness of a network, which measures the impact of the change in one example's small parameters on in the gradient step in the loss of another example. If the network's weights based on one example help to better classify another example, it generalizes well [167].

A way to characterize DNNs is by analogy with Bayesian models. It has been shown that inference with dropout of a DNN approximates a Gaussian process and, hence, that the Bayesian model uncertainty explains the dropout of DNNs [174]. Bayesian models estimate the uncertainty of a DNN model. However, they do not scale well. Hence, a Spectral-normalized Neural Gaussian Process (SNGP) has been proposed that replaces the output layer with a Gaussian process and includes weight normalization in the training [251].

3.5.4 Explanations

Explainable AI has been studied for black-box algorithms of all kinds [187]. Selecting borderline examples or showing the feature importance according to the learned model explains the learned model without looking into the training process. If, however, the explanation refers only to a surrogate model and not to the learned and deployed model itself, they actually do not explain the learning result [177]. The model agnostic methods are complemented by methods for verifying and explaining DNNs. In particular, for scientific data, where we want to model true processes, the modeling procedure itself must be trustworthy.

A survey of verification and explaining DNNs has been framed theoretically [206]. There exists a large variety of methods that explain DNNs [337]. A most prominent method is the layer-wise relevance propagation and its visualization. It shows which areas of an image have been used for classification by a trained model [275]. For image data, this helps users to understand the learned model. Layer-wise weight change helps to understand the training of DNNs [41]. The training process of DNNs can be inspected at each layer—each intermediate representation—in order to find the most influential examples and determine the classes that attributed most to the classification [305].

3.5.5 Hardware Considerations

AI accelerating hardware is a booming market. Many companies offer specialized chips, which are built into phones, tablets, and many other devices. We address several of these developments in Volume 1 of this book series.

Regarding DNNs in particular, a central aspect of accelerating hardware is energy consumption. One example of a processor dedicated to DNNs has been designed by Google for the TensorFlow software, especially for its matrix multiplications. The Tensor Processing Unit (TPU) delivers an order of magnitude better-optimized performance per Watt for machine learning.[4]

Even when using TPUs, DNNs still demand large amounts of energy. In particular, for edge computing and for the Internet of Things, using a Field-Programmable Gate Array (FPGA) as a platform is advantageous. Automatic synthesis of FPGA programs for CNNs has been developed, see, e.g., [176].

In general, Binarized Neural Networks (BNN), i.e., those that calculate with binarized weights and activation function values, consume less energy and require less memory. The use of approximate memory provides DNN training with even lower energy consumption. However, the saving comes at the price of the memory sometimes flipping a bit. A novel approach to BNN training using approximate memory includes robustness against bit errors in the optimization of BNN learning, thus combining the advantages of approximate memory and BNN directly [110].

Examples of how neural networks leverage research in astroparticle and particle physics are described in Chapter 9. Resource consumption of learning methods is a central theme of all chapters.

4 https://cloud.google.com/blog/products/ai-machine-learning/google-supercharges-machine-learning-tasks-with-custom-chip/.

4 Data Acquisition and Data Structure

4.1 Introduction

Wolfgang Rhode
Bernhard Spaan
Hans Dembinski

The common approach to scientific questions in astroparticle and particle physics also results in great technical proximity in the required low-level techniques for data acquisition and the resulting data structures to be analyzed. In this chapter, the general approaches to data acquisition are discussed (Section 4.3.3)

In particle physics experiments at colliders, the event rate is typically very high since rare processes are sought after in most analyses. The event rate is driven by the desire to discover rare processes such as the production of a Higgs boson or a rare decay that is suppressed in the standard model and may be enhanced by physics beyond the standard model. Typically, particle bunches are made to collide at certain interactions point. This means that the event rate is not constant; it repeatedly peaks and falls back to zero. The acceleration with radio-frequency electrical fields enforces this structure. The average rate of collisions of individual beam particles $\dot{N}$ is defined by the number of particles per bunch, the particle density profile in the bunch overlap, the frequency of the bunch crossing, and the total cross-section σ for the collision of two beam particles. All bunch parameters can be combined in a single variable called luminosity $\mathcal{L}$, which yields the simple relation

$$\dot{N} = \mathcal{L}\sigma. \tag{4.1}$$

The cross-section σ is basically an effective area, typically given in units of a barn, $1\,\mathrm{b} = 10^{-24}\,\mathrm{cm}^2$. The total cross-section is a measure of the probability that any collision occurs, but one is often more interested in collisions with a particular final state. Then, $\dot{N}$ refers to the rate of this process, and σ is the corresponding (partial) cross-section.

A data set taken over a period of time t is characterized by the integrated luminosity, $L = \int_t \mathcal{L}\,\mathrm{dt}$, measured in units of inverse barn, 1/b. At the LHC, bunches are 7.5 m apart and move with the speed of light, and thus collide every 25 ns or with a rate of 40 MHz. The peak luminosity and total cross-sections at the LHC are so large that several proton-proton interactions occur per bunch crossing. For example, the ATLAS experiment observed, on average more than 30 interactions per bunch crossing and up to 80 in Run II of the LHC. At LHC energies, each interaction produces tens of particles so that hundreds to thousands of secondary particles are produced in a single bunch crossing.

It is not technically feasible to record all particles of every bunch crossing. For example, the ATLAS experiment in Run I had 140 million channels, resulting in an average event size of about 1.5 MB. At 40 MHz, the full readout of the detector would

https://doi.org/10.1515/9783110785968-004

result in a data rate of about 60 TB per second to be stored, which is impossible even for a laboratory like CERN. It turns out, however, that the vast majority of collisions are not of interest and do not have to be stored. The cross-section for a strong interaction with low momentum transfer makes up the largest fraction of the total cross-section, but studies of these interactions are not the prime priority for the physics program of experiments such as ATLAS, CMS, and LHCb. Thus, the experiments make use of a so-called trigger to reduce the number of events to record while ensuring that the events of interest are kept. A trigger is an algorithm that quickly (within 25 ns) decides whether a bunch collision produced an interesting final state. Examples for such interesting final states will be given later.

Ideally, a thorough analysis of the event would yield perfect information for the trigger to make a decision, but this is challenging due to the short time window between two bunches. Within this window, the full detector needs to be read out, and a computer farm has to process the raw data in order to reconstruct particle tracks from detector hits.

Section 4.2.1 describes the classic trigger concept that guided the design of readout electronics and data acquisition of the LHC experiments in the previous Run I and Run II of the LHC. Until recently, it was deemed impossible to do this for every event at the LHC, but a newly developed readout and trigger scheme employed as part of the upgrade of the LHCb experiment in Long Shutdown 2 will make it a reality. Section 4.4 will give an overview of this new architecture, where the first stage of the trigger will be based on GPUs. The resulting data structures for LHCb will be discussed in Section 4.3.1.

Compared with the situation in particle physics, astroparticle detectors do not have a fixed rhythm in which events occur since the astroparticles originate from natural sources. The instruments are read out continuously during data-taking, which would also produce unmanageable data volumes if stored permanently. The challenge here is to reject the background of atmospheric leptons from low-energy cosmic ray interactions, which are registered in the instruments but are usually not of interest. For this reason, also the astrophysical experiments employ triggers, which here serve the purpose of identifying that an event occurred. In comparison with the triggers in particle physics, these triggers are usually less complex. For example, they look for coincident light emissions in different parts of the detector, e. g. multiple DOMs in the case of IceCube or multiple pixels in several telescopes for CTA. After the trigger decision is made, the raw data volume is still too large for transfer (IceCube) or permanent storage (CTA), so the next step is data volume reduction, where lossy compression is applied to further reduce the amount of data.

Details on the data acquisition are presented in Section 4.2.2 for the Imaging Air Cherenkov Telescopes and in Section 4.2.3 for IceCube. The resulting data structures are described in Section 4.3.2 and Section 4.3.3 respectively.

4.2 Data Acquisition

The data acquisition systems in astroparticle and particle physics experiments are designed to handle increasing data volumes and data rates. In both fields, this is driven by searching for rare events in a large amount of uninteresting background. Since the volume of data that can be reasonably stored and further processed by analysts is limited, the experiments employ trigger algorithms that select interesting events and discards large amounts of background. An experiment can have several triggers to select different kinds of interesting events.

In the past, the trigger was often implemented in hardware since it had to cope with a large data rate, but there is a common development in both fields to implement more and more parts of the trigger in software. This allows one to add new triggers on-demand and implement more refined trigger algorithms.

Another common aspect is the use of temporary buffers, which store events that arrive in the time window while the trigger is processing the previous event. Since the time between two events of interest follows an exponential distribution, a significant fraction of interesting events would be lost without buffers.

The following three subsections describe the data acquisition systems of LHCb, as an example of a modern particle physics experiment, of Imaging Air Cherenkov Telescopes (IACTs), and IceCube, as an example of a neutrino observatory.

4.2.1 Data Acquisition for LHCb

Bernhard Spaan
Hans Dembinski

The LHCb data acquisition system was designed as a scalable computing network with a powerful software trigger stage to select interesting events and to provide analysts with high-level objects that can be directly used without further processing. These capabilities have been further expanded in recent years by moving to a full software trigger and a full online reconstruction, which also performs the calibration and alignment of the detector online. These steps make it possible for LHCb to move from full persistence of raw data to selective persistence of only higher-level physical objects like decays that can be used directly in physics analyzes. We first discuss the trigger concept and then the readout.

The classic trigger concept for experiments at an accelerator uses several successive stages. Each stage requires more time than the previous stage to form a decision, which is compensated by a reduction of the event rate after each stage. The first stage of the trigger at LHCb is called L0 ("Level 0"). It is very fast and reduces the event rate by a factor of 40. Dedicated hardware is used to analyze calorimeter information as well as

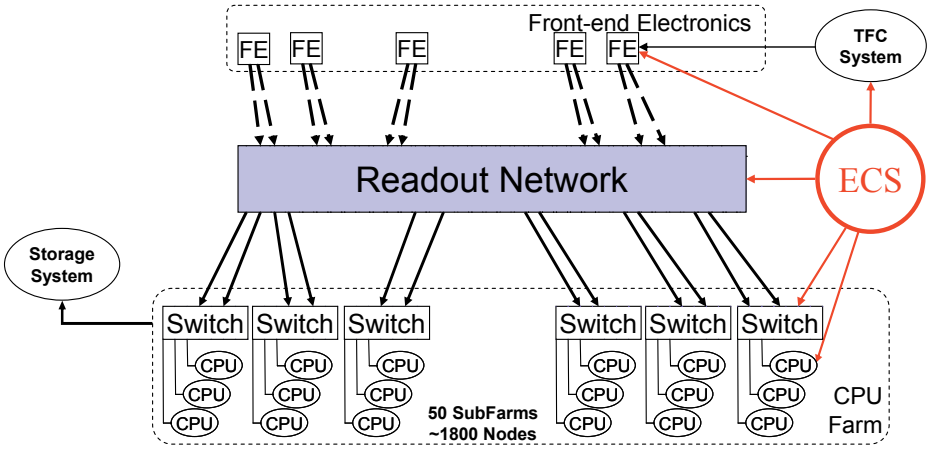


Fig. 4.1: General architecture of the LHCb Online system with its three major components: Timing and Fast Controls (TFC), Data Acquisition, and Experiment Control System (ECS) (Reprinted with small changes under CC BY 4.0 license from [243]).

hits in the muon system, similar to the first-level triggers of ATLAS and CMS. Hits in the muon system are commonly used as a trigger since events that produce muons are comparably rare. Since B hadrons are the primary focus of the LHCb experiment, the calorimeter trigger is also important. It exploits the fact that decays of the heavy b and c quarks produce particles that have a higher transverse momentum, p_T, with respect to the beam axis than the average particle stemming from the primary proton-proton interaction. The L0 trigger thus requires either a calorimeter cluster with several GeVs of transverse energy, a single muon candidate with a p_T of more than $1.4\,\mathrm{GeV}/c$, two muons that are back-to-back in the transverse plane and have a p_T of several hundred MeV/c each. The readout electronics are designed to temporarily store information from the most recent events that occurred after the last readout until the next L0 trigger signal is issued. Temporary storage is achieved either by amplifying analog signals as needed (as in the case of the Vertex Locator) or by digitizing the detector response for each of the active channels every 25 ns and storing it in a circular buffer.

The next stage, usually called L1, is called HLT1 in LHCb ("high-level trigger 1") because it runs in software on a computer farm and already processes the full event after the L0 stage triggers a readout. In the first years of operation (Run 1), the event filter farm was gradually developed to produce more output events per second. From initially planned 2 kHz, the output rate was gradually increased to 5 kHz. This led to a similarly increased luminosity, which was usable at LHCb. For the second phase of the LHC operation (Run 2), the event filter farm was significantly upgraded to consist of 62 sub-farms with more than 50 000 logical cores in total, which is almost doubling the number of logical cores compared with the first phase.

A plan of the LHCb online system that was used in the first and second phases of the LHC before the upgrade in 2018–2019 is shown in Figure 4.1. A description of the LHCb experiment and its sub-detectors (represented by the top nodes in the graphic) was previously given in Section 2.3.1. The purpose of the online system is to transport data belonging to a given bunch crossing from the detector front-end electronics to permanent storage if the bunch crossing is selected by a trigger condition. The design is based on the principles of simplicity, scalability, and, wherever possible, the use of commercial off-the-shelf products and established components and protocols. As a consequence, the LHCb online system is reliable and robust, with enough flexibility to cope with new requirements motivated by experience with real data.

On readout, the on/near-detector electronics (*FE electronics* in the figure) for each LHCb sub-detector multiplex the information and send it either via optical fibers or as an analog signal to the so-called electronics, which consists of standardized *readout boards*. In the case of optical transport, the information of many channels is transported via a single optical fiber at a rate of 1.6 Gbit/s, and about 6000 optical links are used for LHCb. The readout boards can handle up to 64 analogs or up to 24 optical links. They process the optical or analog signals with FPGAs to perform error checks and apply forms of data compression (in the simplest form by not forwarding information from channels without hits), depending on the needs of the individual detectors. The resulting data fragments are collected by another FPGA and formatted into a raw IP packet that is sent to the *Readout Network* via fast Ethernet. Multiple events are packed into a single frame to minimize I/O overheads further down the line. Clock and synchronization signals (as such triggers) are received from the Timing and Fast Controls (TFC) system. The readout boards can also send commands back to the TFC if there is congestion and the trigger rate needs to be throttled. The readout network assembles the data streams from the readout boards and passes those to the event filter farm, on which the High-level Trigger (HLT) runs in software. The HLT algorithms are run on a farm with several thousand CPUs and receive events at a rate of up to 1 MHz from the hardware L0 trigger. The events selected by the farm are then stored permanently (*STORAGE* in the figure).

The quality of the acquired data is automatically checked in a separate monitoring farm (*MON farm* in the figure) with user-defined algorithms to track the efficiencies of the detector channels or the mass resolution for certain decays. Random triggers issued by the L0 trigger are also accepted with a low rate to monitor the trigger itself. All components of the LHCb online system are connected to the Experiment Control System (ECS), which allows for control and monitoring by scientists.

The LHCb experiment has since moved to a trigger-less readout system and a full software trigger [245] because the largest inefficiencies in the entire trigger chain occur in the L0 trigger decision. The new readout system has a similar architecture but skips the hardware L0 trigger and uses a readout network with a much larger bandwidth. It is able to process the full inelastic collision rate of 30 MHz with the event filter farm and a

bandwidth of 4 TB/s. Full track reconstruction is performed at 1 to 2 MHz. Higher-level objects like decay vertices are reconstructed at a rate of 20 to 100 kHz.

During Run 2 of the LHC, real-time alignment, and calibration were introduced to make it available for the HLT algorithms [248]. As of Run 2, the HLT is split into two stages. The first stage, HLT1, receives events at a rate of up to 1 MHz from the L0 trigger. A first full track reconstruction is performed at this stage, and events are selected. The intermediate data are buffered to disk and used to obtain high-quality calibration and alignments in an asynchronous process. The second stage, HLT2, then uses this data to obtain the best track reconstruction possible. This scheme not only improves the efficiency and purity of the HLT selection but also allows for the offline reconstruction to be skipped.

Consequently, offline data processing at this stage is limited to data filtering, which is known in LHCb as *stripping*. In the stripping process, many selection algorithms (*stripping lines*) are executed in parallel. Each line selects events and produces a stream of higher-level reconstructed physics objects that are directly used in data analyzes (e.g., fully reconstructed decays). Stripping reduces the event size to simplify the analysis, hence the name.

The next step forward, also introduced during Run 2, was to perform the stripping online to produce *turbo streams* in addition to the traditional full stream that contains the raw event data. Turbo streams maintain only the higher-level objects that a particular analysis requires, along with monitoring information, but not the raw data and not other unrelated high-level objects. Turbo streams are produced online at the HLT2 stage and saved in an internal compressed format optimized for writing speed. Offline processing of turbo streams requires very limited CPU resources, at the level of 1 in 1000 of the total, because it only amounts to a conversion of the internal data format to ROOT and the inclusion of luminosity information.

The size of an event depends on the amount of information that is stored, ranging from a few kB for a turbo event up to 200–250 kB if the full event is persisted. In 2018, full events were written with a rate of 7 kHz and turbo streams with 3.1 kHz, reaching a throughput of 0.65 Gbit/s with an average full event size of 70 kB [248]. Several PB of data were produced.

4.2.2 Data Acquisition for Imaging Air Cherenkov Telescopes

Maximilian Linhoff

In general, data acquisition for Imaging Air Cherenkov Telescopes (IACTs) consists of two parts, a buffer system keeping a time range of up to several microseconds, and a trigger system that decides when an air shower was likely registered, which leads to the readout of the buffer system around the trigger time. In the most common IACT

trigger system, used in different variants for FACT, MAGIC, and some CTA telescopes, the pre-amplified analog signal is duplicated and split into both systems, and the buffer stores analog charges.

The trigger system combines the analog signal of multiple pixels into what is called a trigger patch and applies a threshold to these patches, transforming the trigger signal of each patch into binary. Different rules can be applied on how many patches at the same time have to be above the threshold and how long the time overlap has to be. When these rules are fulfilled, the buffer is read out, and the analog values are digitized.

For telescope arrays, there is usually the second step of an array trigger that requires that at least a certain number of telescope triggers report an event. Due to the distance between the telescopes and the required processing time, stereo arrays, especially larger arrays like CTA, require the telescopes to have larger buffers. In the following, the general hardware of IACTs and the trigger system is described.

Imaging Air Cherenkov Telescopes most commonly comprise a single, segmented main reflector dish and a very fast camera able to detect down to single photons with sub-nanosecond resolution. The size of the reflector varies widely over the different currently operating and planned facilities.

Two main technologies for the photodetectors are in use: photo-multiplier tubes (PMTs) and silicon photo-multipliers (SiPMs). While today, like photon detection, efficiency in the wavelength range is relevant for IACTs, other properties are very different. PMTs require high voltages, typically several kV, and are susceptible to damage when exposed to excessive light levels, which can already occur during moon-lit nights. For a long time, PMTs had an advantage in photon detection efficiency over SiPMs, which have only recently reached comparable levels, especially in the ultraviolet to blue regime important for detecting Cherenkov light. SiPMs require only a voltage in the range of 30 V to 100 V and are virtually indestructible at high light intensities. A major drawback of SiPMs is their limited size, which makes it hard to tile large cameras as required for the field of view of, say, the CTA LSTs without complex electronics or light guides. See [186] for a detailed review of these technologies.

The cameras are continuously read out during observations. However, the data volume of storing all camera signals far exceeds the amount possible for even short-term storage. An IACT with a 2000 pixels readout at 1 GSample/s digitized at 16 bits would create a staggering data rate of 4 TB/s. Thus, all IACT cameras have a trigger that observes the continuous signal and decides when data should be digitized and stored permanently. This requires a buffer for either analog or digitized values to be able to store the relevant data until the trigger can make its decision. The trigger usually works by combining the signal of multiple pixels into groups and then applying thresholds on the height of the group signal and the number of groups above that threshold. Figure 4.2 shows the individual pixels and the trigger groups for the FACT camera. In the case of multiple IACTs operating as an array, there is also a stereo trigger, combining trigger information of all telescopes. For more details about the FACT trigger system, see [53]; for information about the CTA trigger system, see [172]. Larger buffers are required in

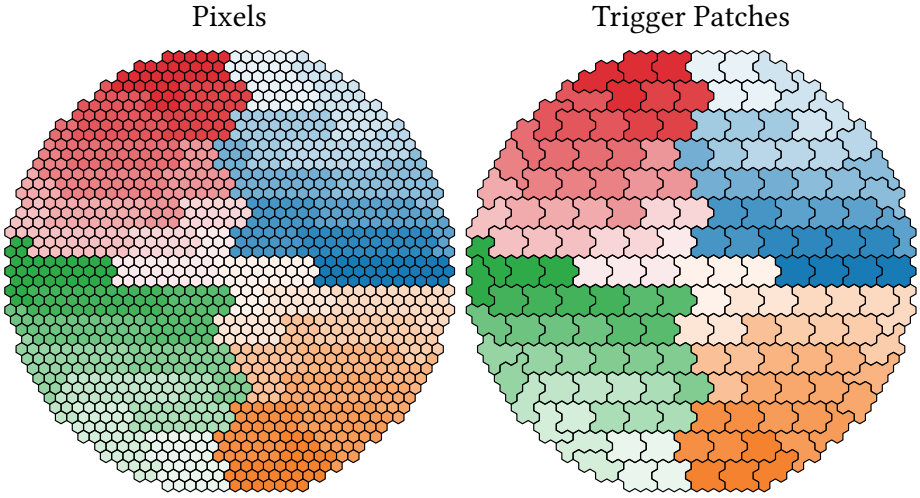


Fig. 4.2: FACT pixels (left) and trigger patches (right). The trigger patches calculate the analog sum of 9 pixels each.

Tab. 4.1: Key properties of currently observing and planned IACTs

Telescope	MAGIC	FACT	CTA-LST	CTA-MST	CTA-SST
Mirror diameter	17 m	4 m	23 m	12 m	4 m
No. of pixels	1039	1440	1855	1855	2048
Technology	PMTs	SiPMs	PMTs	PMTs	SiPMs
Field of view	3.5°	4.5°	4.5°	8°	9°

this case since the signal delay between different telescopes leads to a larger delay between the recording of the data and the final trigger decision. In case the trigger decides that data should be recorded, all pixels of all telescopes with a trigger signal are typically read out for a duration of several tens of nanoseconds around the trigger time, digitized, and stored to disk along with the necessary metadata. This is the first data level accessible to software analysis: time series information of the photo-sensors for each individual pixel—basically a very short video. The typical data structures of IACT data are discussed in Section 4.3.2.

Table 4.1 lists the key properties of currently operating and planned telescope types.

4.2.3 Data Acquisition for IceCube

Tim Ruhe

Data acquisition in IceCube is based on the utilization of the Cherenkov effect, which causes the emission of photons at blue wavelengths. Said photons are then collected by Digital Optical Modules (DOMs), where they are transformed into a time series of charges, often referred to as a waveform, by the use of multiple digitizers with overlapping dynamic ranges, and different sampling speeds. Trigger algorithms continuously search for patterns matching predefined criteria and combine the waveforms returned by individual DOMs to events. These events are then handed over calibrated and processed by basic reconstruction algorithms in order to check whether a certain hypothesis (track or shower) can be met. The data is then further processed to search for events that are likely of astrophysical origin, which then triggers a community alert, to allow follow-up observations with other instruments. Furthermore, the data is compressed in order to meet the criteria for satellite communication and prepared for long term archival.

Although IceCube is a neutrino observatory [21], it does not actually detect neutrinos, as these can only be observed via their leptonic partners, created in a neutrino-nucleon interaction. These neutrino-nucleon interactions can proceed either as so-called, charged-current (CC) or neutral-current (NC) interactions. CC interactions are governed by the following equation

$$\nu_l + X \rightarrow l + X', \tag{4.2}$$

while a neutral current interaction is governed by

$$\nu_l + X \rightarrow \nu_l + X'. \tag{4.3}$$

In both cases the ν_l is an incoming neutrino of a specific leptonic flavor l and l denotes the emerging lepton. The nucleus with which the interaction takes place is denoted as X, whereas X' represents a hadronic or electromagnetic cascade, which also emerges as a consequence of the interaction. In contrast to Equation 4.2, no emerging lepton is produced in a neutral current interaction (Equation 4.3).

Due to their electric charge, the generated leptons, along with the secondary leptons emerging within the cascade, cause a polarization of the medium, which is relaxed by the emission of a characteristic radiation after the lepton has passed. For energies relevant for IceCube, the velocity of these leptons exceeds the speed of light in the detection medium and the radiation emitted by the medium, will interfere constructively. This effect is known as the Cherenkov effect, and the emitted radiation is referred to as Cherenkov radiation. In water this radiation has wavelengths between 400 and 700 nm. In order to observe neutrinos with IceCube, one therefore has to detect light and the detector needs to be designed accordingly.

Fig. 4.3: Top view of the IceCube detector with different colors indicating different construction seasons. Graphic courtesy of the IceCube collaboration.

Photosensors for the detection of light that are embedded in Digital Optical Modules (DOMs) are the fundamental units of the IceCube detector. In total, 5160 of these DOMs are deployed in the deep ice at the geographic South Pole at a depth of between 1450 and 2450 m. The DOMs are arranged on 86 vertical cables called strings, which are themselves arranged on a triangular grid. While the DOM-to-DOM distance is 17 m, the distance between individual strings is approximately 125 m. Each DOM houses a 10" PMT, high-voltage supply, flasher-LEDs, and readout electronics. Data taken by the DOMs is transferred to computers in the IceCube Laboratory (ICL) via twisted wire pairs, which also provide the necessary power to operate the DOMs. Signals sent by DOMs on the same string are combined and collected on dedicated computers, referred to as DOMHubs [21]. As the construction of the IceCube detector was carried out over multiple polar seasons, data was taken with incomplete instrumentation during that time. In particular, data was taken with 22, 40, 59, and 79 strings before the detector was completed in 2010. This incomplete instrumentation leads to a different geometrical shape between individual seasons. To distinguish between different geometries, the number of strings is used. For example, IceCube-22 refers to IceCube in the 22-string configuration. In case the number of strings is not specified, the fully instrumented detector is meant. A top view of the detector with different colors indicating different construction seasons is given in Section 4.3.

In IceCube two different types of event patterns are observed. The first one is referred to as *track like* and originates from ν_μ CC interactions, whereas the second one originates from ν_e and ν_τ CC interactions as well as from NC interactions of all neutrino flavors and is referred to as *cascade like*. Due to the long range of muons in ice, the interaction vertex of a track-like event can be outside the instrumented volume and even occur in

the bedrock below the detector. Cascade like events, however, have to occur inside or at least in very close vicinity to the detector. Events where the entire amount of light is deposited inside the detector are referred to as fully contained. Neutrinos interacting inside the instrumented volume are of special interest, as these *starting events* have a high probability of being of astrophysical origin, especially for cases where their energy exceeds a certain threshold. High Energy Starting Events (HESE) were the basis for the discovery of the diffuse flux of high-energy astrophysical neutrinos (see [211], [19] and [31] for details).

Within the DOMs the arriving Cherenkov photons are transformed into charges by the PMT, which then finally forms a waveform, which is basically a time series of charges. The digitization of the waveforms is also carried out within the individual DOMs. As the photons may arrive from up to 500 m away, and due to the large range in the energy of incident particles (typically 10 GeV up to several PeV), the waveform amplitudes in the DOM have a large range as well (several mV up to ≈ 2 V). In order to handle this variety of signals each DOM is equipped with multiple digitizers, with overlapping dynamic ranges and different sampling speeds. When a DOM registers one or more photons, this is generally referred to as a hit and each DOM detects photons individually. The majority of these photons are due to dark noise, however [21].

As hits due to dark noise will occur as a stochastic pattern throughout the detector, they can be distinguished from photons originating from particle interactions via coincidence requirements. In IceCube two coincidence requirements can be distinguished. The first one is a *hard local coincidence* (HLC), which is fulfilled in case two neighbouring or next to neighbouring DOMs are hit. HLC hits often have complex waveforms indicating that individual DOMs are hit by multiple photons, and are thus saved in full detail. Isolated hits are marked as *soft local coincidences* (SLC) and more aggressively compressed [21].

DAQ trigger algorithms continuously look for clusters of HLC hits in space and time, which indicate a particle interaction instead of dark noise. All algorithms search for a given multiplicity of HLC hits and may also include geometric requirements. The time scale of the trigger is set by the speed of light in ice and the geometric requirement of the respective trigger algorithm. Certain triggers are restricted to a subset of optical modules, e.g. all in-ice[1] or all Deep Core DOMs [21].

The *Simple Multiplicity Trigger* (SMT) requires N or more HLC hits within a time window of several µs, without any constraints on the locality of the hits. N is tuned to the energy threshold of the specific sub-detector, which is set by the string spacing. The *Volume Trigger* defines a cylinder of fixed size around each hit DOM and requires a given multiplicity within this cylinder. This enables the triggering of low-energy events that do not fulfill the SLC conditions. In addition, the Volume Trigger has a simple multiplicity

1 The IceCube neutrino observatory also contains the surface array IceTop [33], which is not covered in this book. The selection of *all in ice DOMS* equals the exclusion of the DOMs in IceTop.

parameter that fires, when a certain number of hits is reached within the defined volume. The *String Trigger* requires a certain number of hits within a span of optical modules along a given string. This enables the triggering of low-energy muons that pass vertically through the detector. While the aforementioned triggers are designed for particles moving through the detector with approximately the speed of light, the *SLOP* trigger aims at triggering slow-moving particles such as magnetic monopoles. For details on IceCube's searches for relativistic and non-relativistic magnetic monopoles, we refer to [20] and [27], respectively.

The *Fixed-Rate Trigger* reads out 10 ms of hit data from the full detector at fixed intervals. As many events fulfill more than just one trigger condition, subtriggers are merged and a global trigger combines time windows and forms non-overlapping trigger requests, which are then used by the Event Builder as templates to retrieve the full information on the hits and to assemble an event. The Global Trigger also ensures that the same hit does not appear in multiple events [21].

Once the Event Builder receives a request from the Global Trigger, it extracts the read-out time windows. The time windows are then forwarded to the appropriate StringHubs, which return a list of all hits inside the given time window. When all hubs have returned a list of hits, these hits are bundled and an event is formed and written to a temporary file, which is renamed in a standardized way, once it reaches a preset size. The files are then passed on to *Online Processing and Filtering* (PnF) [21].

Online Processing and Filtering handles all triggered events and reduces the data volume to a size that can be handled in satellite transfer. The data treatment includes the application of calibration constants, as well as event characterization and selection. Furthermore, information on data quality is extracted. Real-time alerts are generated for astrophysically interesting events (see [22] for details on real-time alerts by IceCube). Last, data files and metadata information are created for long-term archiving [21].

The first data-processing step consists of the waveform calibration. This step requires information on the geometry and the detector status, which is stored in a database. In a second step, the waveform is deconvolved on the basis of the known DOM response to single photons. The deconvolution is required in order to extract amplitudes and arrival times [21]. The extracted time series of waveform amplitudes is the basis for all further reconstructions and stored in a specific format (SuperDST), which uses only 9 % of the storage, compared with the full size of the waveforms. For DOMs where the SuperDST representation is found to be not sufficiently accurate, or which recorded high levels of light, the full waveform information is stored as well [21].

Events are then further processed by reconstruction algorithms, in attempting to match the observed light patterns with track and shower hypotheses. The extracted vertex positions, the reconstructed direction and energy, and the goodness of fit are used to extract interesting events by various filters. Filter selections are set by the collaboration and tuned on a yearly basis. In 2016, 25 filters were in operation and approximately 15 % of all events are selected by one or more filters. Dedicated filters,

for example, search for events that are likely of astrophysical origin and send out alerts, used for follow-up observations by other experiments [21, 22].

The PnF also aggregates information regarding quality and stability of the data-taking, which are utilised for detector monitoring. Furthermore, files for satellite transmission and long-term archiving are generated [21].
The total amount of raw data acquired by the IceCube neutrino telescope is 1 TB/d. Using SuperDST and online filtering reduce the data to 170 GB/d and 90 GB/d, respectively [21].

4.3 Data Structures

Data has to be stored in memory and on disk and organized for easy retrieval. It is common among particle and astroparticle experiments to store data in different stages of processing. The lowest stage contains raw data from the detectors, while the higher stages contain reconstructed physical objects, such as tracks of individual particles or particle showers.

Object orientation is a common theme. Data are typically organized in logical units. For example, a track object stores the position and direction. Related classes are usually linked, and a track object may contain a pointer to the lower level information from which it was formed.

Apart from these common general concepts, there is a large diversity in the details since each experiment typically develops its data structures from scratch for its particular purpose. The format in which data are persisted also differs from experiment to experiment. In the following, the data structures used by LHCb, IACTs, and IceCube are described.

4.3.1 Data Structures for LHCb

Bernhard Spaan
Hans Dembinski

The LHCb software mainly uses a classic object-oriented design to encapsulate data and complex transformations in objects. There is a clear separation of concern between data objects and transformation objects, the latter of which are called algorithms. Data objects cannot transform themselves, while algorithms cannot store event data. There is also a clear separation between data objects produced by the simulation and by the reconstruction. Data objects are passed from algorithm to algorithm in so-called Transient Data Stores, which are flexible and generic. Objects in the store are accessed via keys (strings), and multiple instances of the same class can be stored under different keys. Algorithms can only add new data objects to the store and not modify objects

already in the store. Related data objects can also be stored in data containers, typically vectors. Recently, some containers have been converted from the array-of-structs to the struct-of-arrays configuration to improve the performance of vectorized algorithms that process large amounts of data in parallel.
Reconstructions of higher-level objects used by analysts (tracks, reconstructed decays, etc.) are performed by the LHCb online system from raw event data in real time since these objects are also an input for the high-level trigger. During Run 1 and 2 of the LHC, offline reconstructions were also performed to make use of calibration and alignment data. Traditionally, the reconstructions were processed offline, but this changed with the introduction of online alignment and calibration so that high-level physics objects of the highest quality are now produced online. Only a fraction of the events is now persisted with full raw event data, called *FULL stream*, while many analyses use the so-called *Turbo stream*, which persists only the selected high-level objects from an event that are needed for a particular analysis. A further *TurCal stream* is persisted to monitor the online calibration and alignment.

The LHCb software architecture distinguishes between transient (in-memory) and persistent (on-disk) data representations [57]. This separation has several advantages. The persistent data format can be changed without affecting the physics code, and the two representations can be optimized following different criteria. Making use of this, the previously mentioned streams are written to disk in different internal data formats, which are further converted offline into the ROOT-based DST (Data Summary Tape) and MDST (micro-DST) formats for user analysis. Both formats contain reconstructed physics objects such as tracks, vertices, etc.), while the DST can also contain the raw data.

Data processing in LHCb is performed with a pipeline of *algorithms*, the basic data-processing blocks in the LHCb software architecture that perform reconstruction, filtering, or analysis. Data is passed from algorithm to algorithm in *Transient Stores*, which are organized in a tree-like structure similar to a Unix file system. Each node of the tree is the owner of everything below it, so deleting a node also deletes all its children. Associations between otherwise independent nodes are implemented with links, which extends the lifetime of the linked node to the nodes which hold the link.

There are three categories of data with different access patterns and lifetimes. Each has a corresponding store. The stores are used to pass data between algorithms and act as an intermediate buffer for derived representations such as persistent or graphical representations. Zero or more persistent or graphical representations correspond to one transient representation.

– The *Transient Event Store* (TES) contains the event data obtained from particle collisions (real or simulated) and their successive processing, which are valid only for the time it takes to process one event. By convention, algorithms may not modify data that is already in the TES nor add new objects to existing containers. This ensures that algorithms can be run in any order (the algorithm sequence will

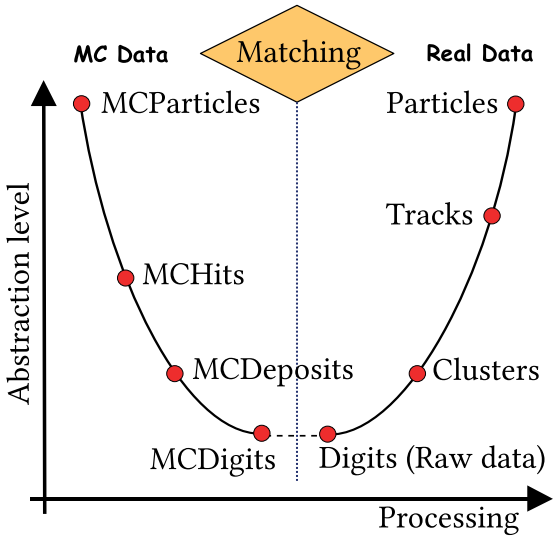


Fig. 4.4: Processing of objects in LHCb. Direct links exist only between adjacent objects in this graph. The processing of the simulated and the real event is fully disconnected, as indicated by the vertical dashed line. The reconstructed event is matched to the simulated event at the lowest level of abstraction, i.e. the digits.

either fail or succeed to produce the same outcome, instead of potentially different outcomes depending on the order).

- The *Transient Detector Store* contains data that describe parameters of the LHCb detector that can vary during a period of data-taking, such as its calibration and alignment parameters. It exists for a period of data-taking in which many events are processed.
- The *Transient Histogram Store* contains statistical data (histograms or tables) generated over the lifetime of a job from processing a set of events over several periods of data-taking.

The stores contain software objects that typically model physical objects. A physical particle, track, detector hit, cluster, vertex, etc., is represented by a corresponding software object. Related objects can be grouped together into a *data container*, typically a vector of contiguous memory. The data objects in the TES are organized into a number of sub-trees.

- MC: Output of the detector simulation, hits, and deposits, particles, and vertices from material interactions or particle decay.
- Raw: Raw data from the real or simulated detector in the same format.
- Rec: Output of (sub-detector) reconstruction (clusters, tracks, etc.)
- Phys: Highest-level objects for physics analysis (particles, vertices, etc.)

Simulated objects are presented by classes other than reconstructed objects and are strictly separated, as illustrated by Figure 4.4.

The LHCb architecture foresees that data objects have a minimal interface that is necessary to conveniently access their state. Code that transforms data in a non-trivial way is instead put into the aforementioned algorithms, which are also objects, but with the complementary purpose of transforming data objects. The LHCb software makes use of vector units in modern microprocessors, which can perform mathematical operations on whole vectors of numbers within a single CPU cycle. To facilitate the use of vectorized math, traditional arrays of structs (AoS) are converted into structs of arrays (SoA) where appropriate [247]. Some algorithms are converted to run on GPUs to make use of massively parallel execution.

4.3.2 Data Structures for IACTs

Maximilian Linhoff

The raw data recorded by most modern IACTs consists of time series information for each pixel in the camera, typically with a resolution between 1 and 2 GSample/s and a duration between 30 and 150 nanoseconds. Some telescopes use two different signal amplification gains for each pixel to increase their dynamic range. This results in a single telescope event being a regular array of dimensionality $(N_{\text{gains}}, N_{\text{pixels}}, N_{\text{samples}})$, where the first dimension might not be present for telescopes not using more than one amplification (e. g. FACT and MAGIC). The data is stored in a flat array since the pixels of IACT cameras are usually not on a regular square (hexagonal grids are more common).

This raw data is accompanied by additional metadata, such as the timestamp of the observation, information about the trigger decision, and information needed to perform calibration for detector properties. IACT array events comprise several of these telescope events that belong to the same shower along with additional array-wide metadata, such as the timestamp of the stereoscopic trigger.

In a typical IACT analysis, this raw data must first be calibrated, e. g. to mediate manufacturing differences between pixels. In case of multiple gains, the appropriate channel is selected. Then the time series information is reduced to just two aggregated values per pixel: the estimated number of photons and their mean arrival time. This data is commonly called the “image level”. For analyses based on classical feature extractions, see Section 7.3. These images are then parametrized, resulting in a list of numbers for each event or a 2d table for multiple events, which is the input for most classical machine learning algorithms. For more information, see Section 8.4.

In recent years, Deep Learning approaches starting at earlier data levels, such as the image level or the calibrated time series, have gained traction, as explained in Section 9.4.

In addition to the event data itself, more slowly changing data describing the state of the telescope and the environment is needed. This includes the telescope pointing direction, calibration coefficients, and atmospheric transmission.

The basic unit of IACT data is the array event, the data belonging to one air shower recorded by the telescopes. An array event is composed of several telescope events, data recorded by a single telescope of the array, and array-wide metadata.

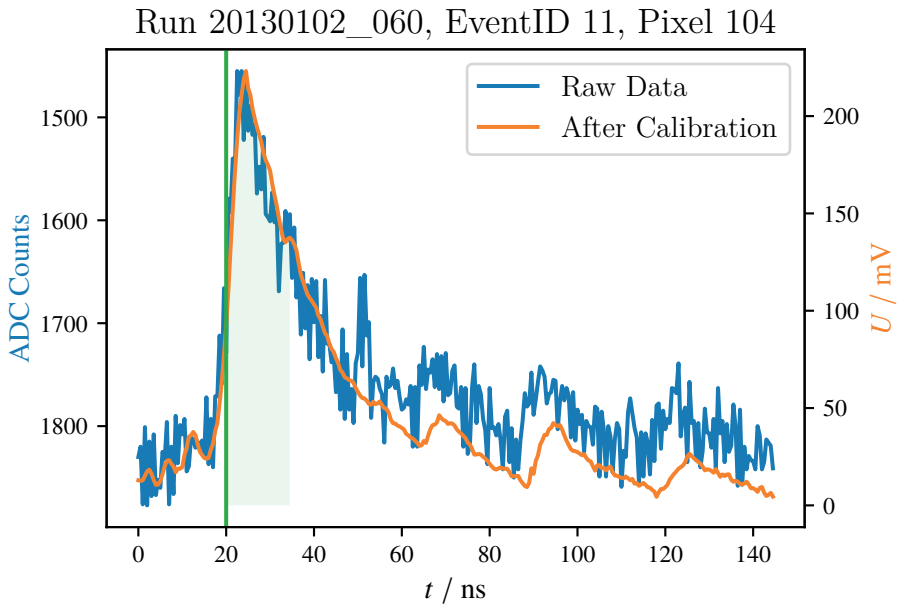


Fig. 4.5: Time series of a single camera pixel of FACT before and after calibration has been performed. The vertical line is the inflection point which is used as the arrival time for the Cherenkov pulse, and the shaded area shows the integral performed to estimate the number of photons. These two properties are displayed in Figure 4.6.

We will follow CTA terminology in this section, which defines several *data levels* for progressively higher processing stages of the analysis. The first data level, `R0`, is the raw digitized signal as produced by the camera electronics and readout server. For several IACTs, including MAGIC and FACT, this is the data level that is stored for long-term preservation and the start point for off-site data processing. The main content at this data level is the voltage time series of each camera pixel in each telescope, contributing to the array event. In general, this is stored as a three-dimensional array of shape $(N_{\text{gains}}, N_{\text{pixels}}, N_{\text{samples}})$. N_{gains} is the number of pre-amplification channels used to read out the signal, usually 1 or 2. N_{pixels} is the number of pixels, and N_{samples} is the number of samples in the time series of each pixel. The R_0 data is specific to each telescope and has to be calibrated for multiple effects, including production differences

between the photo sensors, temperature dependencies of the electronics, aging of PMTs, and more. In the case of multiple gain channels, the most appropriate channel is selected. The result is the `R1` data level, pre-calibrated, homogenized time series data with only one channel, resulting in a shape of $(N_{\text{pixels}}, N_{\text{samples}})$. In the case of CTA, the calibration from $R0$ to $R1$ is performed on the telescope server before the data is transmitted to the central array computing center, while for most other telescopes, $R1$ is an intermediate data level not stored but only produced as an intermediate step of the processing.

The pixels containing the Cherenkov signal from an air shower are usually only a very small sub-sample of the camera. To reduce the data amount, CTA plans to not store time series information for all pixels of a telescope event but only for those likely to contain the Cherenkov signal. The data level after the *data volume reduction* is called `DL0` and is the first data level foreseen for the long-term preservation of CTA. The shape is, in general, the same as for `R1`, but the array now is sparse, i. e. it does not contain values for all pixels.

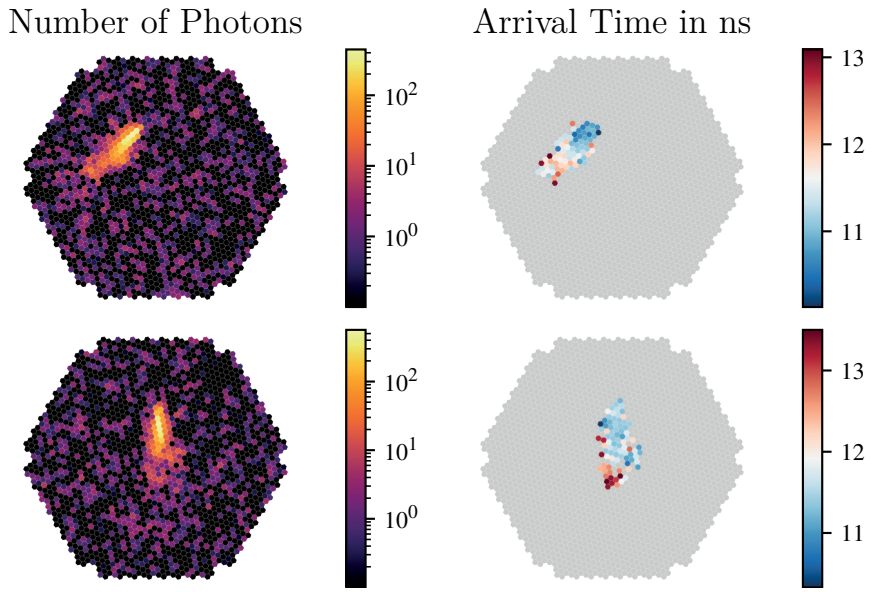


Fig. 4.6: A single 500 GeV shower as observed by two CTA Large-Sized Telescopes.The time series in each pixel has been transformed into the estimated number of photons (left) and their mean arrival time (right). On the right side, pixels not selected by the image cleaning are shown in gray.

In the common, "classical" analysis approach, this data is now reduced in three steps:

1. Image extraction: the number of photons and the arrival time of the Cherenkov pulse are estimated from the time series in each pixel. The number of photons in

each pixel is called the *image* and is comparable with our classical understanding of the term: a monochromatic brightness image.
2. Image cleaning: pixels likely containing Cherenkov signal are selected.
3. Image parametrization: from the selected pixel values, several aggregations are calculated, resulting in a number of *image parameters* for each event.

These steps are detailed in Section 7.3. In the context of data structures, it is enough to highlight that we get two additional representations of the IACT data: the image level with two arrays of shape N_{pixels} for each event and the image parameter level with a collection of single aggregations of all information discussed before. CTA combines these two intermediate data levels into `DL1`.

The `DL1` parameters are the entry point for classical machine learning-based analyses to reconstruct the properties of the primary particles, which will be discussed in Section 8.4. Deep learning approaches, discussed in Section 9.4, usually start at the `DL1` image level but approaches directly starting at the time series stage are also under development.

The result of the event property estimation step is the `DL2` data level. At this stage, possibly multiple values for each property are estimated by different algorithms.

The final event-based data level is `DL3`, which contains the reconstructed event lists of gamma-ray candidates accompanied by the instrument response functions . This is the input data level for the final step of a gamma-ray analysis, the solution of the inverse problem discussed in Section 10.5. It is also the data level CTA will provide to the science users of the observatory. All previous data levels are internal or intermediate data levels and are processed by CTA to produce the `DL3`.

Previously, we discussed the abstract data structure of IACT data, not a concrete data or file format in which this data is stored. While the data structures and analysis approaches are essentially the same for all IACTs, there are currently no common data formats for low-level IACT data. Each collaboration of the currently operating IACTs uses its own software stack and associated data formats, most often built on top of the `ROOT` framework [56]. This also includes higher data levels and scientific analysis, but in this area, an initiative has been formed in the last years to develop a common high-level data format and open-source analysis tools for gamma-ray astronomy, mainly motivated by the fact that CTA will operate as an open observatory, but also by the wish to combine data from the currently operating telescopes to enable multi-instrument analyses and to have a format more suited for long-term archiving and independent of the often proprietary analysis software. The initiative “Gamma-ray Astronomy Data Formats” (GADF) [138] is developing a specification for `DL3` stored in `FITS`[121] files, which is the input format for the open-source analysis framework for gamma-ray data `Gammapy` [139]. `ctapipe` [296] is an open-source library and set of command-line tools for the processing of IACT data from `R1` to `DL3`, currently in prototype stage developed for CTA but with a plugin system also enabling data analysis for non-CTA telescopes or prototypes. `ctapipe`

uses `hdf5`[2] files for the intermediate data levels. The FACT-Tools / aict-tools analysis chain for FACT, discussed in Section 7.3 uses both `FITS` and `hdf5` files at different stages of the analysis.

4.3.3 Data Structures for IceCube

Tim Ruhe

IceCube utilizes its own file format, the i3-files. Inside the file, all required information is stored in an event-wise fashion in frames. Dedicated frames also contain meta-information on the detector geometry, calibration constants, and details on the physics configuration of the data-taking. In addition, the data is hierarchically organized in levels. The basic features of i3-files, as well as the meaning of the different levels, are explained in the following.

The IceCube collaboration utilizes its own file format, which is used for experimental and simulated data. This standardized file format is the i3-file, which is also used for parallel file processing. As every physics run in IceCube usually lasts for approximately eight hours, this run is split into a number of sub-runs to speed up the entire processing. The fundamental unit in the file is the frame, where every frame contains exactly one IceCube event, which corresponds to about 10 ms of IceCube data. In IceCube, an event is formed by a what is known as a trigger and combines the information available from the Digital Optical Modules (DOMs) in a physically meaningful way (see Section 4.2.3 for details). Different types of frames are present in a file, which can be broken down into two larger categories.

The first category is metadata, which is usually required once per file. *I* frames (also referred to as TrayInfo) contain information on how the file has been previously processed. *G* frames (Geometry) contain the basic detector geometry, in particular, the geometric coordinates of all DOMs. The nominal distance of regular IceCube strings is approximately 125 m, but the actual distance may differ by a few meters. The infill array DeepCore has a much denser string and DOM spacing. Precise knowledge of the DOM positions is required for an accurate reconstruction of the events. A reconstruction in this context is a chain of algorithms. For example, say, time residuals are to be computed, which require DOM positions as an input (see Section 7.2 for more details on event property reconstruction in IceCube). The *G* frame was introduced so that the geometry could be updated during the construction of the experiment and to allow for refinements of the position information. The DOM positions are measured relative to the South Pole coordinate grid (Northings and Eastings). In the first stage, x and y coordinates are derived from the position of the drill tower. z coordinates are derived

2 https://www.hdfgroup.org/solutions/hdf5/.

by utilizing pressure sensors during the deployment of the DOMs. After deployment, flasher data is used to correct the initial measurements of the geometry.

C frames (Calibration) contain the calibration constants of the photo-multiplier tubes, the core components of the DOMs. The detector status at data-taking is saved in the *D* frame (Detector), generally for the entire 8 h run. Among other things, this frame stores which DOMs and strings were active during data-taking. Although IceCube DOMs are generally very stable and most DOM failures occurred during deployment, individual DOMs or entire strings might fail temporarily, so that their information is not available for reconstruction.

Information on individual events is saved in *P* and *Q* frames. While *Q* frames contain the recorded waveforms of an event, event reconstruction results are saved in the *P* frames (Physics). Several reconstructions can be applied independently on the same raw data, resulting in several *P* frames. Within *P* frames, the intermediate results of individual reconstruction algorithms can be accessed.

In addition, IceCube data is organized hierarchically in levels. The higher the level, the closer the data are to a physics analysis, and the more reconstruction algorithms have been run on it. Level 0, which is often also referred to as the *trigger level*, is designed to be very fast and aims at separating any kind of particle interaction from noise events in the detector. This is achieved via the application of dedicated trigger algorithms (see Section 4.2.3).

Level 1, sometimes also called *filter level*, contains data taken at a rate of approximately 3 kHz and is dominated by atmospheric muons. At this stage, atmospheric neutrinos, which are the second most abundant type of particle detected by IceCube, make up only about 10 to 20 mHz. A certain amount of background rejection can already be achieved at this level by using dedicated filters that select data with different energies and topologies. The DeepCore filter, for example, selects events below 100 GeV, which interact at the bottom of the detector.

Level 2 is dedicated to collaboration-wide processing, and no events are discarded at this level. Instead, reconstructions are applied, and the events are processed for each filter. The processing is continued at level 3, where it becomes specific to a particular working group. Three main filter chains (cascades, muons, low energy) exist at level 3, where cuts are applied on events that pass a subset of filters. In addition, new reconstructions are applied, which are generally more sophisticated and resource-consuming than the ones applied at level 1. At level 3, which is the output level for the reconstructions run on level 2, the data rate is reduced to approximately 1 Hz.

At levels four and above, event selection and processing become analysis-specific, and data is, for example, filtered to obtain a neutrino-dominated sample. The numbering of the levels and which stages of the cleaning process are summarized into a level depends on the personal preferences of the analyzer and are not necessarily meaningful in themselves. The same holds for the applied cuts and the utilized (machine learning) algorithms. However, over time certain selections have been maintained and

established, which serve as input for different kinds of analyses. An event selection for IceCube is presented in Section 8.3.1.

4.4 GPU-Based Trigger Decisions

Holger Stevens

The LHCb experiment is undergoing an upgrade at the moment. A major change is the use of a pure software trigger. All sub-detectors will be read out trigger-less at 40 MHz, which results in an incoming data rate of 40 Tbit/s at the data center. This data rate will be reduced by a fully GPU-based implementation of the first trigger stage. The corresponding project is named Allen. The LHCb collaboration will be the first experiment with a complete high-throughput GPU trigger in the high-energy physics community. In the first stage of the trigger, complex raw data from the sub-detectors needs to be decoded with the highest speed and prepared for the subsequent further stages, where particle trajectories will be derived that are the basis for the determination of their production vertices and for the identification of particles as muons or hadrons. As Dortmund is participating in the construction of the SciFi tracker, the initial focus was on the development of corresponding decoding algorithms.

The Allen project is a GPU-based implementation of LHCb's first trigger stage (HLT1). The trigger is an essential tool of LHCb's overall data acquisition system, as described in Section 4.2.1. The general necessity for the trigger and the second stage (HLT2) was introduced in Section 4.1 and Section 4.2.1. For the trigger decision, the raw channel information of the various sub-detectors is combined to build tracks and pseudo particles. The different data types within the experiment are explained in Section 4.3.1. This section first presents the general Allen structure and the different approaches to parallelizing tasks. Then the performance in the sense of event throughput is presented for different GPUs.

A flowchart of the trigger process is shown in Figure 4.7. The HLT1 has to reduce the incoming data rate of 40 Tbit/s down to 1 Tbit/s. The limiting factor for the output rate is the disk buffer. In the foreseen trigger scenario, the GPUs are located in the Event Builder (EB). The EB servers are equipped with 32-core AMD EPYC (7502) CPUs, and around 173 of these are needed. Because these servers have empty PCIe slots anyway, they are ideal for hosting GPUs. In the initial plan for the upgrade trigger, no GPU was foreseen. Thereby the HLT1 was located in the EFF similar to HLT2. Due to the transfer of HLT1 into the EB, the first reduction is made before the EFF. This enables the connecting of the EB and the EFF via the onboard network cards of the server. GPU usage leads to a cost reduction; for the pure CPU scenario, an additional 100 GB card would have to be purchased. A detailed comparison of the pure CPU trigger and the hybrid GPU and CPU version are published in [246].

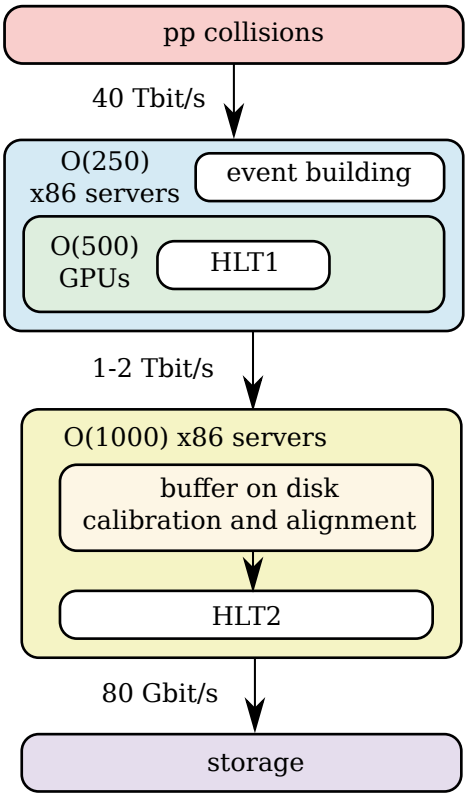


Fig. 4.7: Data acquisition system with a GPU-based HLT1 stage.

Modern instructions such as SIMD and SIMT are used within the Allen project, which allows one to perform the same mathematical operation on multiple data values in parallel. A single LHCb event is rather small, about 100 kB. Within the EB, multi-event packages are built; containing 1000 events. These packages are then copied to a GPU, and the system is not I/O bound. As the events in the package are totally independent, they can be processed in parallel. Hereby it is also mentioned that no communication between the various GPUs is needed. The HLT1 sequence is shown in Figure 4.8. Within this sequence, the different kernels use multiple threads. Algorithms are illustrated as rectangles and reducing steps as rhombi. The raw data is copied from a host CPU to a GPU, and a first cut is applied. The Global Event Cut deletes very crowded events with many hits. This is done as no significant track separation is possible here. In the following steps, the hits from the different sub-detectors are decoded, and track-finding algorithms are applied. Thereby the flight direction of the particles is adapted. One of the most important features is the primary vertex, the place where the proton-proton interaction takes place. The VELO is the only component where no clustering is done within the detector readout, and therefore the clustering is in the trigger. Triplets of

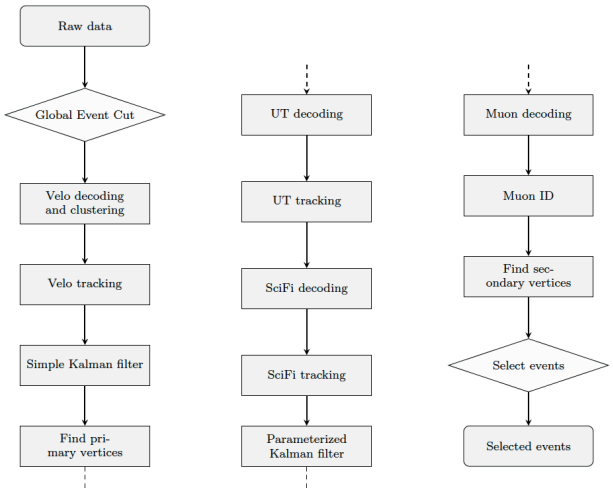


Fig. 4.8: Sequence of the HLT1 Allen application.

hits are built and used as seeds for track candidates. As all tracks are independent, the VELO tracking is parallelized for multiple seeds. The found tracks are used within the next step known as UT tracking. The tracks are extrapolated to the UT, and found hits are matched to the tracks. Of course, the different tracks are handled parallel. The next algorithms correspond to the SciFi tracker. At first, the hits are decoded, and second, the hits are used to extend previously found tracks. Silicon photomultipliers are used as a sensor in the SciFi. The smallest readout unit is a channel of 96 pixels, and in total, this sub-detector has more than 500 000 channels. These are distributed over three stations with four layers each. Within the readout electronic, the signal of the channels is processed, and clusters are formed. This improves the signal-noise ratio. The cluster data is packed in so-called raw banks. The task of the decoding is to transform the cluster from the various raw banks and transform them into physical hit coordinates. An important requirement of the overall trigger sequence is the fact that the hits need to be sorted in x. The SciFi decoding is split into three algorithms: calculate cluster count, pre-decode, and raw bank decoder [359]. For thread-based decoding, the number of clusters is essential. The clusters are counted because the amount is not known beforehand. Due to special data handling, directly neighboring clusters are merged, and only the start and end are stored. In the decoding, the merged information is recovered. For the recognition of clusters to recover, a special bit in the actual cluster data needs to be evaluated. This is done in the pre-decoding. The data is also sorted here. This is needed as the data alignment direction is different for every second layer. In the last step, the actual transformation of the cluster channel identifier into x, y, z coordinates takes place. The values are written to a predefined memory location. Due to the sorting in the pre-decoder, the hits are sorted. Within the SciFi

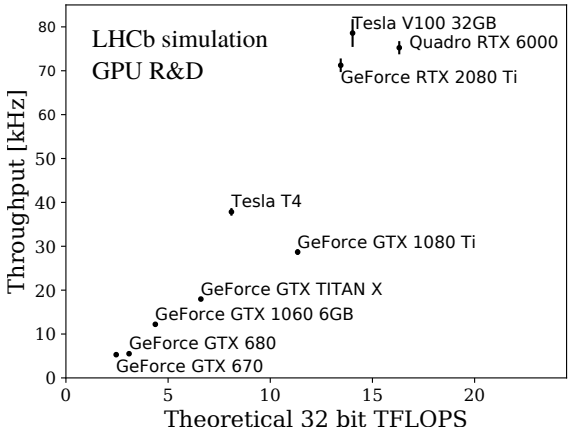


Fig. 4.9: Allen sequence throughput against theoretical GPU performance [11].

tracking, the found VELO tracks are extended. Hereby the momentum of the charged particles is taken into account as the SciFi is placed behind the bending magnet. The muon stations are the only particle identification system that is evaluated in the HLT1. This is due to the special interest in muons in the physics field of LHCb. The muon data is decoded, and the probability for particles to be a muon is calculated. All generated information is used to find secondary vertices. In the end several selection lines are applied, and the selected events are copied back to the host CPU and stored in the buffer until HLT2 starts. The key property of the HLT1 is its throughput of events, as the computing time is limited. The throughput of the Allen HLT1 application is tested for different GPUs. Figure 4.9 shows the theoretical GPU performance in 32-bit TFLOPS against the achieved HLT1 throughput. It can be seen that the throughput scales well with the theoretical performance of a GPU, and a significant performance growth is expected with future GPU generations.

The throughput values shown in Figure 4.9 are average numbers, but due to the random nature of particle processes, the number of hits in the detector varies significantly. Most of the computations within the trigger are combinatorial, and therefore the processing time per event strongly depends on the event size. Figure 4.10 shows the event throughput in bins of the SciFi raw data size.

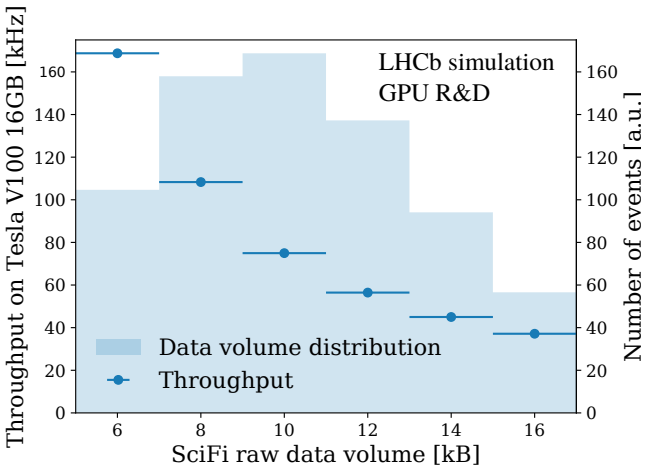


Fig. 4.10: Allen sequence throughput in bins of the SciFi Raw bank size [11].

5 Monte Carlo Simulations

5.1 LHCb: Monte Carlo Simulations and Libraries in Particle Physics

Bernhard Spaan
Hans Dembinski
Gerwin Meier

The LHCb simulation is a software chain that utilizes several specialized software packages. The proton-proton collisions are generated with PYTHIA [353, 354]. The hadronic particle decays are simulated with EVTGEN[239], and the radiation of photons with PHOTOS [133, 180]. The interaction of particles with the detector material is simulated with GEANT4 [40, 50]. Finally, the internal response of the electronics in the detector is simulated with the software BOOLE [128]. After this point, a simulated event is processed in the same way as a real event. The LHCb data acquisition system is described in Section 4.2.1. In the following, the individual steps are discussed further.

The task of PYTHIA is to simulate the high-energy proton-proton collision and the resulting particles as accurately as possible. Like any program in the simulation chain, it uses the Monte Carlo method, which generally speaking breaks a complex process into small randomized processes occurring in succession. In a proton-proton collision, the simulation starts with one or several hard scatterings of the quarks and gluons from the colliding protons. At the beginning, the partons (parton is an umbrella term for quark or gluon) have high energy. Some of this energy is radiated as gluons, and some gluons convert into quark-antiquark pairs. These processes cascade and form a parton shower. The parton-parton interaction and interaction with the other protons in the beam, which have not collided, is also considered. When energy per quarks produced in the parton shower drops below a certain threshold, the shower ends and the quarks bind to form color-neutral hadrons, as mandated by the confinement property of the strong interaction. This hadronization is a complex process for which several effective algorithms exist. To get the properties of the hadrons right, local parton-hadron duality is used, which implies that the flow of the momentum and quantum numbers of the hadrons is mainly the same as the corresponding partons. For more details, we refer the reader to the Lund string fragmentation model [54].

Most of the produced hadrons are unstable. Hadron decay is simulated by EVTGEN. The program aims to accurately describe all underlying physics for the decay that are relevant for an experiment designed to study flavor physics, such as angular and time-dependencies, *CP*-violation, interactions with the quantum-vacuum, and the mixing of neutral particles into their antiparticles. The program makes sure that the momenta and spins of the decay products are properly correlated. A correct simulation of the correlations is important, since they are exploited by certain physics analyses, e. g. for

https://doi.org/10.1515/9783110785968-005

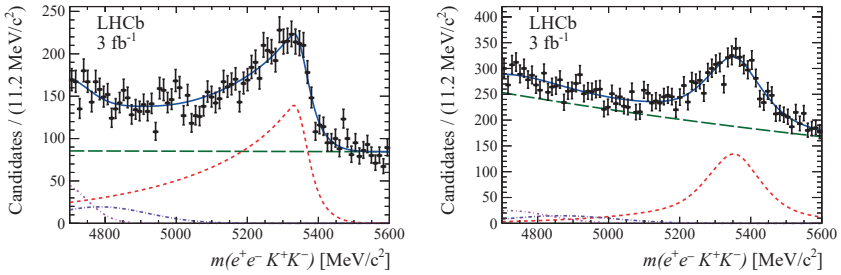


Fig. 5.1: Distribution of $m(e^+e^-K^+K^-)$ for selected $B_s^0 \to J/\psi\phi$ candidates with (left) zero and (right) both electrons with bremsstrahlung correction. The blue solid line shows the total fit composed of the signal (red short-dashed line) and the background contributions. The combinatorial background is indicated by the green long-dashed line while the partially reconstructed background from the $B_s^0 \to \psi(2S)\phi$ and $B_s^0 \to \chi_{c1}(1P)\phi$ decays is indicated by pink and purple dash-dotted lines, respectively. [13].

so-called flavor tagging, an algorithm to determine whether the mother hadron is a particle or antiparticle. For more details, we refer the reader to Ref. [239].

The next step is to simulate radiation of photons by charged particles with PHOTOS. The program is responsible for simulating the energy and direction of emitted photons, as well as the change in momentum and energy of the emitting particle. Bremsstrahlung is important for several LHCb analyses and thus an accurate simulation is required. The LHCb trigger use of photons is detected in the calorimeter. Furthermore, the bremsstrahlung in production and decay of particles and resonances can change the shape of measured distributions. Moreover, the LHCb reconstruction includes an algorithm to combine photons from bremsstrahlung with their electron emitters, which approximately restores the energy and momentum of the electron before emission. This reduces the biases in the reconstructed mass of the ancestor particle from the energy and momentum of its children, as shown in Figure 5.1.

Simulating the propagation and interaction of all particles with the detector material is the next step in the chain, which is done with GEANT4. The geometry of the whole detector needs to be modeled and the various materials in the different subdetectors. This includes not only the active material that is used to detect particles, but also passive support structures, electronics, cables—every bit of matter with which the particles can interact. The effect of external electromagnetic fields, in particular the LHCb magnet, are also accounted for. These processes can result in losses when a particle gets absorbed, but also in newly created particles such as when a photon converts into an electron-position pair in the electric field of a nucleus. In the end, these interactions result in hits in the different sub-detectors.

Following that, the detector response is simulated with BOOLE. The program uses data from the previous step (mainly energy deposits in the active material) to simulate the detector response to these inputs. It also links recorded signals to simulated deposits.

This link at the lowest level is used to match reconstructed tracks to the original particle in the simulation, as described in Section 4.3.1. Different sources of background need to be simulated as well, such as spill-over from previous events or noise from electronic fluctuations, long-term radiation or cross-talk, and the possibility of turning a sensitive volume radioactive. Since the LHCb sub-detectors are independent, the simulation of each sub-detector can be run in parallel.

At this point, the simulation of an LHCb event is complete. The following steps are applied identically to simulated and real events. The LHCb DAQ is described in Section 4.2.1.

5.1.1 Astro: Monte Carlo Simulations, Libraries

Jean-Marco Alameddine

Abstract: In astroparticle physics, the cascading sequence of all interactions from the outer edge of the atmosphere to the signal in the detection electronics must be described with Monte Carlo simulations. The first particles that trigger such cascades can be high-energy photons, electrons, neutrinos, or the atomic nuclei of cosmic rays. Depending on the question under consideration, any of these particles or any of its derivatives can be the signal to be analyzed. What is signal and what is background thus depends on the physical question. Thus, these signals have to be simulated physically completely from theory. Because of the changing measurement conditions with the weather and, for Cherenkov telescopes, also with the zenith angle, especially the simulation of the interactions in the atmosphere requires many resources. Resource consumption can be minimized by intelligently combining computer science methods and simulation. In this section, the applied and developed simulations programs are presented and the linkages of machine learning and Monte Carlo simulations developed in the CRC are motivated.

In astroparticle physics, particles are produced in faraway galactic or extragalactic sources, which are inaccessible to direct observation and have a priori unknown properties. They reach the Earth with energies inaccessible to any currently operated (and, in fact, technically feasible) man-made particle accelerator. Therefore, simulations play a crucial role in the field of astroparticle physics. Astroparticle physicists need to rely on these simulations to be complete and accurate in their description of both signal and background events in order to perform meaningful analyses. However, as described in this section, the challenges physicists face, especially regarding resource management and efficiency, are legion.

As the Earth's atmosphere is opaque to high-energy photons and cosmic rays, they interact with nuclei in the air, creating a cascade of secondary particles called an extensive air shower. While the principles of the underlying physics are, of course, identical to those used to describe the interactions in particle physics experiments, there are important differences that need to be considered. First, interactions in air showers are generally strongly boosted into the forward direction, while the highest-energetic particle physics experiments operate and observe in a head-to-head collision configuration, as in the Large Hadron Collider. This also means that the region of interest for astroparticle physics can't be easily measured with accelerator experiments as particles in the forward direction are lost in the beam pipe. Second, the energies involved can surpass the energies reached with collider experiments by several orders of magnitude, which means that reasonably motivated extrapolations of physics to higher energies need to be made.

In principle, each physical interaction of this cascade needs to be sampled individually to create a complete simulation of an air shower. This idea is implemented by the air-shower simulation tool CORSIKA, which is the most commonly used simulation framework for this purpose and has been utilized by many experiments in astroparticle physics for over 30 years now. The description of the underlying physics is mostly provided by external tools, and can be divided into two different components: The first is the electromagnetic component, which describes the interactions of electrons, positrons, and photons in the particle shower. This component can be described either by a modified version of EGS4, providing direct Monte Carlo simulations, or by the NKG formula, which provides an analytical approach to describe the electromagnetic component. The second is the description of the interactions of hadronic particles, which consists of low-energetic and a high-energetic components. Low-energetic hadronic interactions can be described by the FLUKA, GHEISHA, or UrQMD model, while the high-energetic hadronic interactions can be described by the DPMJET, HDPM, QGSJET01, QGSJET II-04, SIBYLL, VENUS, NEXUS, or EPOS LHC model [194]. Using different models to describe hadronic interactions can yield significantly different results for the shower simulations. This is the effect of different models providing different extrapolations of results from collider experiments into energy and rapidity regimes that are currently technically inaccessible with particle physics experiments. This shows an important connection between challenges in air-shower physics and particle physics, which is highlighted even more in what is known as the muon puzzle: the highly-significant discrepancy of muon numbers measured in extensive air-shower experiments compared with air-shower simulations. This muon puzzle and its implications on particle and astroparticle physics are described in detail in Section 5.4.

Since every interaction in an extensive air shower needs to be sampled individually, their complete simulation is a highly runtime-intensive task. In addition, this random nature makes it possible that even showers with identical initial conditions can have different developments and therefore look entirely different. This means that an enormous number of showers must be simulated to obtain a statistically complete

description, especially if a full-sky description of cosmic rays is needed. In the context of CRC 876, there has been a major effort to find ways to improve the efficiency of the simulations. One possible approach is to limit the time spent on the simulation of showers that are not going to be important for a specific purpose because, say, they are not hitting the detector. For this, it is possible to train models, determining at which point the simulation of a shower, or part of a shower, can be terminated early. This strategy is described in Section 5.2.1. When simulating sets of air showers for classification tasks such as for signal-background separations, the distribution of the underlying initial states can be chosen freely. However, to optimize the necessary computational resources, it is desirable to find a distribution that is optimal for the training of the underlying classification models. An approach to solve this problem called Active Class Selection, which uses machine learning techniques, is described in Section 5.2.2. While the above-mentioned approaches focus on finding improvements on the software side, it is also possible to find potential improvements in the underlying hardware. This is especially important as the electricity costs are a major limitation when it comes to computationally expensive tasks. For this, improvements in resource efficiency by using low-power hardware in CORSIKA simulations are investigated and presented in Section 5.2.4.

As the improvements of the software framework CORSIKA become more and more advanced, the underlying code base, parts were originally developed 30 years ago, reaches its limits. The development of CORSIKA 8 represents the necessary transition to a modern codebase. The aim of this process is to intrinsically provide modern features such as hardware acceleration or parallelization, as well as to create an easier way for developers to implement new features by providing a modular, extensible code structure. This transition, which can be seen as an exemplary effort to modernize scientific software, is described in Section 5.2.3.2.

While in particle physics it is often possible to control the environmental conditions of the experiments to a very high extent, this control can be difficult, if not impossible, for experiments in astroparticle physics. This holds especially true for experiments using the atmosphere as a detection medium, as in IACTs or fluorescence detectors. These experiments are highly dependent on effects such as temperature, exposure to background light, or local magnetic fields. Not taking these effects into account for simulations can lead to significant biases, but recreating full simulations for each possible environmental condition is too runtime-extensive to be feasible. Instead, it is possible to adapt the existing simulation results to the changing environmental conditions by, say, adding noise from the night sky background to noiseless simulations of IACT images. This approach has been developed within this CRC and is described in Section 5.3.2.

A special aspect of astroparticle physics are experiments located underground, such as the Neutrino Observatory IceCube. For these experiments, particles on the Earth's surface—such as muons from air showers—or particles created within the vicinity of the detector—such as particles from weakly interacting neutrinos—need to be propagated

until they interact inside the detector. In this context, muons play a special role, as they can travel large distances through dense media, and are therefore still abundant in underground detectors. Their simulation is therefore crucial, especially as a background that needs to be taken into account. As the number of muon interactions in dense media can be very high, the requirements for muon simulations are both a high precision and a high performance. One tool for this purpose is the software simulation framework PROPOSAL, which provides up-to-date parametrizations of muon physics, as well as an algorithm that allows the finding of a trade-off between speed and accuracy. PROPOSAL is described in more detail in Section 5.2.3.2.

In addition to the above-mentioned simulation tools—frameworks that are used by many different experiments in astroparticle physics—many experiments use their own, often detector-specific, simulation tools. These tools are in particular used to describe very low-level parts of the detector such as the response functions of specific detector parts.

Especially for highly complex or newly installed detectors, much effort needs to be made to get an acceptable level of agreement between real, measured data and simulated events. This data/MC agreement is important since otherwise, machine learners might be trained on features that can only be seen on simulated data, but not in real data. Since calibration measurements in astroparticle physics experiments are, in most of the cases, only possible to a limited degree, it is necessary to directly compare the observed and simulated features. However, simply comparing one-dimensional features is not always sufficient as mismatches might only be visible in high-dimensional correlations. It is thus worthwhile that methods of machine learning be used to find data/MC mismatches by, say, analyzing models that are trained to distinguish between real and simulated events. This approach is described in Section 5.3.3.1, where it has been applied to IceCube simulations.

5.2 Simulation Efficiency Studies

5.2.1 Corsika – Active Learning

Dominik Baack

Abstract: The simulations of high-energy cosmic rays hitting the atmosphere are a requirement for many astroparticle experiments. For most experiments, those simulations consist of tracking the individual incoming particles through the atmosphere, including random effects such as interaction, which can create new particles. Those must be handled in the same way, creating an expanding cascade of particles that must be propagated through the atmosphere. With the high complexity of those Monte

Carlo simulations, the computing time of real-world examples can range from several CPU seconds up to several CPU months for each individual shower, depending on specific settings. For a complete set with a statistically significant number of events, the cumulative computing time is even higher.

Due to the inherent random nature of Monte Carlo simulation, the initial particle's fixed constraints must be kept wide to allow the simulation of boundary cases. This leads to the effect that a large number of simulated particles or even whole events are not triggered inside the experiment and therefore discarded. Looking several steps later in the simulation and analysis pipeline, the training of classifiers or unfolding methods can favor specific events that rarely occur, unlike the previous ones.

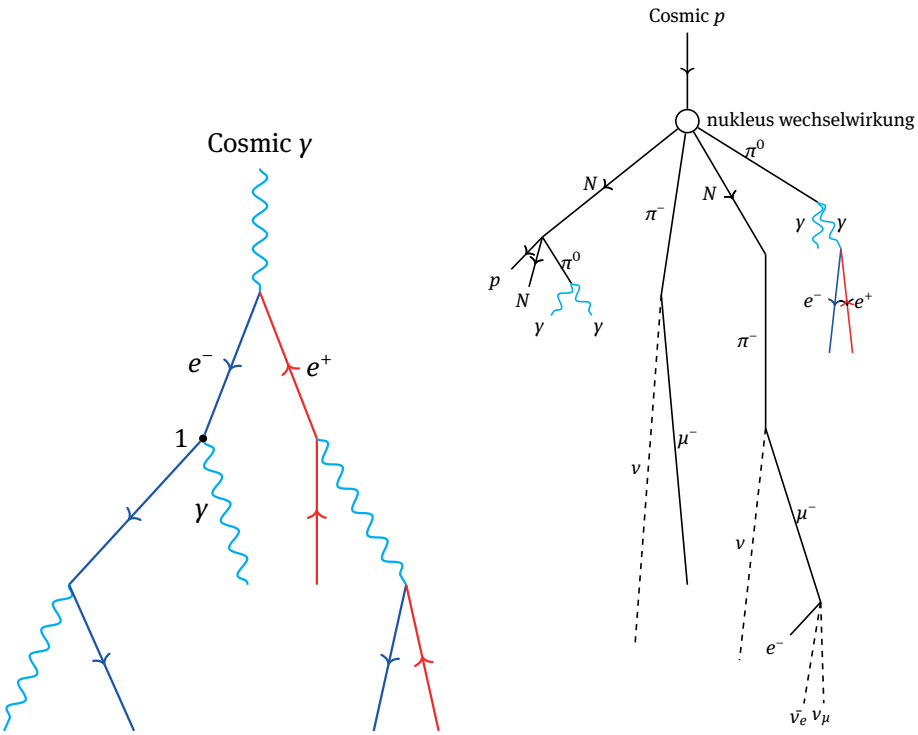
A learning algorithm executed after batches of the simulation makes it possible to train a model to reduce the simulation phase space, which is to say, the number of simulated but not triggered events. In an extension, not only can whole cascades be stopped early; individual parts of the generated shower, called a sub-shower, can also be checked against the model.

5.2.1.1 Introduction - Cascades

Astroparticle physics is a relatively new field of research based on particles with interstellar origins. This covers a wide field of energy from several eV up to several PeV. In the higher energy regime starting around GeV, individual particles are far too rare to observe them directly with comparable small satellite experiments. The difficulty of earth-bound detectors is that cosmic rays do not pass through the atmosphere but rather interact with it. At each interaction point, the energy is transferred to several more particles, which themselves interact in the atmosphere. This leads to a cascade of particles as displayed in Figure 5.2.

5.2.1.2 Introduction - Experiments

With the large variety inside the cascade, several different experimental detection methods are available. Secondary particles can be measured directly or through their emission of electromagnetic waves in the form of fluorescence light, Cherenkov light, or radio emission. The methods of optimizations described later are tailored for experiments where the size of the shower (see Figure 5.3) is an order of magnitude larger than the instrumented area. This condition can be met by Cherenkov telescopes such as MAGIC, CTA, or FACT and smaller water detectors such as HAWK. Furthermore, there are not only experiments, like the already mentioned ones, which observe and measure showers directly. For a much larger number of experiments, the resulting particles do form a disturbing background in their actual measurements. One example is neutrino telescopes. In both cases, a complete simulation is necessary to provide a solid foundation for the subsequent analyses.

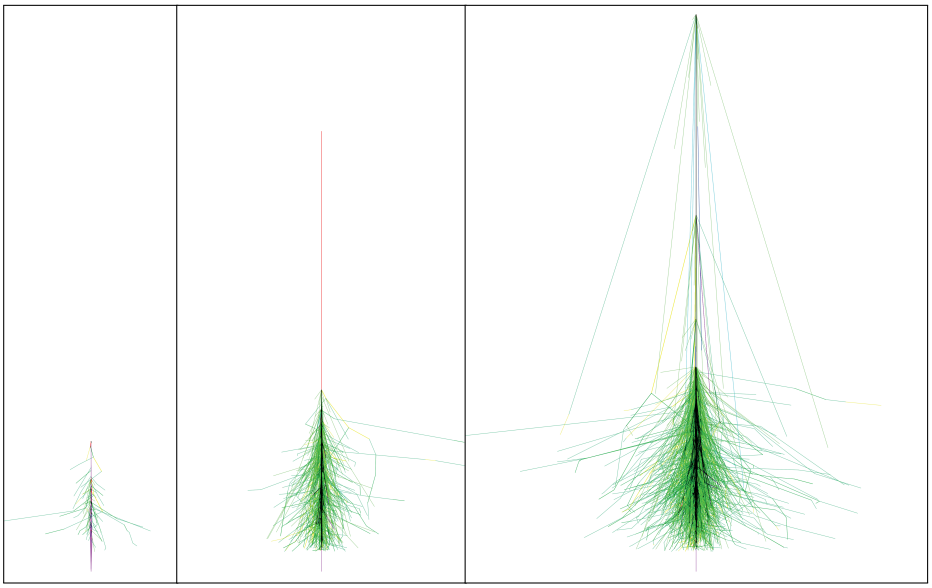


(a) Electromagnetic **(b)** Hadronic

Fig. 5.2: Schematic representation of an electromagnetic and hadronic cascade inside the atmosphere. A realistic representation of the shower is displayed in Figure 5.3.

5.2.1.3 Introduction - Simulations

The signal of interest for the physical analysis is in almost all experiments of this kind superimposed by several orders of magnitude by background signals. In order to be able to draw conclusions from measurements, a classification of the signal events is necessary. Due to the complexity and quantity of the signals, this can only be done automatically on a reasonable scale. For training these automatic classifiers, datasets are required where all input variables are already known. However, since these cannot be determined in the experiment, we can only predict individual events based on theoretical considerations. Due to the large number of statistical processes underlying each interaction within particle showers and other non-linear effects in the experiments, it is impossible to give a closed-form analytical solution to predict the distributions of observables as a function of the input variables. Therefore simulations were developed that describe individual interactions via Monte Carlo methods. Each interaction is sampled from random distributions derived from physical measurements or theoretical results. With this step-by-step process, all particles are tracked, and physical effects



(a) Gamma **(b)** Proton **(c)** Iron

Fig. 5.3: The cascade with different primary particles at an energy of 10 TeV. The shower is displayed with a orthographic projection, which means that the height/width of the image corresponds to a real height of 64 516 m and an accumulated width of 32 000 m.

are applied. The required computing time is equally large given the high number of particles and complexity. It is possible to propagate averages or simplified distributions in the form of cascade equations from which individual parameters can be sampled at the observation level. This enables a faster calculation time, but the simulation can not be utilized for an experiment where observable parameters are generated during the propagation, as with radio emission or Cherenkov light.

5.2.1.4 Cascade Cutting Strategies

In order to reduce the computational time required for each cascade, the number of physical effects applied could be reduced, but this would also reduce the realism of the simulation and, consequently, the possible results. Another simple-sounding method is to reduce the required calculations in the cascade itself. Until now, the air-shower simulation, the simulation of the properties of the experiment such as the optics, and the construction of the observable parameter vectors for the subsequent calculation have been treated entirely separately. Therefore, knowledge about the detector or analysis properties could not be applied to the simulation of the showers itself. Consequently, the complete air shower always had to be simulated.

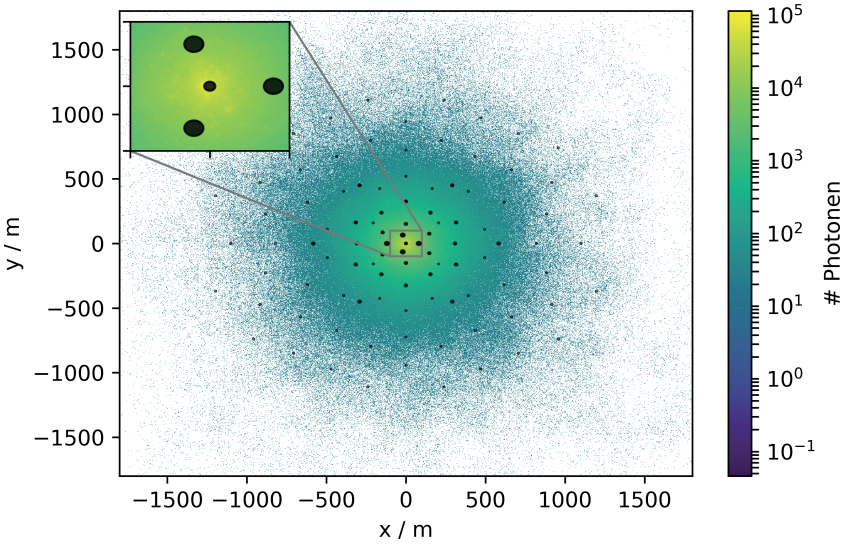


Fig. 5.4: A possible telescope configuration for CTA South with the real telescope size in gray and the photon distribution for a 1 TeV shower.

With this new approach, information such as, which photon contributed to the reconstruction, can be utilized to train a specialized machine learning process to enable the early removal of cascade parts that do not contribute to the later stages of the simulation or provide any information-gain to automatic classifiers.

5.2.2 Control of the Simulation – Active Sampling

Mirko Bunse

Abstract: Simulations in astroparticle physics are label-dependent data generators; they produce the observed features depending on the type of the primary particle, its energy, direction, and other latent quantities to be predicted. When generating training data through such a simulation, we are therefore free to choose the distribution of labels in the resulting training set. In fact, we *have* to choose this distribution prior to the data production. For classification problems, such as signal-versus-background separation, this problem is known as active class selection. We survey existing heuristics for choosing the proportions of classes in this setting and we present a recent strategy

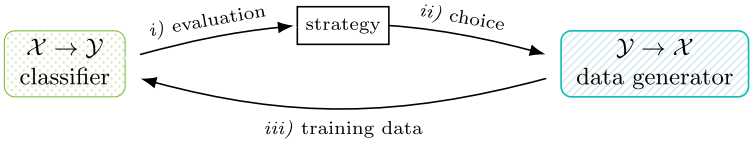


Fig. 5.5: Strategies for active class selection optimize class-conditioned data acquisition [103].

for this choice. This strategy fully accounts for uncertainties about the class proportions, e.g., unknown signal-to-noise ratio, and it is justified in PAC learning theory. Hence, our strategy is a promising candidate for controlling Monte Carlo simulations in terms of the class distributions they generate.

5.2.2.1 Introduction

Active Class Selection (ACS) [254] allows machine learning practitioners to actively choose the label proportions of their training data. This freedom of choice is due to a *class-conditional* data generator, e.g., an experiment or a simulation, which acquires feature vectors for arbitrarily chosen classes. In particle and astroparticle physics, we must define an initial particle for which the response of a particle detector is simulated. This input is what we later need to predict from the detector response [88].

Lomasky et al. [254] have put forward the idea that such a generator can be leveraged in a sequence of multiple acquisition steps, as sketched in Figure 5.5: (i) in each step, a classifier is trained and evaluated on all examples that have been acquired so far, starting from a small initial dataset; (ii) based on the classifier's performance, a data acquisition *strategy* is then allowed to choose the label proportions of the next acquisition step; (iii) the desired proportions are realized by a generator, e.g., a simulation of a particle or astroparticle detector, which produces a batch of labeled data according to the choice of the strategy. This batch adds to the training set from which the classifier will be trained in all subsequent iterations. The promise of such a sequential and informed data acquisition is that the classifier can benefit in terms of data acquisition cost and performance, as opposed to being trained with some predetermined proportions of classes. The ACS framework is considerably different from classical active learning [346], which assumes that unlabeled data can be labeled after its acquisition by, say, a human labeler.

5.2.2.2 Strategies for Class-Dependent Data Acquisition

Lomasky et al. [254] introduced five heuristics for the class-dependent data acquisition in their seminal work on ACS. These heuristics are based on the inverse relative utility of the current prediction model $h : \mathcal{X} \to \mathbb{R}^{|\mathcal{Y}|}$ for each class $y \in \mathcal{Y}$. Namely, each class y

is sampled according to the weight.

$$w(y) = \frac{u(y)}{\sum_{y' \subseteq \mathcal{Y}} u(y')},$$

where $u : \mathcal{Y} \to \mathbb{R}$ is a utility heuristic that might or might not depend on the current prediction model h.

Uniform Assign the same utility $u(y) = 1$ to all classes.

Natural Sample with the class proportions that also occur during deployment, i.e. $w(y) = u(y) = \mathbb{P}(Y = y)$. While the natural strategy usually performs highly competitively in terms of accuracy, it is sometimes beaten by one of the other heuristics.

Inverse Sample by the inverse class-wise accuracy, $u(y) = Acc_h(y)^{-1}$. This strategy is motivated by the assumption that a low accuracy is exclusively caused by an insufficient amount of training data.

Improvement Like the inverse method, but based on the improvement of accuracy during the last iteration $u(y) = (Acc_h(y) - lastAcc_h(y))^{-1}$. This strategy assumes that classes with a "stable" accuracy will remain stable in future iterations.

Redistricting Count the number of examples $u(y) = n_h^{(y)}$ for which the prediction changed during the last iteration. The idea behind this strategy is that examples close to the decision boundary may be more informative than others. Classes for which predictions frequently change are assumed to contain many examples close to that boundary.

Several follow-up studies [113, 204, 299, 391] have applied these heuristics in different settings. One additional heuristic has been proposed as "PAL-ACS" [232], which embeds a classical active learning strategy into ACS. The idea is to evaluate a set of pseudo instances $\hat{D}$, over which the class-wise utilities are then aggregated from an active learning utility measure $u_{\text{AL}} : \mathcal{X} \to \mathbb{R}$. This approach requires a generative model to produce the pseudo instances and an active learning utility function like entropy.

$$u_{\text{PAL-ACS}}(y) = \frac{1}{|\hat{D}_y|} \sum_{\vec{x} \in \hat{D}_y} u_{\text{AL}}(\vec{x})$$

Quite often, however, these sophisticated ACS strategies are not able to outperform the *natural* strategy, which does not even take into account the model performance as proposed by the ACS framework from Figure 5.5 [232, 254]. In fact, we have theoretically assessed that the natural class proportions yield optimal classifier performance in the limit of data acquisition [106]. These findings show that sophisticated strategies are losing their potential benefit over the natural class proportions during the acquisition process.

5.2.2.3 Certification

The existing ACS strategies [232, 254], except for the *natural* strategy, do not account for the class proportions that a trained model needs to handle during deployment; they

solely focus on the perceived *difficulty* of classes. The *natural* strategy, however, requires the practitioner to know the deployment class proportions precisely in advance. What if we know the deployment class proportions not precisely, but with some degree of uncertainty? For instance, astroparticle physicists can estimate the ratio between their signal and their background class only roughly, as being approximately $1 : 10^3$ or even $1 : 10^4$ [88].

One way to tackle these uncertainties is through the *certification* of ACS-trained models [104]. To this end, we certify the set of class proportions $\mathcal{P}$ to which a fixed hypothesis h, trained on the ACS-induced source distribution $\mathcal{S}$, is safely applicable. By "safely", we mean that during the deployment on the target distribution $\mathcal{T}$, h induces only a small ACS-induced error with a high probability.

Definition 5 (Certified hypothesis [104])**.** *A hypothesis $h \in \mathcal{H}$ is (ε, δ)-certified for all class proportions in the set $\mathcal{P} \subseteq [0, 1]^{|\mathcal{Y}|}$ if with probability at least $1 - \delta$ and $\varepsilon, \delta > 0$:*

$$|R_{\mathcal{T}}(h) - R_{\mathcal{S}}(h)| \leq \varepsilon \quad \forall\, \vec{p}_{\mathcal{T}} \in \mathcal{P},$$

where $R_{\mathcal{D}}(h) = \int_{\mathcal{X}\times\mathcal{Y}} \mathbb{P}_{\mathcal{D}}(\vec{x}, y)\ell(y, h(\vec{x}))\,\mathrm{d}\vec{x}\,\mathrm{d}y$ is the expected risks of h, as according to the distribution $\mathcal{D}$ and the loss function ℓ.

In binary classification, $\mathcal{P}$ is simply a range $[p_{\mathcal{T}}^{\min}, p_{\mathcal{T}}^{\max}]$ of class proportions. This range is defined by the largest $\Delta p^{\star}$ for which

$$\Delta p \cdot \Delta\ell \leq \varepsilon \quad \forall\, \Delta p \leq \Delta p^{\star}, \tag{5.1}$$

where $\Delta p = |p_{\mathcal{T}} - p_{\mathcal{S}}|$ denotes the difference between class proportions and $\Delta\ell = |\ell_{Y=2}(h) - \ell_{Y=1}(h)|$ denotes the difference between class-wise losses. The latter of these terms is constant for both distributions $\mathcal{S}$ and $\mathcal{T}$. In turn, $\Delta p \cdot \Delta\ell$ is constant with respect to the random draw of the training set D and is, therefore, independent of ε, δ, and N. It reflects the interplay between the classifier h, the data distribution, and the loss function.

Here, the true difference $\Delta\ell$ is unknown, but we can estimate an upper bound $\Delta\ell^{\star}$ of this quantity from ACS-generated data. We have proposed a PAC bound [104] that is the smallest upper bound of $\Delta\ell$ that holds with a probability of at least $1 - \delta$. The probabilistic nature of this upper bound stems from the fact that $\Delta\ell^{\star}$ is estimated from finite amounts of data.

5.2.2.4 A Strategy For Uncertain Class Proportions

A certificate can help practitioners in assessing the practical value of an ACS-trained model. However, it has no immediate implication on how to acquire data—in terms of an ACS strategy—when the deployment class proportions are uncertain. Therefore, we have proposed an ACS strategy [103] that aims at decreasing the inter-domain gap $\Delta p \cdot \Delta\ell$ from Equation 5.1 as much as possible, as according to a prior distribution $\hat{\mathbb{P}}$

of the deployment class proportions $p_{\mathcal{T}}$. This goal will allow any binary classification algorithm to learn accurate predictions for the target domain, as according to the prior beliefs of a domain expert. In astroparticle physics, for instance, this belief could be a prior distribution about the proportions of gamma and hadronic particles.

Formally, we assume a prior $\hat{\mathbb{P}} : [0, 1] \to [0, 1]$ of the positive class prevalence $p_{\mathcal{T}} \in [0, 1]$ to be given. We incorporate $\hat{\mathbb{P}}$ by marginalizing the inter-domain gap over this prior according to Equation 5.2. Since we do not know the true $\Delta\ell$, we are using the estimated upper bound $\Delta\ell^\star$ instead. Consequently, the marginalization according to $\Delta\ell^\star$ is an upper bound, with probability $1 - \delta$, of the marginalization according to the true $\Delta\ell$.

$$\varepsilon^\star = \int_0^1 \hat{\mathbb{P}}(p_{\mathcal{T}} = p) \cdot \underbrace{|p_{\mathcal{S}} - p|}_{=\Delta p} \cdot \Delta\ell^\star \; \mathrm{d}\, p \tag{5.2}$$

In each ACS iteration, we are free to alter the class proportions $p_{\mathcal{S}}$ of the ACS-generated training set to some degree, depending on how much data we acquire in each batch and on how much data we already have acquired. In fact, we can understand $p_{\mathcal{S}} = \frac{N_2}{(N_1+N_2)}$ as a function of the class-wise numbers of samples N_1 and N_2. The upper bound $\Delta\ell^\star$ also lends itself for interpretation as a function of sample sizes: the more data is acquired in both classes, the tighter will our estimation of this quantity will be. Ultimately, we consider $\varepsilon^\star$ to be a function of N_1 and N_2, so that we can minimize $\varepsilon^\star$ via an optimal choice of N_1 and N_2 in each data acquisition batch.

Our strategy decreases $\varepsilon^\star$ in the direction of its steepest descent, i.e., it takes a simple gradient step with respect to the acquisition vector $\vec{N} = (N_1, N_2)$. The gradient that defines the steepest descent is computed via the product rule:

$$\begin{aligned} \nabla_{\vec{N}}\, \varepsilon^\star &= \nabla_{\vec{N}} f \cdot \Delta\ell^\star + f \cdot \nabla_{\vec{N}} \Delta\ell^\star \\ \text{where } f(\vec{N}) &= \int_0^1 \hat{\mathbb{P}}(p_{\mathcal{T}} = p) \cdot |p_{\mathcal{S}}(\vec{N}) - p| \; \mathrm{d}\, p \end{aligned} \tag{5.3}$$

We will come back to the function f shortly. For now, we plug $\Delta\ell^\star$ and $\nabla_{\vec{N}}\Delta\ell^\star$ into the equation above. These functions are defined by

$$\begin{aligned} \Delta\ell^\star(\vec{N}) &= \hat{\ell}_{Y=2}(h) + \sqrt{\frac{\ln \delta_2}{-2N_2}} - \hat{\ell}_{Y=1}(h) + \sqrt{\frac{\ln \delta_1}{-2N_1}}, \\ [\nabla_{\vec{N}} \Delta\ell^\star]_y &= \left(-\frac{\ln \delta_y}{N_y}\right)^{\frac{3}{2}} \cdot (2\sqrt{2} \ln \delta_y)^{-1}, \end{aligned} \tag{5.4}$$

where the δ_y are probabilities of violations of $\Delta\ell^\star$ that occur from either one of the class-wise losses $\ell_{Y=y}(h)$ in $\Delta\ell$. In fact, finding a suitable assignment of δ_y values within a given probability budget $\delta = \delta_1 + \delta_2 - \delta_1\delta_2$ is the central difficulty in ACS model certification; there, the sample size $\vec{N}$ is fixed, so that $\Delta\ell^\star$ can be optimized over this assignment [104]. Here, we keep the δ_y fixed instead of values that are obtained with a

certificate from previous ACS acquisitions. This change allows us to optimize $\Delta\ell^\star$ over $\vec{N}$ to acquire new data and guarantees that $\Delta\ell^\star$ remains an upper bound of the true $\Delta\ell$ also in the next batch, at least with probability $1 - \delta$. The class-wise estimates $\hat{\ell}_{Y=y}(h)$ in Equation 5.4 are the average values of losses in the training data; they are also part of our certificate.

Plugging a Beta(α, β) prior into the f function from Equation 5.3 yields the following components, where I is the regularized incomplete Beta function:

$$\begin{aligned} f_{\alpha,\beta}(\vec{N}) &= \frac{2p_{\mathbb{S}}(\vec{N})^{\alpha}(1-p_{\mathbb{S}}(\vec{N}))^{\beta}}{(\alpha+\beta)B(\alpha,\beta)} + \left(p_{\mathbb{S}}(\vec{N}) - \frac{\alpha}{\alpha+\beta}\right)\left(2I_{p_{\mathbb{S}}(\vec{N})}(\alpha,\beta) - 1\right) \\ \nabla_{\vec{N}} f_{\alpha,\beta} &= \frac{2I_{p_{\mathbb{S}}(\vec{N})}(\alpha,\beta) - 1}{(N_1+N_2)^2} \cdot \begin{pmatrix} N_2 \\ -N_1 \end{pmatrix} \end{aligned} \tag{5.5}$$

Plugging Equation 5.4 and 5.5 into Equation 5.3 provides us with a gradient that we can compute analytically from a certificate with a δ_y assignment, from sample sizes N_1 and N_2, and from the prior parameters α and β. The negative gradient $-\nabla_{\vec{N}}\,\varepsilon^\star$ of the marginalized error $\varepsilon^\star$ defines the class-wise numbers of samples that our strategy acquires in the next data acquisition batch.

5.2.2.5 Synopsis

With small data volumes or with highly imbalanced classes, our ACS strategy is dominated by the $\Delta\ell^\star$ component; small classes need additional data until this upper bound holds with some desired probability $1 - \delta$. By contrast when the total data volume is large, our strategy is dominated by the f component. To this end, a Beta prior favors class proportions that are close to its mean $\frac{\alpha}{\alpha+\beta}$. The turning point between these two behaviors is well founded in the PAC learning theory that underlies the estimation of $\Delta\ell^\star$.

Due to these properties, our ACS strategy is a promising candidate for controlling physical simulations in terms of the class distributions they generate. Instead of simulating signal events and background events in fixed proportions, our strategy has the potential to improve signal-versus-background classification performance through optimized class proportions that are justified in PAC learning theory. Uncertainties of the practitioner, regarding the signal-to-background ratio in the real world, are fully accounted for through a Beta prior.

5.2.3 Corsika 8 New Modular Library

Dominik Baack
Jean-Marco Alameddine

Abstract: The classification and reconstruction algorithms in astroparticle physics depend on an accurate description of simulated events, as calibration measurements with extragalactic sources are not feasible. Due to the stochastic behavior of the particles during their propagation, huge amounts of Monte Carlo-based simulations are required to cover most parts of the phase space and describe the measured event signatures.

For air-shower events, the simulation framework CORSIKA has become the most used tool. The transition of CORSIKA to a modern and modular framework allows for further optimization methods such as parallelization, GPU programming, and more usage of external physical interaction libraries. Among these new external libraries is PROPOSAL, a simulation library used in neutrino telescopes to propagate leptons.

In this section, we describe the transition of the simulation library PROPOSAL, developed in Dortmund, from a simple muon propagator to a modular library simulating electromagnetic and weak interactions of high-energy leptons and photons. Furthermore, the development of the CORSIKA framework to use external libraries for core processes is described regarding the use of PROPOSAL for the electromagnetic shower.

5.2.3.1 Monte Carlo Simulations of Extensive Air Showers

Extensive Air Showers are particle cascades initiated by high-energy cosmic rays interacting with the Earth's atmosphere (see Section 5.2.1.1). An accurate understanding of air showers is relevant for all experiments that are either interested in the reconstruction of properties of the primary particles or need to consider them as a background for other observations.

In 1989, the first version of CORSIKA (Cosmic Ray Simulations for KASCADE), a Monte Carlo software to simulate air-shower events, was published [194]. Originally developed to provide simulations for the air shower array KASCADE, CORSIKA has evolved into a tool that is now widely used by physicists worldwide. At its core, the CORSIKA code consists of four components: a physics description of hadronic interactions for low and high particle energies, a physics description of electromagnetic interactions, and a general framework responsible for all other physics and the steering of the program. Over time, CORSIKA has been improved, and its functionalities have been continuously extended. However, new developments are impeded by two main constraints: CORSIKA is written in FORTRAN 77, whose language features are limited compared with modern programming languages. Furthermore, the codebase of CORSIKA has not been developed with today's requirements in mind, which imposes severe

conceptional limitations. This means that the current codebase is hard to maintain, and new developments are challenging to include. With these factors in mind, a white paper stating the requirements of a next-generation CORSIKA was formulated in 2018, which ultimately led to the development of CORSIKA 8 [153].

5.2.3.2 CORSIKA 8: The Future of Air-Shower Simulations

CORSIKA 8 is the newest version of the shower simulation framework CORSIKA. With the development of CORSIKA 8, the code was completely rewritten in modern C++, which provides a future-proof and state-of-the-art framework. Considering the many computational resources spent on creating Monte Carlo simulations, which will also be used for CORSIKA 8, one key focus during the development lay in the intrinsic efficiency of the codebase. This means that runtime-efficient designs such as static polymorphism are used, and techniques such as parallel computations and GPU support are an inherent part of the code.

For the design of CORSIKA 8, the driving idea is to provide a flexible codebase that not only fulfills the current requirements but is also highly adaptable for future developments. Previous versions of CORSIKA were developed solely with the idea of creating a program to simulate extensive air showers, resulting in a very specialized, monolithic code structure. CORSIKA 8, however, is designed to be a general framework for particle cascade simulations, not limited to air shower simulations. This idea of flexibility is also represented in the code structure, which consists of four main parts: the particle stack, storing and providing access to all information about the propagated particles; the process sequence, which is the composition of all modules providing the physical input of the simulation; the environment, which stores all medium and geometrical properties of the environment; and the transport code, which combines all parts of the code and simulates the particle development. Each part of the code is highly modular and can be adapted, extended, or completely exchanged. One relevant example is PROPOSAL, an external library used in CORSIKA 8 to simulate the electromagnetic shower component.

PROPOSAL (PRopagator with Optimal Precision and Optimized Speed for All Leptons) as a scientific software went through several transitions during its development. Originally, PROPOSAL was developed as a tool to propagate high-energy muon and tau leptons, i.e., providing information about energy losses as these particles travel through dense media such as ice. As the development continued, this concept was generalized. PROPOSAL is now also capable of simulating electron, positron, and photon interactions. One major advantage of PROPOSAL is that in its current version, physics parametrizations can easily be switched, and the user can add new parametrizations in a modular way. Furthermore, up-to-date parametrizations of particle interactions are directly provided and are regularly updated.

A restructuring of the code has been made to prepare PROPOSAL for the integration into CORSIKA 8 as an external module. Initially, PROPOSAL provided a single

`Propagator` class, which was responsible for the complete simulation of a single particle. While this class is still available, physical calculations have been separated into individual modules. Each module is responsible for solving an individual task, such as sampling stochastic or continuous energy loss, calculating time steps, or performing particle decays [45]. CORSIKA 8 uses these autonomous modules provided by PROPOSAL to simulate the electromagnetic and muonic components of particle cascades. Thus, PROPOSAL replaces the former treatment of the electromagnetic shower component in CORSIKA, which has been performed by using a modified version of EGS4, a FORTRAN code released in 1985. The clear advantages of PROPOSAL over EGS4 are its modern code structure, as it is written in modern C++, its flexibility, which allows adapting PROPOSAL for specific use cases, and the included up-to-date physics descriptions.

CORSIKA 8 is actively developed as an open-source community project.[1] A first pre-release version of CORSIKA 8 is available, which can be used for first tests and validations [52].

5.2.4 ARM-Cluster for Corsika

Jens Teubner
Thomas Lindemann

Abstract: For several years now, *power* has become the limiting factor in virtually every computing system. *Electricity cost* has become an economic issue in modern data centers; enormous efforts have to be invested in dissipating *excessive heat*; and—often overlooked—all modern microprocessor designs are limited in terms of the energy amounts that can be managed within the very small chip sizes [89].

Brawny chip designs, optimized for single-thread performance (Intel's x86 line being the most popular), suffer these limitations even more than lean-core designs (ARM-based chips are well-known representatives). The latter have been designed with a much closer eye on their power consumption, typically at the price of an inferior single-thread performance. As such, lean core designs can be a perfect match for many simulations problems, which typically do not depend on single-thread performance but carry intrinsic parallelism.

As we show in this section, systems built from lean processor cores can indeed outperform their brawny counterparts, which still dominate the market. With *Eriador*, we constructed a massively parallel cluster-in-a-box solution based on ARM processors. For many of the relevant (simulation) problems, *Eriador* provides runtime characteristics

1 https://gitlab.iap.kit.edu/AirShowerPhysics/corsika.

comparable to those of state-of-the-art (brawny) server systems but at a significantly lower energy footprint. This is all the more remarkable since *Eriador* is also cheaper in terms of its raw hardware cost.

5.2.4.1 Introduction

Over several decades, hard- and software advances were driven by the miniaturization of integrated circuits. In line with *Moore's Law*, every hardware generation would double the amount of compute resources, resulting in exponentially growing hardware performance. Even more importantly, that performance growth would come "for free" from an energy perspective: as a consequence of *Dennard scaling*, new hardware generations would still not need more electrical power than their predecessor generation, yielding twice the performance-per-watt ratio.

As time progressed, Dennard scaling came to an end, though, and *energy consumption* has become the problem child of hardware technology. In fact, energy consumption is *the* limiting factor in modern chip designs [89]. Growing energy demands hinder performance advances in two crucial ways. The *monetary cost* of first providing energy and then removing it through cooling, has become a critical cost factor in the design of data centers. Even more importantly, high energy consumption at small chip geometries will result in a massive *heat dissipation*. If the excessive heat cannot be removed through cooling, chips would simply melt.

Hardware manufacturers responded with different strategies to handle the new limitation. Processors primarily used in desktop and server systems are typically optimized for single-thread performance. Toward this end, designers built sophisticated caching and control logic in order to feed a few instruction pipelines in "brawny" processor designs so that they can perform as much work as possible. In the mobile and embedded markets, by contrast, processors have emerged that prioritize energy efficiency. "Lean" processor designs are significantly more energy efficient than their brawny counterparts, but they offer onlylimited single-thread performance.

The favorable energy characteristics of lean processors provide incentives to use them also for applications that are traditionally dominated by server-class, brawny CPUs. At least conceptually, the slow speed of lean processor designs can be compensated by increased *parallelism*.

CORSIKA simulations are a good candidate to follow the route of high parallelism degrees in order to improve the energy efficiency of simulations. Figure 5.6 illustrates the distribution of processing effort spent on individual events. Except for some high-effort events, most events require a relatively short and predictable processing cost. Thereby, individual events can be processed independently of one another, enabling the use of parallelism. Under these premises, we evaluated the energy efficiency gains that can be harvested by replacing the prevalent server-class computing infrastructures (relatively few but brawny processors) with low-power,energy-efficient alternatives (that is, a large count of lean processors instead).

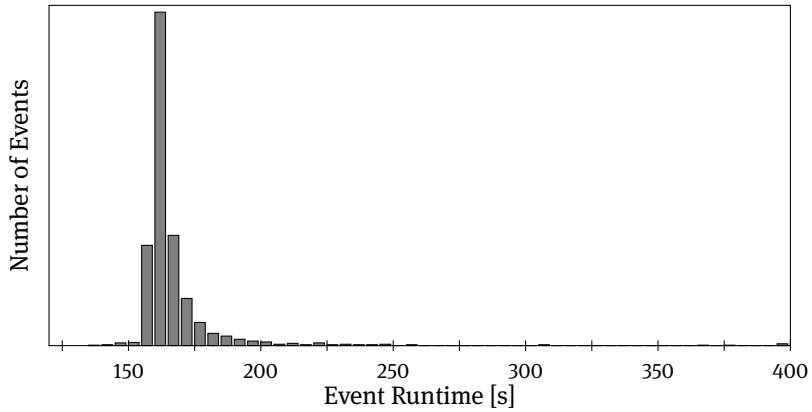


Fig. 5.6: Distribution of CORSIKA FACT event count vs. their processing time (on Intel Xeon).

Specifically, with *Eriador* we propose a platform that can replace a commodity two-socket server system by a cluster of 40 low-energy ARM processors. The low-energy alternative provides the same form factor as the commodity system (two rack units), a much smaller energy envelope, and comparable (sometimes significantly better) application performance when compared with the commodity system.

In the following, we sketch the design decisions behind the *Eriador* platform and demonstrate the energy advantages that can be obtained by running compute-intensive workloads on low-energy hardware.

5.2.4.2 Related Work

The idea of using energy-aware hardware for certain compute-intensive tasks is not entirely new. In 2021, Ou *et al.* built a cluster out of ARM-based PandaBoards [298]. Their system turned out to be highly efficient for web servers, in-memory databases, and video transcoding applications but less for compute-intensive workloads. Göddeke *et al.* [179], too, observed that compute-heavy applications were a weakness of (their) low-power hardware design.

Kruger [234] demonstrated that a combination of an ARM A9 CPU and an Epiphany-based co-processor could match the performance of an Intel i5 at only a third of the Intel processor's power consumption.

5.2.4.3 CPU Architectures for Simulation Tasks

ARM processors are often used as a poster child for energy efficiency. In a student project at TU Dortmund, we evaluated various ARM-based architectures but also low-power offerings with other architectures.

Table 5.1 illustrates several key characteristics for four popular single-board computer offerings (the ARM-based Raspberry Pi 3 probably being the most widely known).

Tab. 5.1: Selected performance and power characteristics of ARM- and Intel Celeron-based single-board computers [184].

Board	RAM	CPU	Network	Power
Odroid-C2	2 GB DDR3	4 × 1.5 GHz ARM A53	1 GBit	2.0–4.2 W
Odroid-XU4	2 GB LPDDR3	4 × 2 GHz ARM A15 4 × 1.4 GHz ARM A7	1 GBit	3.8–10.5 W
Rasp. Pi 3	1 GB LPDDR2	4 × 1.2 GHz ARM A53	100 MBit	1.3–4.2 W
ASRock	4 GB DDR3L	4 × 2.2 GHz Int. Cel. J3160	1 GBit	10.6–16.0 W

As can be seen already in this small selection, performance characteristics (including CPU and network speeds) vary just as much as the average power consumption.

A set of raw performance experiments, primarily based on the *Phoronix Test Suite*, confirmed what Table 5.1 suggests: ARM processors are indeed more energy-efficient than their Intel Celeron counterparts. Given the minimalistic resources of the popular Raspberry Pi and the relatively high power consumption of the Odroid-XU4, we decided to build our own *Eriador* platform based on Odroid-C2 single-board computers.

5.2.4.4 ARM-Based *Eriador* Cluster System

Fig. 5.7: *Eriador* cluster.

Eriador is a platform that packages 40 Odroid-C2 single-board computers, *i.e.*, 160 ARM Cortex-A53 cores, into a single chassis. Mounted into a standard 19" rack, *Eriador* occupies two height units, similar to a commodity (*e.g.*, Intel-based) "brawny" server. An image of *Eriador* is shown here in Figure 5.7.

The platform logically provides a *cluster* of 40 independent machines. A dedicated cluster management software helps to schedule workloads within the cluster and gives access to hardware profiling features.

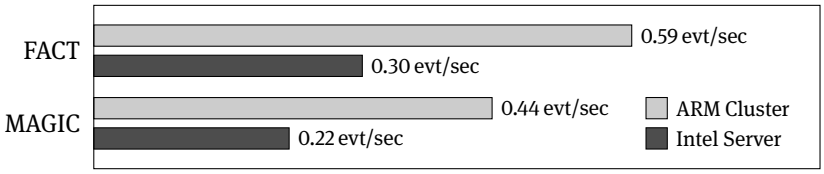


Fig. 5.8: CORSIKA throughput observed when processing FACT/MAGIC events on 160 ARM cores ("*Eriador*") vs. 48 Intel Xeon cores.

Besides the actual processing boards, *Eriador*'s chassis also includes networking hardware, fans, and heat management, as well as a separate, Odroid-based, *management server*. The management server has access to a tailor-made energy measurement component, enabling fine-grained power measurements without interference to the workload processors. Finally, the management server drives an LCD-display, which reports the current system state at the front of the unit.

5.2.4.5 *Eriador* vs. Conventional Server Architectures

We designed *Eriador* to study whether the performance and power-efficiency of particle simulations can indeed be improved by using low-power hardware.

To assess the *performance* aspect, we benchmarked the CORSIKA FACT and MAGIC particle simulators and compared our measurements on *Eriador* with a commodity Intel server. Specifically, our reference platform was a dual-socket Intel Xeon E5-2695 v2 system, clocked at 2.4 GHz and equipped with 256 GB of main memory. As such, our reference platform is not only a good representative of the hardware deployed in data centers at the time of our experiments. Moreover, the machine also likens *Eriador* in terms of form factor and monetary cost ($\sim$ 6000 EUR).

Figure 5.8 shows the throughput (measured as simulated events per second) for both platforms. It is easy to see that *Eriador* outperforms the commodity Intel server by about a factor of two.

To evaluate *energy consumption*, we could rely on the power measurement functionality directly built into *Eriador*. Through dedicated logic, *Eriador* can measure the energy consumption of individual cluster regions with a high resolution. On the Intel side, we used an EnerGenie EGM-PWM-LAN Energy Meter.

On both platforms, we measured *full system power*, *i.e.*, including components such as RAM or storage. Although we turned off as many hardware components in the Intel system as possible, this may somewhat underestimate the achievable power efficiency. However, we would like to note that neither the off-the-shelf Odroid-C2 building blocks of *Eriador* nor the infrastructure developed by our group (networking, power distribution, etc.) has been aggressively optimized for power efficiency. So there likely is room for improvement on the *Eriador* side, too.

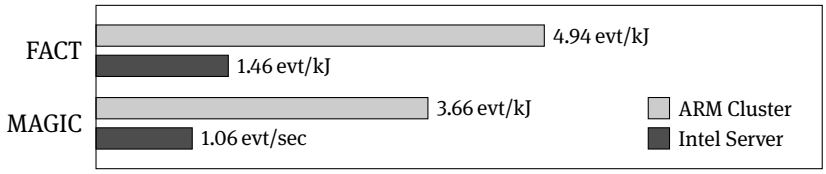


Fig. 5.9: CORSIKA energy efficiency observed when processing FACT/MAGIC events on 160 ARM cores ("*Eriador*") vs. 48 Intel Xeon cores.

Tab. 5.2: Runtime performance and energy consumption of *Eriador* vs. a commodity Intel server when evaluating *k*-means.

System	Exec. Time	Energy
Eriador	2.194 sec	314.36 kJ
Intel Xeon E5-2695 v2	1.139 sec	462.35 kJ

Figure 5.9 illustrates the results of our measurements. The higher throughput of *Eriador* comes at a lower power consumption when the system is under load. Consequently, the advantage of *Eriador* becomes even more pronounced when looking at the energy efficiency of the two platforms.

What is important to note is that *Eriador* consumes less energy *per simulated event*. In a parallelizable workload like CORSIKA, throughput can always be controlled by provisioning more or less compute resources. The low-energy footprint of *Eriador* could therefore be used to simulate at the same speed with less power and to simulate faster while staying within a given power budget.

5.2.4.6 Beyond Particle Simulation

Workloads like CORSIKA favor parallel architectures like *Eriador*, since independent simulations can almost trivially be parallelized over individual machines.

Not all real-world use cases behave so well. To illustrate how *Eriador* may fare in other data analysis settings, we benchmarked the cluster with an MPI-based *k*-means implementation and a dataset that included 1.59 M samples with 20 features each ($\sim$ 780 MB data set size in total; see [184] for details).

Table 5.2 shows the measurements we obtained. This time, *Eriador* is notably slower than the Intel-based reference system. Despite the long execution time, *Eriador* consumes less energy for the overall benchmark.

The key reason for this is that *k*-means is notoriously hard to parallelize. In our case, phases where the implementation can fully leverage the available parallelism, take turns with phases where intermediate results are exchanged between parallel units.

5.2.4.7 Summary

Our evaluation demonstrates that low-power hardware can indeed improve the resource efficiency of particle simulation by a significant margin. Our prototype platform *Eriador* can execute CORSIKA simulations at a higher speed while consuming less energy than a commodity server.

5.3 Validation of the Simulation

Abstract: Analyses in astroparticle physics heavily rely on the use of Monte Carlo simulations, which are utilized in basically all analysis steps from detector calibration to the reconstruction of energy spectra. A good agreement between simulated and experimental data is thus crucial for the scientific success of these experiments. Despite this necessity, a complete agreement between data and Monte Carlo cannot always be guaranteed. As a result, some simulated features disagree more than the experimental ones, while others disagree less. Capturing, quantifying, and possibly eliminating these disagreements, which can arise from a multitude of sources, is the purpose of simulation validation. This section explains how simulations can be validated and further points towards a machine learning-based possibility of minimizing their impact on the analyses for cases where their sources cannot be fully resolved.

5.3.1 Introduction

Creating meaningful Monte Carlo simulations is an iterative process, which is complex, especially at the beginning of new instruments. Today's physics instruments such as particle detectors or Cherenkov telescopes are extremely complex machines with many subsystems resulting in myriads of tweakable parameters that influence the validity of the simulations.

Many of these parameters are directly accessible from design documents, lab measurements of components, or calibration data, but many are not. The main challenge of validation of the simulations is to identify the low-level "screws" that need to be turned once an issue is identified in the higher-level data outputs of the simulations or even after processing the simulated data.

In this section, we offer a small introduction to identifying and tackling these issues, and present a machine learning-based method to identify mismatches between simulated and observed data.

5.3.2 Mismatches Between Observed and Simulated Data

Maximilian Linhoff

Differences between observed and simulated data in physics can arise from four main areas: insufficiently understood physics (see Section 5.4), insufficiently understood properties of the detector system, approximations and simplifications in the simulation, and changing environmental conditions and detector properties over time.
Any simulation will be affected to at least some extent by these problems. In the context of machine learning that uses simulated data for the labeled training dataset, these problems introduce differences between the training data and the data to which a model is applied, resulting in biases in the predictions, if the models are susceptible to differences. Concerning the inverse problems, these systematic errors will affect the detector response, which is also calculated from the labeled simulated data and produces biased results.

A simple example of such a difference is an imaging air Cherenkov telescope that is simulated with a higher reflectance of the mirrors than the actual telescope. This flaw results in brighter images in the simulations, which in turn result in a negative bias of the energy estimation when applied to observed data. Three complementary approaches can be taken to mitigate these issues:

- adapting the simulation,
- transforming the observed data, or
- transforming the simulated data.

The first approach is the most fundamental one and requires re-running the simulation; it is thus the most computationally expensive approach but also the one that likely yields the best results.

The second approach is mostly taken to calibrate for the low-level properties of detector electronics, which would be very expensive to simulate and are thus usually over-simplified in the simulation software.

A prime example of the third approach is the amount of night sky background photons contributing to the noise in IACT images. The amount of moon- and starlight in the field of view does not affect the air-shower physics and thus the amount of Cherenkov light produced. This is, however, the most resource-intensive part of the simulation. Thus, simulating IACT images with low amounts of or even entirely without night sky background photons and then only adding this additional noise in the data processing so that it fits the noise characteristics of the observed data can be much more computationally efficient than doing multiple air-shower simulations and detector simulations with different noise levels. Additionally, instead of simulating the additional noise, one can use noise from observed events without showers (known as random trigger or pedestal events). This can dramatically improve the agreement of

the noise characteristics between simulated and observed data. This approach was developed as part of the CRC works [102, 111].

Validation of simulations and the transformations applied to observed data is done by comparing parameter distributions at different levels of the data processing. This becomes much harder in end-to-end analyses, e. g. Deep Learning (Chapter 9), where intermediate, interpretable representations of the data might not be easily accessible.

Modern particle and astroparticle physics instruments comprise lots of subsystems and especially experiments in astroparticle physics do not have easily controllable environmental conditions. This results in a sizeable combinatorial parameter space for the simulations, which cannot be fully explored to find the correct combination. The computational complexity of creating simulations with sufficiently statistical uncertainties results in very slow feedback loops in the iterative process of improving the simulations. This makes creating suitable simulated datasets one of the major tasks when new experiments are planned and commissioned.

In the end, mismatches between the observed and simulated data will result in systematic biases of the results obtained by applying algorithms trained on these simulated datasets to the observed ones. Sometimes, these systematic errors will be relatively straightforward, as in the example of the IACT with reduced reflectance above. But most of the time, due to the combinatorial nature and complexity of the detectors, the cause, and effects are much more indirectly linked, making it very hard to identify the root cause once a mismatch is identified.

5.3.3 Detection of Mismatches

Tim Ruhe
Maximilian Linhoff

For well-simulated data, simulated events should be indistinguishable from observed events by their properties. Classically, investigations of possible mismatches between simulated and observed datasets rely on one-dimensional parameter distributions. However, systematic errors in the simulation will also affect the correlation between parameters, so looking at single parameter distributions is insufficient. With the high dimensionality of typical intermediate data representations in particle and astroparticle physics, it is infeasible to inspect all possible combinations of parameters manually.

A way to quantify the agreement of simulations and observations is thus to try to classify the datasets with the goal of predicting for each event if it was simulated or observed and then applying the usual quality metrics for supervised classification tasks such as accuracy or the area under the ROC curve. For a well-simulated dataset, this classification task should be hard to impossible, and thus the classification metrics should be compatible with randomly guessing.

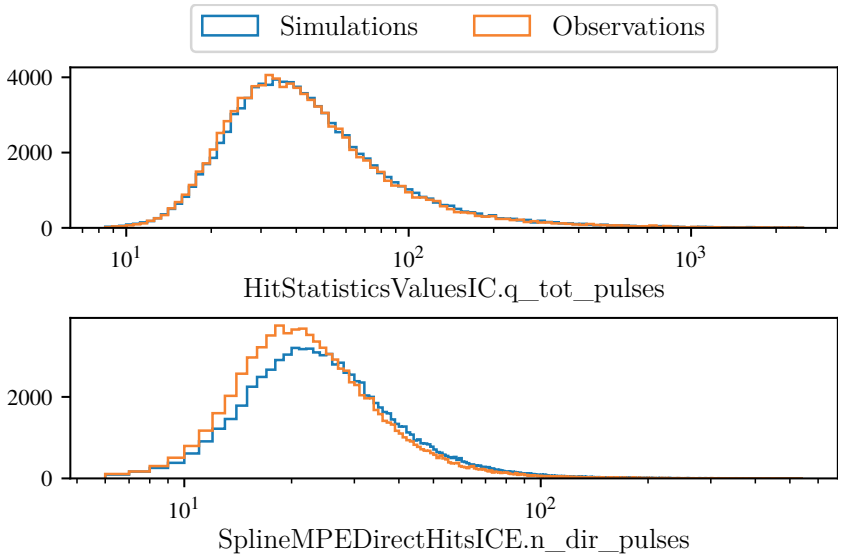


Fig. 5.10: Two example features from the IceCube dataset for *observations* and *simulations*. Top: total charge in the DOMs, one of the well simulated features. Bottom: number of pulses in the DOMs, one of the not-so-well simulated features.

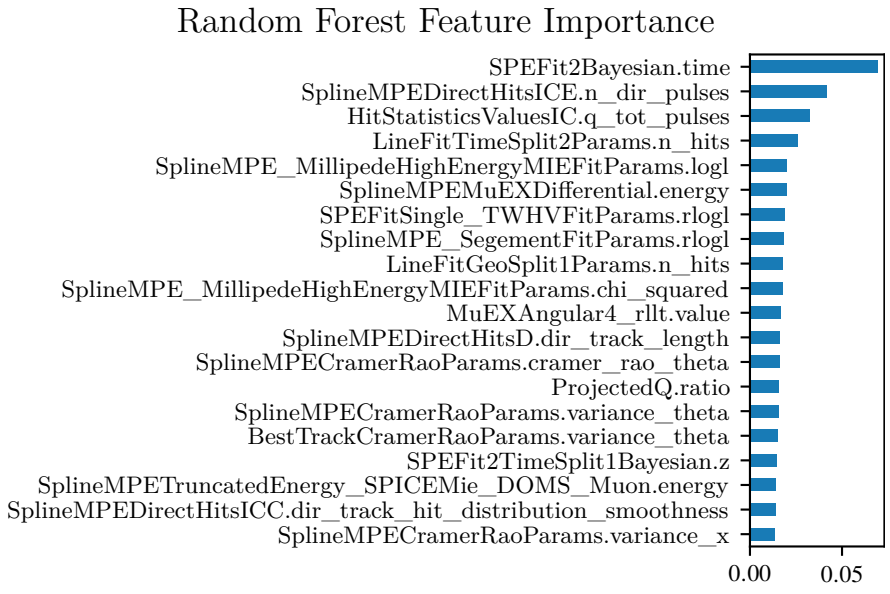


Fig. 5.11: Feature importance of the 20 most important features for the classification into *observations* or *simulations*

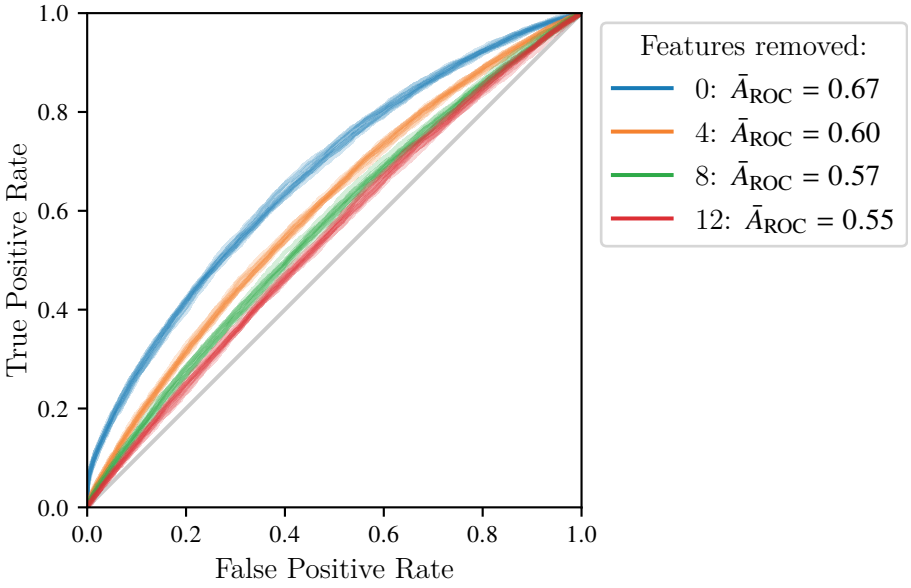


Fig. 5.12: ROC curves for the classification into *observations* or *simulations*. After removing the 12 most important features for this classification from the dataset, the task is much harder to solve.

In addition to quantifying the match between simulations and observations, this approach also allows the identification of badly simulated features by calculating how important an individual feature was to the overall classification. Decision tree-based algorithms can provide individual feature importances via the total reduction of the loss function achieved by the splits in the respective features. For all algorithms, a permutation-based approach can be used [97, section 10]. To quantify the importance of a single feature, the values of this feature are permuted, and the classifier is evaluated on the permuted dataset, resulting in a worse metric than the baseline if the feature is important to the classification.

After identifying problematic features, two approaches are possible: removing badly simulated features until the classification is no longer possible or using the gained insight to improve the simulations. These two approaches are akin to treating symptoms or the underlying root condition in medicine. If possible, the latter will always produce better results, and the former is easier in the short term. Again, both approaches complement each other. This approach was developed as part of [91], and more details can be found in [90].

5.3.3.1 Application using an IceCube high-level dataset

In the following, we will apply this approach to a high-level IceCube dataset, which generally has an excellent agreement between simulated and observed data. Using the

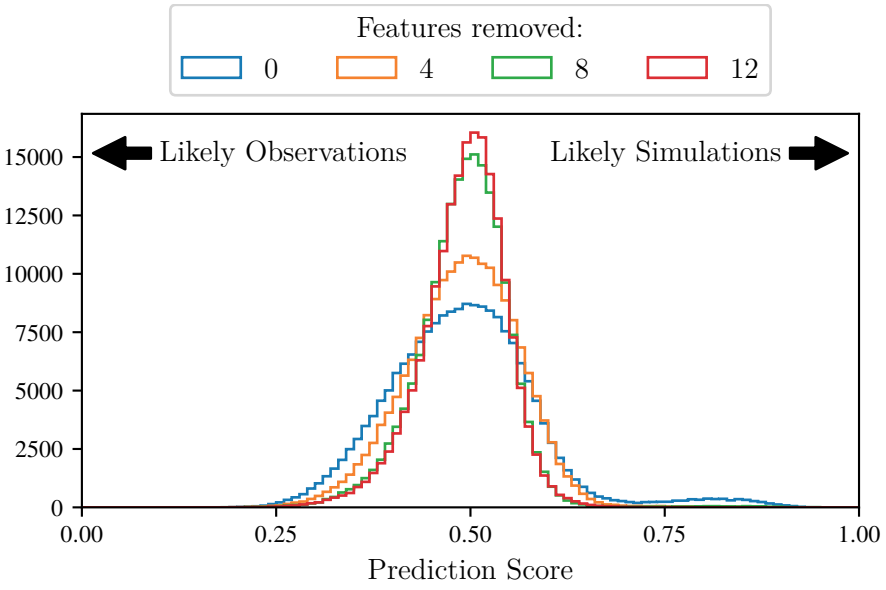


Fig. 5.13: Score distributions for the classifications *observations* or *simulations*.

approach, we will identify a few columns in the dataset that show mismatches and make the classification impossible by iteratively removing features with high feature importance with respect to the classifications *observed* and *simulated*.

The dataset consists of simulated, neutrino-induced events and observations after suppressing the atmospheric muon background to very low levels. As the simulations do not directly follow the expected energy spectra and the observation durations are not equivalent, sample weights for the simulated dataset are necessary to obtain a realistic comparison. Due to the low cross-section, neutrino interactions in IceCube are very rare relative to atmospheric muons. Primary neutrinos are also simulated with an artificially increased cross-section to force interactions inside the detector to obtain sufficient statistics for the neutrino events (the searched-for signal in most IceCube analyses). This artificially increased cross-section must be considered when calculating the sample weights for these events. While most of the features in the dataset show very good agreement between simulations and observations, a few do not, like those shown at the bottom of Figure 5.10.

As quality metric, area under the ROC curve (A_{ROC}) is used, with $A_{\mathrm{ROC}} \approx 0.5$ constituting random guessing, the desired outcome. Both loss-reduction feature importance for tree-based models and permutation-based feature importance for all classification algorithms are implemented in `scikit-learn`. For this example, we will use a random forest classifier and rely on the loss-based feature importances, as these are provided by `scikit-learn` out of the box after training. We train a random forest classifier using tenfold cross-validation.

In the first iteration, we train with all available features. The mean feature importance over the cross-validation iterations is shown in Figure 5.11. It can be seen that a few features stick out, including the one shown in Figure 5.10. We now remove the four features with the largest importance and repeat the procedure three more times. Figure 5.12 shows the resulting ROC curves for each cross-validation iteration along with the mean area under the curve. As expected, the problem gets progressively harder to solve for the classifier; after removing 12 features the mean area under the ROC curve is down to $\bar{A}_{\mathrm{ROC}} = 0.55$ from $\bar{A}_{\mathrm{ROC}} = 0.67$ on the full dataset.

The same behavior can be seen when looking at the score distributions in Figure 5.13. While for the entire dataset, there is a considerable number of events that the model assigned a large likelihood to be simulated, the distributions approach a normal distribution around a mean of 0.5 for the reduced dataset.

For further investigations, looking at the events that were classified as simulations with high likelihood might give valuable insights into why those events were clearly identified, meaning they belong to a class of events that is not or only much more rarely observed in measured data. The same could be said for the opposite case: events identified with high confidence as observed might be a missing class of events in the simulations, but this seems not to be the case here.

As a final remark, it should be stated that this approach is only feasible for datasets with an excellent general agreement between observations and simulations. For example, we did not consider the importance of the features towards our main analysis goal. Suppose the algorithm would remove important features for the following science analysis. In that case, our only choice is to improve the quality of the simulations (treating the underlying condition) instead of keeping the dataset with a reduced feature set (treating the symptoms).

5.4 Keynote: The Muon Puzzle

Hans Dembinski

Abstract: High-energy cosmic rays are studied indirectly through extensive air showers, which are hadronic cascades in the atmosphere. Simulated air showers based on hadronic interaction models tuned to fixed-target and collider data show a muon deficit. This is called the muon puzzle. This is a challenge for applying machine learning to astroparticle physics experiments since machine learning needs accurate simulations to generate training samples. Progress has been made in recent years to experimentally measure the muon discrepancy with high precision. Theoretical studies have connected muon production to the basic properties of hadronic interactions. The studies show that the discrepancy can only be resolved by reducing the energy that goes into neutral pion production relative to long-lived hadron production. The ALICE experiment at the Large Hadron Collider (LHC) recently discovered an enhancement of strangeness production in proton-proton and proton-lead collisions at the LHC, which qualitatively matches this requirement. Further studies on strangeness enhancement in the forward region at the LHC have the potential to finally resolve the muon puzzle.

5.4.1 Introduction

At cosmic ray energies above about 100 TeV, the flux of cosmic rays is so low that direct observation with satellite and balloon experiments becomes infeasible. At these high energies, cosmic rays are observed indirectly via air showers, hadronic cascades initiated by high-energy cosmic rays in the atmosphere. The indirect observation requires accurate air-shower simulations since properties of the cosmic ray have to be inferred from features of the air shower. This is true for conventional analyses in which human-made models are fitted to air-shower data and for modern analyses that use machine learning to create models directly from simulations. The cosmic ray community has built sophisticated simulation packages that integrate state-of-the-art models of electromagnetic and hadronic interactions. Air showers simulated with these programs describe real showers quite successfully, but there is a long-standing discrepancy between simulated and observed muon production in air showers.

Tensions between simulation and experiment are observed in several aspects of muon production, but the most prominent and extensively studied deviation is that of the total number of muons N_μ that arrive at the ground. The muon abundance is significantly lower in simulations than in experiments, and this is called the muon puzzle. Air-shower simulation codes such as CORSIKA use a variety of models for hadronic interactions, which all fail to reproduce the large number of muons observed

in ultra-high-energy air showers. This is a major obstacle for the field since the muon number and the depth of shower maximum $X_{\max}$ are the two main features to infer the mass composition of cosmic rays as a function of their energy. Because of this discrepancy, the estimates for the mass composition obtained from these two air-shower features do not agree. It is also a challenge to use machine learning to analyze air-shower data employed by the Pierre Auger, and IceCube Neutrino observatories [3, 137, 314]. The mismodelling of air showers may introduce subtle biases to the model that the machine learns.

The muon puzzle is also interesting from the point of view of high-energy particle physics because it could indicate missing physics in current state-of-the-art models of hadronic interactions, which describe the hadronic cascade that leads up to the muon production. The dominant interactions in air showers have low momentum transfer and cannot be computed from first principles with perturbative quantum chromodynamics. Effective theories are used to describe these interactions, and an important effect may be missing in these theories.

In this light, the recently discovered universal strangeness enhancement at mid-rapidity in proton-proton and proton-lead collisions at the LHC by the ALICE collaboration [36] is of particular interest. This effect was previously only observed in heavy-ion collisions, where the standard model to interpret these data is based on the formation and subsequent freeze-out of a quark-gluon plasma (QGP). The theory community is divided over whether a QGP can also form in smaller collisions systems such as proton-lead or proton-proton. Even though the mechanism that causes strangeness enhancement is currently unclear, the effect phenomenologically changes the hadron composition so that the muon number in air showers is increased. The phenomenology has been explored with a toy model [77, 312] with the conclusion that this effect could be the missing piece in the muon puzzle.

This guest article is based on a recent review on the muon puzzle and its connection to the LHC [46], which summarizes the current knowledge about the muon discrepancy in air showers, the state of air-shower simulations, what we have learned about the connection of microscopic hadronic physics and muon production in air showers, and finally what we have already learned from the LHC and what kind of measurements are still needed. We refer the reader to the review for further details.

5.4.2 Meta-Analysis of Muon Measurements in Air Showers

Several deviating aspects exist between the simulated muon component in air showers and measurements. The most striking and best explored is the deviation in the mean number of GeV muons as observed by ground arrays of particle detectors. This quantity, or proxies thereof, has been measured by multiple experiments, with initially contradicting results. These apparent contradictions have been largely reduced by a meta-analysis from the Working group on Hadronic Interaction and Shower Physics (WHISP), which

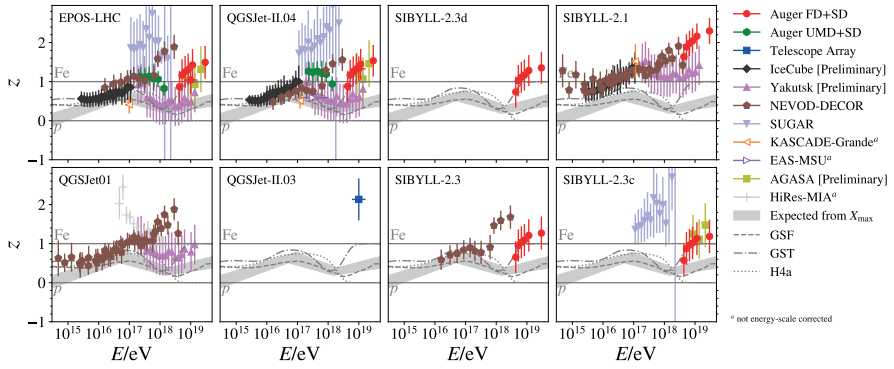


Fig. 5.14: Measurements of muons produced in air showers from nine experiments after adjusting for energy-scale offsets (points) converted to the z-scale as a function of shower energy for different hadronic interaction models. For reference, the figure shows the expected values from global fits to air-shower data (lines) and from optical measurements of the depth of the shower maximum (band). The image was taken from [356].

was formed in 2018 by members of eight (now nine) air shower experiments [116, 141, 356].

Muon measurements from different experiments are not directly comparable since the experiments and data analysis methods differ in many details. They are therefore converted to the logarithmic z-scale, defined by the formula

$$z = \frac{\ln N_\mu - \ln N_{\mu,p}}{\ln N_{\mu,\mathrm{Fe}} - \ln N_{\mu,p}}, \tag{5.6}$$

where $\ln N_\mu$ is the logarithm of the muon number, or any measurable quantity proportional to the muon number, as measured by the detector, while $\ln N_{\mu,p}$ and $\ln N_{\mu,\mathrm{Fe}}$ are the corresponding simulated numbers (on detector-level) from an air-shower simulation with a particular hadronic interaction model. While N_μ depends strongly on the experimental conditions, z is approximately independent of experimental conditions. Furthermore, the results from different experiments are cross-calibrated to avoid relative offsets between the independent energy calibrations of each experiment. This is important since the muon number scales almost linearly with the (apparent) shower energy.

The energy-scale corrected z-values from nine experiments and for eight hadronic interaction models are shown in Figure 5.14. Above 10 EeV, the z-values exceed those of pure iron showers for all models, which is astrophysically not plausible. Furthermore, the values should be consistent with predictions based on the optical measurements of the depth of shower maximum X_{max}, which constrain the cosmic-ray mass composition,

the only other aspect that z values depend on, apart from the shower energy. The measured z values exceed these predictions above 10^{17}eV.

An experimental mistake can be ruled out as the cause of the muon discrepancy since the effect is seen by several independent experiments. A statistical fluctuation can be ruled out as well with very high confidence. Other potential origins for the discrepancy have also been ruled out; see [46] for details. A major error in the propagation of GeV muons through the air can also be excluded. However, there are ongoing efforts to improve the precision for TeV muons propagating in dense media like water, and ice [148, 159, 229, 338, 339], which will also benefit these studies. Attempts to explain the muon discrepancy with exotic physics such as Lorentz-invariance violation [34] or exotic astroparticles are challenged by the fact that this would change the depth fluctuations of the shower maximum [1, 2] and the muon number [4], which are well described by current simulations. The early onset of the discrepancy, at about $4 \cdot 10^{16}$eV and corresponding to a center-of-mass energy $\sqrt{s_{\mathrm{NN}}} \approx 8$ TeV in the first interaction, is another challenge for exotic theories. Hadron collisions at such center-of-mass energies are observed at the LHC, and signatures of exotic physics at these energies would have been found if they had played a role.

The most plausible remaining possibility is a subtle deviation in the simulated hadronic cascade leading to muon production. Progress has been made in recent years on understanding air-shower physics and the possible microscopic origin of this discrepancy, as discussed below. A relatively small deviation in the simulation, potentiated by the hadronic cascade, cannot be ruled out by current LHC measurements and may even be suggested by them.

5.4.3 Muon Production in Air Showers

An air shower is a hadronic cascade in which the kinetic energy of the cosmic ray is successively converted into secondary particles in inelastic interactions with the atmosphere. The Heitler-Matthews model [262] is a simplified model of an air shower. A schema is shown in Figure 5.15 on the left-hand side. Only pions, the most abundant long-lived hadrons, are produced in this model. Charged pions have long lifetimes and interact again to form a hadronic cascade. Neutral pions decay immediately into photons, which form a decoupled electromagnetic cascade. There is a small feedback from the electromagnetic into the hadronic cascade via photo-nuclear interactions, but this effect is neglected here.

In the model, the energy of the parent pion in the model is distributed equally to its children. The energy of each pion, therefore, decreases after each interaction. After k steps of the cascade, it falls below the critical energy, where decay becomes more probable than another collision. At this point, the pions decay into muons. The number of muons N_μ as a function of the energy E and nuclear mass A of the cosmic ray can be

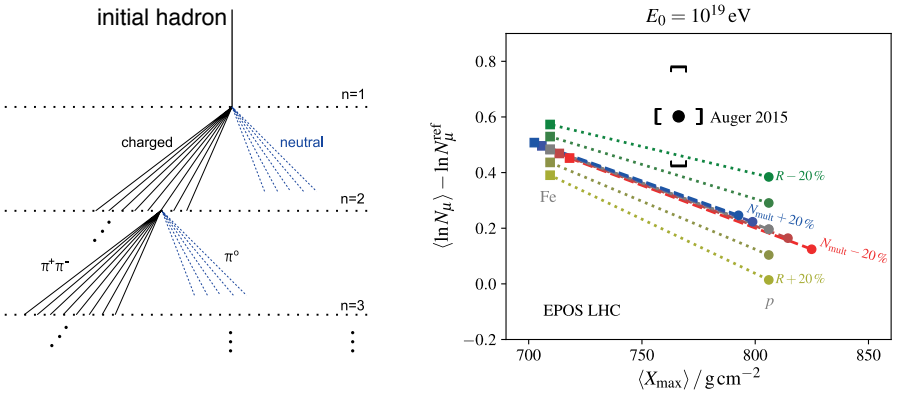


Fig. 5.15: *Left:* Sketch of the Heitler-Matthews model [262] of an air shower initiated by a cosmic ray, in which only pions are produced. Image from [373]. *Right:* Impact of changing the hadron multiplicity and the energy ratio R = $\langle E_{em} \rangle / \langle E_{had} \rangle$ on the muon number and the depth of shower maximum for 10^{19} eV air showers. Lines indicate predictions from air-shower simulations with EPOS-LHC [311] for any cosmic-ray composition between pure proton (bottom right) and iron (top left). The gray line indicates the baseline model, while the colored lines indicate predictions from an ad-hoc modified model, with modifications in steps of 10 %. The data point is from the Pierre Auger Observatory [5]. Image from [77].

computed in this simplified model as

$$N_\mu(E, A) = A^{(1-\beta)} \left(\frac{E}{\xi_h} \right)^\beta \text{ with } \beta = \frac{\ln(\alpha N_{mult})}{\ln N_{mult}}, \tag{5.7}$$

where α is the fraction of charged pions produced, N_{mult} is the hadron multiplicity (average number of hadrons produced per collision), and ξ_h is the critical energy for pions (when decay becomes more likely than another interaction).

The muon number N_μ is very sensitive to α. For an EeVair shower, a 10 % change in α would roughly introduce a 50 % change in the muon number. Accordingly, α needs to be known over a wide energy range to calculate the muon number correctly. In the simplified model, α is 2/3 due to isospin symmetry, but the value differs in real air showers. Protons, neutrons, and strange hadrons are produced, which have lifetimes large enough to participate in the cascade. The results of the simplified model approximately carry over if α is considered more broadly as the energy fraction carried by long-lived hadrons. It is experimentally convenient to measure is the closely related quantity

$$R = \frac{E_{em}}{E_{had}} = \frac{1-\alpha}{\alpha}, \tag{5.8}$$

where E_{em} is the electromagnetic energy flow from photons and electrons, while E_{had} is the hadronic energy flow, and the average is taken over the phase space of the secondaries.

These basic analytical results are confirmed by full air-shower simulations [373]. To increase the muon number in simulations, one must increase the hadron multiplicity N_{mult} or decrease the fraction of neutral pions produced. However, it turns out that an increase in the hadron multiplicity changes the depth of shower maximum in such a way that the discrepancy cannot be resolved. This is illustrated in Figure 5.15 on the right-hand side. Modifications of the hadron multiplicity shift the model line parallel to itself; one cannot close the vertical gap to the data point. This is only possible with a modification of R. A reduction in the energy fraction carried by neutral pion (a reduction in R) is, therefore, the only solution, according to this study, though it may be accommodated by a moderate change in the hadron multiplicity N_{mult}. Currently available data on R from the LHC does not yet constrain the value to the required precision in the phase space relevant for air showers. However, there are strong hints that R may be lower than the value currently used in hadronic interaction models.

5.4.4 Related Measurements at the LHC at CERN

A comprehensive overview of existing LHC measurements at CERN that are relevant for air-shower simulation is given in [46]. The LHC mainly accelerates protons and lead ions, and for a short time, also xenon ions have been accelerated. The approved plans to accelerate oxygen beams in the following years are essential for air-shower physics to measure p-O and O-O collisions [99, 125]. The most common interaction in an air shower is π-N for which p-O collisions are an excellent reference. Current state-of-the-art models show considerable variance in their predictions of hadron production in p-O collisions, despite being tuned to pp data, reflecting the theoretical uncertainties in extrapolating from a pp reference. Together with the essential direct measurements in p-O, the study of both pp and p-Pb data is important to detect potential scaling laws and the universal features of hadron-ion collisions.

A general challenge for air-shower physics is that hadrons emitted in the forward region dominate the air-shower development, but data is sparse in the forward region. This is illustrated in Figure 5.16, in which the particle density is shown as a function of pseudorapidity (related to emission angle as described in the figure caption). The forward emitted particles are less numerous but more energetic and therefore create more secondary particles in the next step of the hadronic cascade. Eventually, the bulk of particles (and muons) is produced by the most energetic particles in the interaction, which are those produced in the forward region. The LHC experiments are well instrumented at mid-rapidity ($|\eta| < 2$), while the very forward region $|\eta| > 5$ is covered only by calorimeters. LHCb is the most forward general-purpose hadron spectrometer, covering the range $2 < \eta < 5$, thanks to its focus on measuring decays of charm and beauty quarks, and is therefore of particular interest. Important in the very forward region are CMS with its CASTOR calorimeter system, the forward counters of TOTEM and ALICE, and the LHCf zero-degree calorimeters for photons and neutrons.

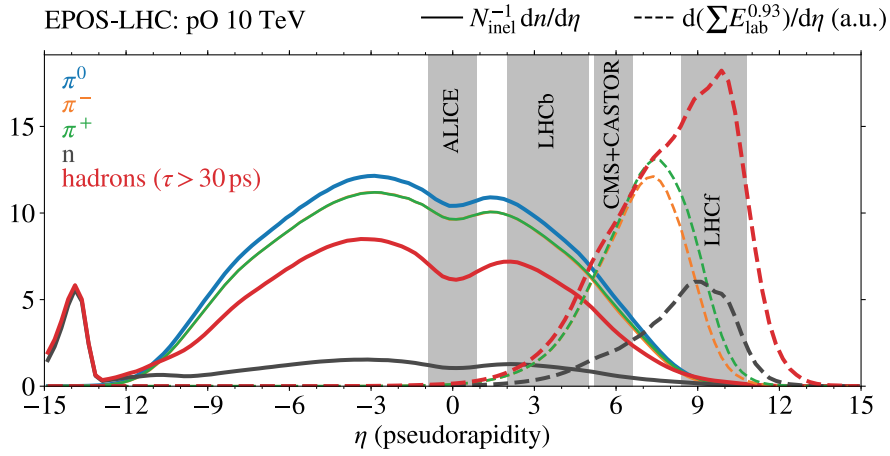


Fig. 5.16: Density of pions, neutrons, long-lived hadrons, photons and electrons as function of pseudorapidity η in p-O collisions (solid lines) at 10 TeV, which is related to the emission angle θ relative to beam axis, $\eta = -\ln\tan(\theta/2)$. Shown for comparison is the expected muon yield from these particles if they form a shower in the atmosphere (dashed lines) and the acceptances of several LHC experiments (bands). Image taken from Ref. [46].

As discussed in the previous section, muon production in air showers depends on hadron multiplicity and the ratio R of electromagnetic to hadronic energy flow—or, more generally speaking, the hadron composition.

Experimental proxies for the hadron multiplicity are the charged particle multiplicity and the energy flow, which have been measured in pp, p-Pb, Pb-Pb, and Xe-Xe collisions at the LHC by ALICE, CMS, LHCb, and TOTEM up to $\eta = 6.4$. See [46] for details. These forward measurements are important since the current generation of hadronic interaction models deviate by less than 5 % at $|\eta| < 1$ in pp collisions (where they have been tuned to older LHC data), but up to 20 % in the forward region. The latest data strongly constrains the model predictions in the forward region in pp collisions. A model variance of 20 % is still found for p-O collisions, which is expected to be strongly reduced with the upcoming p-O data.

The hadron composition has the largest influence on muon production. High-precision data on the relative yields of pions, kaons, and protons in pp, p-Pb, and Pb-Pb collisions is available at mid-rapidity $|\eta| < 1$ from ALICE and CMS. In particular, ALICE studied the production cross-sections of strange hadrons at mid-rapidity $|\eta| < 1$ and discovered an universal (independent of collision energy and system) rise in the production ratios of strange hadrons to pions as a function of the charged-particle multiplicity [36, 382], as shown in Figure 5.17. This behavior was previously known only from heavy-ion collisions and is not expected in p-Pb and pp collisions. The hadron density in the central region rises rapidly with the collision energy; thus, the

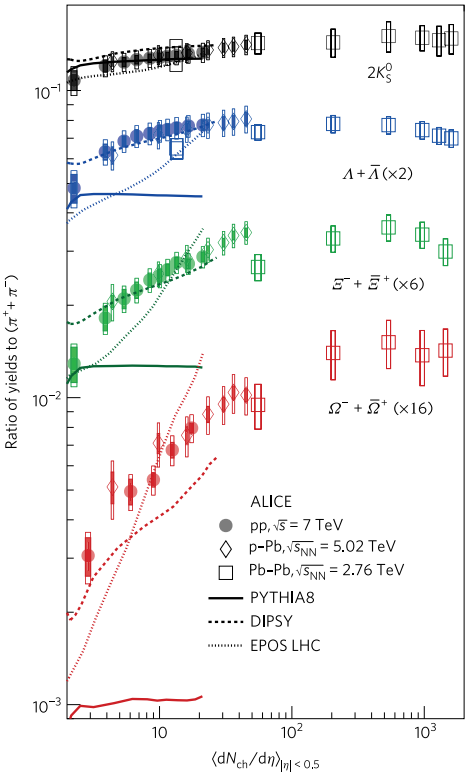


Fig. 5.17: Ratios of strange hadrons to pions measured at mid-rapidity $|\eta| < 0.5$ in pp, p-Pb, and Pb-Pb collisions as a function of the charged particle density $dN_{ch}/d\eta$. Image taken from [36].

relative amount of strangeness production also rises. The strangeness enhancement is accompanied by a reduction of neutral pions, which reduces R. The impact of this effect on air showers was studied with toy simulations [77], and it was found that this modification has the potential to resolve the muon puzzle in air showers.

The ALICE datasets are essential for model tuning and validation. However, they do not directly constrain the hadron composition in the forward region, where two hadron production mechanisms, string fragmentation and remnant fragmentation, contribute. In the forward region, yields of pions, kaons, and protons can be studied only by LHCb since the other detectors lack particle identification capabilities. LHCb has published data from pp collisions up to 7 TeV. The analysis of pp collisions at 13 TeV and p-Pb collisions at 8.16 TeV is ongoing and expected to have an important impact on air showers.

CMS with CASTOR has studied the energy ratio R of electromagnetic to hadronic energy flow as a function of the charged particle multiplicity at mid-rapidity in pp collisions [77, 351] in the very forward region. CMS observed a high value of R in pp,

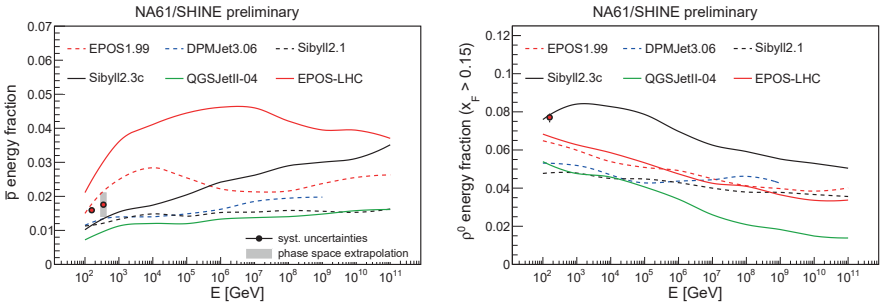


Fig. 5.18: Energy fraction transferred to anti-protons (left) and ρ^0-mesons (right) in π-C collisions as measured by NA61/SHINE (data points) and as predicted by hadronic interaction models over the whole range of beam energies relevant for air showers (images from [316, 374]).

which is larger than what is predicted by all current models, but the value also has a relatively sizeable systematic uncertainty of about 20 %. The considerable value of R in pp found by CMS is surprising because it favors air-shower simulations with even lower muon content than the current state of the art. By contrast, the opposite is required to resolve the muon discrepancy. No significant trend of R was observed as a function of the charged particle density at mid-rapidity.

A follow-up study with p-O and p-Pb collision is critical to understand whether R is reduced in ion collisions compared with pp. Unfortunately, a measurement of future p-O collisions with CASTOR is not possible since CASTOR has been decommissioned. In the very forward region, R is also constrained by LHCf with measurements of photon-production, π^0 production, and neutron production in pp and p-Pb collisions. LHCf plans to study strangeness production at zero-degree angles based on the decay $K^0_S \to 2\pi^0 \to 4\gamma$ with its upgraded detectors in the upcoming high-luminosity LHC run [265].

5.4.5 Fixed-Target Experiments at SPS and LHC

Since hadronic interactions in an air shower span many orders of magnitude in energy, there are also opportunities to improve our knowledge at lower values of the center-of-mass energy in the nucleon-nucleon collision system, $\sqrt{s_{\mathrm{NN}}}$, which are reached by fixed-target experiments [268]. One advantage of fixed-target experiments is the flexibility regarding the target, which makes it ideal for studying nuclear effects in various systems.

The NA61/SHINE experiment at the Super Proton Synchrotron, the pre-accelerator of the LHC, has measured hadron production in pp, π-C, and p-C collisions, where carbon is used as a proxy for air. Among the many measurements, we highlight measurements of the forward ρ^0 and anti-proton production [38, 317] in Figure 5.18. The ρ^0 production is important since it is an alternative to producing a π^0 meson in the

charge-exchange reaction $\pi^- + p \to \pi^0 + n + X$. An enhancement of the ρ^0/π^0 ratio decreases R and increases the muon number in air showers. Anti-protons are a measure of baryogenesis in the air shower, which reduces R. Although the energy flow towards these particle species is small in each interaction, the compounding effect over many interactions leads to an increase up to 60 % in the muon number produced in air showers run with the recent version of SIBYLL-2.3d compared with SIBYLL-2.1 without these effects [324]. While this change is imposing, it has not resolved the muon discrepancy since SIBYLL-2.1 formerly produced particularly muon-poor air showers. The muon production in air showers simulated with SIBYLL-2.3d is now on par with those simulated with EPOS-LHC.

At the LHC, fixed-target experiments are performed by LHCb with the SMOG device [15], which injects small amounts of gas into the detector. The original system was designed to study the beam profile but has been used to place limits on the intrinsic charm inside the proton [12] and to measure the anti-proton production cross-section in p-He collisions [14], which is needed to model cosmic ray interactions in space. The original device has been replaced as part of the LHCb upgrade for Run 3 with an open storage cell that allows for higher gas densities and more target gasses, including nitrogen and oxygen [76]. With LHCb in fixed-target mode, it will be possible to study hadron production at $\sqrt{s_{\mathrm{NN}}} = 115\,\mathrm{GeV}c^{-1}$ at mid-rapidity $-2.5 < \eta_{\mathrm{cms}} < 0.5$ in the nucleon-nucleon center-of-mass frame.

5.4.6 Summary and Outlook

The cosmic ray community has found a discrepancy in the muon production in air showers initiated by ultra-high energy cosmic rays. A muon deficit is observed in air-shower simulations, which after careful evaluation, can be attributed only to a mismodelling of the hadronic cascade in the air shower, leading up to the muon production. The most plausible way to resolve the discrepancy is to moderately reduce the forward energy flow to neutral pions relative to long-lived hadrons as the collision energy increases.

The relative energy flows to neutral pions, and long-lived hadrons in the forward region can be constrained with measurements of proton-proton and proton-ion collisions at the LHC. The upcoming pilot run with proton-oxygen and oxygen-oxygen collisions at the LHC is particularly important. Experiments with forward acceptance, ALICE, CMS, LHCb, LHCf, and TOTEM, provide the best experimental constraints. LHCb is the only fully instrumented general purpose spectrometer with hadron identification capabilities in the forward region. The other experiments provide calorimetric and particle counting information in the very forward region.

The discovery of a universal multiplicity-dependent strangeness enhancement in high-density collisions at mid-rapidity by ALICE could be a potential solution to the muon puzzle. The exact mechanism behind this effect is currently unclear. Further measurements of a possible strangeness enhancement in the forward region with LHCb

and LHCf are underway. Further opportunities for air showers will be offered by running LHCb in fixed-target mode to the study collisions of the LHC beams with nitrogen and oxygen.

Data from the LHC is expected to improve air-shower simulations and resolve the muon puzzle, which will likely boost the prospects of machine learning in astroparticle physics. An intriguing idea to be explored in the coming years is to constrain the parameters of hadronic interaction models with collider and air-shower data simultaneously. This is technically challenging since full air-shower simulations are required to connect the effect of a parameter change to a change in air shower features, and these simulations are very time-consuming. Currently, only collider data is used to constrain the models, and air-shower data is predicted. The air-shower simulation step could potentially be accelerated in the future with fast simulation techniques based on neural networks.

6 Data Storage and Access

6.1 Introduction

Wolfgang Rhode

In Chapter 1, we recognized that the analysis cycle must be considered as a whole in modern knowledge discovery. Data storage can occur at different points in this cycle, from huge volumes of raw data at the beginning, through data selected for specific analysis purposes, to the deconvolved data as the solution to inverse problems. The more fundamental this stored data is, the more analysis can and must be performed to obtain physically meaningful results. Therefore, the data is always inextricably linked to the programs and analyses that are needed.

With this in mind, in this chapter, we will first look at the framework of research data management from the perspective of resource-efficient data analysis. Then two examples will be discussed, and an open approach to raw data will be presented for the example of the FACT Open Data project. The example of the DeLorean architecture will show how fast data access to experiment-internal data structures can be achieved.

6.2 Research Data Management

Wolfgang Rhode
Dominik Elsässer

Abstract: Astroparticle and particle physics analyses usually rely on processing a large number of observables or features that are recorded and processed at a high frequency. The data may have been either recorded experimentally or calculated using Monte Carlo methods. The data may contain information that is scientifically irrecoverable either because the triggering event is unique (such as a supernova explosion) or because the experiments no longer exist in the same form. Thus, regardless of their nature, the data also represents a large material value. In this chapter, the path of data through data analysis is discussed keeping in mind that appropriate means of access must be found, depending on the particular purpose of the data analysis: from real-time analysis during data collection to precision analysis in the weeks and months after data collection and possibly to re-analysis many years after data collection.

https://doi.org/10.1515/9783110785968-006

6.2.1 Management of Large Amounts of Research Data

It seems quite natural to the modern scientist to be able to access the observation results of Ptolemy, Galileo, or Brahe—but at the same time, the contents of research done a few years ago, possibly even on data from an ongoing experiment, cannot be reproduced because the necessary information is no longer available or the programs required for this are no longer executable.

In this book, we discuss data analysis derived from experiments designed and conducted by hundreds of scientists over decades. In addition, a roughly equal amount of work is needed to determine the virtual reality (Monte Carlo simulation) that describes the experiment in question. Here, the material value of the work may well lie in the many CPU hours spent generating and testing the simulations. Both the real and ideal financial contributions to such an experiment will eventually be about the same, and are likely to total several hundred million euros.

A material idea is appropriate here because it immediately shows that experiments that require considerable time and financial resources, and thus are realizable only by large international collaborations, must be repeatable, in principle. However, practically no one will want to or be able to finance them. This is also because a future experiment would undoubtedly consider technological advances that have taken place in the meantime. Especially in astrophysical observations, transient phenomena occur anyway, which can be observed practically only once. This raises the question of how to handle these valuable data in such a way that they remain accessible to mankind for the most benefit in an optimal form for as long as possible. This is a question of research data management.

The long-term vision for the coming centuries or millennia here can obviously only be to leave to posterity measurement points in physical units and with the best possible calculation of the uncertainties in such a way that these measurements can be interpreted physically even without detailed knowledge of the experiments or analysis methods. These are the optimal solutions to the inverse problems posed by the measurement, including the corresponding error considerations.

A comparatively modest requirement is the stipulation of many research funding organizations that research data be retained for reanalysis for at least ten years in such a way that it is accessible to and useable by the scientific community. The time period can be seen as the time when re-analyses of the original experiment are initiated to consider new or improved findings. New questions may be raised about the research data from among the receiving scientists, and it must be possible to substantiate the course of the analysis for litigation that, unfortunately, can occasionally arise. Within this time, the experimentally recorded data go through a gradual publication process in which the circle of researchers who have access to the data continues to expand, and the range of user qualifications changes accordingly. This transition process requires careful planning.

The flow chart of typical analyses in the research field considered here, outlined in Figure 1.5 in Chapter 1, contains three meta-levels relevant for research data management. This is, first, the data reduction by many orders of magnitude that necessarily occurs in the experiment or during the analysis. There is, second, the level of increasing publicity of the data. Third is the level of the usually very specialized software, which not only has to be made increasingly accessible to other researchers so that they have access to the analyses, but which also has to be designed in principle from the outset in such a way that it remains executable despite changing conditions (computer hardware, operating systems, further development of software, and libraries used).

In this book, we address the situation of large international research consortia such as the LHCb detector in particle physics or the IceCube, MAGIC, and Cherenkov telescope array (CTA) experiments in astroparticle and particle physics. Even though the question of how to manage the totality of all research data from the large research consortia can be answered only by the international consortia themselves, we develop approaches to solutions for the contribution of the individual analysis or the individual researcher. Even in the experiments, the management of the analyzed data is the responsibility of the researchers at the universities or institutes where the analyses were performed.

The situation thus differs fundamentally from those scientific fields in which the volumes of data collected are still very moderate. There, it is technically possible in principle to store all the data required for an analysis for the required period of ten years. Naturally, this also simplifies the aforementioned associated secondary tasks of data management.

Below we offer an abstract discussion of the management of large volumes of electronically stored and machine-processed research data regarding data reduction, publicity, and software.

6.2.1.1 Data Sources and Streams

Data analyses in experiments in the natural sciences and engineering are usually based on processing data sets containing a large number of features collected in parallel for each measurement. For storage space reasons, the individual measurements are recorded at a high frequency and must be processed in real time, at least at the lower levels. In addition to a large volume, the data is characterized by structures of very different complexity.

Data is collected and stored in a multi-step process. In the first step, several thousands to millions of electronic sensors are typically read out with potentially very high frequencies (GHz). The primary data rate here can be a few 1000 petabytes/year. This signal can be interpreted as the beginning of a continuous data stream. The storage of data volumes that predominantly contain a noise signal is neither possible nor reasonable. Within nano- to microseconds, therefore, an initial decision must be made as to which spatial or temporal signal configuration may contain meaningful information.

The signal selected in this trigger decision consists of remaining noise and physical signals that are of interest for different analyses. The different signatures are separated step by step in the following steps, usually using machine learning methods, and put in the form of a scientifically interpretable functional relationship as the solution to an inverse problem. The time available for the analysis depends on the compression of the data and the available computing capacities.

The decision models for data reduction developed with data mining methods are based on analyses of simulation results (Monte Carlo events), which, as "virtual reality", represent the expected physical situation with sufficient precision.

The aim of the management of the research data is to store the relevant data for a long time. The analyses performed should be reproducible. Re-analyses in the form of a new run through the analysis chain under possibly improved knowledge of mathematical or physical boundary conditions should also be feasible.

At least in the first analysis steps, data is necessarily lost (trigger). Also, it can be assumed that the importance of individual research papers (dissertations) does not necessarily justify petabyte-sized data storage. In such scenarios, data of the entire stream can be stored, at least exemplarily, before and after the selection and deconvolution steps. Thus, although a selection cannot be undone, the effect of a selection algorithm is traceable. This is especially true for the simulated Monte Carlo data and the models used for the decisions.

6.2.1.2 Data Formats and Software

The data formats, the operating systems used for analysis, and the specialized software rapidly evolve, just like computer hardware, and can change even during the runtime of the experiments. This usually leads to the fact that after a very short time, the data can no longer be read, and the analysis programs can no longer be run on the then more modern computers. One solution is to store virtual machines with the complete software used for the analysis along with the data. Such a procedure is necessary especially because the specialized software often documents the special characteristics and treatment of the data more exactly than any written documentation (e.g., efficiency corrections of detector components). For this purpose, suitable criteria catalogs must be developed for the joint storage of data and software.

6.2.1.3 Privacy

Scientific data is preferentially available to the scientists (collaborations) involved in its collection. The data (e.g., telescope images) should be made available to the interested scientific public after an appropriate period of time. However, during the analysis of the data in a collaboration, different access rights may be required. Initially, to avoid psychologically induced effects in analyses, only small parts of the entire data are used in what is known as blind analyses where scientists investigate, discuss, and test the analysis methodology for errors.

The "blindness" of the data is only lifted when all participating scientists in a collaboration are in agreement. Ph.D. students may be allowed to see different parts or properties of the data, depending on the state of their analyses. Because of the sheer size and complexity of the research data, manual problem-solving is increasingly bound to become error-prone in the future.

Other features of the data, as well as the programs used for analysis, which may be the personal, unpublished intellectual property of the scientists, may or may not be released to the public.

A publication system must therefore be integrated into the management of research data that can also manage access rights at the same time in a personalized and time-dependent fashion. Such a system must ultimately meet the same requirements that are also necessary for using medical data for research.

6.2.1.4 Sustainability

Many experiments in basic research are not only linked to large and sometimes singular expenditures; they also have runtimes of several decades. At the same time, a laboratory calibration of the entire detector is regularly not possible. Therefore, there is a great interest in being able to compare measurements and observations of well-understood events or sources—astrophysical objects with temporally constant fluxes and spectra—across experiment generations and also for the purpose of "cross-calibration" between experiments running at the same time. In principle, this opens up exciting possibilities such as separating previously unrecognized systematics on the detector or analysis side from "new physics" and thus either improve existing methods or directly pave the way to new discoveries. Likewise, in principle, an increase in integral sensitivity is possible by combining many data sets from different experiments. Such a sustainable use of research data directly requires the long-term preservation of data sets and associated programs and documentation analogous to the criteria and procedures outlined above. At the same time, however, it ideally includes open access to data sets, instrument response functions, and analysis tools. A best practice example is the development of a common data format for the high-level data of ground-based Cherenkov astronomy, as well as the open availability of some observational data of the Crab Nebula (Messier 1) analyzed in [1] with the open-source package gammapy. Individual fits and joint fits for comparing the results of the Fermi-LAT, MAGIC, VERITAS, FACT, and H.E.S.S. experiments have been produced (see Figure 6.1). This represents a significant step into the era of the long-term reuse of existing observations, such as during future observatory operations of the Cherenkov Telescope Array.

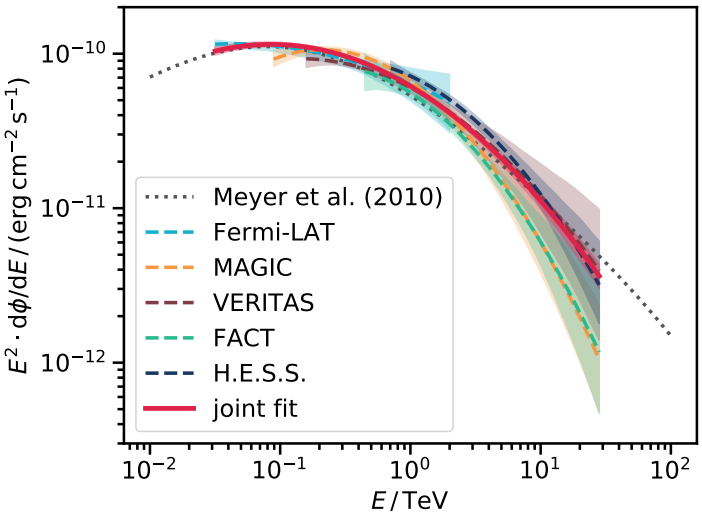


Fig. 6.1: Fits of the spectral energy distribution of the Crab Nebula (Messier 1); comparison of individual instruments, and common fit. The figure is taken from page 5 (Figure 2) of Nigro et al. [292], reproduced with permission ©ESO.

6.3 The FACT Open Data Project as an Example of Public Data Access

Maximilian Linhoff

Abstract: In particle and astroparticle physics, the data observed or simulated by the collaborations who are operating the instruments is almost always treated as proprietary, a closely guarded secret. No collaboration has made its raw data publicly available. This can also be explained by the sheer amount of data, which makes simple solutions like webservers infeasible, and by the complexity of the data, the large amount of domain knowledge, and often by proprietary tools required to analyze the data.

However, in 2017 the FACT collaboration (Section 2.2.1.2) published[1] a large dataset consisting of 17 hours of Crab Nebula observations and simulated data needed to perform the analysis. The data is available at all the data levels described in Section 7.3 and can be used for teaching and machine learning (ML) research and has facilitated many student projects and lectures that would be much harder or less interesting

1 https://factdata.app.tu-dortmund.de.

without this publicly available, complex data set.

In 2017, the FACT collaboration published a set of Crab Nebula observations comprising a total of 17.7 h of observation time together with accompanying simulations. The data is available at several processing steps, from the raw data of the telescope over intermediate steps of the extracted Cherenkov images up to the feature level, where each image is parameterized by a set of attributes suitable as input for classical ML algorithms.

The steps from raw data to image parameters are discussed in Section 7.3, the estimation of event properties using ML is described in Section 8.4 and the final step of unfolding the energy spectrum of the Crab Nebula from these observations is performed in Section 10.5 using this data sample as an example.

6.3.1 Available Data

Tab. 6.1: Key properties of the simulated data sets released as part of the FACT Open data set.

Primary	E_{min} / GeV	E_{max} / TeV	R_{max} / m	$N_{\text{simulated}}$	$N_{\text{triggered}}$
Gamma	200	50	270	12 000 000	1 914 812
Proton	100	200	400	780 046 520	509 652

The observed data consists of 218 runs taken under mostly favorable environmental conditions, with a small number of runs suffering from cloud coverage or other bad weather conditions. The accompanying simulation data is used as the labeled dataset required at several points in the analysis, mainly for training the machine learning models and calculating the instrument response needed for the unfolding.

The observed raw data is stored in FITS files using a custom compression extension, readers are available for Python[2], Java[3] and C++[4].

Simulated raw data is stored in standard-conform FITS files. Processed data is available either in FITS format for single runs or as HDF5 files for the whole dataset. The Python module `pyfact` provides utility methods to read the HDF5 files into `pandas DataFrames`.

Simulations of the particle cascades in the atmosphere were carried out using the software CORSIKA [194] resulting in the Cherenkov light reaching the telescope. The following simulation of the telescope was performed using CERES [98]. The most important properties of these simulations can be found in Table 6.1.

2 https://github.com/fact-project/zfits.

3 https://github.com/fact-project/fact-tools.

4 https://trac.fact-project.org/browser/trunk/Mars.

In total, 1.1 TB of simulated raw data and 0.9 TB of observed raw data were released. The amount of raw data belonging to this relatively brief observation time exemplifies the challenges connected with making raw scientific data accessible. In general, it is not feasible or even helpful to publish the raw data to a wider audience. Several challenges scientific instruments face when it comes to publishing raw data are:

- The sheer data volume.
- The required level of expert knowledge to do anything with these low-level data.
- Non-standard data formats that are needed to efficiently store the large data volume.

Due to these reasons, it is normally only feasible to publish data at higher abstraction levels. These are accessible to the majority of researchers in the field.

For gamma-ray astronomy, the abstraction level are lists of gamma-ray candidate events where each event has a single estimated value of the relevant quantities like energy and direction. To obtain scientific results from this, the inverse problem needs to be solved. This requires knowledge of the instrument response (the convolution kernel). Hence, this is also included in data releases at this abstraction level.

One example, which also includes the discussed data level of the FACT open dataset, is the combined analysis of Crab Nebula data from several currently operating gamma-ray observatories as shown in Figure 6.1. This is also the plan for the upcoming Cherenkov Telescope Array, the main users of the observatory will be able to openly obtain this abstraction level of the data.

For the FACT dataset, also intermediate data levels between the raw data and the high-level event lists for published. This proved very fruitful and most of the examples in this book make use of this dataset at the different levels of processing.

6.4 The DeLoreanSystem Architecture as Example of Experiment-Internal Access

Jens Teubner

Abstract: Modern research in high-energy physics depends on the ability to analyze massive volumes of data in a short time. *DeLorean* is a new system architecture for high-volume data processing in the domain of particle physics. It combines the simplicity and performance of relational database technology with the massive scalability of modern cloud execution platforms (Apache Drill, for that matter). The key ingredient to *DeLorean* is a compact *data synopsis* that can be used to approximate the result set of complex analysis queries. The approximation allows a guided search to obtain the actual query result with minimal I/O cost for large data sets.

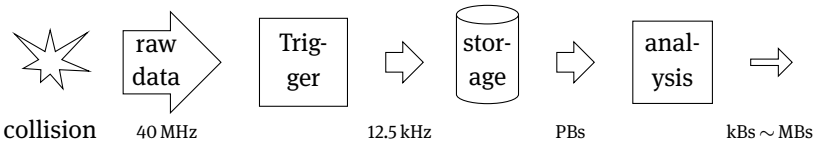


Fig. 6.2: LHCb processing pipeline. Data is collected at a rate of 40 MHz, then pre-filtered using a trigger system, and persisted. Stored data is then made available to various forms of analyses.

6.4.1 The Data Volume Problem at LHCb

The experiments at the Conseil Européen pour la Recherche Nucléaire (CERN) are often cited for their stunning data volumes and rates. And in fact, the detectors of the *LHCb* experiment (cf. Section 2.3.1) collect about 1 TB worth of data every second—a data volume far exceeding what could be analyzed in full with current hardware. Although an elaborate *trigger system* (cf. Section 4.2.1) disposes of more than 99.99 % of the raw data volume, data in the order of 10–20 petabytes remains left to be stored permanently and made available for analyses.

As their daily task, physicists around the world access these data to verify their hypotheses. When doing so, they are generally after *rare* events, which may occur with probabilities as low as 10^{-15}.[5] That is, out of the vast data volume stored at CERN, often only very few events have to be identified and retrieved efficiently.

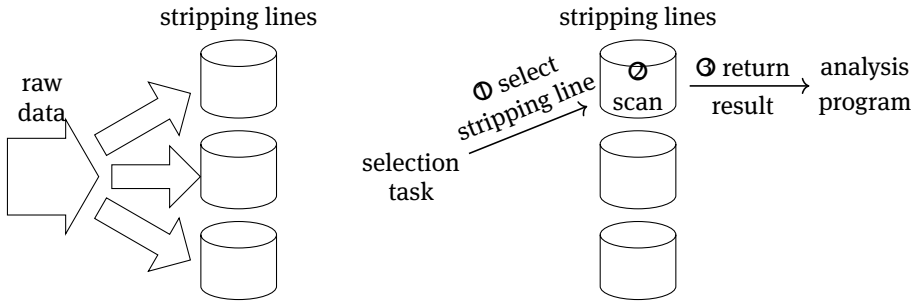
The *criteria* that characterize a (potentially) relevant data item can be diverse, ranging from simple predicates such as "return all collision events that produced a muon particle with an energy of at least x" to complex conditions on particle trajectories that can be inferred from observations. With the *DeLorean* system, whose concepts we will detail in this section, we aim to provide answers for simple classes of predicates with extreme efficiency—expecting that simple predicates can cut down the full data set to volumes where compute-intensive criteria become feasible even with conventional methods.

Our processing pipeline is summarized in Figure 6.2. Event data is collected at a rate of 40 MHz and fed through a trigger system. 10–20 PB of data are persistently stored every year and made available for analyses. Out of the large data volume, analyses typically select only a few kilo- or megabytes.

6.4.1.1 LHCb's "Stripping" Mechanism

Sifting through the complete data set for every analysis is not feasible with the data volumes involved. Therefore, LHCb physicists installed a mechanism that they refer to as "stripping."

5 To illustrate, as few as ten $B^0 \to \mu^+\mu^-$ decays were extracted from the entire data set of the first LHC run from 2010 to 2013 [225].



(a) Preprocessing. Acquired data is partitioned into "stripping lines."

(b) Data selection. Read stripping line(s) as a whole.

Fig. 6.3: LHCb "stripping" concept. The input stream is partitioned into a few hundred "stripping" lines after data acquisition (events may also be stored in multiple stripping lines). The only access mechanism allowed is to scan a full stripping line.

This mechanism is illustrated in Figure 6.3. After data acquisition, a separate preprocessing stage segregates all event data into so-called *stripping lines* on the storage cluster; each stripping line corresponds to a predefined selection criterion. In Figure 6.3a, the high-volume input stream coming from the left is partitioned and stored into multiple files ("stripping lines").[6] After "stripping," individual analysis tasks read one or more stripping line(s) as a whole (Figure 6.3b).

To date, a few hundred stripping lines are registered with the LHCb-system, which was found to be a compromise between selectivity and the cost of preprocessing. In fact, stripping lines need to have a selectivity below 0.5 %, so the cost of materializing data into stripping lines stays manageable.

A key limitation of the "stripping" approach is that arbitrary access to the full data set is still not feasible. Any filter criterion must be declared ahead of time, so a dedicated stripping line can be populated during preprocessing. It would be highly desirable to permit flexible *ad hoc querying* to scientists instead.

6.4.1.2 Database Technology to the Rescue?

Database systems can be extremely efficient at navigating huge amounts of data. This is achieved by providing efficient data representations, access methods, and distribution of available resources. In particular, indexing techniques promise the fast identification of relevant data. Searching for small subsets in extensive data collections is inherently challenging whenever the number of dimensions is high and the queries unknown or *ad hoc*, i.e., when query-relevant dimensions change frequently. However, even without considering a particular indexing technique, the hardware limitations of the underly-

6 Some events may appear in multiple stripping lines. This happens when they satisfy the predicates stated for more than one stripping line.

ing storage media are already a sizeable obstacle when trying to achieve reasonable speedups compared to a simple *scan* of the data.

A Primer on Database Indexing On a huge scale, when data far surpasses the amount that could be reasonably held in main memory, the cost of database access is measured by the number of required *I/O operations* on the storage devices. The performance of already *slow* (compared to main memory) bulk storage media, such as disk- or tape devices, degrade even further when data is not accessed in an orderly or *sequential* order. For example, reading a particular byte from a disk-based hard drive requires the retrieval of a whole block of multiple kB or MB and, more importantly, the physical movement of the disk's arm into the correct position on the disk. Because the data of interest is rarely located in a single, consecutive location and is more likely to be scattered *randomly* across the whole storage, the number of items identified by an index has to be very small in order to even compensate the hardware's performance degradation due to this *random access*.

Indexes for Multi-Dimensional Data In addition, the cost of *evaluating the index* in the first place has to be compensated as well. Popular multi-dimensional indices, such as the R*-tree [80] or k-d-tree, can reduce the number of data points to the relevant set at a reasonable access cost, provided that the dimensionality is sufficiently low. However, when the dimensionality becomes too high, or only a subset of the indexed dimensions is used in a query, the cost of their access, combined with the cost due to random access during result retrieval, quickly outweighs the gains [386]. Due to this *curse of dimensionality*, multi-dimensional indices either require super-linear search time or they grow super-linear in size, often both [361].

Weber *et al.* [386] recognized that this dilemma leaves *scans* as the only viable route to answer queries on high-dimensional data. In this spirit, it has been shown that even a simple scan on a vertically partitioned (in-memory) database, where each column is stored separately, outperforms multi-dimensional indices with selectivities as low as 1 % [196]. Due to the vertical partitioning, or *columnar storage*, dimensions not involved in a query can be ignored, effectively reducing the scan volume. When only a subset of the indexed dimensions in a multi-dimensional index is used, such a scan is almost always preferable [361].

An essential approach to this indexing problem is bitmap indices, such as FastBit [392]. They use a set of bitmaps per attribute, with a number of binary entries equal to the number of tuples in the database to indicate which tuple is associated with a particular attribute value. The bitmaps associated with qualifying values are retrieved for each attribute and then logically combined to identify data that qualifies for a query. The index result is then used to access the storage and retrieve the actual query result. Because they treat dimensions individually and bitmaps can be efficiently combined via bit-wise operations, bitmap indices are said to *beat the curse of dimensionality* [360].

However, the *size* of bitmap indices poses another important issue, which is also shared by many multi-dimensional indices. The number of bitmaps in a bitmap index is proportional to the number of indexed, distinct values of the attributes. To deal with high-cardinality attributes, most notably floating-point data, quantization strategies are used to reduce the effective number of different values and, therefore, bitmaps. In addition, compression is an integral part of bitmap indices. Despite the usage of compression, and although their space requirements do not increase super-linearly with the number of dimensions, they still often consume about the same space as the indexed data [237]. Furthermore, the quantization, also termed *binning* in this context, introduces false-positive candidates that increase the number of I/O accesses by inflating the selectivity of the index access.

6.4.1.3 Efficient Database Scans

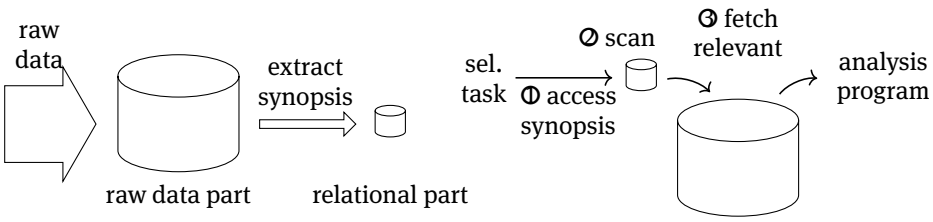
Because a simple, columnar scan is one of the most efficient ways to evaluate queries on high-dimensional data, many techniques were developed to accelerate it.

A straightforward way is via parallelization. Machines in large clusters can sift through data independently, which eventually motivated frameworks such as MapReduce [136], which allow for a straightforward and failure-tolerant implementation of scan-heavy workloads on large clusters of commodity machines. However, with limited project budgets, the ability to scale the performance simply by extending the cluster is also limited.

When many queries are issued to a system, the scanning effort can often be shared between *concurrent workloads*, which allows utilizing the limited I/O bandwidth of the storage more efficiently [238, 397].

Other approaches try to reduce the effective scan volume for a query. One way to achieve this is by first considering compact summary statistics about a hardware block (on disk). For example, the BlockIndex [393], which was implemented in the ADIOS I/O system [252], stores the ranges of values in each block as meta information. Comparing the value ranges with the predicates of a given query allows skipping blocks during a scan that can not contain tuples that might qualify. However, this technique can only be efficient when ranges of values in a block exhibit a statistical dependency on the data storage order. Although this property is available for some attributes of some spatio-temporal data sets, data from independent events in projects such as the LHCb is unlikely to offer it.

Another way to reduce the considered data volume during a sequential scan is by scanning a *reduced representation* of the data points. ColumnSketches [196] represent the values of a column by quantized values, such that each value requires fewer bits but still allows the evaluation of predicates. Again, this comes at the cost of an inflated selectivity because some of the data points might be false-positively identified as qualifying by the ColumnSketch. However, the effective data volume that must be considered



(a) Preprocessing. *DeLorean* extracts a *synopsis* or summary from the raw data set.

(b) Data selection. Synopsis is used to fetch only relevant items from the raw data store.

Fig. 6.4: *DeLorean* architecture. At data acquisition time, *DeLorean* extracts a compact *synopsis* of the raw data. At analysis time, *DeLorean* permits ad hoc selection criteria and fetches only a relevant data subset from the raw data part.

may be considerably reduced compared to a plain, columnar scan. The same technique was also used before for nearest-neighbor queries in the form of VA-files [386].

6.4.2 DeLorean: Optimized Scans for Efficient Data Access

To make the aforementioned advances in database technology accessible to physics, we prototyped the *DeLorean* platform at TU Dortmund University. *DeLorean* assumes a high-volume data store—such as embodied by the ROOT-based storage of the LHCb experiment at CERN—and pairs it with a novel *acceleration* engine, which is based on relational database technology.

The latter "relational part" of *DeLorean* is obtained by extracting a *synopsis* or *summary* from the original data set. Figure 6.4a illustrates this idea (where we refer to the existing ROOT store as "raw data"). The synopsis holds information that we can later use to access the "raw data part" in a very deliberate way, picking only a few selected items from the potentially huge data set (cf. Figure 6.4b). We will detail possible synopsis implementations in the following.

Feasibility and Required Selectivities For the *DeLorean* architecture to be feasible, the *size* of the synopsis must be small enough to not overstretch the available storage budget. *DeLorean* may replace space-consuming stripping lines, making the concept typically feasible from a storage space perspective.

The approach is much more sensitive than the achieved *selectivity* when accessing the raw data store at data analysis time. Every "pick" from the data store will incur a relatively high *access latency*. In practice, this means that *DeLorean* can only be cheaper than reading the complete (!) data set when $\ll 1\%$ of the stored events are actually fetched (cf. Figure 6.4b).

Tab. 6.2: Observed filter selectivities for a very small LHCb data excerpt and initial cuts to analyze $J/\psi \to \mu+ \mu-$ decays (extracted from `JPsi2MuMu` stripping line).

Cut	Candidates	Survivors	Selectivity
First	7 325 531	170 858	2.33 %
Second	170 858	2 542	1.49 %
Total	7 325 531	2 542	0.03 %

This requirement is given for all analyses that we encountered.[7] To illustrate this, Table 6.2 lists the selectivities of the initial cuts that implement the `JPsi2MuMu` condition at LHCb. Taken together, both predicates correspond to a selectivity as low as 0.03 %. That is, an efficient accelerator mechanism based on these two cuts can significantly reduce the query time that the analysis program will observe.

Scan-Based Acceleration As detailed before, multi-dimensional data sets favor scan-based processing. This is particularly true when the query patterns on the data sets vary strongly, as is the case for LHCb. Therefore, the accelerator for *DeLorean* is built mainly using scans.

The software frameworks at CERN heavily rely on ROOT, which—for our current context—provides a persistence mechanism for C++ objects. That is, the existing storage layer at CERN consists of serialized C++ objects. For analysis, these objects are de-serialized, and then handed over to C++/Python code for processing. To save disk space, ROOT files are aggressively compressed.

The beauty of ROOT is its seamless interplay with the existing analysis code. Over time, a very large library of analysis routines has evolved that is mostly written in C++. It is, however, challenging to make the approach run efficiently at very large scales. This is mainly for two reasons:

1. C++ object (de)serialization results in relatively complex data structures on disk and in memory. The strategy can, therefore, poorly benefit from modern hardware advances and deep memory hierarchies. By contrast, relational database engines intentionally stick to a very rigid and well-defined data model—one of the key ingredients to their excellent scalability.
2. The ROOT de-serialization mechanism will always read in C++ objects as a whole. In practice, many kilobytes have to be read from storage, even when a very simple characteristic (e.g., a *charge* value) would be enough to decide that the object can be skipped for a particular analysis.

7 In fact, the existing "stripping-line" mechanism has a similar limitation. Registered stripping lines must have a selectivity $\lesssim 0.5\,\%$ to not exceed storage budgets at LHCb.

To avoid—or at least mitigate—the above two problems, the synopsis of *DeLorean* uses a relational storage model. Thereby, the synopsis includes those fields of the dataset that are queried frequently (using simple, "sargable" predicates) and with high selectivity. With a scan on the relational side, we select *candidate matches*, and then de-serialize from the ROOT part only those C++ objects that are still promising.[8]

6.4.2.1 Optimizing Scans

Parallelization Using Drill/HDFS An advantage of scan-based processing is that scans can be parallelized easily, even to a massive scale. In *DeLorean*, we use Apache Drill to realize the relational part. Based on a Hadoop Distributed File System (HDFS), Drill provides a natural way to parallelize typical analysis queries at very large scales. Drill's column-oriented *Parquet* storage format can assist in optimizing the type of scans typical for *DeLorean*. A big bonus of using a Hadoop-based approach is its tolerance to failures, which allows the use of cost-efficient consumer hardware.

Columnwise Storage The characteristics in Table 6.2 make selection queries in *DeLorean* an excellent candidate to apply *column-store technology*. Storing data in a columnar fashion has two important advantages:

1. Queries must read from disk only those columns that are actually relevant for the particular filter task (such as *charge* and the *position* vector).
2. When a query consists of multiple selection predicates ("cuts"), data for later attributes must only be fetched from disk for rows where earlier predicates were satisfied. The query optimizer may optimize the order of predicate evaluation accordingly.

Both properties result in a reduction of the data volume that has to be scanned (read from disk) for filtering. For simple queries like those that we discuss here, reducing the I/O volume may directly translate into improved overall performance.

Lightweight Compression Column-oriented storage goes well together with *lightweight compression*. With a reduced overall disk memory footprint, the system may be able to read the relevant data from disk with fewer I/O requests and faster. Such an improvement will, of course, only be beneficial as long as the implied overhead—CPU cost for decompression in particular—does not outweigh the reduction in I/O cost. To this end, earlier work has developed compression schemes that are particularly lightweight and can provide high throughput. The most notable example is the PFOR family of compression schemes of Żukowski *et al.* [398]. In *DeLorean*, we opted for

8 ROOT supports the efficient, tree-based seeking of selected events.

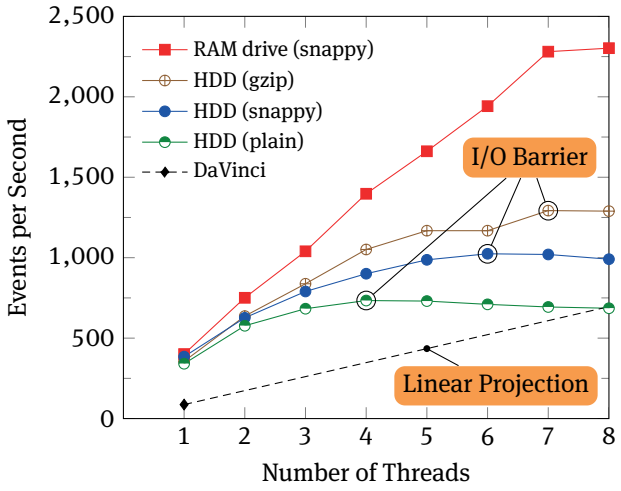


Fig. 6.5: Single node scalability of *DeLorean* compared with *DaVinci*.

Google's *Snappy* library and *gzip* because they integrate particularly well with our base platform Apache Drill.

6.4.3 Evaluating *DeLorean*

To test whether Apache Drill is a viable back-end option for *DeLorean*, preliminary experiments were conducted to be able to compare *DaVinci* (the existing software stack used in the LHCb collaboration) and *DeLorean*. In this experiment, both approaches execute the "indexing part" to show the scalability of *DeLorean*. All experiments have been conducted on an Intel Xeon E5-2609v2 dual-socket workstation (8 physical cores, no Hyper-Threading) with 64 GB of RAM and a 7200 rpm SATA HDD running Scientific Linux 6.7.

Using an out-of-the-box embedded Drill instance, we were able to achieve an event throughput increase of up to a factor of 4.6 (single-threaded). Unfortunately, a multi-threaded configuration of *DaVinci* is currently not available in our laboratory setting. For the sake of fairness, we assume a linear scalability for *DaVinci* (best case). Figure 6.5 shows the scalability for different compression algorithms compared with the *DaVinci* projection. One notable result of the experiment is that *DeLorean* even outperforms the linear projection of *DaVinci*.

Figure 6.5 clearly shows that the HDD-bound experiments hit the I/O barrier.[9] Here, we can see that compression can leverage the I/O bottleneck significantly. *gzip*, the "heaviest" of the compression algorithms, performs best in this scenario, so it might be worthwhile to invest in higher compression ratios using domain-specific compression algorithms (like run-length encoding or delta encoding).

6.4.4 Looking Ahead

The massive data volumes at LHCb are a prime example of the size and the complexity of modern, data-intensive applications. The example also shows that classical database techniques, indexing in particular, can hardly support the given complexity; but naive approaches cannot deal with the data sizes encountered here.

DeLorean illustrates a possible way out of the dilemma based on proven techniques from the database literature. The combination of a "naive" (i.e., scan-based) access mechanism and a compact data representation can help to master the size and complexity of the data at hand. Thereby, scans are trivial to implement, parallelize, and scale. Our compact synopsis makes the approach viable for LHCb's data volumes.

It is easy to see that the pattern behind *DeLorean*—scan-based access to a compact data structure—is highly versatile. Not only could it be applied equally well to other application classes. We are also actively working on variations to the *DeLorean* scheme, e.g., by applying (lossy) compression (to reduce scan volumes even further, at the price of low false-positive rates) or by breaking up *DeLorean*'s strictly column-oriented view. Combined representations of small groups of attributes can, in fact, result in even further volume reduction, with negligible impact on the result quality. Conversely, extending the synopsis by pre-materialized information may increase scan volumes somewhat; the return can be better response times and reduced CPU complexity when asking queries whose bits and pieces were pre-materialized before.

9 The RAM drive experiment is there to verify the I/O bottleneck. Regarding the massive amount of data, it is unrealistic to process the whole data set in memory.

7 Monitoring and Feature Extraction

7.1 Introduction

Wolfgang Rhode

First, the data recorded by an experiment consists of the individual detections of electromagnetic interactions that are spatially and temporally limited in the smallest subdetectors. These signals are used for data analysis in two ways.

In particular, using Monte Carlo methods (as presented in Chapters 1 and 5) to develop and optimize machine learning methods for data analysis requires a perfect description of the detector in Monte Carlo. A dedicated analysis of unprocessed or barely preprocessed (e.g., calibration, if necessary) data can provide spatial and temporal monitoring of the detector function. Failures, malfunctions, or gradual changes in parts of the experiment are detected and can be either corrected or integrated into the Monte Carlo simulation.

Usually, the low-level data contains the information initially sought for signal background separation or later for finding a correlation with a physical property only indirectly. Therefore, the second analysis path leads to the extraction of these features. What will inspire the chosen path will be, of course, the domain knowledge of the analyzing physicist. Also, on this basis for feature extraction, different machine learning methods are evaluated and optimized problem-specifically. Conversely, this procedure also leads to the fact that all recorded signals that do not carry any information that can be used later on may be removed from the dataset. This significantly reduces the amount of data to be stored. Both parts of the feature extraction save resources from all areas (CPU, memory, storage, network traffic, ...) for further analysis.

The examples presented in this chapter are taken from gamma astronomy for monitoring and from gamma and neutrino astronomy for feature extraction.

7.2 Feature Extraction and Selection in IceCube

Tim Ruhe

Abstract: In neutrino astronomy, data can be represented in the form of simple tables with a large number of columns and an even larger number of rows. Columns represent event properties extracted by so-called reconstruction algorithms and include, for example, the estimated energy and direction of an incoming particle. At later stages of an analysis, these event properties or *features* serve as input for machine learning

https://doi.org/10.1515/9783110785968-007

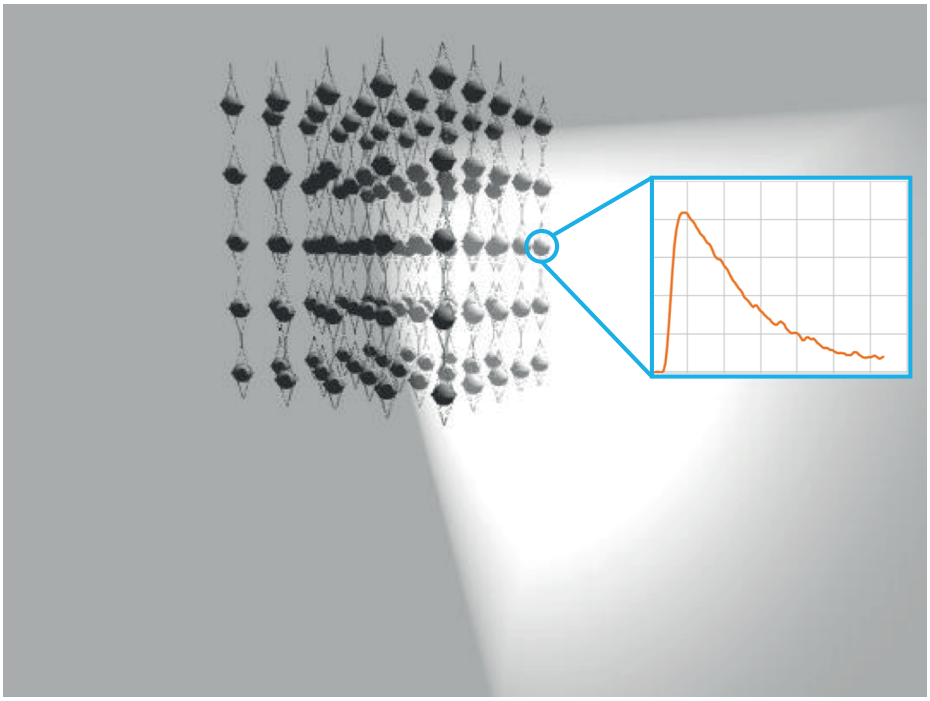


Fig. 7.1: Schematic view of a Cherenkov light cone, depicted in light grey, entering the IceCube detector from the right-hand side. While a particle traverses the detector, several optical modules register the Cherenkov light and transform it into a time series of charges. Graphic: courtesy of the IceCube collaboration.

classifiers. However, not all extracted features can be used for several reasons. Certain features may be entirely useless in classification, especially if they are constant, while others are expected to cause a bias in the automated classification. Furthermore, the information contained within certain features may be redundant and may be caused, say, by the extraction of the same event property (e.g., the zenith angle) by multiple reconstruction algorithms. Last but not least, the distributions between simulated and experimental data may be different. The aforementioned arguments ask for a detailed feature selection to be carried out prior to the classification of events via machine learning models. The reduction of dimensionality while maintaining an amount of information that is sufficient to obtain a stable and reliable classification is an active field of research. This chapter will provide an overview of feature selection techniques utilized in analyses of data obtained with the IceCube neutrino telescope.

In IceCube (see Section 2.2.1.1), the features later utilized in physics analyses are extracted from the light-pattern produced from Cherenkov radiation from charged secondary particles. Figure 7.1 depicts a Cherenkov light cone (shown in light grey), entering the detector from the right-hand side. While the charged particle initiating

the cone traverses the detector, it is followed by the light cone. Optical modules are hit by the Cherenkov photons and, as a consequence, register so-called *hits*. These hits are transformed into a time series of charges, generally referred to as a wave form (see inset of Figure 7.1). The wave form itself is a consequence of the fact that South Pole ice is a natural medium, which contains a depth-dependent amount of impurities. The impurities, mainly dust, cause a significant amount of light scattering. As a consequence, the peak of the Cherenkov radiation spectrum is smeared out to a so-called wave form. The depth-dependent optical properties are, in fact, one of the largest sources of systematic uncertainties for the IceCube detector. For details on the determination of the optical properties of South Pole ice and their effect on the photon detection in IceCube, we refer to Refs. [35] and [26]. In summary, the optical modules thus transform an optical into an electrical signal, which is then used for feature extraction.

Some of the extracted features are relatively simple and do not require a sophisticated reconstruction. The total charge (Q_{tot}) accumulated by all optical modules during an event is a straightforward example, as is the number of optical modules N_{Ch} (sometimes referred to as the number of channels) hit in an event.

The second category of features is extracted via the use of simple algorithms, which assume the particle's trajectory through the detector to be a straight line. Accordingly, these algorithms are referred to as line-fits. The line-fit ignores the optical properties of the detection medium, as well as the geometry of the Cherenkov cone, and attempts to minimize a χ^2 of the form [44]:

$$\chi^2 = \sum_{i=1}^{N} (r_i - r_{\text{LineFit}} - v_{\text{LineFit}} \cdot t)^2 \,. \tag{7.1}$$

In Equation 7.1, the r_i correspond to the positions of the optical modules and r_{LineFit} denotes the extracted particle trajectory. v_{LineFit} describes the speed of light in the medium and is also extracted from the line-fit algorithm. The three variables are connected via the time t required for photons to travel from their point of origin somewhere along the particle trajectory r to an optical module where it is finally detected [44]:

$$r_i = r_{\text{LineFit}} - v_{\text{LineFit}} \cdot t. \tag{7.2}$$

Although the assumptions for the line-fit algorithm do simplify the problem, its outcome already provides valuable insight into the detected event. The extracted speed of light, for example, can be utilized to discriminate between track- and cascade-like events. Due to the geometry of a cascade-like event, basically a ball of light expanding within the detector, the reconstructed values of v_{LineFit} are significantly smaller than for track-like events. Furthermore, the line-fit algorithm provides a first guess, utilized as a seed value for more sophisticated reconstruction algorithms.

Using a likelihood description, the task of feature extraction can be generalized to estimate a set of parameters $\vec{a}$ given some measured observables $\vec{x}$. For independent components, x_i of $\vec{x}$, $\vec{a}$ can then be estimated by maximizing a likelihood of the

form [44]:

$$\mathcal{L} = \prod_i p(x_i|\vec{a}), \tag{7.3}$$

where $p(x_i|\vec{a})$ is the probability density function of measuring x_i given $\vec{a}$. For simplicity, a Cherenkov cone initiated by an infinite muon track is assumed. Said muon track is then described by [44]:

$$\vec{a} = (\vec{r_0}, t_0, \hat{\vec{p}}, E_0). \tag{7.4}$$

In Equation 7.4 $\vec{r_0}$ is an arbitrary point on the muon's trajectory, which is passed by the muon at time t_0 with energy E_0 along a direction $\hat{\vec{p}}$. Along this trajectory, Cherenkov photons are emitted at a fixed angle θ_C (also referred to as the Cherenkov angle), with respect to $\hat{\vec{p}}$. The actual feature extraction is then performed by minimizing $-\log \mathcal{L}$ with respect to $\vec{a}$.

The measured observables include the times t_i at which a certain optical module is hit. In that case, the likelihood in Equation 7.3 becomes [44]:

$$\mathcal{L} = \prod_i p(t_{\text{res},i}|\vec{a}), \tag{7.5}$$

where the so-called time-residual $t_{\text{res},i}$ is given as the difference between the arrival time of unscattered photons (in the context of IceCube, often referred to as a *direct* photon) and the actual arrival time. As part of the reconstruction, different likelihoods are evaluated. These include the *hit and not hit likelihood*, the *amplitude likelihood* and the zenith-weighted *Bayesian likelihood* [44].

In addition to the arrival direction of the incident particle, its energy is important in many analyses. As the energy of the particle is not directly accessible as a consequence of the indirect nature of the measurement, energy-related observables are derived instead. These energy-dependent observables are especially important for reconstructing neutrino energy spectra (see Chapter 10). In IceCube, various algorithms are used for the reconstruction of energy-dependent variables (see for example [32]). The total charge (Q_{tot}) is a particularly straightforward example. The amount of charge collected by a Digital Optical Module (DOM) (see Section 2.2.1.1) is proportional to the amount of Cherenkov radiation observed by said DOM, which is itself proportional to the energy of the incident particle. Q_{tot} thus provides an intuitive approximation for the particle's energy, which is relatively simple to reconstruct. This proxy does, however, not account for the detector geometry or the properties of the detection medium. It becomes clear that a long track, which traverses the entire detector, will emit more Cherenkov light within the detector than a so-called *corner clipper*, which only traverses a small portion of the detector. The same holds when comparing particles traversing the deep clear ice with particles traversing large layers of dust, where a significant portion of the light is absorbed. To account for these effects, more sophisticated reconstruction algorithms are required.

One example of a more sophisticated energy reconstruction is the so-called *truncated mean* [32], which discards large energy losses that occur stochastically along the

track (mostly via *bremsstrahlung*) and thus distorts the energy reconstruction. Another example is the use of the muon energy as a proxy, which is more directly accessible than the energy of the neutrino but also suffers from the limitations due to the detector geometry and the detection medium described above. A more accurate reconstruction of the particle energy and certain other properties is obtained by utilizing deep neural networks (see Section 9.2.2).

For an machine learning-based analysis, it is often beneficial to generate new features in addition to the existing ones. These new features can be based on the existing ones, which has the additional advantage that the information contained in two or more reconstructed variables is condensed into one feature. This approach has the advantage that a certain amount of resources can be saved during training, testing, and application of the utilized classification algorithm. The zenith angle θ of incident particles in IceCube is a relatively simple example. As this property is extracted by several reconstruction algorithms, the agreement between the different algorithms with respect to θ does provide a valuable handle to discriminating between well and poorly reconstructed events [16]. More sophisticated approaches to feature engineering in IceCube are discussed in [91].

Feature Selection While the wave forms obtained by the DOMs can be directly used in Deep Learning applications (see Section 9.2), a more conventional feature extraction requires the events to be processed to what is internally referred to as level 3 [91, 329, 341, 342], via standardized reconstruction algorithms. The type of algorithm used for processing and, thus, also for feature extraction, strongly depends on the observation channel (see Section 4.2.3). We refer to Section 4.3.3 for details on IceCube data levels. At level 3, which forms the starting point for the machine learning-based analyses discussed in this section, each event is characterized by more than 1200 attributes [91, 329, 341, 342]. In order to reduce the computational cost of the subsequent machine learning analysis, it is beneficial to reduce the dimensionality of the feature space. The challenge is, however, to achieve such a dimensionality reduction with a minimal loss of information. A method is needed that takes into account feature interactions, enhances the learning performance, scales well for large feature sets, and is stable with respect to unseen data of the population. The *FastEnsemble* algorithm offers a fast computation of such a method [344]. It has been successfully applied to the IceCube data [332]. In the following, the method is described in detail focusing on the impact of feature selection and stability for the analyses in neutrino physics.

Comparing the feature selections carried out on data taken with IceCube in the 59-, 79- and 86-string configuration, one finds that certain recurring patterns can be identified and that differences mainly arise from improvements in the available feature extraction algorithms and from improvements in the detector itself. Compared with the 59-string configuration, the 79- and 86-string configurations exhibit a more homogeneous instrumentation (see Section 4.2.3 for details on the different detector

configurations). Furthermore, an improved understanding of certain detector properties such as that of ice has been gained over time.

In a first step, attributes that are either constant or might become a source of potential *bias* are removed from the feature set. Constant attributes can obviously not be utilized to discriminate between signal and background events, and potentially biasing features are excluded to achieve an unbiased selection. A potential bias in the selection might, for example, arise from including timing information or identifiers. If these are different between simulated signal- and background events, a classifier might pick up these differences. Such behavior could result in a classifier that does not generalize to unseen events. The same step excludes features that only exist for simulated events, e.g., the true zenith angle or the true particle energy, and attributes with running numbers (mainly IDs).

In a second step, NaNs (not a number) and other missing values are handled. The strategies for handling these, however, differ between different analyses. The straightforward approach is to simply exclude features above a certain threshold with respect to missing values [16]. Although NaNs are often a nuisance in data analyses because many numerical algorithms cannot handle these effectively, they can also be a source of valuable information. In IceCube, for example, the failure of a reconstruction algorithm is often encoded as a NaN. This failure can, however, provide valuable information on the quality of an event, especially if more than one reconstruction failed or the failure occurred for a significant fraction of events. In order to keep this information, which can later be handled by a machine, it is thus beneficial to replace the missing values by a real number. When doing so, however, one should aim at choosing numbers outside the range of the feature, as a bias is constructed otherwise.

In a next step, attributes carrying identical or almost identical information are excluded. Encoding the same information in two or more different features does not add any value to the feature space. This exclusion is easily achieved by computing the correlation between different features via, say, the Pearson correlation coefficient. After the correlation is computed, one ends up with n sets of highly correlated features, where only one such feature per set is retained [91, 329, 341, 342].

Applying the feature selection steps discussed above reduces the dimensionality of the dataset from more than 1000 to 200–400 features. The exact number depends on the detector configuration, as well as on the chosen thresholds. This number can, in principle, be handled by machine learning algorithms, but one should keep in mind that in most real-life analyses, the algorithm will be trained, validated, and tested multiple times. Even in the rare case where the setup of the machine learning process does not require any further optimization or debugging, the process will have to be carried out on example sets generated with different simulation settings, in order to avoid large uncertainties in the event selection, due to a specific choice of simulated events. It is thus advisable to aim for an efficient machine learning process and a feature set that does not contain useless information with respect to the target variable. Furthermore,

certain types of learning algorithms are bound to fail when trained with such a large number of variables.

From this point on, a manual feature selection becomes rather challenging, as, in addition to the discriminative power of an attribute, its correlation to other attributes should be taken into account to avoid redundant information. One naïve example includes reconstructed features, which are returned by various reconstruction algorithms and may individually provide a relatively strong separation between signal- and background events. Adding more than one of these attributes will, however, not aid the selection because the information encoded in the attributes is more or less redundant. Hence, automated feature selection methods are generally applied at this point of the analysis [344].

Numerous feature selection algorithms exist, and using a simple forward selection is in many cases a feasible solution. In the context of astroparticle physics, however, the use of a forward selection (or backward elimination) is not sufficient. The reason for this does not lie with the aforementioned algorithms but with the data, which exhibit numerous features with similar information. As already discussed above, a selection of such *redundant* features, which will occur in a native way when a forward selection is used (simply because of their strong correlation with the target variable), does not—or at least not significantly—improve the event selection. The Minimum Redundancy Maximum Relevance (MRMR) algorithm [144, 190] is an iterative algorithm that addresses this issue by taking into account the relevance of a feature with respect to the target variable, as well as its redundancy with respect to the variables already selected in previous iterations. For details on MRMR we refer to Section 3.1.4 and [144, 190, 344]. MRMR has been applied in analyses of atmospheric muons [170, 171] and muon neutrinos [16, 18, 91].

Feature Selection Stability As any feature selection is carried out on an example set with a finite number of events, the selection can be affected by statistical fluctuations. In order to avoid a bias, the stability of the feature selection should be carefully monitored via a stability measure like the Jaccard index [213] or Kuncheva's index [235]. The Jaccard index is given as [213]:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}, \tag{7.6}$$

whereas Kuncheva's index is defined as [235]:

$$I_C(A, B) = \frac{rn - k^2}{k(n - k)}. \tag{7.7}$$

In both equations, A and B represent subsets of features. In Equation 7.7 k is the size of the subset, whereas $r = |A \cap B|$ denotes the cardinality of the subset. The total number of features is given as n. While the Jaccard index can take values between zero and one, Kuncheva's index is bound below by -1 and can take a maximum value of 1. It should be noted that Equation 7.7 is not defined for $k = 0$ and $k = n$. As these are the trivial

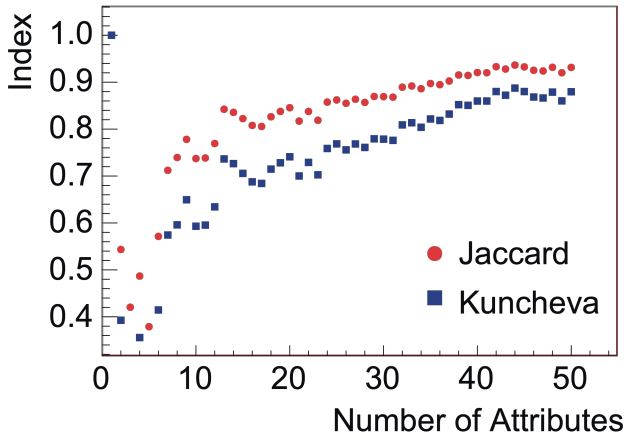


Fig. 7.2: Feature selection stability for a MRMR . The stability of the selection increases with an increasing number of attributes and starts to saturate for $n_{\text{Att}} \geq 20$. Figure taken from [329].

cases of selecting either no or all features, which render the comparison of feature sets useless, this does not pose a problem. A good agreement between the selected subsets is achieved when the indices are close to one, whereas a poor agreement is observed in case the indices are close to 0 (Jaccard) or −1 (Kuncheva). Examples regarding the stability of the feature selection carried out with MRMR, and a simple forward selection are depicted in the Figures 7.2 (MRMR) and 7.3 (forward selection). One finds that for MRMR, the stability of the selection increases with an increasing number of selected attributes. The opposite case is observed for the forward selection, and both indices are found to decrease with an increasing number of selected attributes. For selection algorithms, both indices are equal to one in case only one attribute is selected. This indicates the existence (and selection) of a feature with a very strong correlation to the class variable.

Further Remarks The best and most stable feature selection becomes useless in case the selected features are subject to disagreements between simulated- and experimental data. As experiments in astroparticle physics are sometimes subject to such mismatches, all selected features should be monitored with respect to this agreement. This will allow for training a well-generalizing classifier, which can be applied to experimental data. It is mostly a matter of taste where in the feature selection such a check for potential disagreements should occur. One may, for example, analyze the selected features after the selection is completed. This has the advantage that relatively few attributes must be closely studied, and features with detected mismatches can be manually excluded from the selected set. However, by doing so, one runs the risk that an entirely new

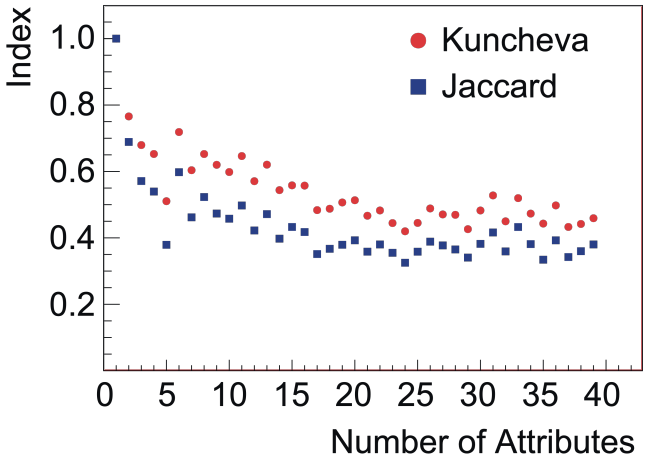


Fig. 7.3: Feature selection stability for a simple forward selection. A decreasing stability is observed with an increase in the number of attributes. Figure taken from [329].

feature selection becomes necessary if too many selected features have to be excluded. By contrast, the opposite case of investigating and excluding attributes before running an automated feature selection can be time- and resource-consuming. For more details on the data/MC mismatches and their detection via the use of machine learning, see Section 5.3.3.

As an additional sanity check on the feature selection, it is often helpful to study the importance of the individual features with respect to the classification task they have been selected for. When using MRMR with IceCube data, it has been observed that the features selected in the later iterations of MRMR show little to very little importance with respect to the classification task. Whether some of those less important features can be manually excluded from the feature set depends on the available computational resources and the classification task at hand. Moreover, to some extent, such an exclusion can also be considered a matter of taste.

7.3 Feature Extraction for IACTs

Maximilian Linhoff
Jens Buß
Lukas Nickel

Abstract: The classical approaches for Imaging Atmospheric (or Air) Cherenkov Telescope (IACT) data analysis provide physicists with a multi-stage data analysis chain that is responsible for the estimation of the physical properties of each air shower from the raw telescope data. An essential step in this chain is the reduction of the time series for each pixel to a set of features characterizing the events by an IACT. In this section, we describe how raw data from an IACT is generally handled to determine a parametrization of the shower image. These features are used in subsequent stages of the analysis chain to estimate the physical properties of an incoming cosmic particle (see Section 8.4). In order to give an example for such a low-level analysis, we discuss the implementation in the analysis package *fact-tools*. This package was developed within CRC 876 as an extension to the *streams* framework [87] for the low-level analysis of data from the FACT experiment [88].

Typically, each IACT's raw data event contains the digitized and uncalibrated electronic signal of its camera's pixels, which is often a time series. An event is defined as a single trigger of the IACT's data acquisition, as described in Section 4.2.2. In this section, we present the consecutive steps performed by the *fact-tools*:

i) calibrating the raw data and removing artifacts from the time series,
ii) reducing the pixel's time series to the number of photons and their mean arrival time per pixel,
iii) selecting those pixels that belong to the shower image (aka image cleaning), and finally
iv) reducing the information from the selected pixels (photons arrival times) to an ensemble of features per event.

These features characterize the spatio-temporal distribution of photons in the shower image using descriptive statistics, fits to the photon distribution, and hand-crafted features known to domain experts as Hillas parameters. These features are used in subsequent steps to determine the class of the primary particle, its energy, and direction. The main goal of the feature extraction is to reduce the amount of data while keeping most of the information relevant to the task at hand, i. e. engineering features that are relevant to the classification or regression tasks and not redundant to other features.

7.3.1 Introduction

The "classical" analysis approaches for IACTs aim at reducing the dimensionality of telescope data by using a multi-stage data analysis chain. Following CTA's terminology, this chain is responsible for the transition from data level `R0` to data level `DL2`. The goal is the extraction of meaningful parameterizations of the observed particle shower images. The provided features are used to estimate the properties of the gamma events (`DL3`) by applying machine learning models trained on simulated data. The machine learning applications are discussed in Section 8.4.

As described in Section 4.3.2, the measurements of the photo sensors of an IACT are digitized and stored (or streamed) as data level `R0` for further steps of the analysis chain. These data must be calibrated to get voltage time series at data level `DL0`. However, the operations that are necessary for the calibration are specific to the hardware of each IACT. Therefore, the calibration is not further explained in this section. Examples for raw data calibrations of some of the currently operating IACTs can be found in: [334, 352].

7.3.2 Image Extraction

The first step in the event reconstruction of IACT data consists of calibrating the raw wave forms in order to compensate for differences in the electronic modules. Inherently, different pixels will have varying properties regarding electronic noise, time offsets and signal gain. Using calibration measurements with lasers or a closed camera shutter, these can be corrected, and charges can be translated into photo electron counts.

Reducing data from a series of normalized charges to images is done by integrating over a selected window of the wave forms. Since the Cherenkov pulses are only seen for a few nanoseconds, the readout window will generally be larger than the part of the wave form containing the photon signal. While the width of the integration window can often be seen as a constant, the position needs to be chosen for each pixel either individually or by averaging the wave form with surrounding pixels.

Integrating the wave forms over the specified window then reduces the data from $n_{\text{readout bins}} \times n_{\text{pixels}}$ to $2 \times n_{\text{pixels}}$ with the two values per pixel corresponding to the total intensity and the time of the Cherenkov signal. The position of the inflection point in terms of the n-th bin of a time series is translated to a time relative to the trigger giving a measure for the temporal evolution of the shower, referred to as signal arrival time. The integral value over the calibrated wave form charge is expressed in terms of photoelectron counts, giving a measure for the light intensity. This process is illustrated in Figure 7.4 for the case of a pixel containing a signal in the readout window.

Thus, two images are obtained for every recorded event: One of the photon counts and one of arrival times.

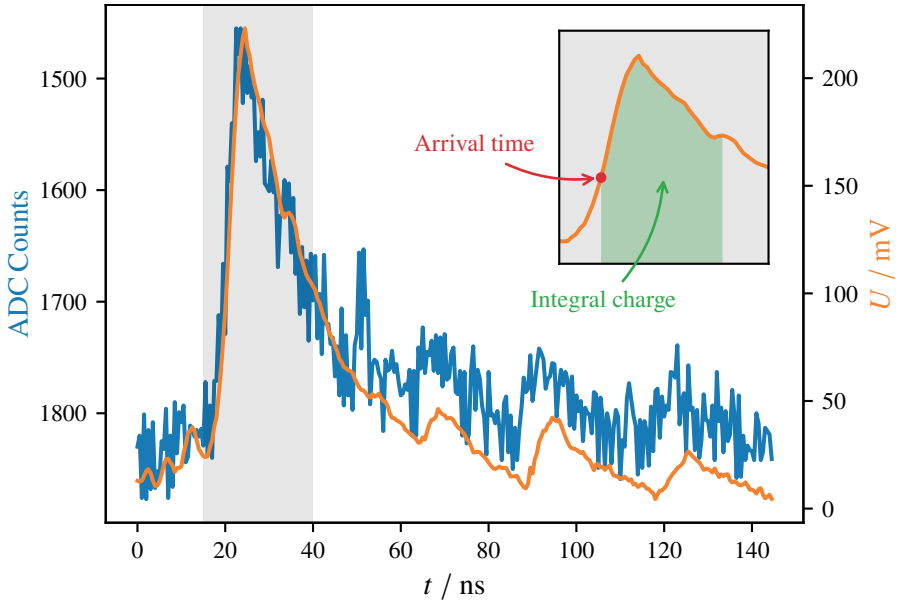


Fig. 7.4: Image extraction for a single pixel containing a pulse. As a first step, the raw wave form (blue) is calibrated. The calibrated wave form (orange) is then integrated in the part containing the signal peak (upper right). The start of the integration window is identified with the arrival time, and the integral under the wave form gives the photon counts.

7.3.3 Image Cleaning

A typical gamma-ray shower does not extend over the whole camera. In the image cleaning step, a subset of pixels likely to contain the Cherenkov signal is selected. The selection is based on the image values obtained in the extraction step, assuming that the pixels containing the signal are brighter than those only containing noise. Additionally, signal pixels are required to be correlated in terms of their arrival times as is evident from the shower development. A complete image cleaning usually consists of some combination of the following steps:

1. Selecting pixels above a certain charge threshold
2. Removing pixels with less than a given number of selected neighboring pixels
3. Selecting pixels above a second, lower threshold if they are adjacent to selected pixels
4. Removing pixels with less than a given number of selected neighboring pixels arriving at a similar time

As a result of this step, pixels not likely to contain the Cherenkov signal are discarded and subsequent analysis steps focus only on the selected part of the image(s). Figure 7.5

shows the selection of signal pixels by constructing a mask on the full (photon counts) image.

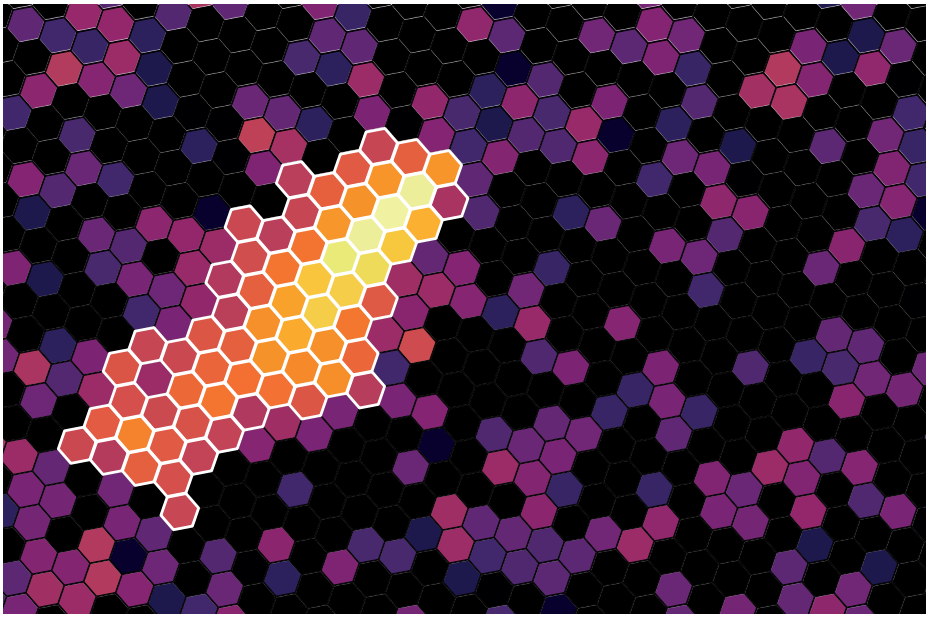


Fig. 7.5: Zoomed-in view of a recorded Cherenkov image showing pixels in the vicinity of the Cherenkov signal. Out of the full image, a subset of pixels with high photon counts is selected (white border).

7.3.4 Image Parametrization

In comparison to shower images produced by hadronic primaries (which constitute the main background class in gamma-ray data analysis), the shower images created by gamma rays are rather homogeneous, single-blob images elongated along the shower direction. To parametrize these properties, Michael Hillas introduced a set of features when working for the first IACT, the Whipple telescope [199]. His initial set of parameters has been extended over time. However, the classical set of parameters is still used regularly and considered the most important feature also in modern machine learning-based analyses. The Hillas features can be calculated by performing a principal component analysis on the two-dimensional images. Let x_i and y_i be the coordinates of the i-th pixel of the camera in the focal plane and w_i the number of photons of that pixel, and $\mathcal{C}$ the set of pixels selected by the image cleaning. The first Hillas parameter

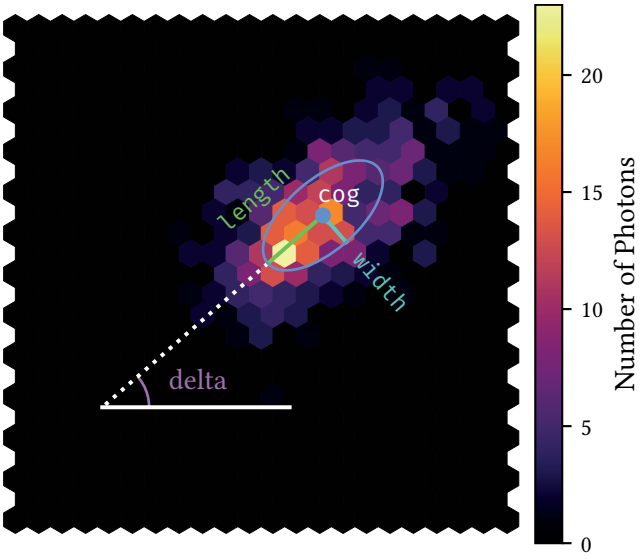


Fig. 7.6: The classical Hillas features calculated for a simulated gamma-ray event in the FACT camera.

is just the sum of the weights:

$$\text{size} = \sum_{i \in \mathcal{C}} w_i$$

We start by calculating the *center of gravity* (cog) of the shower by calculating the weighted mean of each coordinate:

$$\text{cog}_x = \frac{1}{\text{size}} \sum_{i \in \mathcal{C}} w_i \cdot x_i \qquad \text{cog}_y = \frac{1}{\text{size}} \sum_{i \in \mathcal{C}} w_i \cdot y_i$$

and translate into coordinates with the origin at the cog

$$\Delta x_i = x_i - \text{cog}_x \qquad \Delta y_i = y_i - \text{cog}_y$$

Now we have to calculate the weighted covariance matrix of the pixel coordinates:

$$\text{Cov}(x, y) = \frac{1}{\text{size}} \sum_{i \in \mathcal{C}} w_i \cdot \Delta x_i \cdot \Delta y_i$$

and accordingly calculate $\text{Cov}(x, x)$ and $\text{Cov}(y, y)$ resulting in the 2×2 covariance matrix $\boldsymbol{V}$.

As the final step, we calculate the eigenvalues and eigenvectors of $\boldsymbol{V}$. Let λ_0 be the larger and λ_1 be the smaller eigenvalue of $\boldsymbol{V}$. We then calculate the Hillas parameters length and width as

$$\text{length} = \sqrt{\lambda_0} \qquad \text{width} = \sqrt{\lambda_1},$$

which correspond to the standard deviations along the principal components of our light distribution.

The rotation angle of the main component with respect to the x-axis is obtained via the corresponding eigenvector v_0 and usually called δ:

$$\delta = \arctan(v_{01}/v_{00}).$$

Especially for estimating the shower direction, it is helpful to calculate skewness and kurtosis along with the principal component. Often, a linear regression of the arrival times in the rotated system is also performed. A simplified python function to calculate the Hillas parameters is shown in Listing 7.1.

Listing 7.1: Simplified python code to calculate Hillas parameters given pixel coordinates and the image. This code does not handle edge cases such as size = 0 or width = 0 and does not calculate higher order moments along the main axis. Adapted from ctapipe [296].

```
import numpy as np

def hillas(pixel_x, pixel_y, image):
    '''Calculate hillas parameters'''
    size = np.sum(image)

    cog_x = np.average(pixel_x, weights=image)
    cog_y = np.average(pixel_y, weights=image)

    delta_x = pixel_x - cog_x
    delta_y = pixel_y - cog_y

    cov = np.cov(delta_x, delta_y, aweights=image)

    # eigh assumes a hermitian matrix (here real and symmetric)
    # and guarantees the order of the eigenvalues
    eig_vals, eig_vecs = np.linalg.eigh(cov)
    width, length = np.sqrt(eig_vals)

    delta = np.arctan(eig_vecs[1, 1] / eig_vecs[0, 1])

    return size, cog_x, cog_y, length, width, delta
```

Additional features describing the image morphology can be calculated. This includes the number of pixel groups in the set of selected pixels, the proportion of light at the edge of the camera, and proportions of light in some regions of the image with respect to size, e. g. the proportion of light inside the ellipse with semi-major axes length and width as shown in Figure 7.6.

7.4 Monitoring the Telescope via Data Summarization

Sebastian Buschjäger

Abstract: In astroparticle physics, we naturally face machines with long lifetimes and long-running processes since a detector instrument such as a telescope is often built, used, and maintained for several decades. Thus, it is imperative to monitor the behavior of the detector in order to assure a certain data quality during its lifetime. While operators closely monitor the behavior of their detectors, they cannot possibly review, e.g., 60 events per second in real time. However, a small and well-chosen sample of the measurements can help to monitor the behavior of the detector and pinpoint unexpected results early in the processing pipeline. To be meaningful, such a data summary must be small, comprehensive, and computed on time. Due to their compelling theoretical properties, submodular functions have been the focus of data summarization algorithms as they naturally capture many aspects of data summarization while offering theoretical guarantees. However, one challenge in applying submodular functions for data summarization in astrophysics is the data representation. As we will discuss, submodular functions require a notion of similarity between measurements to select the most informative events from the data stream for the summary. Thus, an appropriate representation of the events must be obtained before the summary selection. Such a representation can be obtained through autoencoders, which embed the high-dimensional raw feature space into a meaningful lower-dimensional embedding space. However, there is no guarantee that the embedding space is well suited for data summarization. In this section, we will first review submodular functions in the context of data summarization and then present an autoencoder algorithm that computes embeddings that are tailored to the summarization tasks.

7.4.1 Introduction

Astroparticle physics experiments naturally deal with large volumes of data measured over years of continuous observation. Over time, the measurements are subjected to a natural concept drift by, say, broken sensors, changes in the environment, or changes in the telescope's operation. Thus, the health status and the actual observations must be closely monitored to detect changes and interesting events early on.

While numerous concept drift detection mechanisms exist, they do not give direct feedback to the operator by, say, characterizing the type of concept drift occurring. One way to visualize large amounts of data and possible concept drifts is to give a human operator a small, comprehensive summary of the measurements. This way, the operator

can interact with the data by looking at the data summary and directly reason about possible changes in the data or detector.

Submodular optimization has become a valuable tool in machine learning and data mining. Submodular functions reward adding a new element to a smaller set more than adding the same element to a larger set. This makes them ideal for solving data summarization tasks in which a submodular set function f assigns a utility score $f(S)$ to a summary S. Then, using submodular maximization, the optimal summary can be selected. Submodular functions designed for data summarization utilize a notion of distance between two observations x_i and x_j. Unfortunately, distance metrics lose their expressiveness with higher dimensions, so data summarization through submodular optimizations becomes more and more difficult with increasing numbers of dimensions [82].

One way to circumvent this problem is to embed the high-dimensional data into a lower-dimensional embedding space. To do so, we can use a specialized Deep Learning architecture (see Chapter 9) called autoencoders that consists of two parts, namely the encoder and the decoder. The encoder tries to encode the data into a lower-dimensional space, whereas the decoder tries to reconstruct the original data from the output of the encoder. Both parts are trained jointly in an end-to-end fashion so that the encoder tries to encode most of the information about the data into the lower-dimensional space. By contrast, the decoder tries to reconstruct the original data as much as possible (see [183], Chapter 14). Autoencoders and their variations have been studied extensively with great practical success for embedding tasks. Unfortunately, while autoencoders perform well, there is no guarantee that the embedding space retains the same notion of similarity as the original data space. Simply put, there is no way to guarantee that objects that are close in the original space are still close to each other in the embedding space, making it difficult to use them with data summarization techniques.

In this section, we discuss how to apply autoencoders for subsequent data summarization. First, we introduce the framework of submodular functions in the next section. Then, we discuss autoencoders in general and their application to the related tasks of cluster analysis. From these clustering autoencoders, we derive submodular autoencoders specifically targeted for data summarization tasks. Finally, we present experimental summaries of the FACT data and conclude the section.

7.4.2 Data Summarization with Submodular Functions

Submodular optimization offers a well-established mathematical framework to select small and comprehensive summaries for various tasks in linear time. Formally, we consider the problem of selecting K representative elements from a ground set V into a summary set $S \subseteq V$. To do so, we maximize a non-negative, monotone submodular set

function $f\colon 2^V \to \mathbb{R}_+$ which assigns a utility score to each subset:

$$S^\star = \underset{S \subseteq V, |S|=K}{\arg\max} f(S) \tag{7.8}$$

For the empty set, we assume zero utility $f(\emptyset) = 0$. We denote the maximum of f with $OPT = f(S^\star)$. A set function can be associated with a marginal gain which represents the increase of $f(S)$ when adding an element $x \in V$ to S:

$$\Delta_f(x|S) = f(S \cup \{x\}) - f(S) \tag{7.9}$$

We call f submodular iff for all $A \subseteq B \subseteq V$ and $x \in V \setminus B$ it holds that:

$$\Delta_f(x|A) \geq \Delta_f(x|B) \tag{7.10}$$

The function f is called monotone, iff for all $x \in V$ and for all $S \subseteq V$ it holds that $\Delta_f(x|S) \geq 0$. In general, the maximization of a submodular set function is NP-hard [160], which makes solving Equation 7.8 difficult. Therefore, a natural approach is to find an approximate solution. Nemhauser et al. presented in [164] a simple $(1 - (1/\exp(1))) \approx 63\,\%$ greedy approximation algorithm called `Greedy` for solving Equation 7.8, which runs in linear time and requires a fixed memory budget. `Greedy` offers a constant approximation guarantee and only requires $\mathcal{O}(K)$ memory. As the name suggests, it works in a greedy fashion. Starting from an empty summary, it computes the marginal gain $\Delta_f(x|S)$ of each element in the ground set and picks the one with the largest gain. This process is repeated K times until the summary is full. The algorithm is depicted in Algorithm 7.1.

Algorithm 7.1: `Greedy` algorithm.

```
S ← ∅
for 1, . . . , K do
    x = arg max{Δ_f(x|S)|x ∈ V}
    S ← S ∪ {x}
end
return S
```

The disadvantage of `Greedy` is that it requires K iterations over the entire dataset, which is costly if the ground set is very large. By contrast, submodular streaming maximization algorithms (see Section 3.1 in Volume 1 for an overview) review each item in the dataset exactly once and decide on the fly if a new item should be added to the summary or not, reducing the overall runtime from $\mathcal{O}(NK)$ to $\mathcal{O}(N)$. As an example consider the `ThreeSieves` algorithm presented in Section 3.1 in Volume 1. The main idea of this algorithm is that we can simulate the `Greedy` algorithm if we would stack K copies of

the entire dataset on top of each other and process them one by one. If we had known the gains of each item in summary ahead of time (i.e., the gains of the elements the `Greedy` algorithm would select), then we could review each item in the stacked dataset and compare it against the optimal gain to decide if we should add it to the summary. Of course, it is impossible to know the exact gains ahead of time, and in fact, this is part of the problem we would like to solve. Submodularity allows us to estimate that the optimal gains should be in the interval $[m, K \cdot m]$, where $m = \max_{x \in V} f(\{x\})$ is the maximum singleton element. Given this interval, we can sample different thresholds from it into a set O and hope that one of the thresholds is close enough to the true gain. The question now becomes which threshold to pick from this set O for comparing the individual gains against each other. Note, that when the threshold is too large, we will never pick any item; if it is too small, we will quickly fill up the summary with sub-optimal items. As stated, we cannot determine with absolute certainty that a given threshold is exceeded. However, we can do so with high probability.

More formally, we aim at estimating the probability $p(x|f, S, v)$ of finding an item x that exceeds the gain v for a given summary S and function f. Once p drops below a user-defined certainty margin τ, we pick the next smallest threshold in O and repeat the process. The estimation of $p(x|f, S, v)$ is straightforward since we observe with each item regardless whether it exceeds the current threshold or not. However, this estimation comes with its own uncertainty, so we need to estimate its confidence interval to make a confident decision. The computation of confidence intervals for estimated probabilities is a well-known problem in statistics. Assume we start with the largest threshold in O. In this case, most items will not have a gain exceeding the threshold, so we reject the item. In this case, we have a heavily one-sided binominal distribution with probabilities near 0. We can use the Rule of Three to estimate the confidence interval for these types of distributions. It states, that the confidence interval of $p(x|f, S, v)$ after observing T events is $\left[0, \frac{-\ln(\alpha)}{T}\right]$. For example, with 95 % certainty the confidence interval of $p(x|f, S, v)$ is $\left[0, -\ln(0.05)/T\right]$ which is approximately $[0, 3/T]$ leading to the term "Rule of Three" for this estimate [218]. We can use the Rule of Three to quantify the certainty that, with high probability, there will not be a novel item in the data stream after observing T items. Once `ThreeSieves` determines with enough credibility that the current threshold will likely not be out-valued, it picks the next smallest threshold and continues the estimation. The complete algorithm is depicted in Algorithm 7.2.

Until now, we have characterized the utility function as being non-negative, monotone, and submodular, but we left open which particular function would be well suited for a data summary on the fly. For a data summary, a diverse set of items is desired to capture the data stream fully. In other words, we want the observations in S to be most dissimilar to each other. One way to write this more formally uses a kernel function $k(x_i, x_j)$, which expresses the similarity between two items $x_i, x_j \in S$. A common

Algorithm 7.2: `ThreeSieves` algorithm.

$O \leftarrow \{(1+\varepsilon)^i \mid i \in \mathbb{Z}, m \leq (1+\varepsilon)^i \leq K \cdot m\}$
$v \leftarrow \max(O)$; $O \leftarrow O \setminus \{\max(O)\}$
$S \leftarrow \emptyset$; $t \leftarrow 0$
for *next item* x **do**
  if $\Delta_f(x|S) \geq \frac{v/2 - f(S)}{K - |S|}$ *and* $|S| < K$ **then**
    $S \leftarrow S \cup \{x\}$; $t \leftarrow 0$
  else
    $t \leftarrow t + 1$
    if $t \geq T$ **then**
      $v \leftarrow \max(O)$; $O \leftarrow O \setminus \{\max(O)\}$; $t \leftarrow 0$
    end
  end
end
return S

example of such a kernel function is the radial-basis-function (RBF) kernel

$$k(x_i, x_j) = \exp\left(-\frac{1}{2l^2} \cdot ||x_i - x_j||_2^2\right) \tag{7.11}$$

where $l \in \mathbb{R}$ is a scaling constant and $|| \cdot ||_2$ denotes the Euclidean norm. A kernel function gives rise to the kernel matrix $\Sigma_S = [k(x_i, x_j)]_{ij}$ containing all similarity pairs of all points in S. This matrix has the largest values on the diagonal because the items are the most similar to themselves. By contrast, values on the off-diagonal indicate the similarity between distinct elements and thus are usually smaller. Since we seek a comprehensive summary, we are more interested in the pairs with values near 0 on the off-diagonal of Σ_S. This intuition has been formally handled in the context of the Informative Vector Machine (IVM) [197]. The IVM is a Gaussian Process (GP) [322], which greedily selects a subset of data points and keeps track of the GP's posterior distribution. Based on a diversity argument, the authors propose to select the point that covers the training data best, iteratively. The intuition can be formalized in maximizing the logarithmic determinant of the kernel matrix:

$$f(S) = \frac{1}{2} \log\det(\mathbb{J} + a\Sigma_S) \tag{7.12}$$

where $a \in \mathbb{R}_+$ is a scaling parameter for numerical robustness and $\mathbb{J}$ is the $K \times K$ identity matrix.

In [345], this function is shown to be submodular. Its function value does not depend on V, but only on the choice of the kernel function k and the summary size K. This makes it an ideal candidate for summarizing data in a streaming setting. In [108], it is proven that $m = \max_{x \in V} f(\{x\}) = 1 + aK$ and that $OPT \leq K \log(1 + a)$ for kernels

with $k(\cdot, \cdot) \leq 1$. This property can be enforced for every positive definite kernel with normalization [185].

7.4.3 Dimensionality Reduction with Autoencoders

Submodular functions offer an excellent framework for selecting a well-chosen data summary. The log-determinant in Equation 7.12 is a submodular function that focuses on a diverse set of candidates, making it ideal for monitoring tasks. However, this function requires using a similarity kernel that must fit the task at hand. While submodular functions guarantee a mathematically sound solution to the summarization problem, the kernel encodes the similarity between items and thus determines how valuable the data summaries are in practice.

Different kernel functions for different application scenarios have been studied extensively in the literature. However, they sometimes yield sub-optimal results for high-dimensional, unstructured data such as images or time series. In these cases, Deep Learning-based approaches often show better results. Recently, the combination of both approaches has also been successfully explored [219, 236]. One approach to combine both methods is to first project the original data in a lower-dimensional *embedding* space by a deep autoencoder and then take these embeddings as an input for a kernel to compute the actual similarities. This way, the embeddings might be able to capture high-level interactions between data points, whereas kernels offer a mathematically proven way to express similarities. An autoencoder is a special neural network architecture designed to encode the input into a compressed embedding, which is then used to decode the original input. Figure 7.7 gives an overview of the general structure of an autoencoder, where $w \ll r$ is the embedding size and r is the dimensionality of the original data. Formally, we jointly train the encoder $e\colon \mathbb{R}^r \to \mathbb{R}^w$ and the decoder $d\colon \mathbb{R}^w \to \mathbb{R}^r$ so that the reconstruction loss $\ell\colon \mathbb{R}^r \times \mathbb{R}^r \to \mathbb{R}$ is minimal:

$$L(e, d) = \frac{1}{N} \sum_{i=1}^{N} \ell(x_i, e(d(x_i))) \tag{7.13}$$

where $\{x_1, x_2, \ldots, x_N\}$ are samples drawn from the data distribution $\mathcal{D}$. In practice, e is the desired encoding architecture and usually d mirrors e in reversed order. Both are jointly trained via stochastic gradient descent.

Autoencoder is a simple, unsupervised method that can reduce the runtime and improve the results of successive summarization methods that use embeddings instead of the unstructured input data. However, a fundamental assumption in data summarization is that similar events have a similar feature representation. While this might be true for the original data, this is not necessarily true for the latent embedding space. Put differently, there is no guarantee that similar observations are placed in the same region of the latent space and thus have a similar embedding. Clustering tasks are closely related to data summarization tasks, in which we want to find clusters of

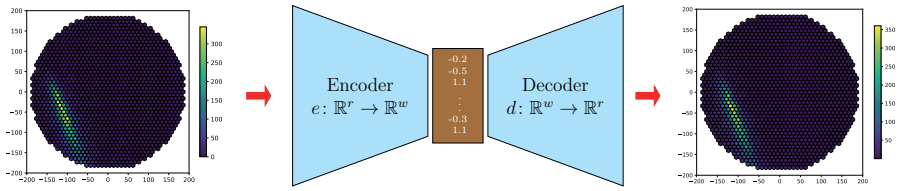


Fig. 7.7: An overview of the autoencoder architecture. The encoder e receives the original data as an input and embeds it in a lower-dimensional space. Then the decoder d uses this embedding to reconstruct the original input.

data. The main difference between both tasks is that in clustering, artificial centroids represent a cluster of data, whereas, in data summarization, the actual observations are reported. This makes data summarization more challenging because we cannot freely optimize the centroids to improve the summary as we would do in clustering. However, we can still borrow some insights from cluster embeddings for our desired summary embeddings.

Cluster embeddings are embeddings that are specifically trained for successive cluster analysis [394]. The main idea is that once an autoencoder with sufficient reconstruction capabilities is trained, we can refine it for the cluster analysis. For clustering, it might be desirable that the distances between a point and the cluster centroids should roughly follow a Student's t-distribution where one centroid is very close, and the others are far away [376]. Given an initial guess of centroids $S = \{\mu_j | j = 1, \dots, K\}$, the authors of [394] use a soft cluster assignment and measure the distance between an embedding point $e(x) = z \in \mathbb{R}^w$ and the centroids $\mu \in \mathbb{R}^w$:

$$q(z, \mu) = \frac{\left(1 + \|z - \mu\|^2/\alpha\right)^{-\frac{\alpha+1}{2}}}{\sum_{j=1}^{K} \left(1 + \|z - \mu_j\|^2/\alpha\right)^{-\frac{\alpha+1}{2}}} \tag{7.14}$$

Here α is the degrees of freedom of the Student's t-distribution, and $q(z, \mu)$ can be interpreted as the probability of assigning sample x to cluster μ. Given these assignments, the authors now propose minimizing the KL divergence between these assignments and a target Student's t-distribution p:

$$KL(P(S, e) \| Q(S, e)) = \sum_{i=1}^{N} \sum_{j=1}^{K} p(z_i, \mu_j) \log \left(\frac{p(z_i, \mu_j)}{q(z_i, \mu_j)} \right) \tag{7.15}$$

where S is the set of chosen clusters and $p(z_i, \mu_j)$ is given by

$$p(z_i, \mu_j) = \frac{q(z_i, \mu_j)^2 / c_i}{\sum_{k=1}^{K} q(z_i, \mu_k)^2 / c_k}, \text{ with } c_j = \sum_{i=1}^{N} q(z_i, \mu_j) \tag{7.16}$$

Let

$$c(x) = \arg\min_{\mu \in S} \|x - \mu\|_2^2$$

be the closest cluster center to x. The cluster embeddings follow a three step procedure

$$e, d = \arg\min_{e,d} L(e, d) \tag{7.17}$$

$$S = \arg\min_{S \subseteq \mathbb{R}^w, |S|=K} \sum_{j=1}^{N} \|x - \mu_{c(x)}\|_2^2 \tag{7.18}$$

$$e, S = \arg\min_{e,S} KL(P(S, e) \| Q(S, e)) \tag{7.19}$$

In the first optimization step, an encoder/decoder pair is trained via stochastic gradient descent. Then, k-means clustering is obtained (via Lloyds algorithm [253], say). Finally, the encoder is refined again via stochastic gradient descent. Note, that after the initial encoder/decoder pair is trained, the decoder is *not* changed anymore.

7.4.4 Submodular Autoencoders

Cluster embeddings work well in situations where a good autoencoder is already available that only needs refining for later cluster analysis. However, they have four individual shortcomings, which we will now tackle in detail.

Selecting Meaningful Representatives The joint optimization of S and e in the last optimization step may lead to cluster centers that do not correspond to an actual measurement from the data distribution. Put differently, μ_k might be the result of an optimization step (e.g., averaging in the case of k-means) that—once decoded—would have never occurred in the original data distribution. Data summaries offer a way to select meaningful representatives from the dataset, which naturally do not have this problem. The initial cluster assignment can directly be replaced by a data summarization algorithm such as the maximization of a submodular function via `Greedy`. In this case, the last optimization step should not alter the selected centroids leading to:

$$e, d = \arg\min_{e,d} L(e, d) \tag{7.20}$$

$$S = \arg\max_{S} f(S) \tag{7.21}$$

$$e, S = \arg\min_{e} KL(P(S, e) \| Q(S, e)) \tag{7.22}$$

Joint Optimization of the Reconstruction Loss Cluster embeddings use a post-processing step to refine embeddings after being trained for a minimal reconstruction loss. However, there is no reason to directly train the autoencoder to minimize the reconstruction loss and compute good cluster embeddings at the same time. We propose

a regularized training objective that balances the reconstruction error and the clustering performance. Let $\lambda \in \mathbb{R}$ be a regularization strength, and

$$e, d = \arg\min_{e,d} L(e, d) \tag{7.23}$$

$$S = \arg\max_{S} f(S) \tag{7.24}$$

$$e, d = \arg\min_{e,d} L(e, d) + \lambda KL(P(S, e) \| Q(S, e)) \tag{7.25}$$

Incorporating Domain-Specific Knowledge Like clustering, data summaries assign similar items to the same representative. Thus, as already discussed, an autoencoder should place similar items into the same region of the latent space $\mathbb{R}^w$. We can leverage similarities and structures from the original domain $\mathbb{R}^d$ to further enforce these types of similarities. As discussed in Section 7.3 we can often represent the measurements of a detector as an image (cf. Figure 7.7). The content of these images is rotation-invariant, e.g., a very similar shower can come from different directions. This results in vastly different images where a shower is depicted, say, in the upper left or upper right corner of the image, while both might be very similar from a physicist's point of view. In addition, similar-looking showers can have vastly different intensities. While one shower might contain up to 400 photons per pixel, other showers with very similar morphology can be comparably weak, with only up to 100 photons per pixel. Again, both events give very different images but with similar physical interpretations. We can utilize this insight more formally by introducing a noise process Θ that applies domain-specific alterations to the original images and demands that embeddings for the original image $e(x)$, as well as the embeddings for the altered image $e(\tilde{x})$, are close together. Let $\lambda_1, \lambda_2 \in \mathbb{R}$ be regularization parameters. Then we may use:

$$e, d = \arg\min_{e,d} L(e, d) + \lambda_2 \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{\tilde{x} \sim \Theta} \left[\| e(x_i) - e(\tilde{x}_i) \|_2^2 \right] \tag{7.26}$$

$$S = \arg\max_{S} f(S) \tag{7.27}$$

$$e, d = \arg\min_{e,d} L(e, d) + \lambda_1 KL(P(S, e) \| Q(S, e)) + \lambda_2 \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{\tilde{x} \sim \Theta} \left[\| e(x_i) - e(\tilde{x}_i) \|_2^2 \right] \tag{7.28}$$

Alternating Minimization So far, we have presented the training of the autoencoder and the summary selection as two separate training steps. If the summary selection algorithm (e.g., `Greedy`) is comparably slow, performing a single summary selection in-between autoencoder training and cluster refinement is beneficial. However, with faster summary selection algorithms such as `ThreeSieves` available, we can perform

multiple summary selections in an alternating minimization-maximization scheme where we alternate between the two objectives:

$$e, d = \arg\min_{e,d} L(e, d) + \lambda_1 KL(P(S, e) \| Q(S, e)) + \lambda_2 \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{\tilde{x} \sim \Theta} \left[\| e(x_i) - e(\tilde{x}_i) \|_2^2 \right] \tag{7.29}$$

$$S = \arg\max_{S} f(S) \tag{7.30}$$

Here, after a first initial guess of S has been made, we train the autoencoder for, e.g., one epoch. Then, we reselect the summary via `ThreeSieves` (or via `Greedy` if the dataset is small) and continue to train the autoencoder for the next epoch repeating the overall process. This allows us to refine the cluster assignments, the encoder/decoder pair, and the actual summary at the same time while simplifying the overall training process.

7.4.5 Experiments

It has been shown that `ThreeSieves` has a good performance under concept drift [110]. Here, we focus on the interpretability of submodular autoencoders in the context of physics data. To do so, we evaluate our submodular autoencoder on the publicly available Crab Nebula observation data in which the FACT telescope was directed at this source [53, 84]. The data consist of 17.7 h of total recording with 3 972 043 recorded events. Before training the submodular autoencoder, we perform the following pre-processing (see Section 9.4 for a more detailed explanation):

- *Sensor calibration:* The detector's sensors behave differently in different environmental situations. For example, the temperature may effect the sensor, which should be corrected.
- *Extracting photon counts:* The FACT telescope produces 1440 time series, each with a length of 150 nanoseconds. We remove noise from the time series and focus on a time window of 55 nanoseconds which contain most of the relevant Cherenkov photons. Calibration measurements for the sensors depict a typical voltage curve when a single photon hits the sensor. This baseline measurement is subtracted as often as possible from the actual measurement until there is no signal left [282]. The number of subtractions can be considered to be the number of photons that arrived during the time series. The resulting image then shows the photon counts for each sensor in each pixel.
- *Image mapping:* The FACT sensors are arranged in a hexagonal form. In hexagonal grids, each pixel has up to six neighbors instead of four, as in regular Euclidean grids. Convolutional autoencoders apply rectangular convolution filters to extract and generate higher-level features. Although the neighborhoods of pixels in rectangular and in hexagonal grids are slightly different, in a series of pre-experiments and student works, no performance difference could be found between using rect-

angular and hexagonal filters for FACT [263, 328]. Therefore, we choose to transform the data into 45×45 images in which the hexagonal grid is slightly rotated into the middle of the image. This allows us to use regular Convolutional neural networks (CNN) architectures and filters together with standard frameworks.

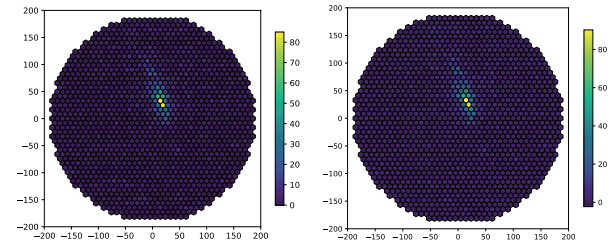
– *Filtering:* Data of the Crab Nebula only contains a small fraction of the interesting gamma-ray events. The measurements are overwhelmed with the hadronic background noise by a factor of 1000–10 000, which is isotropically distributed over the entire sky. Due to their high frequency, these events dominate both the reconstruction errors and the clustering, so we decided to remove events that did not contain at least one pixel with more than 29 photons. After filtering, 157 420 interesting events were left in the dataset.

As an encoder, we use a VGG-style neural network architecture [350] with three convolution blocks followed by a single linear block. Each convolution block contains two 3×3 convolutional layers, each followed by a batch normalization layer and a ReLU activation, as well as a single max-pooling layer of stride 2. The linear block contains a linear layer with 64 hidden neurons followed by a batch normalization layer and a ReLU activation. Finally, we add another linear layer with size 64 to obtain $w = 64$ dimensional embeddings. Our decoder mirrors the encoder in reversed order. We used the mean-squared error for training via stochastic gradient descent using PyTorch [300] and achieved a near-perfect reconstruction loss. However, we suspect that a loss function that better fits the distribution of the pixel values (e.g., a Poisson loss) might further increase the performance. We apply the following noise process to the images during training:

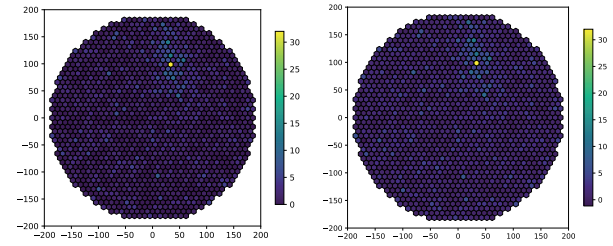
1) The images of the FACT telescope are rotation-invariant so that showers can occur from any direction. We randomly select a rotation angle from $[0, 360)$ degrees and rotate the alerted image by the given angle.
2) We are mainly interested in showers with different morphologies. Hence, showers with different energy levels but a similar morphology should be considered similar, here. Our noise process randomly selects a scaling factor from $(0, 2]$ and scales all photon counts in the image by the given factor.

We train for a total of 100 epochs with an initial learning rate of 10^{-3}, which is halved every 25 epochs on a batch size of 128. We set $\lambda_1 = \lambda_2 = 1$ and $\alpha = 1$ (cf. [257]) in our experiments. As mentioned earlier, we reselect summaries after each epoch. To do so, we use the `Greedy` algorithm maximizing the log-determinant using the RBF kernel (Equation 7.12) and (Equation 7.11) with $a = 1$ and $l = 1$. We found that a summary of size $K = 25$ gives a manageable but reasonably accurate overview of the data. For a better presentation, we focus on a selection of images in this chapter and refer interested readers to https://www-ai.cs.tu-dortmund.de:5443/, which lets people select and view summaries of varying sizes and with varying parameters.

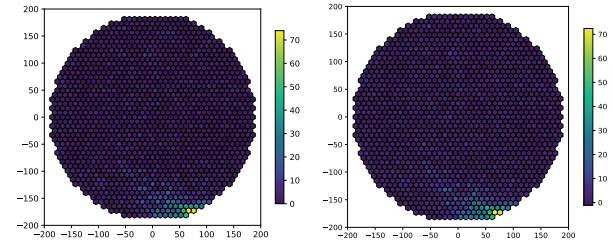
Figure 7.8 displays the summary selected in this experiment. For presentational purposes, we depict 5 of the 25 events here. The left column depicts the original events, whereas the right column depicts the corresponding reconstruction via the decoder. On average, we obtained a reconstruction error of around 1.5 photons per pixel, which can also be seen in this figure. For most events, the reconstructed image is nearly indistinguishable from the original one, although the reconstructions do not seem as nuanced as the original events. The only exception is the event (5), in which the reconstruction seems to overestimate the number of photons a bit. Together with a domain expert, we identified the following types of events: (1) is a small, focused gamma shower with moderate intensity; (2) is a small, gamma shower with low intensity; (3) is a so-called corner clipper, in which only part of an air shower hits the telescope; (4) is a very fragmented shower with moderate intensity, possibly due to a proton event and (5) a large gamma shower with high intensity. The physicist concluded that the images offer some form of interpretability and clearly distinguish different types of events, making them useful for monitoring.



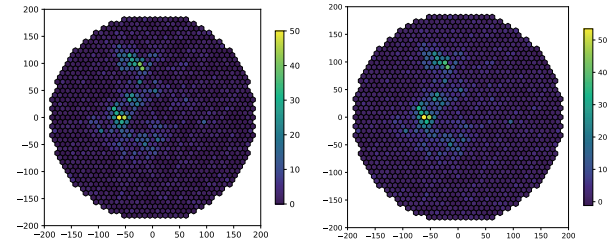
(1, original) (1, reconstructed)



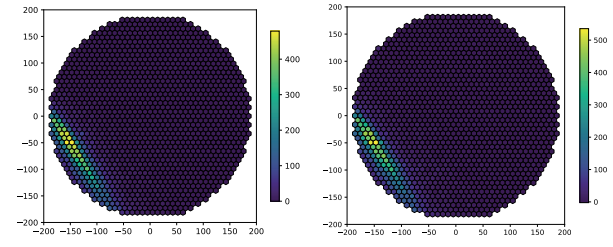
(2, original) (2, reconstructed)



(3, original) (3, reconstructed)



(4, original) (4, reconstructed)



(5, original) (5, reconstructed)

Fig. 7.8: Data summary of Crab Nebula data using submodular autoencoders (excerpt).

7.4.6 Discussion and Outlook

In astrophysics experiments, we face large volumes of data measured over years of continuous observation. This data naturally contains concept drift due to, say, broken sensors, changes in the environment, or changes in the operation of the telescope. While classic concept drift detection methods accurately detect concept drift, they do not display current measurements in a meaningful way so humans can interact with them. Data summarization techniques condense large amounts of data into a small, comprehensive data summary that can then be reviewed by humans in considering current measurements. Submodular functions are at the heart of summarization algorithms as they reward adding a new element to a smaller set more than adding the same element to a larger set. Submodular functions designed for data summarization often utilize the distance between two observations. Unfortunately, distance metrics start to lose their expressiveness with higher dimensions, so data summarization through submodular optimizations becomes more and more difficult. Autoencoder, by contrast is a technique well suited to embed high-dimensional measurements into a lower-dimensional embedding space, which can then be used to compute data summaries.

Unfortunately, while the autoencoder shows excellent performance in compressing information into a lower-dimensional latent space, there is no guarantee that the embedding space retains the same notion of similarity as the original data space. To tackle this problem, we presented submodular autoencoders. Our method alternates between summary selection via submodular maximization and the refinement of an autoencoder for better embeddings. Moreover, we added a novel, domain-specific regularization to further reinforce similarities between similar events in the data. Our submodular autoencoder produces embeddings well suited for data summarization and, thanks to recent advances in submodular streaming maximization, are generally quick and easy to train. The resulting summaries provide physicists with selected events that are easily interpretable.

8 Event Property Estimation and Signal Background Separation

8.1 Introduction

Wolfgang Rhode

Abstract: Imaging Air Cherenkov and neutrino telescopes acquire data at rates up to of several thousand events per second. For most science cases, however, the majority of the triggered events consists of unwanted particles, typically referred to as background, that need to be discarded in order to carry out precision analyses and to answer physics questions.

Although a certain fraction of said background can be rejected by the use of simple cuts, a relatively large fraction remains. The remaining signal-to-background ratio strongly depends on the details of the analysis and ranges from 10^{-9} for tau-neutrino searches to 10^{-3} in spectral analyses of atmospheric muons. In state-of-the-art analyses this remaining background is typically rejected by the use of supervised machine learning algorithms such as Random Forests (RFs).

This chapter describes the utilized algorithms and discusses the practical challenges arising in the application of supervised learning techniques in astroparticle physics exemplified by analysis from the IceCube neutrino observatory and the Imaging Air Cherenkov Telescopes MAGIC, FACT, and CTA. We discuss how the existing methods were altered to match the requirements of large scale experiments in astroparticle physics and present select results obtained over the entire funding period.

After measurement data without information content has been removed from the data sets for analysis and physically meaningful features have been extracted from the information carriers, the next step is to extract the signal events relevant for analysis. At this point, the datasets typically contain several classes of events with physical information content. In order to analyze a dedicated physical question, the event class of interest must be separated as a signal from the sum of all other event classes. This separation can result from a single or multi-stage ML selection process. The choice of the specific machine learner used will also depend on the quality criteria relevant for further analysis.

It is obvious for physical reasons that the features technically used for the separation may be correlated with physically relevant characteristics. Therefore, it often helps to optimize estimators for specific physical characteristics (direction, energy, ...) at this point of the analysis. These estimators can be used here as carriers of do-

https://doi.org/10.1515/9783110785968-008

main knowledge in the separation and later (Chapter 10) as a basis for solving the inverse measurement problem (deconvolution of the energy spectrum or the angular distribution).

Different approaches from the fields of particle physics (LHCb and ATLAS) and astroparticle physics (IceCube and IACTs) are presented in this chapter.

8.2 Boosted Decision Trees LHC

Margarete Schellenberg
Bernhard Spaan

Abstract: The classification of relevant data points is the most common use of data analysis in particle physics, where one observes billions of data points, of which less than a thousand can be interesting. The main part of the data is the combinatorial background, where unrelated tracks are combined and mimic an interesting signal data point. This background can be reduced with requirements on single variables. However, much better efficiencies can be achieved with the usage of boosted decision trees. At the LHCb experiment, this is commonly done with supervised learning. The signal proxy is typically well modeled simulated or recorded data, and the background proxy recorded data outside the signal region. In the analysis of the decay $B^0 \to J/\psi K^0_S$, about 90 % of the signal data is kept, while over 99 % of the background data is removed with the use of a boosted decision tree. The analysis of the decay $B^0 \to D^{\star\pm} D^{\mp}$ requires a more complex selection, keeping 92 % of the signal data and suppressing more than 98 % of the background data. In summary, modern particle physic analysis would not be possible without the use of machine learners such as boosted decision trees.

In this section, the use of Boosted Decision Trees at LHCb is described in the context of the analysis of the $B^0 \to D^{\star\pm} D^{\mp}$ decay. A challenge here is identifying and reducing the background candidates that dominate the recorded data set. Still, with the use of a boosted decision tree, about 92 % of the signal data is kept, while over 98 % of the background data is removed.

In the analysis of the decay $B^0 \to D^{\star\pm} D^{\mp}$ [10], a measurement of time-dependent charge-parity (CP)-violation is performed, which is one of the keys to understanding the matter-antimatter-asymmetry observed in our universe. The reconstruction is done with the decays $D^- \to K^+\pi^-\pi^-$ and $D^{\star+} \to D^0\pi^+$, where the D^0 decays into $K^-\pi^+$.

Freely propagating B^0 mesons can mix into their anti-particle state ($\bar{B}^0$) and vice versa. Also, the charge-conjugated final states, $D^{\star+}D^-$ and $D^{\star-}D^+$, are common to both the B^0 and $\bar{B}^0$ meson decays. The interference between the amplitudes of the direct

decay and of the decay after $B^0 - \bar{B}^0$ mixing results in the decay time-dependent CP-asymmetry is:

$$A_f(t) = \frac{\Gamma(\bar{B}^0(t) \to f) - \Gamma(B^0(t) \to f)}{\Gamma(\bar{B}^0(t) \to f) + \Gamma(B^0(t) \to f)} = \frac{S_f \sin(\Delta m t) - C_f \cos(\Delta m t)}{\cosh\left(\frac{\Delta\Gamma t}{2}\right) + D_f \sinh\left(\frac{\Delta\Gamma t}{2}\right)}. \tag{8.1}$$

The decay time-dependent asymmetry is given by the difference between the decay time-dependent decay widths, $\Gamma(t)$, of B^0 and $\bar{B}^0$ mesons decaying into the final state f, normalized to the sum. An analogous asymmetry exists for the final state $\bar{f}$. $B^0(t)$ and $\bar{B}^0(t)$ denote the initial B flavor, and the parameters Δm and $\Delta\Gamma$ are the differences of the masses and decay widths between the heavy and light mass eigenstates concerning the $B^0 - \bar{B}^0$ system. With some assumptions, the decay time-dependent asymmetries become

$$A_f(t) = S_f \sin(\Delta m t) - C_f \cos(\Delta m t), \qquad A_{\bar{f}}(t) = S_{\bar{f}} \sin(\Delta m t) - C_{\bar{f}} \cos(\Delta m t), \tag{8.2}$$

where S_f, $S_{\bar{f}}$, C_f, and $C_{\bar{f}}$ are the CP observables that are to be measured by performing a maximum-likelihood fit of the B^0 meson decay-time distribution. In order to extract the information from such large data samples, it is necessary to achieve a good control over the different backgrounds. Depending on the kinds of background, it is common to use different selection methods.

Besides the relevant and interesting data points, known as the signal data, the data samples at LHCb contain many uninteresting data points, known as background data. These can be divided into two classes: physical background and combinatorial background. Physical background can be further classified into different kinds. Exclusive backgrounds arise when final state particles are misidentified or misreconstructed and, as a result, mimic the signal. In partially reconstructed backgrounds, single particles of a different decay are not reconstructed, and the remaining tracks are wrongly combined with the signal candidate. Furthermore, the background can arise from different B mesons decaying into the same final state. All of these kinds of physical backgrounds appear as peaking structures in the mass distribution. In most cases, they can be suppressed efficiently with rectangular requirements on single variables. In this context, rectangular means that certain measured quantities of the event have to lie in well-defined ranges and do not vary depending on other quantities in the event, which contrasts with using a multivariate analysis. If contributions cannot be eliminated completely, they must be described by Probability Density Functions (pdfs).

A combinatorial background arises when unrelated particle tracks, which do not necessarily originate from the same vertex, are combined and mimic signal data points. The combinatorial background shows no peaking structures. Rather, it has a flat, exponentially decreasing shape, which cannot be removed entirely and must be described by a pdf. Due to the high track multiplicity at LHCb, the data is dominated by combinatorial background.

To illustrate, Figure 8.1 shows the distribution of the invariant $D^{\star\pm}D^{\mp}$ mass after a selection using a Boosted Decision Tree (BDT). The signal component is shown in dashed blue. To its right, a background component arising from a B_s^0 meson decaying into the $D^{\star\pm}D^{\mp}$ final-state is apparent, represented as the dotted green line. To its left, exclusive background arising from $B^0 \to D_s^+ D^{\star-}$ is present and displayed in brown. Due to a kaon/pion misidentification in the $D_s^+ \to K^-K^+\pi^+$ decay, the signal decay gets reconstructed. Towards lower masses, partially reconstructed backgrounds from $B^0 \to D^{\star+}D^{\star-}$ and $B_s^0 \to D^{\star+}D^{\star-}$ are present and shown in magenta and turquoise, respectively. One of the $D^{\star\pm}$ meson is decaying into $D^{\pm}\pi^0$, where the neutral pion is not reconstructed, leading to a false reconstruction of a $D^{\star\pm}D^{\mp}$ final state. The combinatorial background component is described by the dark green line extending across the whole range.

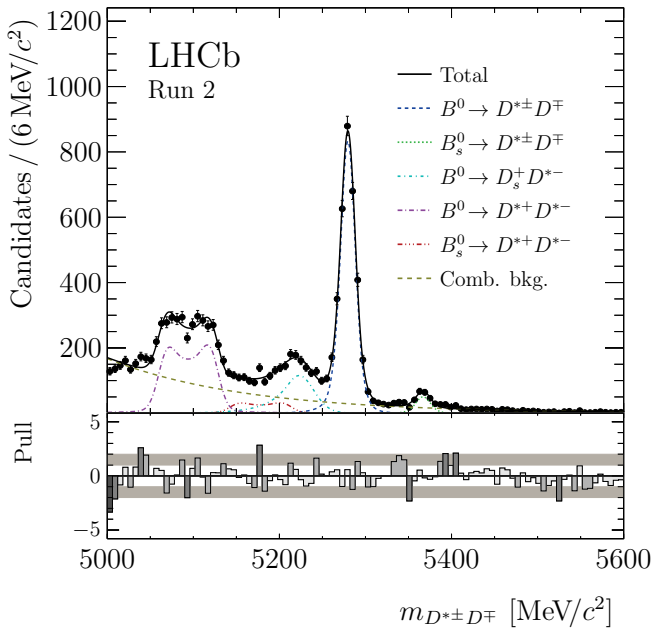


Fig. 8.1: Distribution of the invariant $D^{\star\pm}D^{\mp}$ mass after the application of the BDT requirement [10]. The fitted pdf is overlaid in black; the figure displays projections of the components of the pdf.

Like the physical background, the level of combinatorial background can be reduced by requirements on single variables. However, much better efficiencies can be achieved using Boosted Decision Trees. At LHCb, this is commonly done with supervised learning. The signal proxies are typically well modeled simulated or recorded data, and the background proxy consists of recorded data outside the signal region. In the case of

the $B^0 \to D^{*\pm} D^{\mp}$ analysis, simulated signal decays are used for the signal proxy. The distribution of the invariant $D^{*\pm} D^{\mp}$ mass in Figure 8.1 shows that above 5400 MeV/c^2, the upper mass side-band, only combinatorial background is present. Towards lower masses, physical backgrounds contribute. Thus, the background proxy is represented by reconstructed B^0 candidates with invariant masses higher than 5400 MeV/c^2. For the training of the BDT, the TMVA framework [200, 201] utilizing the Freund-Schapire method [169] is used.

A 2-folding method is employed to avoid overtraining effects in the upper mass side-band as it is not only used in the BDT training but also for the mass fit in a following analysis step. The data set is randomly divided into two equally sized parts. On each half, a separate BDT is trained, but with precisely the same training options. Each of the two BDTs is then applied to the dataset that was not used for the training.

A set of input features is needed to train the BDT. A first BDT is trained using a large set of features that show visible differences in their signal and background distributions. This is done by comparing the distributions of signal data and statistically subtracting the background [313] and the background from the upper mass side-band. After training the BDT, the variable importance is calculated by counting how often the variables are used for a decision at a node and weighting each decision occurrence by the separating power it has achieved and the number of events in the node. All observables showing low importance are discarded, leaving the final selection of features. A total of 26 features are used for the final training. These are listed in Table 8.1.

Tab. 8.1: List of input variables used in the training of the BDT in order of importance.

Variable		
D^+ decay time significance	$K(D^+)$ ProbNN	B^0 IP
χ^2	D^+ FD χ^2	$\pi_2(D^+)$ p_T
D^0 FD χ^2	$m(D^{*+}) - m(D^0)$	$K(D^+)$ p_T
$K(D^0)$ ProbNN	DTF χ^2	D^0 decay time significance
$\cos(\angle(D^+, \pi_2))$	$\cos(\angle(D^+, K))$	D^{*+} p_T
$\cos(\angle(D^0, \pi))$	B^0 p_T	$\cos(\angle(B^0, D^{*+}))$
D^+ IP χ^2	D^+ p_T	$\cos(\angle(D^+, \pi_2))$
$\pi_2(D^+)$ ProbNN	$\pi_1(D^+)$ ProbNN	$\pi(D^{*+})$ ProbNN
D^0 IP χ^2	$\pi_1(D^+)$ p_T	$\pi(D^0)$ ProbNN

Among the set of features are decay-time significance variables of the D^+ and D^0 mesons, which are defined by the decay time divided by its uncertainty, and the χ^2 of the flight distance of the D^+ meson. These features represent how far the D meson has traveled before its decay. D mesons have a finite lifetime, i.e., they travel a certain distance before they decay. By contrast, background candidates have much lower reconstructed decay times. Typically their flight distance is close to zero.

The so-called ProbNN variables of all final state particles are included. They provide the probability that the particle is a kaon or pion and can be used to suppress pion-kaon misidentification.

Another feature is the χ^2 of the impact parameter (IP) of the B^0, D^+, and D^0 meson. The IP represents the shortest distance between the reconstructed track and the primary proton-proton interaction vertex (PV). It ensures that the B^0 meson originates from the PV, while the D^+ and D^0 mesons do not originate from the PV but from a detached vertex.

Additional features are transverse momenta p_T of some final state particles. Typically, the signal has higher transverse momenta than the background. Besides that, correlations between the p_T and the kinematic of other final state particles are present for the signal, whereas the background has no such correlation.

The mass difference of the D^{*+} and D^0 mesons is a powerful feature. The D^{*+} and D^0 mesons are reconstructed by the final state particles $K^-\pi^+\pi^+$ and $K^-\pi^+$, respectively, whereby the reconstructed $K^-\pi^+$ particles are the same for both mesons. Due to this, the difference between the two nominal masses is barely higher than the pion mass. By subtracting the two masses, mass resolution effects are reduced. Correctly reconstructed $D^{*+} \to D^0\pi^+$ decays have a sharp distribution around the true mass difference, while candidates with larger mass differences are probably background caused by mis-reconstruction.

Furthermore, the χ^2 of the fit to the complete decay chain is included. While the typical value is relatively small for signal, it is high for background.

Features describing the cosine of the angles between reconstructed tracks also exploit the different correlations between signal and background.

In order to optimize the hyperparameters used in training, the ROC (receiver operating characteristic) curve, which represents the background rejection as a function of the signal selection efficiency, is utilized. The larger the area under the ROC curve, the better the BDT performance. The hyperparameters are changed iteratively until the area under the curve does not increase in size. In the final BDT training, the BDTs consists of 800 trees, and the depth of the trees is limited to three. At each node of the tree, 3 % of the training data set must be present. Each variable is scanned at 40 points to find the best cut point. For boosting the method by Freund and Schapire [169] is used with a factor of $\beta = 0.1$. To ensure a more robust prediction and avoid overtraining, the bagging method is employed. Here, this method reduces the variance of the BDTs by creating subsets with a size of 80 % compared to the original dataset for every tree. In this resampling, events can be picked several times. In addition, for every tree, only ten random features of the whole set are used.

The BDT classifier distributions, with training and testing samples superimposed, are shown for one fold in Figure 8.2. The training and test samples are in good agreement, and it can be assumed that no overtraining is present. The classifier distributions of the two trained BDTs, when applied to the candidates they were not trained on, are compatible with each other on data. Hence, the BDT classifiers from the two folds can

be combined and are treated as one in the following. The BDT response is added to the data samples. Afterward, the selection is performed by applying a cut on the BDT response.

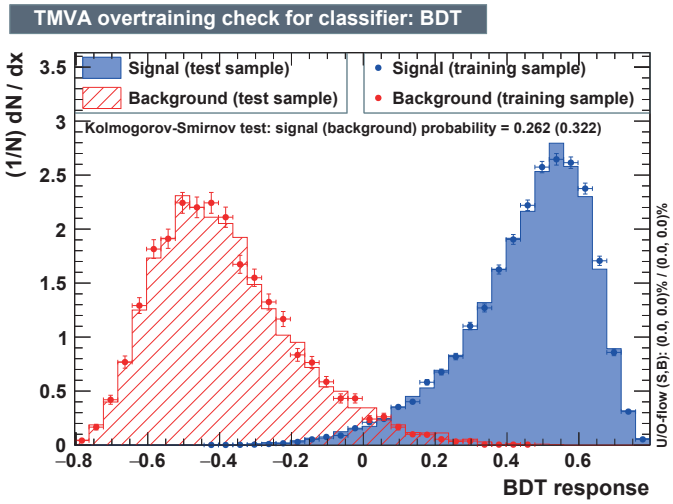


Fig. 8.2: Comparison between the BDT classifier on the (data points) training and the (shaded area) test samples of the (blue) signal and (red) background components.

In order to maximize the sensitivity on the *CP* parameters, the optimal cut point on the BDT response has to be found. A measure called "figure of merit" (FOM), which represents the inverse variance of the *CP* parameter of interest, is maximized with respect to the requirement on the BDT response. As only the sensitivities are reflected in the FOM, the central values of the *CP* parameters are not biased. In this case, the parameter $\sin\phi_d^{\text{eff}}$, which combines the *CP* parameters S and C, is used as a single reference parameter. The FOM takes into account several properties of the data sample that directly affect the statistical power of the sample in a *CP* measurement. Besides a high effective signal size and good flavor-tagging performance, small estimates of the decay time error will lead to a more precise measurement. In addition, the sensitivity of the *CP* parameters depends on the reconstructed decay time of the B^0 candidates. Differences in the distributions of these observables for different BDT cut points are considered.

By summing over all events, the FOM is defined as

$$\text{FOM} = \frac{(\sum_i s_{wi})^2}{\sum_i s_{wi}^2}\overline{\text{D}}\,, \tag{8.3}$$

where s_{wi} denotes the so-called sWeights [313] and statistically substracts remaining background data. They are determined by performing an extended maximum likelihood

fit to the $D^{*\pm}D^{\mp}$ mass distribution. The term $\overline{\mathrm{D}}$ is defined by

$$\overline{\mathrm{D}} = \frac{1}{\sum_i s_{wi}} \sum_i (1-2\omega_i)^2 \mathrm{e}^{-(\Delta m_d \sigma(t_i))^2} \cdot \mathrm{X}_i \cdot s_{wi} \,. \tag{8.4}$$

It is averaged over all candidates considering the decay-time uncertainty estimates $\sigma(t_i)$, where t_i denotes the decay time, and the mistag probabilities ω_i. The parameter Δm_d represents the mass difference in the $B^0 - \bar{B}^0$ system and is an external, constant input. The term X_i includes the parameter of interest $\sin\phi_d^{\mathrm{eff}}$, which is also an external and constant input, and is defined by

$$\mathrm{X}_i = \left[\frac{2 d_i |\lambda| s}{1 + |\lambda|^2 + d_i D_{\mathrm{FT}} \mathrm{e}^{-(\Delta m_d \sigma(t_i))^2/2} \left(-2|\lambda| s \sin\phi_d^{\mathrm{eff}} - (1-|\lambda|^2) c \right)} \right]^2, \tag{8.5}$$

where d_i denotes the decision for the production flavor of the B^0 mesons and the abbreviations $s = \sin(\Delta m_d t_i)$, $c = \cos(\Delta m_d t_i)$, and $D_{\mathrm{FT}} = (1-2\omega_i)$ are introduced for a more compact definition of the term.

At first, the influence of the single contributions is investigated by dividing the FOM into the following individual components. The effective flavor-tagging term efficiency is given by

$$\mathrm{FOM}_{\epsilon D^2} \equiv \frac{1}{\sum_i s_{wi}} \sum_i (1-2\omega_i)^2 \cdot s_{wi}, \tag{8.6}$$

and the effect of the decay time error estimates is described by the term

$$\mathrm{FOM}_{\sigma_t} \equiv \frac{1}{\sum_i s_{wi}} \sum_i \mathrm{e}^{-(\Delta m_d \sigma(t_i))^2} \cdot s_{wi} \,. \tag{8.7}$$

The effective signal size is given by

$$\mathrm{FOM}_{S_{\mathrm{eff}}} \equiv \frac{(\sum_i s_{wi})^2}{\sum_i s_{wi}^2}, \tag{8.8}$$

and

$$\mathrm{FOM}_{B^0} \equiv \frac{1}{\sum_i s_{wi}} \sum_i X_i \cdot s_{wi}, \tag{8.9}$$

denotes the term associated to $\sin\phi_d^{\mathrm{eff}}$.

Afterward, the FOM is calculated completely. Figure 8.3 shows the total FOM along with its individual components determined in the BDT cut scan.

The optimal cut-point is found at a BDT classifier value of 0.18. It is shifted towards a harder cut value than that of the maximum of the component of the effective signal size. This is caused by the flavor-tagging component and the component FOM_{B^0} comprising $\sin\phi_d^{\mathrm{eff}}$. Due to the fact that the oscillation period given by $2\pi/\Delta m_d$ is large compared with the decay time of the B^0 meson, the component containing the decay time error estimates has no impact on the optimal cut value. This becomes noticeable as the term is constant on the entire scan range.

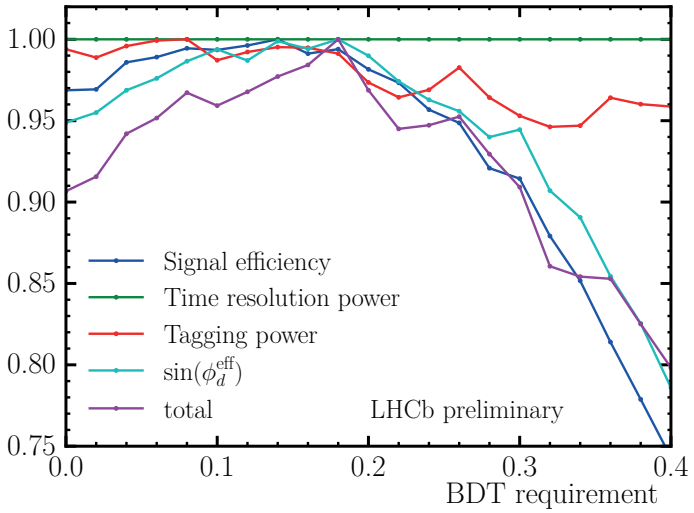


Fig. 8.3: The FOM on recorded data for different cuts on the BDT response. The full FOM is presented in violet. The other graphs show the individual components: in blue, the term of the effective signal size; in green, the term for the decay time error estimates; in red, the tagging power term; and in turquoise, the term depending on sin ϕ_d^{eff}. All graphs are normalized to the highest value.

While the FOM used in this decay time-dependent analysis optimizes the uncertainty of *CP* observables, different FOMs are used depending on the kind of measurement. A FOM commonly used in particle physics maximizes the significance of the signal candidates N_{signal}

$$\text{FOM} = \frac{N_{\text{signal}}}{\sqrt{N_{\text{signal}} + N_{\text{background}}}} . \tag{8.10}$$

For making a discovery, a FOM that is independent of expectations about the presence of signal,

$$\text{FOM} = \frac{\varepsilon_{\text{signal}}}{\frac{3}{2}\sqrt{N_{\text{background}}}} , \tag{8.11}$$

with the signal efficiency $\varepsilon_{\text{signal}}$, is commonly used [318].

The cut on the BDT response is applied to all data samples. The $D^{\star\pm}D^{\mp}$ mass distribution before and after the application of the optimized BDT requirement is shown in Figure 8.4. The comparison demonstrates that the level of the combinatorial background has been reduced by a significant amount while the signal peak remains almost unchanged. Afterward, the signal efficiency is calculated with simulated $B^0 \to D^{\star\pm}D^{\mp}$ samples, and the background rejection is determined with the upper mass side-band of recorded data. With this optimized BDT requirement, it is possible to remove 98 % of the background while keeping 92 % of the signal. This achievement would not be possible without the use of a multivariate analysis.

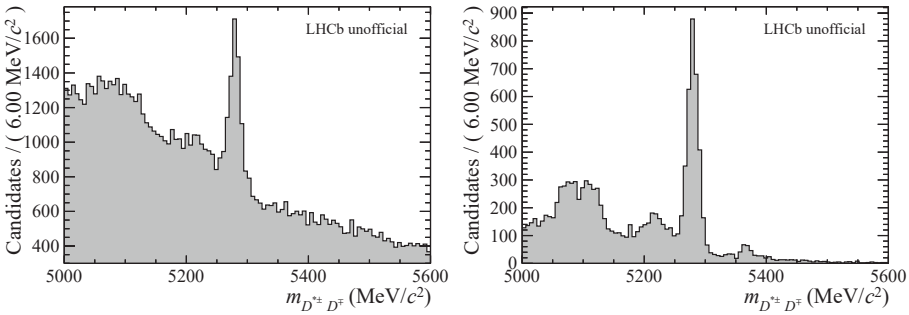


Fig. 8.4: Distribution of the invariant $D^{*\pm}D^{\mp}$ mass before (left) and after (right) the application of the optimized BDT requirement

8.3 Event Selection in IceCube

Tim Ruhe

Abstract: If the IceCube trigger conditions are fulfilled, an event is formed as the waveforms of the DOMs are read out. The information of the individual waveforms is then combined to reconstruct the incident particle's energy and direction. As IceCube is a multi-purpose detector, different types of events are triggered. This introductory section explains the different types of events and their sources.

Before one commences with a classification task, it is advisable to gain a clear understanding of the class variable(s) and the number of classes. When physics analyses apply classification algorithms, the task is usually to discriminate between physically interesting and physically non-interesting events, which are often referred to as *signal* (interesting) and *background* (not interesting). The definition of the signal depends on the experiment at hand and often also differs between individual analyses carried out with data obtained with the exact same instrument. In IceCube, for example, atmospheric muons form a background for many different analyses but are also interesting in themselves and are thus the signal in analyses of atmospheric muons. This chapter discusses the selection of signal events in IceCube using two examples: muon neutrinos (ν_μ) and tau neutrinos (ν_τ). As the different event topology is an essential criterion for distinguishing different types of events, the different event topologies are explained below.

Within IceCube, events with different geometric properties originating from different interactions are observed.

Track-Like Events *Track-like* events originate from muons in the ice, which initiate the emission of Cherenkov light along their trajectories. Accordingly, an elongated light pattern is detected by the optical modules. The majority of the muons in IceCube originate from two sources. The first source is neutrino-induced muons, meaning they are produced in charged-current (CC) neutrino interactions of the form:

$$\nu_\mu + X \rightarrow \mu^- + X'. \tag{8.12}$$

Due to the relatively long range of the muon (500 meters for a muon with an energy of 1 TeV), the interaction in Equation 8.12 may occur in the ice outside of the instrumented volume or even in the surrounding bedrock.

The second source is atmospheric muons created in cosmic ray air showers.

Cascade-Like Events Cascade-like events exhibit a more spherical light pattern and are produced in two types of interactions. The first type is ν_e CCs of the form:

$$\nu_e + X \rightarrow e^- + X', \tag{8.13}$$

which—due to the short range of the electron—results in an electromagnetic cascade around the interaction vertex. Cascade-like events further originate from ν_τ-CC interactions. In rare cases, however, these interactions exhibit a very distinct light pattern, often referred to as a double cascade. Due to their uniqueness, double-cascades are separately addressed below. The second type is neutral-current (NC) interactions of all neutrino flavors.

Cascade-like events can be further subdivided into fully and partially contained events. For fully contained events, the entire Cherenkov light is emitted within the detector, and the whole cascade is visible to the optical modules. For partially contained events, only a part of the Cherenkov light is emitted within the instrumented volume, and only a certain part of the cascade is visible to the optical modules. While fully contained events require an interaction well inside the instrumented volume, partially contained events can have vertices at the edges of the detector.

Double Cascades This particular type of event exhibits a rather unique light pattern, consisting of two cascades, which in an ideal setting are connected by a track. Double cascades are a *smoking gun signature* for detecting a tau neutrino, as this event topology can be produced only via the interaction of a ν_τ. The first of the two cascades originates from the interaction of the ν_τ, where a τ-lepton is formed. The τ-lepton will then travel through the ice and cause a track-like structure, similar to the ones created by muons. Due to the relatively short lifetime of the τ, the lepton will decay shortly after its creation, which in the case of a hadronic decay, will result in a hadronic cascade.

The challenge in the detection of double cascade arises from the interplay of two limiting factors, the first one being the short lifetime of the τ and the second one being the steeply falling energy spectrum of the ν_τ. Due to the short lifetime of the τ, the

length of the track is rather limited (≈ 50 meter per PeV), which implies that the two individual cascades will not be sufficiently far apart to be resolved individually, in case the energy of the ν_τ is smaller than a few PeV. In addition, the expected power law with a spectral index of $\gamma \approx -2$ will lead to only a handful of events, energetic enough to produce a well resolvable double cascade over the entire lifetime of the detector.

The selection of muon neutrinos and tau neutrinos via the application of machine learning techniques is addressed in the following sections.

8.3.1 Muon Neutrino Selection

Tim Ruhe

Abstract: As muon neutrinos cannot be measured directly, they are detected via their interaction with nuclei in the ice or the bedrock. Within these interactions, a muon—the leptonic partner of the muon neutrino—is formed. In IceCube, muons will then initiate the emission of Cherenkov light, which is then detected by the Digital Optical Modules (DOMs). Unfortunately, muons are also created in cosmic ray air showers. As these so-called atmospheric muons outnumber the neutrino-induced muons by several orders of magnitude, they provide an inevitable background in searches for muon neutrinos. Since the Earth is opaque to muons, a significant fraction of the background can be removed by applying geometrical cuts, which effectively use the Earth as a muon shield. Applying such cuts will remove 99.9 % of the background muons, but this is not sufficient due to their sheer abundance. Furthermore, the remaining background of atmospheric muons is falsely reconstructed and significantly harder to reject. This section will describe the tools and techniques utilized for selecting muon neutrinos with the IceCube experiment in analyses of atmospheric neutrino energy spectra. Special focus is laced on the utilization of random forests and their optimization, which were found particularly useful for this type of application.

First, it is helpful to understand the goal of such an analysis. In the muon neutrino analysis presented here, the physics goal is the measurement of a muon neutrino energy spectrum via the use of deconvolution techniques (see Chapter 10 for details). Such an analysis requires not only a high purity (better than 99 %, if possible) of the final set of selected events but also a clear understanding of the remaining background. As the neutrino energy spectrum is expected to follow a power law of the form $\frac{d\Phi}{dE} \propto E^{-\gamma}$, with $\gamma \approx 3.7$ for atmospheric and $\gamma \approx 2$ for astrophysical neutrinos, one expects only a small number of events at high energies. As a certain amount of background examples within the final data set are unavoidable, one has to make sure that these do not populate the high-energy bins to a large extent. A significant background contribution

in the highest energy bins would result in an apparent flattening of the unfolded neutrino energy spectrum. As a certain amount of flattening is, in fact, expected due to a shift in the dominant component of the spectrum (atmospheric neutrinos for small energies, astrophysical neutrinos for high energies), such an additional flattening would significantly alter the physical interpretation of the spectrum.

In atmospheric muon neutrino analyses, which are used as an example in this section, the straightforward definition labels neutrino events as signal and muon events as background. It is not that simple, however. Muon neutrinos can be detected only via their leptonic partner—the muon—which is created in an interaction of the neutrino with a nucleus in the ice or the bedrock. In this type of analysis, muons are thus the signal *and* the background. The question is then: How can these two types of muons, which can be more clearly labeled as atmospheric- and neutrino-induced muons, be distinguished?

Before answering this question, it is useful to consider the data at the starting point of the analysis. Like many other IceCube analyses, the analyses presented here do not commence from *raw data*, which consists of electric pulses recorded by the optical modules as a response to the detected Cherenkov light. Instead, the analyses use data with a higher level of abstraction. Internally, this higher level of abstraction is referred to as level 3. On this level 3, data cleaning, as well as advanced reconstructions and some initial background rejection, has already been applied. Details on the data acquisition in IceCube are given in Section 4.2.3, whereas a brief overview of the applied reconstructions is provided in Section 7.2. An overview of the different data levels is given in Section 4.3.3.

The Earth can be used as a filter to discriminate between atmospheric and neutrino-induced muons. While neutrinos can traverse the Earth up to PeV-energies[1], the Earth is opaque for muons. The first task in a muon neutrino analysis is thus often to apply a cut, which rejects events with zenith angles smaller than $\theta \leq 86^\circ$. Purely geometric considerations would suggest the cut to be placed at $\theta \leq 90^\circ$. However, muons entering the detector at angles $86^\circ \leq \theta \leq 90^\circ$ have a very high probability of ranging out before reaching the detector. Allowing for this additional zenith range results in the acceptance of more neutrino-induced muons while restricting the additional background to only a few events. This relatively simple physical and geometrical consideration already reduces the background by approximately one order of magnitude.
A certain fraction of downward-going ($\theta \leq 90^\circ$) atmospheric muons is, however, falsely reconstructed as upward going ($\theta > 90^\circ$). Due to the overwhelming number of atmospheric muons, compared with the number of interacting neutrinos, falsely reconstructed atmospheric muons still outnumber neutrino-induced muons by more than two orders of magnitude. Accordingly, the classification task carried out by the use

1 The neutrino cross-section increase linearly with energy, which finally results in the fact that the Earth can no longer be traversed unhindered for neutrino energies of a few PeV or higher.

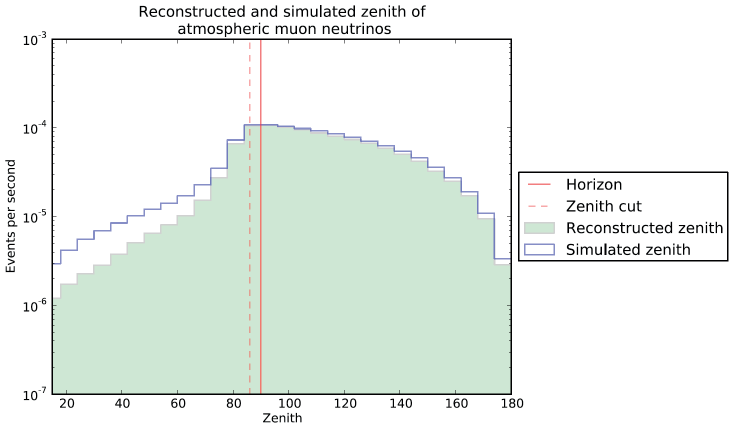


Fig. 8.5: Number of events per second as a function of the reconstructed zenith angle. The dashed and solid lines indicate the applied cut and the horizon. Figure adapted from [91].

of machine learning algorithms is to distinguish between correctly and falsely reconstructed muon events.

Figure 8.5 shows the true (blue) and reconstructed (green) zenith for atmospheric muons. The true and reconstructed zenith of atmospheric ν_μ is shown for comparison. The solid red line depicts the horizon ($\theta = 90°$), whereas the applied cut for the IC86 analysis ($\theta \geq 86°$) [91] is shown as the dashed red line. One finds that the number of detected atmospheric muons exceeds that of detected atmospheric neutrinos by several orders of magnitude. By applying the aforementioned cut, 97.7 % of the background muons are removed while retaining 76.9 % of the ν_μ [91]. Due to the larger overall number of atmospheric muons, the signal-to-background ratio after the application of the cut is still 1:18 [91].

As individual events in IceCube are described by more than 1000 features, the next step in a muon neutrino analysis is the selection of attributes that are considered suitable as input for the utilized classification algorithm. The feature selection is often an iterative process and is described in more detail in Section 7.2.

After the input features have been selected—a dimensionality reduction that makes the entire analysis less resource-consuming—the automated classification commences. In the context of IceCube and this CRC, analyses of muon neutrinos were carried out for data taken in the 59- [16], 79- [18], and 86-string [91, 341] configuration. In all analyses, a random forest [97] has been used as a classifier. While the random forest from Weka [189] embedded in the Rapidminer [321] framework was used for the IC59 and IC79 analyses, the 86-string analysis utilized the random forest from sklearn [302].

As no labeled historical data exist, the machine learning-based event selection in IceCube requires using simulated events for the training and the validation of the selected classifier. For IceCube analyses, atmospheric muons are generated using the air

shower code CORSIKA [194] (see also Chapter 5), whereas neutrino events (in this case, neutrino-induced muons) are generated using the ANIS package [175]. Using simulated examples bears the risk of disagreements between simulated and experimental data, which may result in a classifier bias. Therefore, some care must be taken to validate that there is sufficient agreement between simulated and experimental data or to ensure that the classifier is not affected by possibly observed discrepancies. Details on how to detect and handle such mismatches are given in Subsections 5.3.2 and 5.3.3.

As a result of the use of different data, as well as different implementations of the random forest, the utilized settings are somewhat different for the individual analyses. The number of trees is one of the most important parameters of a random forest, and 500 trees were used in the IC-59 analysis [16]. For IC-79, however, 200 trees were found to be sufficient [18], while for IC-86, 503 trees were used [91]. In all three analyses, the number of attributes selected at each node was chosen according to the default value. Moreover, the depth of trees was not restricted in any of the three analyses.

The individual analyses differ regarding the ratio of signal and background examples utilized for training. While a ratio of 10:1 (signal to background) was found to be optimal for the IC-79 analysis [18], ratios of 1:1 were used for IC-59 and IC-86 [16, 91]. Neither of the ratios, however, represents the true signal-to-background ratio. The differing signal-to-background ratios are mainly used for two reasons. The first reason is a practical one, as the production of enough simulated muon events to mimic the expected ratio on experimental data would be too resource-consuming. The second reason is that using the expected ratio with the available amount of simulated muon events would result in only a handful of neutrino events. This is clearly not sufficient for the training of a machine learning algorithm. The ratios of 1:1 and 10:1, respectively, are then also used for testing and validation. The number of selected neutrino events and the estimated purity are two figures of merit for physics analyses, and the true signal-to-background ratio needs to be taken into account to obtain both numbers. At this stage, it should be noted that these are, of course, not the only figures of merit. Measures like accuracy, precision, and *AUC* are, of course, also taken into account in the optimization process. These measures are, however, not always reported in physics papers, and different event selection methods can thus often be compared only by their achieved purity and the total number of neutrino candidates.

Although the majority of the selected neutrino candidates are expected to be of atmospheric origin, the energy distribution of the simulated examples follows an E^{-2} spectrum rather than an atmospheric one ($\frac{d\Phi}{dE} \propto E^{-3.7}$). For practical reasons, neutrino events are generated according to power laws following either E^{-1} or E^{-2} to generate a sufficient number of events at high energies. Using an E^{-2} instead of an atmospheric distribution was, however, also found to be beneficial for the training of random forests in muon neutrino analyses. In this case, the distribution contains a sufficient number of high-energy events in the training sample, which can then also be selected from the overall set of experimental data. Furthermore, a significant drop in the recall was

observed in IC-59 in case an atmospheric spectrum was used [329]. To obtain reliable numbers for the expected number of events and the purity after selection, weights corresponding to certain atmospheric models, e.g., Honda [255] and ERS [152], are assigned to the events and evaluated when these figures of merit are computed. In all three analyses, training and testing of the classifier were carried out in cross-validation. The usage of cross-validation also allows for the estimation of uncertainties in the expected number of events. Because the number of expected background examples is small, the derived confidence intervals provide valuable information when comparing the number of events observed on experimental data with the number of expected events.

After training, the classifier is applied to simulated data to verify its performance. In most cases, the classifier is also applied to experimental data at this stage to check for possible inconsistencies between data and simulation. In a random forest, all events are processed by every tree, and the final confidence assigned to the i-th event is then given as:

$$c_i = \frac{1}{n} \sum_{j=1}^{n} c_{ij}, \tag{8.14}$$

where n is the number of trees in the forest and c_{ij}, is the classification score assigned to the i-th example by the j-th tree. For the event selections discussed here, the forest settings were chosen such that c_{ij} can take the value 0 and 1, where 1 corresponds to the classification as a signal and 0 corresponds to the classification as a background event. Other settings are, however, possible, and the c_{ij} can then take values between 0 and 1. When using the default settings of the classifier, examples with $c_i \geq 0.5$ are classified as signal events by the random forest.

Figure 8.6 depicts the classification score obtained for signal (upgoing ν_μ, blue) and various possible sources of background. Although the main background component is atmospheric muons (cyan), other possible background sources might also be present in the sample. These components include ν_e and ν_τ, which will generate cascade-like events in the detector. The same holds for ν_μ originating from NC instead of CC interactions. Looking at Figure 8.6 one finds that low classification scores are dominated by background events (especially atmospheric muons), whereas signal events mostly populate high classification scores. The two distinct maxima observed at $c_i = 0$ (atmospheric muons) and $c_i = 1$ (ν_μ) are typical for this kind of analysis.
One also finds that the default setting of the classifier for the applied cut ($c_i \geq 0.5$) is insufficient in this case. A sample selected in such a way would be dominated by atmospheric muons, which are still one order of magnitude more abundant at this score. Instead, a user-defined cut on c_i needs to be placed to achieve a high enough purity ($P \geq 99\,\%$), which is required by the spectral reconstruction. It should be noted, however, that the purity requirement needs to be interpreted correctly, as not the number itself is important. Due to the steeply falling energy spectra of atmospheric and astrophysical neutrinos, the contamination with atmospheric muons must be considered for the

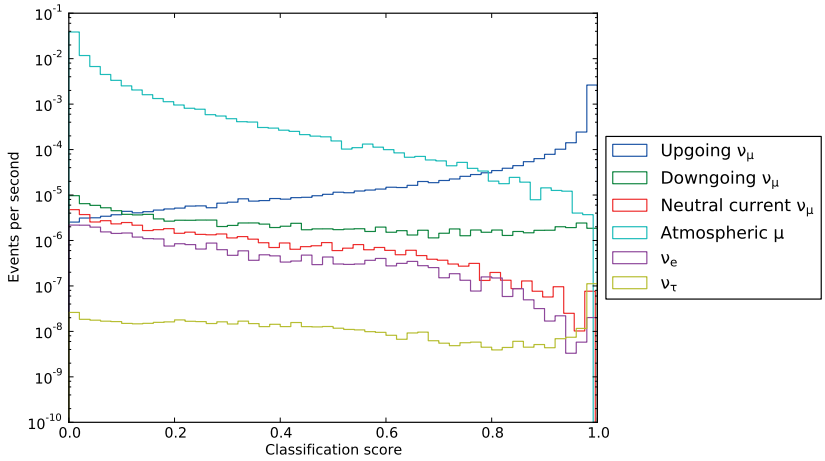


Fig. 8.6: Classification score obtained for signal (upgoing ν_μ) and various possible sources of background. Figure adapted from [91].

individual energy bins. Too many atmospheric muon events would alter the shape of the reconstructed spectrum and thus also change its physical interpretation, especially at high energies. As the number of neutrino-induced muons is significantly larger at low energies, one can allow for a certain muonic contamination in these bins.

From Figure 8.6 one also finds that atmospheric muons are the dominant background component, even at high classification scores. The contribution of all other background components is found to be 1–2 orders of magnitude smaller compared to atmospheric muons and can therefore be neglected in the subsequent unfolding.

As stated above, the trained classifier is usually applied to experimental data at this stage so that the outcome of the classification score can be readily compared with results anticipated from simulations. To obtain the results shown in Figure 8.7 the exact same classifier as for Figure 8.6 was used. The classification score for experimental data is depicted in green, whereas the scores obtained for simulated data are shown in blue. In contrast to Figure 8.6, only the sum of all simulated components is shown. One finds that the two distributions agree well. Especially the two distinct maxima at $c_i = 0.0$ and $c_i = 1.0$ are well reproduced.

Although the necessity of applying an additional cut on the obtained c_i exists in basically all analyses discussed here, the choice of this cut differs between individual analyses. While an extremely strict cut of $c_i = 1.0$ was chosen in the analysis of data obtained with IceCube in the 59-string configuration [212], a somewhat looser cut of $c_i \geq 0.92$ was chosen for data obtained with IceCube in the 79-string configuration [18]. In order to observe a possible flattening of the spectrum at high energies and thus achieve a model-independent confirmation of a flux of high-energy astrophysical neutrinos, one has to ensure that a sufficient number of high-energy events passes the event selection.

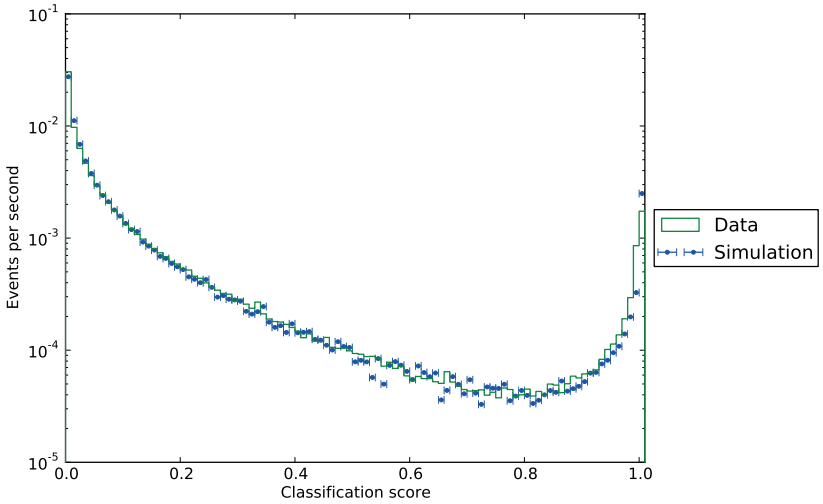


Fig. 8.7: Classification score obtained for data and simulation. For simulations the sum of all contributing components is shown. Figure adapted from [91].

To achieve this, two-dimensional cuts, which also consider the energy dependence of the classifier output, were applied in [341] and [91]. In [341] events were chosen to pass the selection if $c_i \geq 0.87$ for cases where the estimated energy of the event was found to be below 10 TeV. For all other cases, events with $c_i \geq 0.7$ were chosen to pass the selection. The final selection presented in [91] is somewhat more sophisticated. In order to achieve a high enough number of events in the sample, the confidence cut is optimized in N overlapping windows of width w. Cuts are optimized such that a purity of $P \geq 99.7\,\%$ is achieved.

The correlation of the classification score to the estimated muon energy is shown in Figure 8.8. Results for experimental data are shown in the larger left panel, whereas the outcome for different simulated spectral components is depicted in the smaller panels on the right-hand side. The black and the grey solid lines indicate the applied two-dimensional cuts for the zenith bands from $86^\circ \leq \theta \leq 111^\circ$ and $111^\circ \leq \theta \leq 180^\circ$, respectively. Two distinct maxima at $c_i = 0.0$ and $c_i = 1.0$ are also visible in this visualisation. In contrast to Figure 8.6, however, one finds that atmospheric muons with high classification scores are centered around muon energies of approximately 10^3 GeV. This clearly shows the benefit of studying the correlation of the classification score to other variables of interest, which allows the application of more sophisticated cuts and thus the selection of more neutrino candidates. The different cuts obtained for different zenith regions within the optimization process show that it is beneficial to include even more dimensions in the cut optimization.

Although the specific choices between the discussed analyses differ to a certain extent, one can summarise that the basic conceptual steps required for selecting neutrino-

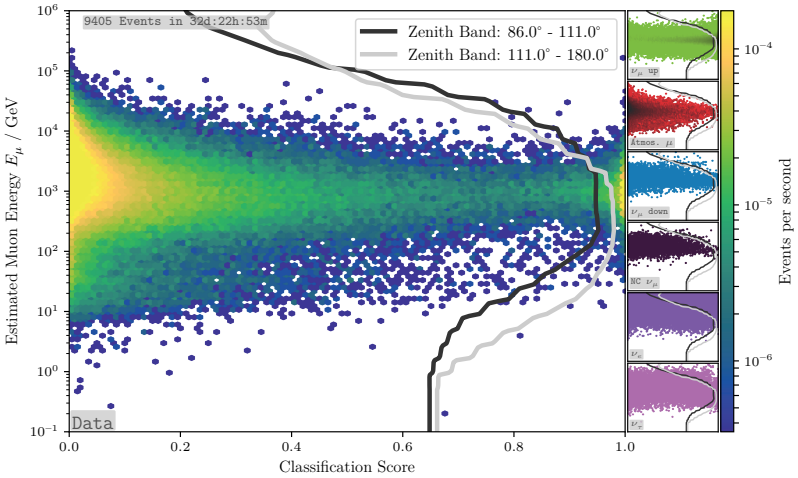


Fig. 8.8: Correlation of the obtained classification score to the estimated muon energy for experimental data (large left panel) and simulated components (smaller panels on the right). Figure adapted from [91].

induced muons remain unchanged. These steps commence with applying a fundamental and straightforward geometry cut on the estimated arrival direction of the incident particles. This cut is followed by a dedicated feature selection, which has not been covered in this section (see Section 7.2 for details), and serves the purpose of selecting the input variables for a machine learning-based selection of events. Finally, a cut on the classifier output must be applied to account for the overwhelmingly large number of atmospheric muons.

8.3.2 Tau Neutrino Selection

Tim Ruhe
Maximilian Meier

Abstract: An ideal tau neutrino interaction in IceCube will produce a very distinct light pattern, consisting of two cascades—the first one originating from the tau neutrino interaction itself and the second one originating from a non-muonic decay of the emerging tau lepton—and a track-like signature emerging along the path of the tau lepton. Such signatures are generally referred to as double cascades. For such an ideal scenario, however, both cascades need to be contained within the detector. In addition, the length of the tau track is 50 m per PeV of neutrino energy. A double cascade will thus become more distinguishable from a single cascade as energy increases, but it will also become less likely, due to the steeply falling neutrino energy spectrum. This challenge can be circumvented by looking at double pulses in waveforms of individual DOMs. In this section we present a random forest-based tau neutrino selection and its optimization via a model rejection factor. The section will put particular focus on the challenges associated with the identification of tau neutrinos. Compared with muon-neutrino searches, these challenges arise from the significantly larger background rate (the signal to background ratio is 10^{-10}) and additional sources of background.

Tau neutrinos are not produced in the Earth's atmosphere, and their detection is thus a direct evidence for the emission of high energy neutrinos in astrophysical sources. For energies in the PeV-range, ν_τ are expected to exhibit a unique event signature inside the detector, which is referred to as a double cascade. This distinct signature consists of two single cascades, connected by a track-like signature. The first cascade originates from the ν_τ CC interaction of the form:

$$\nu_\tau + X \rightarrow \tau^- + X^{'}. \tag{8.15}$$

In Equation 8.15, X denotes a nucleus in the ice or the surrounding bedrock, whereas $X^{'}$ denotes the emerging hadronic cascade. Before its decay, the emerging τ-lepton traverses the detector on a straight line, and the subsequent τ-decay then causes a second hadronic cascade. Unlike the muon, the τ cannot only decay into lighter leptons (e and μ), but also into hadrons, due to its relatively large rest mass.

In spite of this very unique signature, ν_τ are extremely difficult to detect. Due to the relatively short lifetime of the τ, the lepton will decay promptly after its generation, which poses strict limits on the length of the observed track. Comparing a track length of ≈ 50 m per PeV with the string spacing of ≈ 125 m, one finds that in most cases the two emerging cascades will not be far enough apart to be distinguishable. In addition, the expected power law with a spectral index of $\gamma \approx -2$, will lead to only a handful of

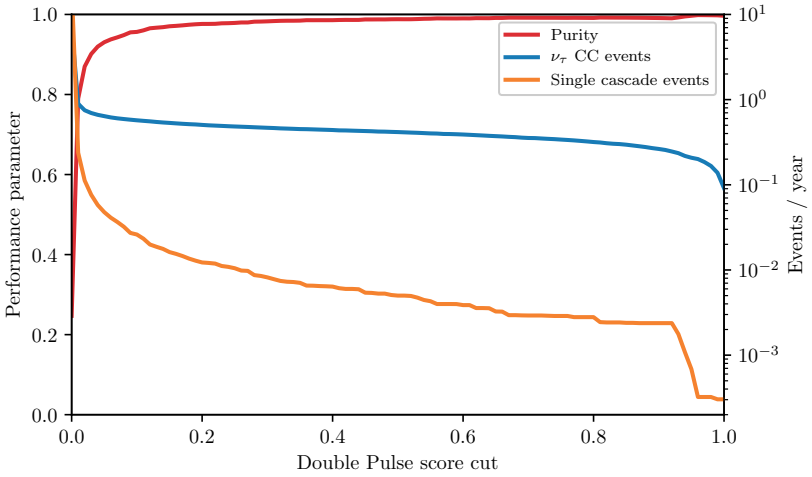


Fig. 8.9: Expected number of events per year (right y-axis) and performance parameter (purity, left y-axis) as a function of the applied double pulse score cut. Figure taken from [264].

events, energetic enough to produce well-resolvable double cascades over the entire lifetime of the detector.

This challenge can, however, be addressed, by looking for *double pulses* instead of double cascades. Double pulses are waveforms with two distinct peaks, which can be observed in individual optical modules, even in cases where the track of the τ lepton is too short to generate a distinguishable double cascade. Accordingly, the first pulse originates from the first, and the second pulse from the second cascade. In order to detect double pulses, many of the analysis techniques derived for the selection of muon neutrinos, such as feature selection via MRMR, can be readily applied.

One of the main differences between the selection of ν_μ and ν_τ are the additional classes of background events, which need to be considered during the analysis. While atmospheric muons are the main background source in ν_μ analyses, ν_e (CC and NC) and ν_μ (NC only) interactions also need to be considered in $\nu_\tau$analyses, as these interactions result in single-cascade events, which need to be distinguished from double cascades. Double cascades can also be mimicked by muons, when the first pulse is due to a neutrino-nucleon interaction and the second pulse originates from a large stochastic energy loss (bremsstrahlung in most cases). These challenges are addressed by utilizing a two-stage event selection, where different classes of background events are rejected at each of the stages.

As there is no obvious label for double-pulse waveforms in IceCube simulations, the first task of the analysis is a manual definition of signal waveforms. This manual definition includes the application of cuts on the total charge per event and per waveform. These cuts need to be applied, as an energy of $E_\nu \approx 10\,\text{TeV}$ is required for a

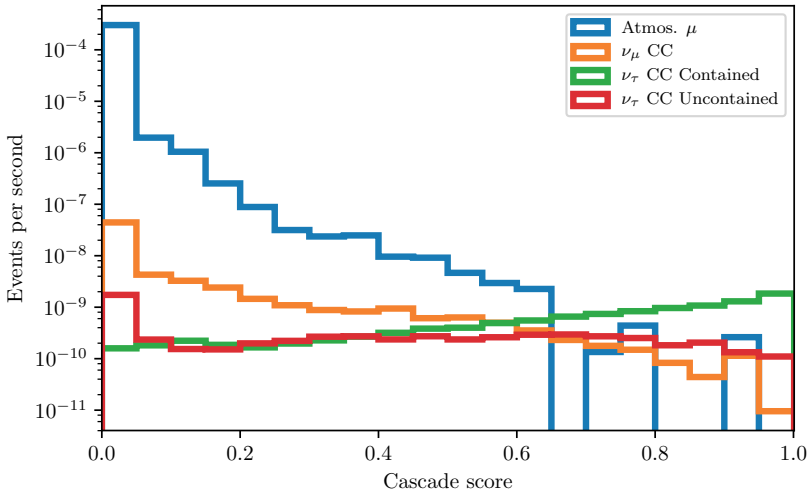


Fig. 8.10: Classification score obtained for the first of two random forests. Figure taken from [264].

double cascade to be recognized as two distinct pulses in one or more of the adjacent DOMs. Whether a double-pulse waveform is detected in a ν_τ-interaction or not, crucially depends on the deposited energies, the vertex positions of the interaction, and the subsequent decay. Therefore, a set of geometrical cuts is applied to simulated events to pick up waveforms with substantial double-pulse features. The application of said geometrical cuts, led to the selection of 6874 waveforms, which were used as input to the machine learning algorithms applied in the next steps of the analysis.

The first of two machine learning-based analysis steps consists of the training, optimization, and application of a random forest, in order to distinguish double and single cascades. Here we use a random forest of 200 trees. The majority of the features used as input to the forest is derivative-based and aims at characterizing the changing slope of the waveform. Figure 8.9 depicts the outcome of the application of the obtained model. While the double-pulse score cut (confidence obtained from the random forest) is shown on the x-axis, the performance parameter—in this case the purity of the event selection (depicted in red)—is shown on the left y-axis. The y-axis on the right-hand side depicts the number of events per year as a function of the double pulse score. Tau neutrino CC events are depicted in blue, whereas single cascade events are shown in orange. One finds that both distributions fall off as a function of the score cut. There is, however, a relatively sharp drop in the number of single cascades between double-pulse score cuts of 0.0 and 0.1.

A rather soft cut of 0.2 on the double-pulse score cut is chosen. This cut suppresses the majority of the single-cascade background. Compared with [387] an increase of 90 % is observed for the retained signal events, which results in an expected event

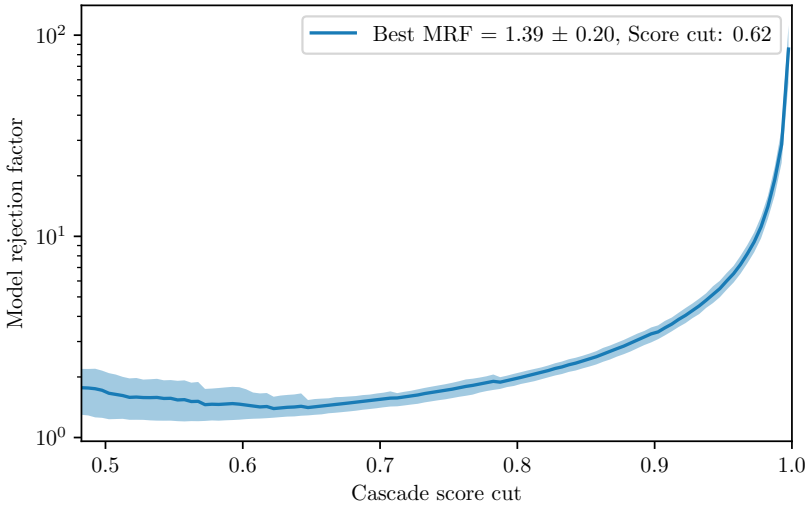


Fig. 8.11: Distribution of scores obtained with a random forest. The forest was trained to distinguish between single and double cascade events. Figure taken from [264].

rate of 0.63 ν_τ-events per year. The selected cut further improves the purity of the sample from 90 % to 97 %. While single cascades are well rejected at this stage, the sample is still dominated by atmospheric and neutrino-induced muons. Therefore, a second selection step is required. Fortunately, the track-like events, which form the irreducible background to the previous analysis step, can be reduced by taking into account their distinct event topology. In order to do so, the data are processed from level 2, which contains only a minimum of reconstructions, to a dedicated cascade level 3, which allows for additional cascade reconstructions. Variables used as input to the model (again a random forest) were selected using the MRMR algorithm. The agreement between simulated and experimental data was validated using the method proposed in [90] (see Section 5.3.2 for details).

In contrast to previous analyses, no veto region is defined for the ν_τ-selection presented here, as the definition of such a region drastically reduces the fiducial volume of the detector. *Not* using a veto region also offers the possibility of retaining partially contained events, which is expected to further increase of the event rate. However, in order to obtain a feasible classification task, two measures are taken. First, selection is limited to events, where both the ν_τ-interaction and the τ-decay vertex are contained within the boundaries of the detector. The boundaries of the detector are given by the outer layer of strings or DOMs. Second, a cut is applied to remove all events, for which the reconstructed interaction vertex is 60 m outside the detector. The background is composed of atmospheric muons, and muons induced in ν_μ-CC interactions. At this point the use of CORSIKA becomes unfeasible, due to computational constraints. Instead, atmospheric

muons were simulated with MuonGun [377], which allows effective parameterization of incoming muons around the IceCube detector to reduce computational cost.

The outcome of the application of the trained model is shown in Figure 8.10. While atmospheric muons are depicted in blue, neutrino-induced muons are depicted in orange. Contained and uncontained ν_τ CC events are shown in green and red, respectively. One finds that the rate of muon tracks falls off as a function of the classification score (x-axis). The rate of atmospheric muons is found to fall off faster than the rate of neutrino-induced muons. After a sharp peak around a classification score of 0.0, the rate of contained and uncontained $\nu_\tau$events increases as a function of the classification score. For scores exceeding ≈ 0.8, the sample is expected to be dominated by ν_τ CC events.

In order to obtain the final event sample, an additional cut on the classification score of the second random forest needs to be applied. Within the analysis discussed here, this final cut is chosen on the basis of a model rejection factor (mrf) [198], which is defined as follows:

$$mrf = \frac{\bar{\mu}_{90}}{n_\mathrm{s}} \tag{8.16}$$

with

$$\bar{\mu}_{90} = \sum_{n_\mathrm{ons}=0}^{\infty} \mu_{90}(n_\mathrm{obs}, n_\mathrm{b}) P_{n_\mathrm{b}}(n_\mathrm{obs}), \tag{8.17}$$

where $\bar{\mu}_{90}$ represents the average upper limit of all possible observable counts (n_obs) weighted with the respective Poisson probability $P_{n_\mathrm{b}}(n_\mathrm{obs})$ assuming that no signal is observed [264].

The model rejection factor is chosen as an optimization criterion, to avoid a potential bias, which may originate from selecting a cut based on the observed events. As the mrf uses expectations only, the selected cut will result in an optimal upper limit for the analysis at hand. Figure 8.11 depicts the mrf of the analysis as a function of the applied score cut. The shaded area represents the uncertainty of the obtained values. Looking at Figure 8.11 one finds that an optimal (minimal) mrf of 1.39 ± 0.20 is obtained for a score cut of 0.62.

After the application of the score cut, two ν_τ-candidates are selected from IceCube data obtained in the 86-string configuration from 2011 to 2019 (7.5 years in total). For the first event, which was observed during the 2014 season, three waveforms pass the double-pulse selection criteria. The waveform obtained in the central DOM achieves the highest possible double-pulse score of 1.0. Double-pulse scores of 0.45 and 0.81 were obtained for the remaining two waveforms. The event was classified with a cascade score of 0.92. The combination of the observed cascade and double-pulse scores make this event the most interesting one in the sample.

Figure 8.12 shows the waveforms with a distinct double-pulse shape, observed for this particular event. One finds that all three DOMs are located on the same string (string 20). The most prominent double-pulse structure is observed for DOM 26 (orange),

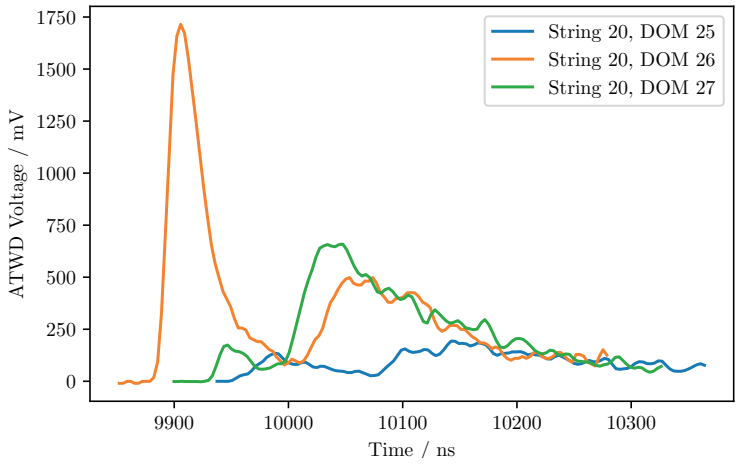


Fig. 8.12: Waveforms exhibiting a double-pulse structure for the ν_τ-candidate observed in the 2014 data-taking season of IceCube. The double-pulse structure is most prominent for DOM 26 on string 20 and becomes less distinct for the two neighbouring DOMs (25 and 27) on the same string. Figure taken from [264].

but becomes less distinct for the two neighbouring DOMs, depicted in blue (DOM 25) and green (DOM 27).

Figure 8.13 shows the event display of this particularly interesting ν_τ-candidate. The event is fully contained within the instrumented volume of the detector, as DOMS near the detector boundaries did not observe a significant amount of light. Colors in this figure represent the arrival time of the observed Cherenkov photons and range from red (early) to blue (late). The size of the color-coded DOMs represents the amount of light collected by a particular DOM. The spherical shape of the event, indicated by large amount of light collect at early times (central DOMs in red) and smaller amounts of light detected by outer DOMs at later times, is clearly visible from the event display.

For the second event, a double pulse score of 0.565 was observed for a single waveform and a cascade score of 0.675 was observed for the event as a whole. This event is thus most likely a background event.

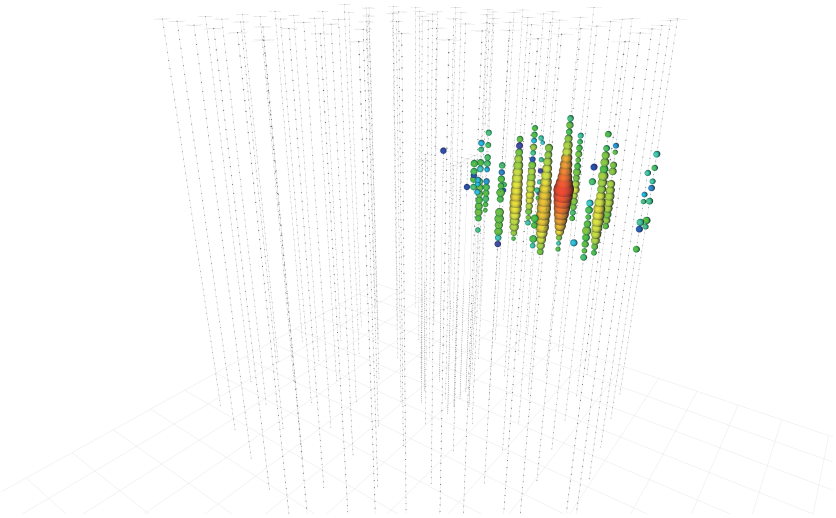


Fig. 8.13: Event view of the ν_τ-candidate observed during the 2014 data-taking season of IceCube. The event is fully contained inside the instrumented volume. Colors in this plot represent the timing of the detected Cherenkov photons from red (early) to late (blue), whereas the size of the DOMs corresponds to the amount of light observed by a particular DOM. Figure courtesy of the IceCube collaboration.

8.4 Estimation of Event Properties for IACTs

Maximilian Linhoff
Jens Buß

Abstract: To be able to do gamma-ray astronomy and obtain spectral information about the sources using IACT data, three properties of each recorded air-shower event have to be estimated: the energy of the primary particle, its type, and its direction. Estimating the energy is a one-dimensional regression task. The type of a particle is in general a multi-class classification. However, for standard gamma-ray analysis it is simplified to a binary classification into “gamma” (the signal class) and “hadron” (the background class). Direction is a two dimensional regression, since the two angles of the spherical celestial coordinate system have to be estimated. In IACT arrays like MAGIC or CTA, geometric approaches utilising the fact that the air shower was observed from multiple angles, usually deliver the best results. However, this task is considerably harder for monoscopically operating telescopes, like FACT or the LST-1 prototype. Here, machine learning instead of the geometric approach delivers the most competitive results.

In this section, we present the analysis chain based on the feature extraction from Chapter 7, using classical machine learning approaches to solve these tasks for FACT, CTA, and the LST-1 prototype.

We present the used algorithms and obtained results of this analysis chain, including energy-dependent benchmarks for the analysis such as the angular resolution of the directional reconstruction, the bias and variance of the energy estimation, the ROC curve of the gamma/hadron classification, and the overall the point-source flux sensitivity.

For the telescope arrays, an additional challenge is how to best combine the data and/or predictions of the single telescopes into a common prediction for the observed air shower.

In this section, we discuss how to estimate the relevant properties of the primary particles which induce the air showers from the features extracted as described in Section 7.3. This section was partly adapted from two Ph.D. theses written in the context of the CRC 876 [101, 294].

To do gamma-ray astronomy and the spectral analysis of gamma-ray sources, three main properties have to be estimated for each recorded air-shower event:

Particle type IACTs measure thousands of air showers induced by charged cosmic rays for each measured gamma ray. To reduce this background as effectively as possible, events are classified as induced by gamma rays or hadrons. This is a classification task.

Primary energy While techniques such as unfolding could directly estimate a spectral distribution from several input features, the condition of the inverse problem usually improves when using a direct estimator of the primary energy. Using a single estimate also makes it much easier to define common data formats for event lists and instrument response functions since the low-level image features will differ between the different observatories and software packages, but an estimated energy is very universal. This is a one-dimensional regression task.

Particle origin To enable gamma-ray astronomy, we need to know from where the particle actually came. In the end, we want to know this direction expressed in the fixed coordinate system of the International Celestial Reference Frame (ICRS). Usually, the estimation is done in the local coordinate system of the detector, say, using horizontal coordinates, which are then transformed into the sky-fixed system. The direction of the particle also plays a crucial role in the background suppression, as the cosmic ray background is isotropic compared with the compact or point-like gamma-ray sources.

The two-dimensional regression to estimate the origin of a gamma ray is especially hard for monoscopic IACTs, as multiple telescopes allow simpler and more precise geometric reconstruction methods. We will discuss both approaches, a machine learning-based solution for monoscopic telescopes and several approaches for stereoscopic arrays.

Historically, these tasks were performed by applying event selection criteria, often called *cuts*, for the particle classification and hand-crafted regression formulas fitted to simulations for the energy estimation and reconstruction of origin, see e. g. [43]. Modern methods of machine learning have improved performance and removed human biases from these steps.

Additionally, a high precision timestamp of the arrival time of the event is needed, which is of special interest for pulsars like the one at the center of the Crab Nebula, which has a period of ≈ 30 ms [79]. This is usually produced by the trigger electronics at the telescope, can be used without further processing and can achieve sub-microsecond precision.

In case of stereoscopic observations, the question arises when and how to combine the data of the single telescopes to form a common prediction for the air shower. Multiple approaches of different complexity exist and are discussed towards the end of this section. This is also a topic of ongoing research to improve the analysis, especially for the complex arrays of CTA.

8.4.1 Labeled Training Data

Since we have well-defined reconstruction tasks and labeled data available from either Monte Carlo simulations (see Chapter 5) or measurements of regions where no (known) gamma-ray source resides and thus only background events are expected, supervised machine learning is the method of choice.

The labeled data for gamma rays always comes from Monte Carlo simulations, while two approaches exist for the background training data. The first possibility is to also use Monte Carlo simulations, but since these simulations are computationally expensive and more prone to mismatches (cf. Section 5.3.2) another possible approach is to use observed data from positions in the sky where no gamma ray source is. But the second approach however comes with its own challenges to make sure the "off data" is measured under conditions comparable with the data of interest.

Two different kinds of gamma-ray simulations are distinguished: simulations of a point source at a certain position in the field of view of the telescope and diffuse simulations, where gamma rays are uniformly sampled in the field of view.

Simulating point sources is more efficient if only known point sources at a specific position in the field of view are to be analyzed. This is usually the case for extra-galactic observations due to the large distances, relatively compact emission regions, and angular resolution of the telescopes. However, there are also numerous extended sources, mainly inside our own galaxy and unknown sources or wide-field observations. For these kinds of observations, the models need to be able to make predictions over the full field of view and the instrument response functions must also be evaluated over the full field of view.

However, since the cosmic ray background arrives isotropically, the estimation of the particle's origin and type should always be trained on diffuse gamma-ray simulations, as to not create biases towards a specific position in the field of view.

Simulations of background particle types are always diffuse. The full background for gamma-ray analyses is formed by all charged cosmic rays and diffuse gamma-ray emission. This includes protons, helium cores, heavier nuclei, electrons, and positrons. Two assumptions reduce the different training datasets we need:

1. Helium cores and heavier nuclei are similar to protons or, more precisely, the features that differentiate gamma-ray- from proton-induced showers are even more pronounced for heavier nuclei.
2. Electron- or positron-induced air showers are indistinguishable from gamma-ray-induced air showers on an individual basis, since the exact same processes take place, save for the very first interaction in the air shower.

With these two assumptions, it is sufficient to train the particle classification model on gamma-ray and proton simulations or observed off data, which includes all types of background particles.

This leaves us with two or three needed datasets for training our machine learning models:

1. diffuse proton simulations or off data labeled as background data for the particle type classification;
2. diffuse gamma-ray simulations for the origin estimation, as signal class for the particle type classification, and possibly for the energy regression in case the assumption of a known point source at a fixed position in the field of view cannot be made; and
3. point-source gamma-ray simulations for the energy regression in case the assumption above is valid.

In the following sections, the analysis developed for FACT on its open Crab data set is presented. The analysis configuration and software dependencies are described in the "Open Crab Sample analysis" repository.[2]

This analysis starts at the feature level stored in `HDF5` files as described in (Section 4.3.2 and Section 7.3). These preprocessing steps were performed using the FACT-Tools software [100]. For the machine learning tasks at hand, the `aict-tools` [295] are used. This python package builds on the machine learning algorithms implemented in `scikit-learn` [302] and provides command-line tools to train, apply, and evaluate machine learning models for the three tasks discussed above. These tools are configured using `yaml` files, specifying which algorithms to use, which features should be included,

2 https://github.com/fact-project/open_crab_sample_analysis.

random seeds, and other important options. This analysis approach was developed as part of [294] and is discussed in more detail there.

8.4.2 Particle Classification

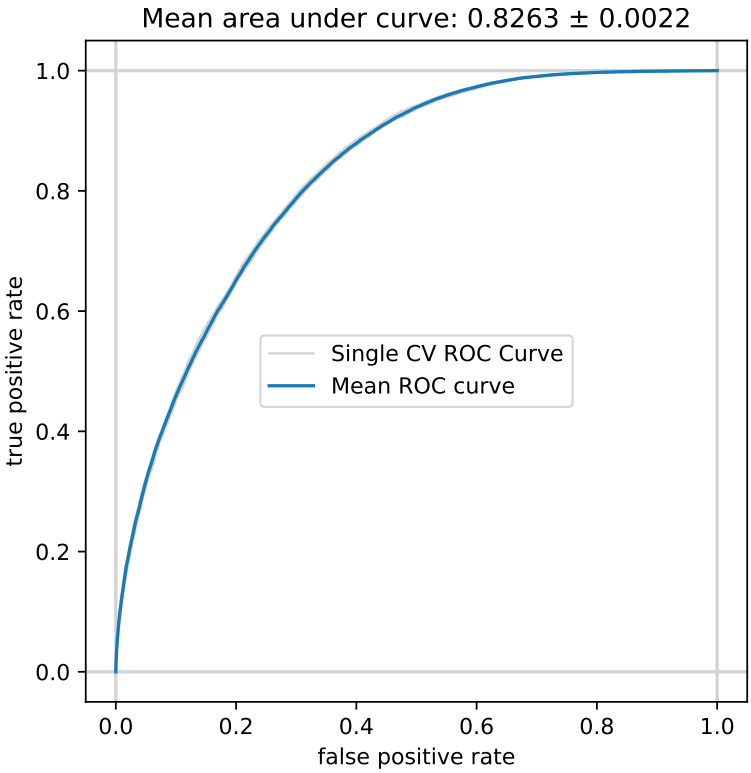


Fig. 8.14: Receiver-Operating-Characteristic curve for the gamma/hadron separation on the FACT Open Crab data set.

Most commonly, particle classification is done using decision tree-based ensemble methods including boosted decision trees and random forests, which are trained to distinguish gamma-ray-induced showers (signal) from proton-induced showers (background). The output score is usually called `gammaness`, with values close to 1 indicating that the shower was likely induced by a gamma ray. For the analysis of the FACT open Crab data set using the `aict-tools`, a random forest classifier is trained in a five-fold cross-validation on simulated diffuse gamma-rays as signal class and protons as background class. The resulting ROC curve for this classification is shown in Figure 8.14.

8.4.3 Energy Estimation

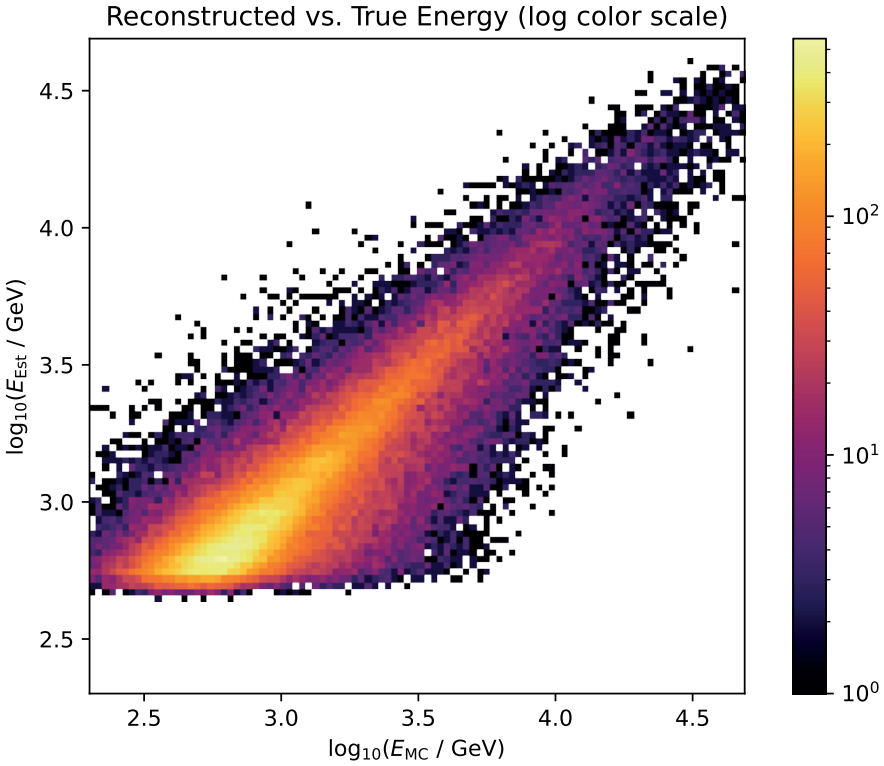


Fig. 8.15: Energy migration for the FACT open Crab data set.

Energy estimation is also most commonly done using decision tree-based ensemble methods, this time of the regression type. Instead of predicting the energy directly, it is beneficial to predict the natural logarithm of the energy with the MSE as a loss function. The choice of using the logarithm as opposed to the value itself is motivated in the large domain of the primary energies. The loss function would be completely dominated by few very high energy events if evaluated on the values themselves. Using the logarithm of the energy, the lower-energy particles also contribute to the loss and overall performance over the whole energy range is improved.

On the FACT open Crab data set, a random forest is also used. Figure 8.15 shows the and the resulting energy confusion, the resulting R^2 score is 0.7811 ± 0.0036.

8.4.4 Origin Estimation

In general, the estimation of the gamma-ray origin is a two-dimensional regression task, as either two coordinates in the detector plane or on the sky have to be estimated. For stereoscopic operations with many telescopes, geometrical methods are possible and perform well. With a single telescope, no stereoscopic reconstruction of the direction is possible and it has to be estimated from the single image or the image parameters. The most straight-forward approach would be a two-dimensional regression. However early in the history of gamma-ray data analysis, the so-called "disp method" was invented that reduces the problem to a one-dimensional regression and a classification task building on the Hillas image parametrization [241]. First solved with simple parametrizations, the regression and classification tasks lend themselves to being solved using supervised machine learning as with the other tasks. The "disp method" can also be used for arrays and was shown to perform better than the geometrical methods in situations with very few telescopes, as for example the two-telescope MAGIC system [289].

To simplify the task, the assumption is made, that the source position lies on the reconstructed shower axis. This simplification introduces a source of error if the shower axis is not properly reconstructed. However, it also enables the decomposition of the origin estimation into two tasks that are more easily solvable. First, the objective of the regression task is to find the absolute distance from the center of gravity of the shower image to the origin of the shower called `disp`. Second, the classification task is to find the direction on the shower axis that can be interpreted as the sign of `disp`. This is also referred to as "head-tail disambiguation" and is shown in Figure 8.16.

This method was developed for the Whipple telescope [241], greatly improving the sensitivity for point sources over the methods used before, but this did not predict a source position; it only used the orientation of the Hillas ellipse towards the assumed source position. Disp was parameterized using

$$\texttt{disp} = \xi \left(1 - \frac{\texttt{width}}{\texttt{length}}\right), \quad [241, (7)] \tag{8.18}$$

which is also the parameterization used by the FACT-Tools analysis. The parameter ξ was found either by optimizing it on observed data of a known, strong source as in [241] or on simulations as for the FACT-Tools. As this parameter ξ is just a fixed number, (8.18) is not able to predict $|\texttt{disp}|$ well over a large energy range. The value used in the FACT-Tools, $\xi = 1.38^\circ$, performed best around an energy of 3 TeV, see Figure 8.17, but much worse at higher energies. The FACT-Tools head-tail disambiguation used features dependent on the known position of a point source. While this greatly increased the efficiency of the prediction—especially for low-energy gamma rays—it also increased the false positive rate as all showers including the diffuse hadronic background were more often predicted towards the suspected gamma source. As the prediction is not independent of the source position, it has to be repeated for the off positions, increasing the computational costs with each off position. It also prevents the creation of skymaps, as there is no single prediction for an event.

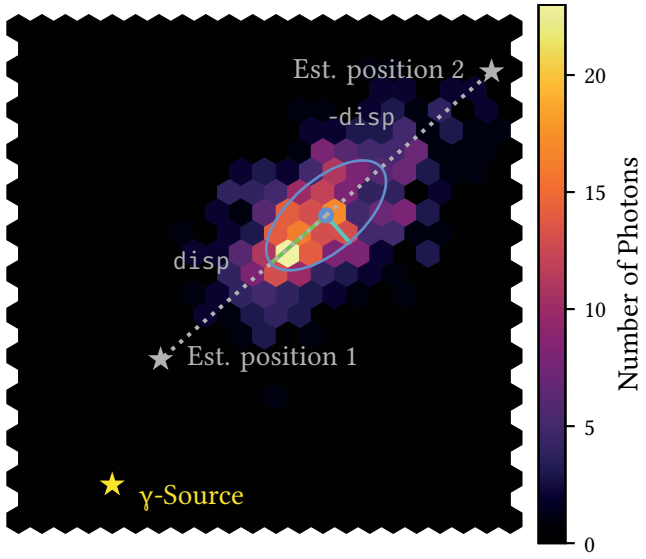


Fig. 8.16: The two possible reconstructed source positions for a given |disp|. The selection of the correct one is the target of the head-tail disambiguation.

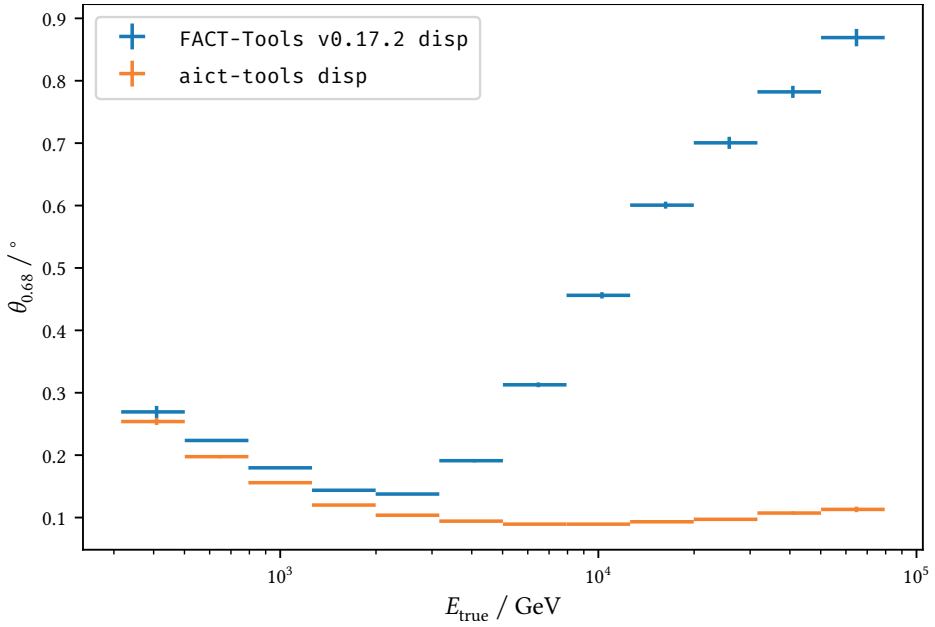


Fig. 8.17: Comparison of the angular resolution, defined as the 68 % containment radius of the distance between estimated and true origin, for the machine learning-based method and the previously used approach. The improvement is dramatic, especially at high energies.

The MAGIC experiment improved on the simple parameterization in Equation 8.18 using parameters depending on image parameters, e. g. `size`, to model energy dependencies and `leakage` to better handle showers not fully contained in the camera. Finally in 2009, before MAGIC-II was built and MAGIC moved to stereo reconstruction techniques, using a random forest regressor to estimate |`disp`| was evaluated and was found to perform better than the previous parameterization. In that case, the prediction for sgn `disp` was done using the difference between the image center of gravity and its brightest pixel, which is related to the skewness of the light distribution. To adapt these methods for the FACT-Tools analysis chain, the `disp` estimation was built into the `aict-tools`. To further improve over previous works, a random forest classifier is employed for the head-tail disambiguation. This results in a single prediction of origin for each event in the detector coordinates frame, which can be transformed to a position in the ICRS. A comparison of the results of the machine learning-based approach to the previously used approach is shown in Figure 8.17.

8.4.5 Combining Multiple Telescopes

With multiple telescopes, the reconstruction of the origin is a much simpler task. The main shower axis in each telescope camera together with the telescope position defines a plane. The direction of the primary particle is now the intersection of all planes defined by the individual telescopes. The solution proposed by [101] is to take the weighted mean of all possible two-telescope intersections, thus averaging over $N \cdot (N-1)/2$ intersecting lines of two planes. As weights for the individual telescopes, $w_i = \text{size} \cdot \text{length/width}$ is used, which puts larger weights on bright (large-size) and more elongated events, as the determination of the shower axis is more precise for those.

This geometrical approach performs best for large numbers of telescopes, i. e. $N \geq 4$. For the stereoscopic arrays with few telescopes such as two MAGIC telescopes or the very-low energy showers in CTA, a stereoscopic version of the disp approach yields slightly better results [289].

In this scheme, each individual telescope estimates disp, yielding two possible source positions per telescope. Now, the least-squares solution only using one source position from each telescope is used.

For the other parameters, weighted averages of single-telescope predictions are used. However, the machine learning models can receive features from the stereo reconstruction and aggregated values of the array, much improving the learner performance over the monoscopic case.

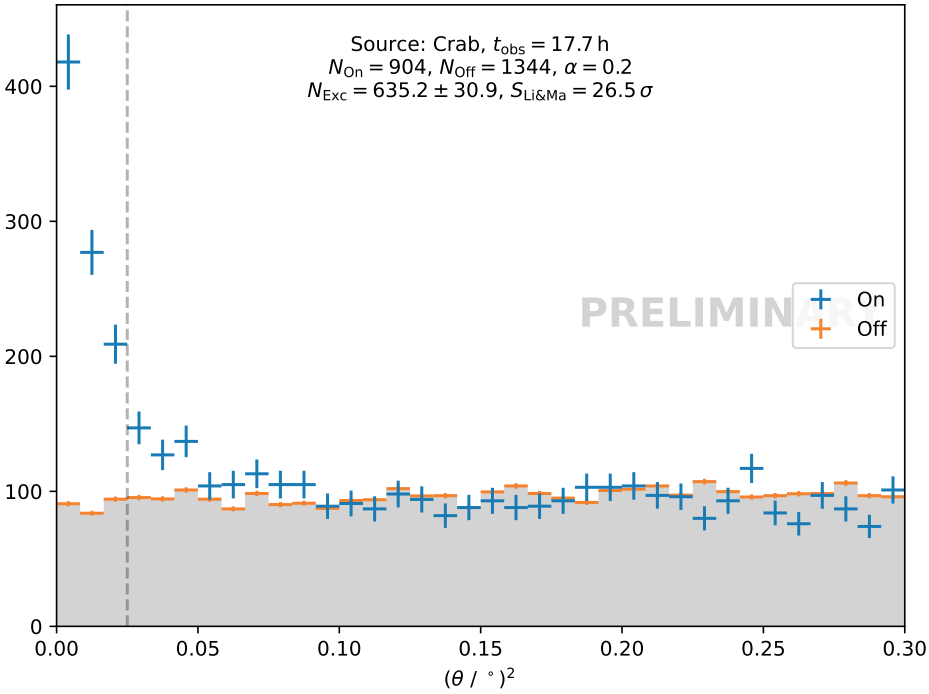


Fig. 8.18: Squared distance between estimated and assumed source position. The Crab Nebula is in the region called “On”, while “Off” is the background estimated from 5 geometrically equivalent positions in the field of view. A likelihood-ratio [250] test is performed to test the null hypothesis of no gamma-ray emission in the on-region. The null hypothesis is here rejected with a p-value corresponding to 26.5 σ.

8.4.6 Final Event Selection

The last step of the event reconstruction—or rather the first step of any following analysis—is to select the events for further processing, discarding events based on the estimated `gammaness` and, if analyzing point sources, the estimated direction.

As always, this is a trade-off between background suppression and signal retention. There are different trade-offs to be made for different kinds of studies. Spectral- and spatial analyses usually can live with a higher background, while the detection of weak sources usually needs stronger background suppression to find a significant signal at all.

Up to a point, the performance of the different reconstruction tasks also increases as energy increases. It is therefore beneficial to not apply a single-selection threshold in gammaness and a reconstructed source position. Rather, one is needed that depends on the estimated energy.

Figure 8.18 shows the distribution of the distance between the estimated source position and the catalog position of the Crab Nabula for all events with gammaness > 0.8. There is a large excess towards 0 over the background estimated from the off-regions, which are positions inside the field of view that are geometrically equivalent to the Crab Nebula position, but where no gamma-ray source resides.

8.5 Keynote: Data Analysis at ATLAS

Johannes Erdmann

Abstract: Machine learning plays a crucial role in many aspects of data analysis at the ATLAS experiment. ATLAS is one of the multipurpose particle-physics detectors at CERN's Large Hadron Collider and targets a diverse scientific program. This section highlights a few aspects of machine learning in ATLAS, which are taken from two scientific areas: the search for rare top-quark processes and the search for new heavy particles. The primary application of machine learning is driven by the need for excellent classification to efficiently suppress the large background processes and allow for better sensitivity to the signal processes. A particular challenge in all of the applications is given by the fact that the classifiers are trained on simulations, while their effect in real collision data must be well modeled in order to be reliably used in the data analysis.

8.5.1 Introduction

The ATLAS experiment [71] is one of the four large experiments at CERN's Large Hadron Collider (LHC). ATLAS is an omni-purpose detector with the typical onion-like structure of collider detectors. The particle collisions occur in its center, and the detector consists of different cylindrical detector parts around the interaction point. The first detector part is made of tracking detectors in a magnetic field. The curved trajectory of charged particles leaves hits in the different layers of the tracking detector, measuring the direction of the particle's track as well as its momentum via the track's curvature. The second detector part is made of calorimeters, which measure the deposited energy of electrons[3], photons, and hadrons. Electrons and photons initiate electromagnetic cascades ("showers") in the first layers of the calorimeter known as the the electromagnetic calorimeter (ECAL). While these showers are typically fully contained in the ECAL, showers induced by hadrons reach into the surrounding layers, which are therefore called the hadronic calorimeter (HCAL) and are deep enough to contain these hadronic showers. The calorimeters are segmented into cells so that the direction of the initial particle is reconstructed in addition to its energy. The calorimeters are surrounded by tracking chambers that are immersed in a magnetic field. These are called muon chambers because muons are the only charged particles that are expected to traverse the calorimeters. The direction and momentum of muons are hence measured via the hits and the curvature in the muon chambers.

The ATLAS detector is designed to pursue a large range of particle-physics goals, which are primarily focused on the analysis of proton-proton interactions that are provided by the LHC. The goals include precision tests of the predictions of the Standard Model of particles physics (SM) as well as the search for new particles that are predicted in theories beyond the SM (BSM). While the latter would show up as resonant deviations from the SM-only hypothesis, the former would result in non-resonant deviations. Each crossing of the proton bunches in the LHC can result in an "event". However, most events that occur in the collisions are not of interest. Even after selecting events for permanent storage with a rate of approximately 1000 Hz with triggers, the majority of recorded events are background events. The main task for analyzing the data, therefore, is making an excellent selection of the events to enrich the event sample with signal events. Classification based on machine learning (ML) is hence of enormous importance for data analysis. Its use is primarily focused (although not limited to) the separation of signal from background events on the event level as well as the discrimination of particles in the detector (electrons, photons, muons…) from misidentified other particles (known as fakes).

3 Following the jargon in collider physics, no distinction is made between reconstructed particles and anti-particles. In the detector, a positron is simply an electron with a track curved in the opposite direction.

When searching for rare processes, all other SM processes with their respective cross-sections are considered background. Actually, if the rare process is only present in a BSM theory (such as the production of a new particle), all SM processes are part of the background. In any case, the modeling of the background is of crucial importance. In many cases, it relies on Monte Carlo (MC) simulations (cf. Chapter 5). These simulate the relevant process at a given order in perturbation theory (for example, next-to-leading order in α_S), and simulate the evolution of quarks and gluons in the parton shower, their hadronization, and the response of the detector, so that the output format of the MC simulations is identical to real raw collision data (digitized voltages or currents in individual detector cells for example). The same algorithms used to reconstruct the particle signatures from the raw detector output are then used for real collision data and simulations. Typically, these MC simulations already provide excellent modeling of the SM processes. However, the quality of the predictions depends on the process, and there are processes for the parts of the generated phase space that are known to be subject to large uncertainties. All MC simulations are improved using data-driven corrections so that, say, the identification of a muon in data and MC simulations is corrected to agree based on measurements in clean reference processes ($Z \to \mu^+\mu^-$ and $J/\psi \to \mu^+\mu^-$ in this case). The MC simulations and the data-driven corrections come with their set of systematic uncertainties so that the full background model includes a nominal prediction and a well-defined set of systematic uncertainties.

When searching for rare processes, the presence of the signal is tested by a hypothesis test. The goal is to reject the background-only hypothesis with large significance. Suppose no such deviation from the background-only hypothesis is found. In that case, however, the signal-plus-background hypothesis is tested with a varying degree of the signal so that an upper limit on the minimal amount of signal is set that would have been incompatible with the observed data if it existed. The hypothesis test is typically based on a profile-likelihood fit with the likelihood ratio as a test statistic [131]. The fit can be binned or unbinned, and systematic uncertainties are included as nuisance parameters. In order to avoid overly optimistic exclusion limits, the CL_s technique is used [323].

In this article, two representative examples of searches for rare processes with the ATLAS experiment, focusing on the analysis strategy and the use of ML[4], will be discussed. These examples highlight typical analysis strategies in searching for non-resonant and resonant deviations from the SM hypothesis with a binned profile-likelihood fit. The non-resonant search shows the use of ML as an event classifier, while the resonant search uses ML to discriminate different particles.

4 The choice of the examples is obviously driven by personal bias.

8.5.2 Rare Top-Quark Processes: Searching for FCNC Processes

Processes with flavor-changing neutral-current (FCNC) interactions are examples of extremely rare SM processes, as these are forbidden at the tree level and highly suppressed at the loop level due to the Glashow-Iliopoulos-Maiani mechanism [178]. For the top quark, these couplings are so weak that they translate into branching ratios (BR, $\mathcal{B}$) in the top quark of the order of 10^{-17} to 10^{-12} [42]. However, BSM theories can introduce new particles that effectively introduce stronger FCNC interactions that could be detectable in top-quark processes. These could result in the decays of the top quark to an up or charm quark plus a neutral boson, such as a photon. However, the same vertex also gives rise to the anomalous production of the top quark in association with the neutral boson. In our example this is the photon, as shown in the Feynman diagram[5] in Figure 8.19.

This is the process searched for in this analysis [67]. It is an example of an analysis with a MC-driven background estimate that is used in a binned profile-likelihood fit to search for a BSM enhancement over the SM background hypothesis. It is also an example of an event-level classifier based on neural networks to distinguish the rare signal process (FCNC production) from the background. As shown in Figure 8.19, the outgoing particles ("final state") consist of a top quark and a high-transverse-momentum (p_{T}) photon.[6] The top quark decays almost entirely into a W boson and a b quark, and the W boson can either decay leptonically, $W \to \ell\nu$, or hadronically, $W \to qq'$. Despite the smaller BR of the leptonic decay, it has the huge advantage of much smaller backgrounds due to the characteristic feature of a high-p_{T} electron or muon.[7]

The SM background processes consist of irreducible and reducible backgrounds. An important irreducible background is $W + \gamma$ production together with additional jets from the hadronization of quarks or gluons, including potentially including b jets from the hadronization of b quarks. An important reducible background is $t\bar{t}$ production in the dilepton decay channel, i.e. both top quarks decay leptonically, where one of the leptons is an electron that is misidentified as a photon. Both electrons and photons, result in electromagnetic showers with the main difference being that the electron shower is expected to have one track pointing to the center of the energy deposition in the calorimeter. If that track is mismeasured, the electron is easily misidentified as a photon. The different background contributions are shown in Figure 8.20, where

5 Feynman diagrams are a visual representation of interaction processes of elementary particles: the interacting particles enter from the left ("initial-state particles"), interact via vertices and the exchange of virtual particles (in this case: the up or charm quark in the middle), and the outgoing particles ("final-state particles") are shown on the right side of the diagram.

6 At hadron colliders, the initial momentum in beam direction, p_z, of the colliding partons within the proton is not known. Hence, momentum conservation of the final-state particles is given only in the transverse direction. This is the reason for the frequent use of transverse momenta, p_{T}, at hadron colliders.

7 Tau leptons are typically not used, because of their many different decay modes.

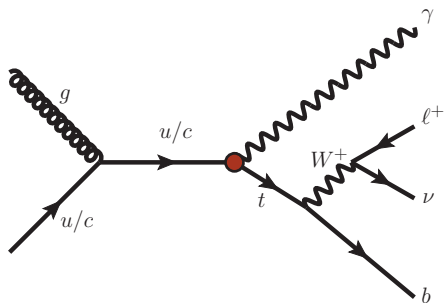


Fig. 8.19: Example Feynman diagram for the FCNC process [67]. The FCNC vertex is highlighted with the red dot.

"$e \to \gamma$ fake" is mostly from dilepton $t\bar{t}$ events. All background contributions are initially estimated from MC simulation but are checked and adjusted using data. In particular, the $W + \gamma$ background is estimated in a control region that is kinematically close to the phase space that is used to search for the FCNC signal (signal region) but enriched in $W + \gamma$ production. The modelling of the $e \to \gamma$ fake probability is checked by comparing $Z \to e^+e^-$ events, i.e. e^+e^- events with an invariant mass (m_{ee}), close to the Z-boson mass (m_Z), with $e\gamma$ events with a similar invariant mass. These $e\gamma$ events with $m_{e\gamma} \approx m_Z$ originate from $Z \to e^+e^-$ production with a misidentified electron. Comparing the ratio of ee and $e\gamma$ events in data and MC results in a correction factor that is applied to the MC simulations and that corrects for the difference in the fake probability between data and MC.

Now that the background contributions are well estimated, the task is to discriminate the backgrounds from the signal process to be sensitive to signal cross-sections as small as possible. The reason is that the signal cross-section is so small in the SM that it is impossible to measure it at the LHC. By targeting signal cross-sections that are as small as possible, however, we address BSM theories that predict cross-sections of at least that value. This means we need excellent discrimination of the signal from the background, which is achieved in this analysis with a fully-connected feed-forward neural network (NN) as a binary classifier. The NN is trained based on the MC simulations that have been corrected using the data in the control regions, using a set of ten well-discriminating input variables. Most of these variables are kinematic variables, such as the transverse momentum p_T of the photon (Figure 8.20, left), a particle that shows, on average, a much higher transverse momentum for the signal than for the background. Another input variable is the sign of the electric charge of the charged lepton (Figure 8.20, right). Most backgrounds are charge-symmetric, in particular $t\bar{t}$ production, which constitutes the main part of the $e \to \gamma$ fake background. The FCNC signal, however, couples a top quark to either an up or a charm quark (cf. Figure 8.19). These couplings to the up and charm quark are searched for separately in the analysis. In the case of the up-quark coupling, there is a preference for the production of top

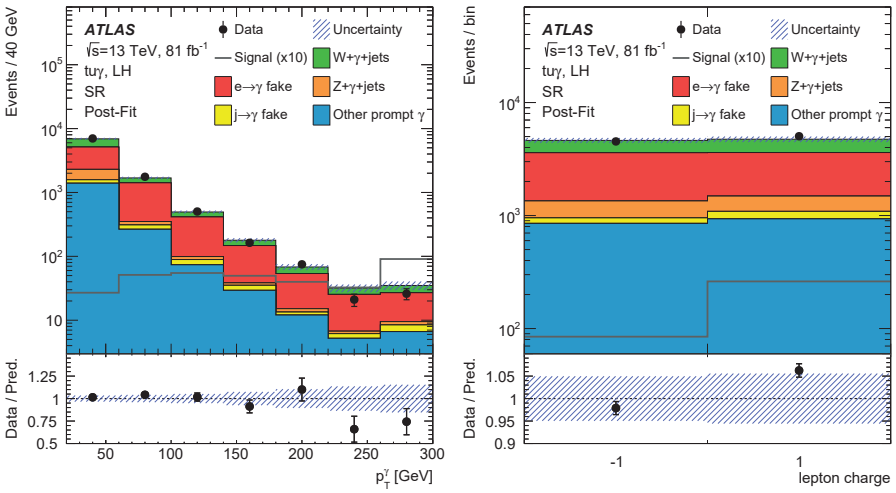


Fig. 8.20: Data-MC comparison of input variables to the FCNC neural network: transverse momentum of the photon (left) and sign of the lepton charge (right). The signal is overlaid with an arbitrary cross-section and the bottom panel shows the ratio of data over MC prediction [67].

quarks as opposed to anti-top quarks because the proton contains up quarks as valence quarks but anti-up quarks only as sea quarks. This results in a preference for positively charged leptons from the top-quark decay, which is shown in Figure 8.20 on the right.[8] What is essential to notice is that the input variables show excellent discrimination of signal and background and that—very importantly—the description of the input distributions in the MC samples agrees with the data within the uncertainties in the background-enriched regions. This is crucial because the MC samples are used to train the NN classifier, and a badly-modeled input variable would likely lead to the NN output distribution also being poorly modeled by the MC samples (cf. Section 5.3.2). This step must, of course, be checked before training the NN and, in particular, before "unblinding" any bin of a distribution with good sensitivity to the FCNC signal. In order to avoid any (unconscious) bias in the design of the analysis (selection, NN training, systematic uncertainties, etc.), a blinding criterion is formally applied, and the unblinding of the sensitive bins is formally agreed upon in the ATLAS-internal review process when the analysis strategy is fully defined. The blinding criterion is typically based on the signal-to-background ratio so that bins with a ratio of $> X\%$ are blinded. For signals with a varying cross-section, such as the FCNC signal in this analysis, the upper limit on the cross-section from the previously best search can serve as the reference cross-section for the blinding criterion.

8 Charm and anti-charm quarks both come from the proton sea. So, there is no preference for a positive lepton charge in the case of the FCNC coupling to the charm quark.

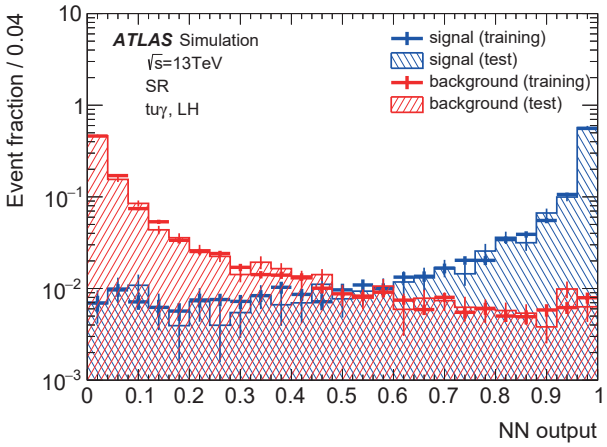


Fig. 8.21: Output of the event-level neural network in the FCNC search for signal and background, for training and test datasets in each case [67].

The available MC datasets are then divided into a training, a validation, and a test dataset, with the training dataset being the largest of the three. The NN is trained on the training dataset with different sets of hyperparameters (number of hidden nodes, number of hidden layers, hyperparameters of the NN optimizer, etc.), and the best set of hyperparameters is chosen based on the discrimination as evaluated on the validation dataset. Possible overfitting is tested by comparing the NN output distributions for the training and test datasets, as shown in Figure 8.21. Background and signal distributions are much better separated in the NN output distribution than for the individual input distributions, with the signal peaking at NN values around one. Also, the training and test distributions agree within the statistical uncertainties of the MC samples for the signal as well as for the background, indicating the absence of overfitting. Actually, this test is only shown for one type of FCNC signal in Figure 8.21, i.e. an FCNC coupling to the up quark with left-handed chirality. Indeed, the whole analysis (including the NN training) is repeated for a right-handed FCNC coupling and for left- and right-handed FCNC couplings to the charm quark. This is necessary because the distributions of the input variables change depending on the choice of the FCNC coupling. The difference between up- and charm-quark couplings, which arises from the coupling to valence up quarks in contrast to charm quarks from the proton sea, leads to different kinematic and lepton-charge input distributions. The effect from the chirality of the coupling is smaller but still influences the kinematic distributions of the top-quark decay products. Due to the specific nature of the weak interaction, which only couples to left-handed particles, the weak top-quark and W-boson decays result in different kinematic distributions of the lepton p_{T}, depending on the handedness of the FCNC coupling.

After unblinding, a binned profile-likelihood fit is performed to the NN output distribution in the signal region and distributions in two control regions. The result of this fit is shown in Figure 8.22 for the signal region and the $W + \gamma$ control region. This fit includes nuisance parameters for the systematic variations constrained to their respective systematic uncertainties. Performing the fit to all regions simultaneously allows these nuisance parameters to be adjusted to fit the best data. The data in the control regions allow us to adapt and potentially constrain the nuisance parameters of the background modeling.

Indeed, some nuisance parameters may be left entirely unconstrained in the fit, which in this analysis was done to normalize the $W + \gamma$ process. Its normalization is entirely determined by the data, so that the $W + \gamma$ normalization in the signal region is primarily determined by the data in the $W + \gamma$ control region. However, one needs to be careful, though, to make sure that such extrapolations make sense. Actually, the $W + \gamma$ events in the signal region are primarily produced in association with b-jets, while this is not the case for the $W + \gamma$ events in the control region. One solution would be to introduce separate nuisance parameters for the $W + \gamma$ normalization in the control and signal regions so that the normalizations are fully decoupled. The solution used in this analysis is to apply an additional nuisance parameter for $W + \gamma$ production in association with b-jets, according to its systematic uncertainties. This additional nuisance parameter is then unconstrained mainly by the data in the control regions.

As seen in Figure 8.22, the NN output distributions in the control and signal region are well described by the post-fit MC predictions, i.e. after all nuisance parameters are adjusted in the profile-likelihood fit.[9] This is good news regarding background modeling but bad news regarding the search for the FCNC signal: no significant deviation from the SM prediction (expected around NN outputs of one) is observed, and no BSM physics is found. But this also allows us to constrain FCNC interactions of the top quark and photon much more strongly than ever before. Testing how strong the signal must have been so that it would have been very unlikely to observe the background-like data that we did observe, we can set upper limits on the FCNC signal cross-section, which we can translate into upper limits on the FCNC couplings' strength. One can then further interpret these upper limits on the couplings as an upper limit on the BR for the FCNC decays $t \to u + \gamma$ of 2.8×10^{-5} and 6.1×10^{-5} for left- and right-handed coupling, respectively, as well as upper limits on the BR for the decay $t \to c + \gamma$ of 2.2×10^{-4} and 1.8×10^{-4} (again for left- and right-handed coupling). These were by far the most sensitive limits on these couplings and BRs when this analysis was published and still are at the time of writing.[10]

9 The sixth bin in the $W + \gamma$ control regions shows a discrepancy that is not significant.

10 That is, January 2022.

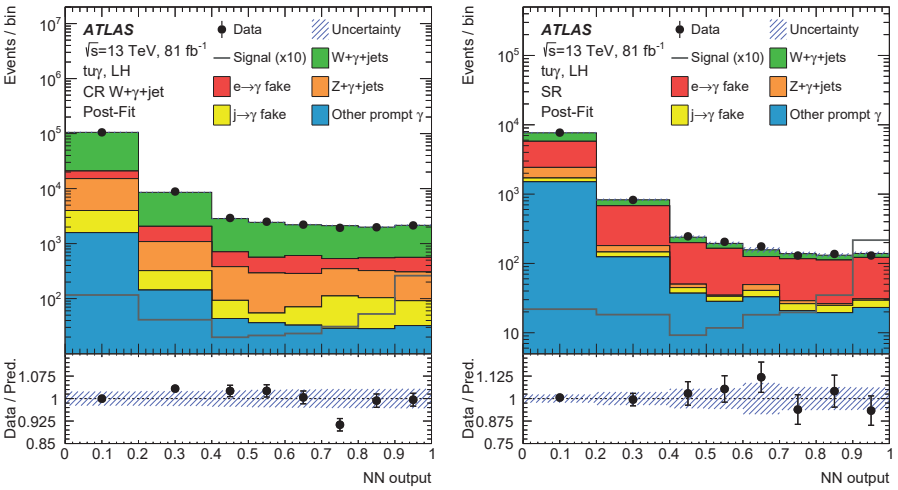


Fig. 8.22: Data-MC comparison of the output of the neural network in the FCNC search for the $W + \gamma$ control region (left) and for the signal region (right). The signal is overlaid with an arbitrary cross-section and the bottom panel shows the ratio of data over MC prediction [67].

8.5.3 Searching for New Heavy Particles: Vector-Like Quarks

The previous section discussed an example of a search for non-resonant BSM contributions on top of the SM background. This means that we were looking for new particles that cannot be resolved at the LHC because, say, their masses are too large. However, the LHC provides the highest center-of-mass energies ever achieved in a laboratory particle-physics experiment, which means that we can search for new, massive particles directly via their resonant contributions to SM spectra.

One example of a type of new, massive particle that could be observed at the LHC is a vector-like quark (VLQ). One of the big questions of the SM is why there are exactly three generations of fermions. A priori, there is no good reason for the number of generations to be three, and one can ask whether there is a fourth generation of quarks. After the discovery of the Higgs boson, however, it has become clear that this fourth generation cannot just be another, heavier chiral quark doublet [145], because the Higgs boson couples to fermions proportionally to their mass, and such a heavy fourth generation would contribute significantly to the Higgs-boson production cross-sections and to several decay-branching ratios via loops. If the heavy quarks are vector-like, however, i.e. if their electroweak couplings do not follow the typical V−A structure but couple as a vector (V), the SM Lagrangian could simply include a direct mass term. Their masses would not originate in the Higgs mechanism, and the constraints from Higgs-boson cross-sections and branching ratios would not apply! Actually, there is much more motivation for these VLQs to exist. In particular, they appear in models

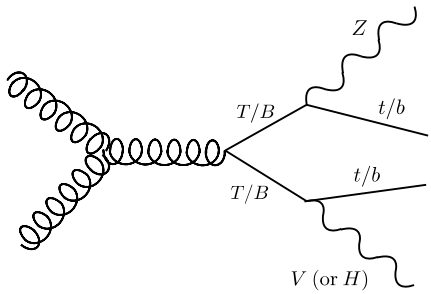


Fig. 8.23: Example Feynman diagram for the VLQ search [70].

that explain the large difference between the electroweak (246 GeV) and the Planck scale ($\mathcal{O}\left(10^{19}\right)$ GeV) [221] and in models that explain the large hierarchy of the fermion masses [220], if they couple primarily to the third quark generation, i.e. the top, and the bottom quark. This means that their primary decay channels would be one third-generation quark plus a boson: tZ, tH, tW, bZ, bH and bW. The focus of the search discussed here is VLQs with the same charges as the SM quarks ($+2/3e$ and $-1/3e$), which are traditionally denoted by the symbols T and B, respectively. The masses of these particles are free parameters of the search, as they are not predicted, but there is good motivation for them to be in order of 1 TeV.

As VLQs are colored particles, they can be produced in pairs at the LHC via strong interaction, just as any other quark. A Feynman graph is shown in Figure 8.23 together with some possible decays. The BRs of the VLQ decays are only predicted in specific models and are, in principle, free parameters of the search. One such model would be a specific electroweak representation of the VLQ. For example, in a model with a singlet T or singlet B, the BRs to quark + Z/quark + H/quark + W ratio is approximately 25 % : 25 % : 50 % for large VLQ masses. This means that many final states of VLQ pair production are possible, and it would be inefficient to design searches for each possible final state ($ZtZt$, $ZtHt$, $ZtWb$, $HtHt$, ...). The ATLAS VLQ search program was hence divided into several analyses that focus on one specific VLQ decay for the first VLQ and cover as much of all decays of the second VLQ in the pair. The analysis discussed here is the so-called "$Zt/b\ + X$ search", which covers final states with at least one VLQ decaying to a Z boson and a third-generation quark [70].

This search is complemented by other searches that focus on at least one VLQ decaying to Ht ("$Ht + X$ search") etc. For the $Zt + X$ search, the most promising strategy is to select events with Z boson decays to e^+e^- or $\mu^+\mu^-$, because of the excellent electron energy and muon momentum resolutions that make for the reconstruction of a sharp Z-boson peak and hence low backgrounds from non-Z-boson events. The final state hence

consists of two oppositely-charged leptons[11] in association with a varying combination of heavy resonances (top quarks, Z/W/Higgs boson) depending on the VLQ BRs. In addition, the final state contains at least two b quarks from the VLQ decay or from the top-quark decays and may even contain additionally charged leptons from the decays of top quarks or one of the bosons. While this results in a signature that is already quite distinct, there is an additional feature that can be exploited to separate the signal from the SM background: a VLQ with a mass that is much larger than the masses of its decay products. This means that the heavy resonances from the decay are strongly Lorentz-boosted and that the decay products of these resonances will have small angles to each other. Given that the main decay modes of these resonances are decays to quarks, this results in collimated streams of hadrons that can be reconstructed as jets.[12]

One can then analyze the structure of the energy depositions in the calorimeter cells to distinguish jets that originate from the heavy resonances from jets that originate from gluons or up, down, charm, strange, or bottom jets ("QCD jets"). Top quarks decay to three quarks ($t \to Wb \to qq'b$) and are likely to have a three-prong-like substructure. Z, W, and Higgs bosons decay to two quarks and likely have a two-prong-like substructure. In addition, Higgs bosons decay mostly to a pair of bottom quarks so that the number of b-tags associated with the jet helps to distinguish it from Z- and W-boson jets. The latter two types are difficult to distinguish, though, and are typically treated together as vector-boson (V-boson) jets.

The main development for the search discussed here was the development of a deep learning-based algorithm to distinguish these multiple classes of jets—hence its name: Multi-Class Boosted Object Tagger (MCBOT).[13] It is based on a deep fully-connected feed-forward neural network trained with MC simulations for the different classes of jets over a large range of jet $p_{\mathrm{T}}$and builds on developments in a VLQ search in fully hadronic final states [69]. The latter point is important because the higher the p_{T}, the larger its Lorentz boost. This results in smaller angles of the resonances' decay products and hence different patterns of the energy depositions in the calorimeters. As we do not know the VLQ mass, MCBOT must work well for a large range of masses and hence p_{T} of the VLQ decay products. MCBOT is not the first algorithm to identify boosted hadronic resonance decays. Much work was done previously for the identification of boosted hadronic top quarks as well as V-boson and Higgs-boson jets (see for example [62, 63, 65] for some work within the ATLAS Collaboration), and not the first to use ML for this task (for a top-tagging comparison of many different ML algorithms see [112]). However, the search for VLQs is different from many other use cases of top-/V-/Higgs-tagging, where only one type of resonance is present in the final state. As opposed to searches

11 Tau leptons are again neglected.

12 Actually, one needs large radius parameters for these jets (of the order of 1.0), as opposed to the relatively small radius parameter of 0.4 that is typically used in the experiment to reconstruct jets from QCD jets.

13 Here, the term "boosted" refers to boosted particles and not to boosted ensemble classifiers.

for resonances that decay to $t\bar{t}$ and for which the identification of boosted top quarks is enough, the VLQ final states are rich in all different classes of hadronic resonances, which calls for a multiclass strategy.

MCBOT is a deep neural network (DNN) with four different output nodes: one for boosted top quarks, one for boosted V bosons, one for boosted Higgs bosons, and one for QCD jets. As an activation function for the four output nodes, the softmax function is used so that each output lies in the range [0, 1], and all four output values add up to one. Instead of designed features, essential variables are used as input to the DNN: the four-vectors of the three jet constituents with the highest p_{T}, the information on whether these constituents are associated with a b-tag, as well as the jet invariant mass and its p_{T}. While constituents' kinematic and b-tagging information encode the decay kinematics of the underlying resonance decays and their quark flavor, the jet mass naturally distinguishes the different resonances due to their different masses. The jet p_{T}, however, is not a discriminating variable but rather a parameter for MCBOT: the algorithm must work well for low and high p_{T}, and hence all different input classes are reweighted to have the same underlying jet p_{T}.

However, as the Lorentz boost strongly influences the patterns within the jet, it is essential to pass this information to the DNN training. Another particularity of MCBOT is that its jets are not built from individual calorimeter depositions but from smaller jets, which are already calibrated [285]. This is important to assess the systematic uncertainties associated with the efficiencies for the identification of the hadronic resonances, which are estimated from MC simulations. Any potential difference in efficiency between MC simulations and data must be accounted for by systematic uncertainties, which are often estimated by comparing data and MC predictions in a reference process. However, it is difficult to identify hadronic resonances with very high p_{T}, which can be so high that any SM reference process runs out of statistics and makes it impossible to compare with MC predictions in a meaningful way. Building the jet from small jets, which have a calibration that comes with well-defined systematic uncertainties, circumvents this problem because the uncertainties from the small jets can be used to estimate the uncertainties for the larger jets straightforwardly. This is well worth the cost of not using the higher resolution within the jet that could be achieved with even smaller jet constituents that lack a proper systematic model.

The final DNN has a structure with four hidden layers, with 32, 27, 14, and 12 nodes each. The output distributions for V-boson jets, Higgs-boson jets, and top-quark jets are shown in Figure 8.24 together with the distribution for QCD background jets. MCBOT clearly discriminates the different resonances from background jets. Several distinct features can be observed in the distributions, such as the peaks around 0.9 and 0.8 in the Higgs-boson distribution. These features have been investigated in detail, and the Higgs-boson peaks around 0.9 and 0.8, for example, originate from Higgs-boson jets with precisely two and only one b-tagged constituent, respectively, i.e. jets in the peak around 0.8 are missing the second b-tag that is expected from the decay $H \to b\bar{b}$. In order to unambiguously assign each jet to one output class, the class with the

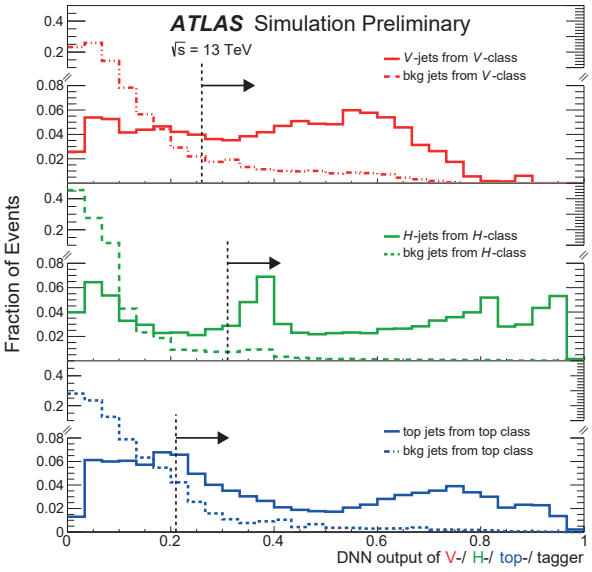


Fig. 8.24: Output of the different DNN output nodes, each for the respective signal particle (solid curves) and for QCD background jets (dashed curves): *V*-boson jet output (top), *H*-boson jet output (middle), and top-quark jet output (bottom). The dashed vertical lines indicate the thresholds that are used in the VLQ search [70].

largest softmax output is chosen. Additionally, a minimum threshold for a successful classification was optimized for each output class, shown as dashed vertical lines in Figure 8.24. Jets that pass these thresholds form MCBOT tags of the top, V, or Higgs type.

How is MCBOT now used in the VLQ search? The VLQ search uses many analysis regions based on the number and type of MCBOT tags, the number of associated b-tags, and the presence of two or at least three charged leptons. An overview of the different categories is shown in Figure 8.25 together with the MC prediction in each region and the observed data. An example signal that consists of a vector-like T quark with a mass of 1.2 TeV and BRs as predicted in the singlet model is also shown. Clearly, the number of expected and observed events differs strongly from category to category, and so due to the contributions from the different background processes as well as the signal-to-background ratio. In general, regions with many MCBOT tags have a better signal-to-background ratio and a lower number of events with the associated larger statistical uncertainty. Combining all categories in an overall hypothesis test gives the best sensitivity for the VLQ signal.

In addition, in each of the different analysis categories, a "final discriminant" is chosen that further separates the background from the signal events. One example of such a final discriminant is shown in a region with two MCBOT tags, two b-tags, and

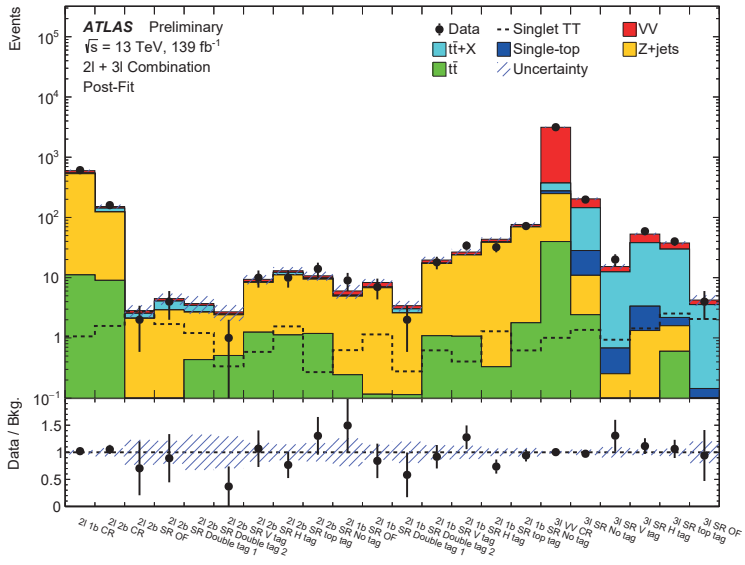


Fig. 8.25: Data-MC comparison for the different analysis regions in the VLQ search. In each region, the inclusive event yield is shown. The vector-like T signal is overlaid for an example mass of 1.2 TeV and singlet branching ratios. The bottom panel shows the ratio of data over MC prediction [70].

two charged leptons in Figure 8.26. The final discriminant is the invariant mass of the Z boson (reconstructed from the two charged leptons) and one of the b-tagged small jets. This is exactly the reconstructed invariant mass of a vector-like B quark. As mentioned at the start of this section, we are searching for a resonance on a falling background spectrum! Actually, the background expectation is very low due to the presence of the two MCBOT tags. The signal that is overlaid in the figure is a vector-like T signal, however, which is not expected to show a sharp resonance in this final discriminant. The T decays to Zt, which results in a final state with an additional W boson from the top quark decay in addition to the Z boson and the b quark. However, the T signal also shows an enhancement at larger values of the discriminant compared with the SM background. In principle, one could use different optimized final discriminants for the different signals. This was tested, and the gain was small compared with the additional complication of basically doing the analysis twice, i.e. once for each signal.

The combined fit to all final discriminants in all categories is then performed under the background-only hypothesis to test for the presence of a VLQ signal. Actually, several of the regions are very background-rich and serve as control regions, just as in the case of the FCNC search. Also, here, the good modeling of the data and the MC prediction give confidence in the overall background modeling. No significant excess above the SM background was found, which means that 95 % confidence-level upper limits are set on the VLQ production cross-section using signal-plus-background fits

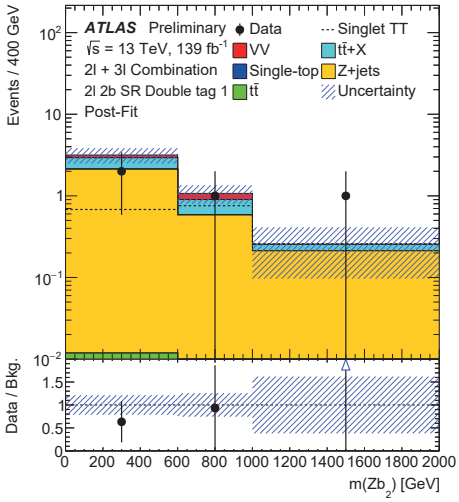


Fig. 8.26: Data-MC comparison of the final discriminant in an example signal region of the VLQ search. The vector-like T signal is overlaid for an example mass of 1.2 TeV and singlet branching ratios. The bottom panel shows the ratio of data over MC prediction [70].

as a function of the VLQ mass (for the T and B signals separately). Since we know the production cross-section for VLQ pair production from QCD, the intersection of the upper cross-section limit, and the theory cross-section results in a lower limit on the VLQ mass at 95 % confidence level. Assuming singlet BRs for the VLQ, the lower mass limit is approximately 1200 GeV for the B quark, for example (Figure 8.27, left). When we want to move away from specific BR assumptions, we can scan all possible BRs into SM particles, which results in a "BR triangle", shown in Figure 8.27 (right) for the B quark. The BR into Wt is shown on the x-axis, the BR into Hb on the y-axis, and the BR for the decay into Zb results from $1 - \mathcal{B}(Wt) - \mathcal{B}(Hb)$, i.e. high values for $\mathcal{B}(Zb)$ populate the lower left "Z corner", where this search is particularly sensitive. For very large values of $\mathcal{B}(Zb)$, masses below 1400 GeV are excluded at 95 % confidence level. At the time of writing[14], this analysis sets the most stringent lower limits on vector-like T and B quarks that decay primarily into Zt and Zb, respectively. For $\mathcal{B}(Zt/Zb) = 100\,\%$, these are as high as $m_T > 1600$ GeV and $m_B > 1402$ GeV, and hence significantly more stringent than the limits from the previous most sensitive limits from the combination of many ATLAS VLQ searches from a smaller dataset [59].

14 That is, January 2022.

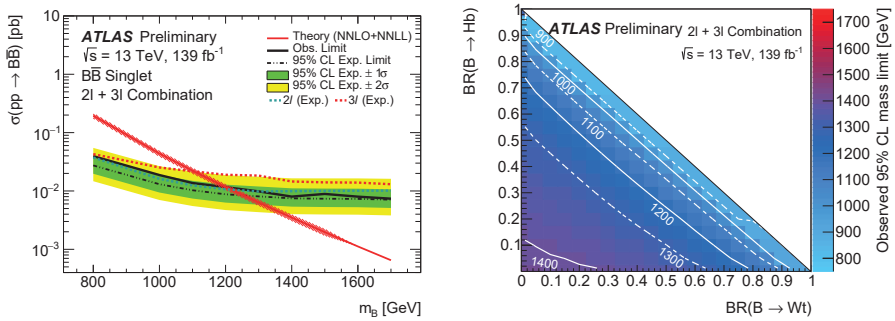


Fig. 8.27: Left: 95 % confidence level upper limit on the cross section for vector-like B production as a function of the B mass. Shown are expected limits from the 2ℓ and 3ℓ channels as well as their combination (dashed/dotted lines) and the observed limit (solid black curve). The theory expectation for singlet branching ratios is also shown. Right: 95 % confidence level lower limit on the B mass as a function of the branching ratios into Wt (x-axis) and Hb (y-axis). The branching ratio into Zb is calculated via $\mathcal{B}(Zb) = 1 - \mathcal{B}(Wt) - \mathcal{B}(Hb)$, i.e. it is 100 % in the lower left corner [70].

8.5.4 Conclusions

The search for rare processes with the ATLAS experiment crucially relies on a good modeling of the background as well as excellent discrimination of the signal from the background using classification algorithms. Shallow neural networks and boosted decision trees have a long tradition in collider experiments and have been successful in ATLAS for, say, the measurement of rare single-top-quark production with early LHC data [8], the observation of the very rare $t\bar{t}H$ production process [6], or the search for new particles, such as leptoquarks [9]. However, more and more advanced machine learning techniques are being exploited to improve the precision of the analyses. The Multiclass Boosted Object Tagger is one example that uses deep learning to separate several classes of jets.

While this chapter was written, the FCNC search was updated with a larger dataset [68], which showed that the multiclass approach is also useful in the case of signal-background discrimination if the signal consists of several components (in this case, the FCNC production and decay processes). Combining the three-class output of a deep neural network into a likelihood-ratio-inspired discriminant improved the sensitivity to the FCNC signal by up to 30 % compared with a binary DNN classifier [68].

Many more approaches have been and are being explored in ATLAS. Here are just some successful examples: the classification of jets as originating from the hadronization of a quark or a gluon with convolutional neural networks treating the energy depositions in the calorimeter cells as an image [66]; b-tagging with recurrent neural networks interpreting the set of tracks associated with the jet as an ordered series [64]; b-tagging with deep sets interpreting the set of tracks as a point cloud [60]; searching for new resonances in dijet events without assuming a specific resonance model

using weak supervision [7]; and simulation of showers in the calorimeter using Generative Adversarial Networks to approximate the precise but rather slow simulation with GEANT4 [61].

Further improvements in machine learning and its application to collider physics have the potential of advancing the sensitivity of the experiment further to better explore the (costly!) collider data. However, it is important to ensure that more complex algorithms do not exploit unphysical features in the training data. These could be correlations that are present in the MC simulations but are different in real collision data or in features of the data in control regions that are different in the actual signal regions. The validation of a very good modeling of the input variables, their correlations, and the output of the machine learning algorithms remains crucial when exploiting new exciting methodology.

9 Deep Learning Applications

9.1 Introduction

Wolfgang Rhode

"Deep Learning" is one of the machine learning methods and is therefore always referred to in this book when they are mentioned. However, deep learning has a special position in that it is based on algorithms called "neural networks" and thus has a very large number of free parameters that can be optimized to fulfill the learning task. Experience shows that the danger of overtraining, which is higher than with other algorithms because of the large number of parameters, can be averted. A frequently occurring uneasiness of physicists about the fact that the physical meaning of the resulting optimal parameter sets is not necessarily obvious can be minimized with statistical tests of the learning success. However, it may be advantageous or even necessary for the learning success if the algorithm is given further structural algorithmic help.

In three application areas, we show examples of how Deep Learning can be used in astroparticle and particle physics. In the neutrino telescope IceCube, convolutional neural networks are applied to data of shallow abstraction level to reconstruct events of different classes under the algorithmic inclusion of domain knowledge. In LHCb, B mesons are identified and interpreted. The construction of a Deep Learning-based analysis chain is presented from the field of gamma astronomy.

9.2 Deep Learning for IceCube

Mirco Hünnefeld

Abstract: Data of high-energy physics experiments, such as IceCube, is rare in the context of machine learning, insofar as its generation and the laws and symmetries it abides by are usually well understood. The goal of reconstruction methods is to utilize this information to infer physical properties from measured data. While traditional methods use this information explicitly, most machine learning approaches learn the underlying patterns implicitly from the training data.

In this section, we illustrate how a deep convolutional neural network (CNN) can be applied to IceCube data for event reconstruction. As systematic uncertainties are inherent to Monte Carlo simulations, emphasis is placed on the investigation of the *robustness* of the employed method and its response to systematic variations of the

https://doi.org/10.1515/9783110785968-009

baseline Monte Carlo simulation. Although successful, the CNN cannot fully utilize available information. To combat this deficiency, a hybrid reconstruction method is proposed that combines the strengths of traditional maximum likelihood-based methods with those of deep learning.

Data in high-energy physics experiments comes along with extensive knowledge of the underlying data-generating processes. Most experiments utilise complex Monte Carlo simulations (see Chapter 5), an approach requiring that inherent patterns, symmetries, and scientific laws exhibited in the data are well understood. The abundance of this detailed domain knowledge provides the opportunity for dedicated deep learning architectures that harness the full potential of the physics detector. Crucial to the success of this task is the exploitation of domain knowledge within the applied method.

The advent of machine learning in physics experiments illustrates a paradigm shift from explicit to implicit utilization of domain knowledge. In traditional data analysis, methods from physics and statistics, most prominently maximum likelihood-based reconstruction methods, are implemented with the aim of exploiting available *domain knowledge* directly. By contrast, standard machine learning applications and deep learning architectures do not directly use this information; rather, they learn it implicitly from the training data. Often these approaches are in direct competition with each other, with strong proponents on either side. Particularly in physics experiments, where *interpretability* and understanding of the inner workings of algorithms as well as their effect on analysis results is crucial, the advent of deep learning was initially perceived more reluctantly than in other domains. However, as illustrated in the following sections, classical and deep learning approaches do not have to exclude each other; instead, they can be used in symbiosis.

9.2.1 Domain Knowledge in IceCube

In this section, deep learning approaches to IceCube data are illustrated by the example of angular reconstruction of cascade events. Cascades are short-ranged particle showers with an expansion of several meters induced by neutrino interactions in the instrumented volume (see Section 8.3). These events are inherently difficult to reconstruct in IceCube as the particle shower appears almost point-like given IceCube's sparse instrumentation [17, 21]. Reconstruction methods must therefore be sensitive to subtle differences in the deposited light pattern in order to achieve high accuracy.

The direct exploitation of existing symmetries and prior knowledge in the neural network architecture may aid the reconstruction task by improving the model's performance. As this domain knowledge is already built into the architecture, it must not be learned implicitly in the training process, which speeds up and facilitates convergence. At the same time, the direct inclusion of symmetries allows the neural network to extrapolate along these symmetries. Therefore, the model can provide meaning-

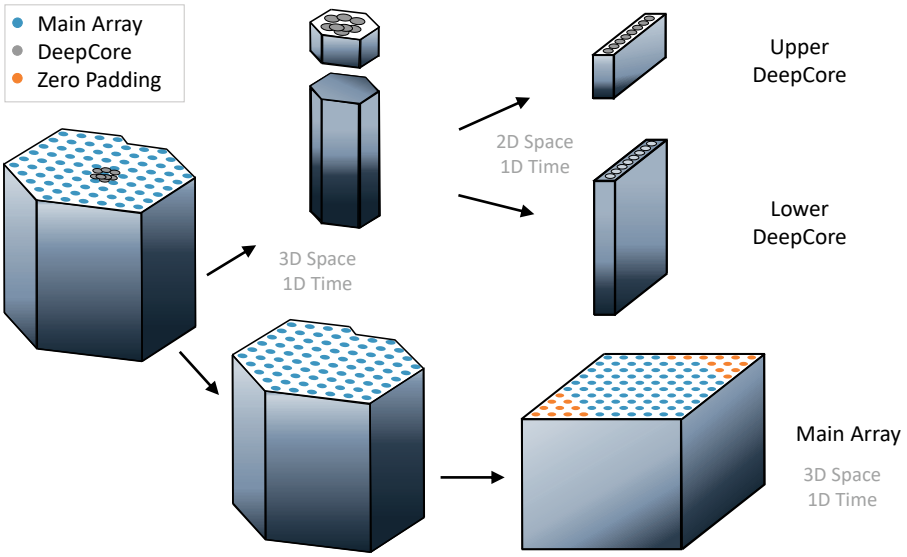


Fig. 9.1: The IceCube detector encompasses three main parts: the main array, consisting of 78 strings arranged on an approximate triangular grid and another eight strings in the center, which comprise upper and lower DeepCore. Data from the three detector components is separated, padded with zeros and transformed to fit into an orthogonal grid. This transformation allows for the utilization as input data to a CNN. [31] © 2021 IOP Publishing Ltd and Sissa Medialab

ful inter- and extrapolations beyond the phase space covered by the training data, as long as these are covered by the built-in symmetries. This leads to a more robust and complete utilization of available information. Standard neural networks are able to interpolate on the training data if this is sampled densely enough, and they can therefore appropriately model the symmetries. However, they are incapable of truly extracting and exploiting these symmetries unless the architecture itself supports this. For instance, convolutional neural networks (CNNs) utilize translational equivariance in order to exploit translational invariance of the input data [240], which has lead to breakthroughs in the field of image recognition [233]. A standard multilayer perceptron can theoretically learn the same mapping. However, in practice, it does not perform as well for image recognition tasks.

Available domain knowledge and symmetries in IceCube include translational and rotational invariance in the particle shower. The underlying physics of the neutrino interaction is invariant under translation and rotation—they do not depend on the location or orientation of the detector. However, this symmetry is only approximately valid in measured data, due to dust impurities and anisotropy in the detector medium [25] and the geometry of the detector grid (see Figure 9.1). Apart from these symmetries, further domain knowledge, such as the linear relation between deposited energy and detected charge or the properties of the detector itself, is known a priori.

As mentioned in Section 7.2, IceCube records photons in the photo-multiplier tubes at each Digital Optical Module (DOM). These are the measurements. The photons are converted to an electrical signal–charge as a function over time also known as a *wave form*. In a subsequent step, individual pulses, each consisting of a time and charge, are then extracted from the wave form. Each DOM may measure a pulse series of arbitrary length. A collection of such pulse series in a specified time window is referred to as an *event*. A reconstruction method aims at extracting the properties of the primary neutrino based on the pulse series and the given domain knowledge.

9.2.2 Convolutional Neural Networks in IceCube

In order for Neural Networks (NNs) to make reliable and robust estimates, the NNs have to learn the underlying patterns in the data. This can be achieved either implicitly during the training procedure by inferring them from the training data or explicitly by choosing appropriate network architectures. One of the dominating symmetries in IceCube is translational symmetry in the x-y-plane (parallel to the ice surface) and to first order also along the z-axis. Convolutional layers are equivariant under translation and therefore can exploit translational symmetries in the input data. Consequently, Convolutional Neural Networks (CNN) are a suitable choice for event reconstruction in IceCube.

There are, however, some challenges in applying CNNs to IceCube data. IceCube data consists of pulse series collected in each of the 5160 DOMs. The DOMs are installed on a total of 86 vertical strings that are broadly grouped into three detector parts, as illustrated in Figure 9.1. The data has to be transformed to fit into orthogonal tensors, as required by CNN applications. This is achieved by separating the data of the individual detector components. The strings of the main array are shifted and padded with zeros in order to transform the triangular into an orthogonal grid (see Figure 9.1). The resulting three input arrays are then provided to three independent stacks of convolutional layers as depicted in Figure 9.2. The result of each of the convolutional layers is combined in a flattened layer and passed on to two small dense NNs, which output the quantity of interest and the estimated uncertainty on the prediction, respectively. The convolution kernels utilized for the main array are similarly transformed to the input data, such that the convolution operation effectively convolves a hexagonally shaped kernel. The hexagonal convolutions are more adequate to IceCube's geometry and, in principle, enable the exploitation of the six-fold rotational symmetry, although this is not further pursued here for simplicity (see [31] for further details on the hexagonal convolutions and the network architecture).

At this point, the spatial coordinates of the DOMs are transformed in a way such that they are suitable to a CNN. Each of the elements in the input arrays from Figure 9.1 corresponds to a DOM. What remains is the transformation of the time dimension, given by the pulse series at a given DOM in a feature vector of constant length.

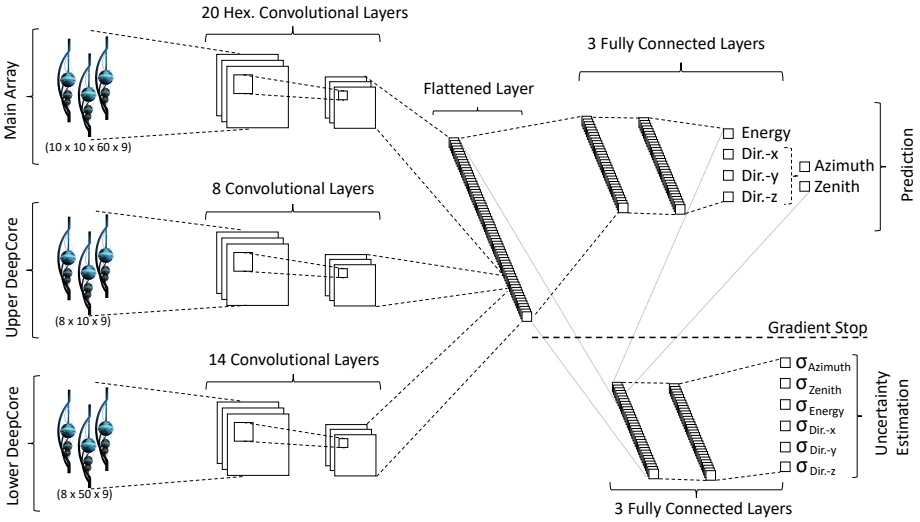


Fig. 9.2: A sketch of the neural network architecture is shown. Data from the three subarrays is sequenced into convolutional layers. The result is flattened, combined, and passed on to two fully-connected subnetworks that perform the reconstruction and uncertainty estimation. The uncertainty-estimating sub-network also obtains the prediction output as an additional input. The figure and caption are taken from [31]. © 2021 IOP Publishing Ltd and Sissa Medialab

The pulse series at a given DOM can have an arbitrary length depending on the amount of deposited energy and the distance of the DOM to the energy depositions in the detector. This leads to varying numbers of pulses across several orders of magnitudes. While variable input lengths are appropriate to recurrent architectures, the CNN applied here requires a constant-sized input tensor. Hence, the pulses are transformed to a feature vector of nine variables that summarize the pulse distribution at a particular DOM. These variables include quantities such as the total charge (correlated to the energy deposition around a DOM) and timing information such as the time of the first pulse. The time of the first pulse is essential for directional reconstruction as the first pulse at a given DOM is least likely to have scattered prior to hitting the photomultiplier. Based on the arrival time, the distance of the emitter to the receiving DOM can be estimated. All nine input features are illustrated in Figure 9.3. Note that these input features are a general selection that is applicable to a wide range of tasks. Specific classification or regression tasks may benefit from dedicated input features that emphasize the information relevant to the task at hand.

In order to obtain per-event uncertainties, the network is trained with a Gaussian likelihood

$$\ell = \ln\left(y_{\text{unc}}\right) + 0.5 \cdot \left(\frac{y_{\text{true}} - y_{\text{pred}}}{y_{\text{unc}}}\right)^2 \tag{9.1}$$

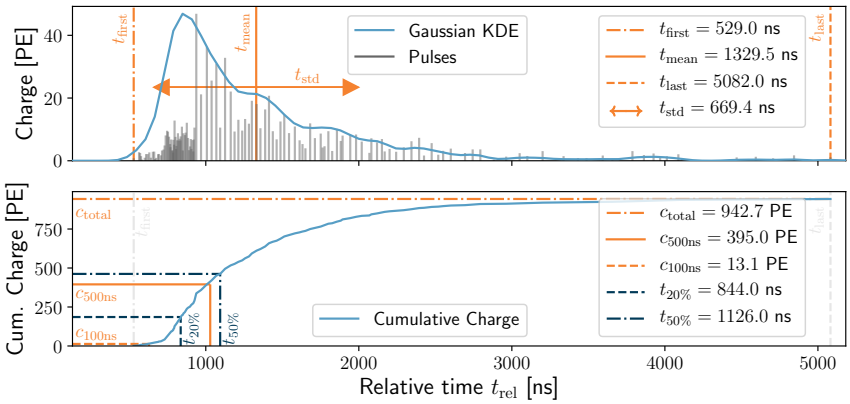


Fig. 9.3: An example pulse series and corresponding input data for a single DOM is shown. The measured pulse series cannot directly be utilized by a CNN due to its varying length. The pulses are therefore reduced to nine input parameters (c_{total}, $c_{500\text{ns}}$, $c_{100\text{ns}}$, t_{first}, t_{last}, $t_{20\%}$, $t_{50\%}$, t_{mean}, t_{std}) that summarize the pulse distribution. The figure and caption are taken from [31]. © 2021 IOP Publishing Ltd and Sissa Medialab

as the loss function (constant terms omitted), where y_{pred} and y_{unc} are the outputs of the two sub-networks for the reconstructed quantity and uncertainty estimation thereof, respectively. As a result, the network not only provides a point estimate, but an estimate of the posterior distribution. This assumes that the residuals $\Delta y = y_{\text{pred}} - y_{\text{true}}$ are distributed as Gaussians with the width y_{unc}.

This assumption can be tested by plotting the pull distribution, which is a histogram of the per-event residuals divided by the estimated uncertainty: $\Delta y / y_{\text{unc}}$. For Gaussian distributed residuals and a correct uncertainty estimate, this should result in a normal distribution. As shown in Figure 9.4, the pull distribution for the three reconstructed quantities is well described by a normal distribution, apart from deviations in the tails of the distribution.

An alternative method to verify the uncertainty estimate of the neural network is by computing the coverage. Based on the assumed Gaussian distribution, the number of events can be computed in which the true value lies within a certain quantile of the predicted value. This can then be compared with the number of events in which the true value should lie within a certain quantile, assuming that the estimated Gaussian distribution is correct. As is illustrated in Figure 9.5, the coverage indicates that the neural network provides an appropriate uncertainty estimation.

In Figure 9.6, the angular resolution of the CNN approach is compared with the default reconstruction method in IceCube, which is based on Maximum Likelihood Estimation (MLE). The CNN is capable of significantly improving the directional reconstruction of cascade events above 10 TeV. This is the interesting energy regime when searching for extraterrestrial neutrino sources because the contribution of astro-

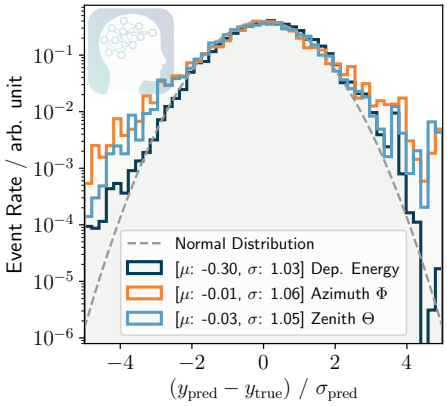


Fig. 9.4: The pull distributions are shown for the reconstructed quantities in comparison with the normal distribution that indicates a good uncertainty estimator. These match well, apart from slight deviations in the tails of the distribution. The figure and caption are taken from [31]. © 2021 IOP Publishing Ltd and Sissa Medialab

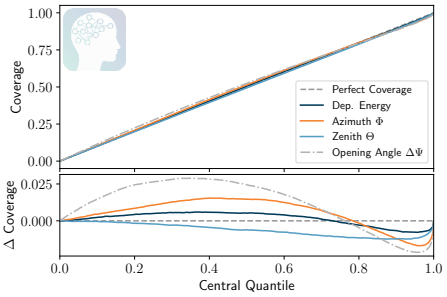


Fig. 9.5: The coverage, i.e. the number of events that fall in a certain quantile, is shown as a function of the computed central quantile based on the uncertainty estimate. Perfect coverage is obtained on the dashed diagonal. The bottom panel shows the difference in coverage of each reconstructed quantity compared with the perfect one-to-one relation. The figure and caption are taken from [31]. © 2021 IOP Publishing Ltd and Sissa Medialab

physical neutrinos becomes more dominant with increasing energy. In addition to the improved resolution, the necessary computation time is reduced by two to three orders of magnitude compared with IceCube's standard reconstruction method. Additional performance comparisons, including energy reconstruction, are provided in [31].

The calculation of the performance gain in Figure 9.6 is based on a Monte Carlo simulation. However, the simulation is an approximation of reality—it includes inherent uncertainties. Some of the sources of uncertainties may be known, such as the ice properties that affect photon propagation in IceCube, but others may remain undetected. For physics analyses, it is therefore important that the effect of these potential systematic uncertainties be well understood. Since these uncertainties are inherent to the physics experiment, analyses are constructed in a way that the impact on the results is minimized. Hence, reconstruction methods, whose outputs are often directly used in analyses, should ideally be robust regarding systematic uncertainties. This is particularly important for machine learning-based methods, which rely on the correctness of the training data.

The investigation of the data/Monte Carlo agreement and the impact of systematic uncertainties play a crucial role in interpreting analysis results. For one, the degree of data/Monte Carlo agreement needs to be quantified. This is typically done by comparing one-dimensional distributions of reconstructed or measured quantities. To also account for correlations in a high-dimensional space, we may train a machine learning classifier to distinguish real data events from simulated ones. Ideally, the classifier should not

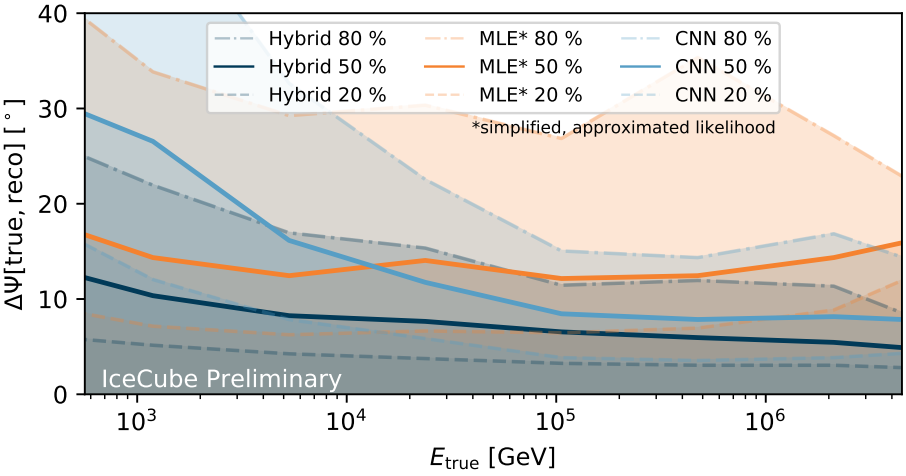


Fig. 9.6: The angular resolution of charged-current ν_e interactions for a typical cascade event selection is compared between IceCube's default reconstruction (MLE) [17], the CNN based method [31], and the hybrid method [208], discussed in Section 9.2.3. The CNN improves the resolution above 10 TeV compared with the standard method in IceCube (MLE). Due to the inclusion of additional domain knowledge, the hybrid method leads to a significantly improved angular resolution over the whole energy range. The plateau towards higher energies is induced by systematic uncertainties. [208].

be capable of separating these. If a separation is possible, this indicates that there is a certain mismatch between simulated and real data. See also Section 5.3.2 for more details.

This test can be performed on the inputs to the CNN as well as the outputs of the CNN. No significant mismatches were found when applying a random forest classifier to the outputs of the CNN [31]. Apart from this test, the impact of known systematic uncertainties on the NN output may be studied. The primary sources of systematic uncertainties in IceCube arise from the optical properties of the glacial ice. The impact on the angular resolution for various realizations of the optical properties is illustrated in Figure 9.7. Monte Carlo simulations are performed in which the optical properties of the ice are modified. The angular resolution is then computed on these systematic datasets and compared with the resolution on the baseline Monte Carlo set. A slight shift in resolution indicates that the reconstruction method is insensitive to the modifications. As shown in Figure 9.7, the CNN (right panel) is more robust towards the studied systematic uncertainties than the standard reconstruction method (left panel). Further tests and modifications are performed in [31] that are omitted here for brevity.

The (CNN) provides a robust reconstruction method that improves upon the current standard in IceCube while significantly reducing the required computational cost.

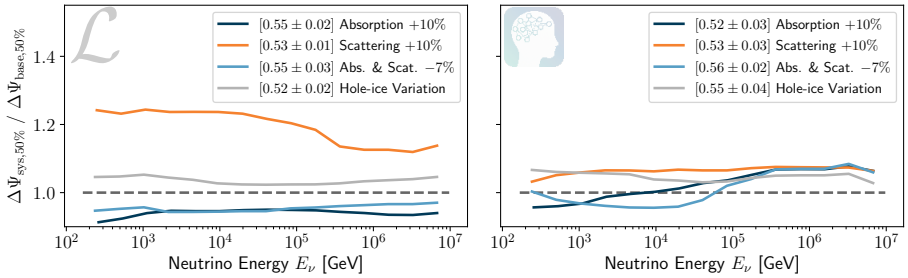


Fig. 9.7: The median angular resolution is compared between the baseline simulation $\Delta\Psi_{\mathrm{base,\,50\%}}$ and the systematic variation $\Delta\Psi_{\mathrm{sys,\,50\%}}$ for the standard reconstruction method (left) and the CNN-based method (right). A ratio close to 1.0 indicates a robust reconstruction method that is insensitive to the varied systematic parameter. For each systematic variation, the data/MC test via the is performed. The resulting Area Under the Curev (AUC) for each variation is shown in brackets in the legend. The figure and caption are taken from [31]. © 2021 IOP Publishing Ltd and Sissa Medialab

Consequently, the CNN approach has been employed in a search for neutrino emission with cascade events [24] as well as for a novel real-time alert stream with high-energy cascade events. These events are likely of astrophysical origin and are therefore of greater interest to the scientific community. Reconstructions via the CNN are performed in real time such that alerts can be sent out to telescopes around the world within seconds to a few minutes of detection at the South Pole (required time is dominated by online filtering and satellite transfer of data). This allows for fast follow-up observations with different messenger particle.

Despite the success of the CNN in IceCube, the approach has inherent limitations. The translational invariance is only approximately given in IceCube data, and the assumption of a regular grid, as required by the CNN, is also violated due to minor irregularities in the detector layout. While the summary features of the pulses at a DOM allow for robust features, they induce information loss and also require a manual selection. Moreover, additional symmetries and domain knowledge exist that cannot be directly utilized by the CNN.

Some of these limitations can be solved using other standard NN architectures. For instance, Recurrent Neural Networks eliminate the necessity of the pulse summary features and enable end-to-end training directly from the extracted pulses. This would avoid potential information loss but might result in the NN being more susceptible to subtle systematic uncertainties in the Monte Carlo simulation. At the same time, graph neural networks can be employed to handle IceCube's irregular geometry more adequately. However, none of the standard deep learning architectures has the ability to explicitly include the wealth of available domain knowledge. As previously mentioned, the detailed knowledge of the data-generating processes in physics experiments like IceCube is by far superior to other domains such as image recognition. Dedicated

solutions are required to harness the full potential. The following section presents a hybrid reconstruction method that tackles these deficiencies.

9.2.3 Combining Deep Learning with Maximum-Likelihood

The previously described CNN approach exploits translational invariance, but lacks the ability to put additional domain knowledge to good use. Current standard reconstruction methods in IceCube are in principle able to fully utilize available information, but they are often limited by computational constraints [17, 29, 30, 122, 188]. These methods infer event properties by optimizing the likelihood of the observed data using maximum likelihood estimation. In this MLE approach, the observed light pattern in the detector is compared with the pattern expected for a certain event hypothesis. One of the major limitations of the MLE approach is the accurate description of the high-dimensional pulse arrival time pdf and expected charge at each DOM for a given event hypothesis. The probability density function (pdf) of the pulse arrival time for IceCube's default reconstruction method is obtained from splines fit to tabulated Monte Carlo simulations [17, 29]. Due to the high dimensionality, simplifications, such as the approximate rotational symmetry in the x-y plane, must be used to reduce the size of the look-up tables. MLE approaches via direct re-simulation exist [122], but they are generally infeasible to run on a large number of events. Neural networks, by contrast, are universal approximators that excel at interpolating high-dimensional data. A hybrid reconstruction method is defined in the following that uses this property by replacing the look-up tables with a generative neural network. The following discussion is adopted literally from [208].

The generative model G

$$G(\vec{\xi}) = \{\vec{\lambda}, \vec{\mathbb{P}}(t)\} \tag{9.2}$$

is trained to map the cascade hypothesis $\vec{\xi} = (x, y, z, \theta, \Phi, E, t)$ to the expected charge $\vec{\lambda}$ and pulse arrival time pdf $\vec{\mathbb{P}}(t)$ at each DOM. The pulse arrival time pdf $\mathbb{P}_d(t)$ at the d-th DOM is parameterized by a mixture model

$$\mathbb{P}_d(t) = \sum_j^K w_j \cdot \mathrm{AG}(t|\mu_{(d,j)}, \sigma_{(d,j)}, r_{(d,j)}) \tag{9.3}$$

of K asymmetric Gaussians [223]:

$$\mathrm{AG}(x|\mu, \sigma, r) = N \cdot \begin{cases} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), & x \leq \mu \\ \exp\left(-\frac{(x-\mu)^2}{2(\sigma r)^2}\right), & \text{otherwise} \end{cases} \tag{9.4}$$

$$N = \frac{2}{\sqrt{2\pi} \cdot \sigma(r+1)} \tag{9.5}$$

where r parameterizes the asymmetry. The mixture model allows for a good description of the pdf while keeping the number of free parameters reasonably low, as illustrated in Figure 9.8.

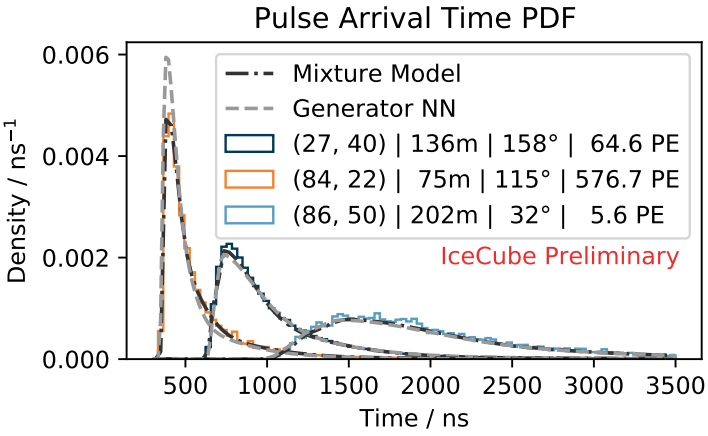


Fig. 9.8: The pulse arrival time pdf at a specific DOM gives insight into the incident angle and the distance of the origin of the arriving photons. The legend indicates the string and DOM number, the distance to the cascade vertex, incident angle, and the average charge collected over 1000 simulations. A mixture model of only three to five asymmetric Gaussians (five shown here) is enough to provide a good description of the pdf. The generator neural network (NN) is able to model the pdf on unseen data. The figure and caption are taken from [208].

The architecture of the generator NN is set up to output the parameters of the mixture model $\{\vec{\mu}_d, \vec{\sigma}_d, \vec{r}_d, \vec{w}_d\}$ and the expected charge λ_d for each DOM. In order to utilize the exact detector geometry and rotational and translational invariance in physics parameter space of the neutrino interaction, relative displacement vectors and angles to each individual DOM are computed and provided as input to the NN, as illustrated in Figure 9.9. The neural network performs a series of convolutional layers with 1×1-kernels. Internally this is implemented in the TensorFlow framework [28] via two-dimensional convolutions. The first layer uses locally connected layers, i.e. it does not apply weight sharing across DOMs. This allows the NN to model the position and direction dependent on the symmetry-breaking optical properties of the ice. Subsequent layers utilize standard convolution operations with weight sharing. Therefore, after the initial locally connected layer, every DOM is treated equally. Additional domain knowledge, such as the linear scaling of the collected charge to cascade energy or the differing quantum efficiency ϵ_d of the DOMs, is directly incorporated into the

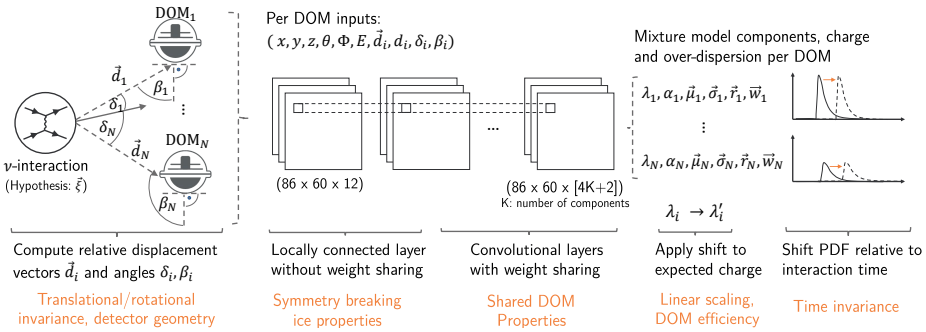


Fig. 9.9: A sketch of the generator neural network architecture is shown. Due to the construction in "forward direction", similar to the Monte Carlo simulation, domain knowledge (examples indicated in orange) can be explicitly included in the architecture. The figure and caption are taken from [208].

architecture, by scaling the expected charge output:

$$\lambda_d^{'} = \lambda_d \cdot \frac{E}{10\,\mathrm{TeV}} \cdot \epsilon_d. \tag{9.6}$$

In general, the architecture may be configured analogously to the MC simulation, while computationally expensive parts are replaced by a neural network approximation. Any domain knowledge that goes into the Monte Carlo simulation may therefore also be utilized in the generator NN. In contrast to standard deep learning architectures, this is possible here because the generator NN is defined in the same "forward" direction as the simulation. Standard deep learning applications, such as the CNN-based method, attempt to directly infer the the quantities of interest from measured data, i.e., in the "inverse" direction compared with the simulation. However, in this "inverse" direction, it is a challenge to exploit information such as the symmetries of the underlying physics interactions because the observed data is already convolved with detector effects, which may break symmetries and lead to degeneracies.

Although the focus is on the reconstruction of cascades, this method can be generalized to arbitrary light sources. In IceCube, any event topology can be built up from a linear superposition of cascades and track segments, such that only two generative models for these elementary source types are required. Systematic uncertainties may also be included in the event hypothesis $\vec{\xi}$ as nuisance parameters. An alternative method to account for systematic uncertainties is to marginalize over them during the training process of the generator NN. This is accomplished by utilizing a training dataset that employs the SnowStorm [23] method, which samples new systematic parameters from a continuous prior distribution for every batch of simulated events.

For the training procedure, an extended unbinned likelihood over the measured pulses is used. For the case without systematic parameters or systematic parameters as

nuisance parameters, the per-event likelihood is defined as

$$\ell_{\text{event}}\left(\vec{x} = \{\vec{c}, \vec{t}\} \mid \vec{\xi}\right) = \prod_{d}^{D} \text{Poisson}\left(\sum_{i}^{N_d} c_{d,i} \mid \lambda_d\left(\vec{\xi}\right)\right) \cdot \prod_{i}^{N_d} P_d(t_{d,i} \mid \vec{\xi})^{c_{d,i}}, \tag{9.7}$$

where $D = 5160$ is the total number of DOMs, N_d is the number of pulses at the d-th DOM, and $c_{d,i}$ and $t_{d,i}$ are the charge and time of the i-th pulse at the d-th DOM. When marginalizing over systematics, one must account for the over-dispersion in measured charge. In this case, the measured charge at a DOM no longer follows a Poisson distribution and a Gamma-Poisson mixture distribution may be used instead. The Gamma-Poisson mixture distribution is a real-valued pendant to the negative binomial distribution. It is capable of modeling the over-dispersion. The parameterization from [336] is chosen:

$$\text{GammaPoisson}\,(z|\lambda, \alpha) = \frac{\Gamma(z + \frac{1}{\alpha})}{\Gamma(z+1)\Gamma(\frac{1}{\alpha})} \left(\frac{1}{1+\alpha\lambda}\right)^{\frac{1}{\alpha}} \left(\frac{\alpha\lambda}{1+\alpha\lambda}\right)^{z} \tag{9.8}$$

which introduces the shape parameter α that leads to over-dispersion when $\alpha > 0$. As a result, the generator NN must also output the shape parameter $\vec{\alpha}$ for each DOM and the likelihood is modified to:

$$\ell_{\text{event}}\left(\vec{x} = \{\vec{c}, \vec{t}\} \mid \vec{\xi}\right) = \prod_{d}^{D} \text{GammaPoisson}\left(\sum_{i}^{N_d} c_{d,i} \mid \lambda_d(\vec{\xi}), \alpha_d(\vec{\xi})\right) \cdot \prod_{i}^{N_d} P_d(t_{d,i} \mid \vec{\xi})^{c_{d,i}}. \tag{9.9}$$

9.2.4 Model Performance and Applications

An additional benefit of the generator NN over the standard deep learning architectures lies in the improved *interpretability* of the model. Individual components of the model may be investigated and cross-checked. Figure 9.8 demonstrates that the model is capable of correctly modeling the arrival time pdf on unseen data. It is also possible to investigate how the pdfs change for individual DOMs when modifying the event hypothesis. In Figure 9.10, an example is shown in which the cascade zenith values and the z coordinate of the interaction vertex are shifted. As expected, the generator NN models a smooth transition from one hypothesis to another.

The trained generative model may be used in a maximum-likelihood setting to reconstruct events via the likelihoods provided in Equations 9.7 and 9.9. The hybrid method is able to improve upon the CNN and the standard reconstruction method over the whole energy range, leading to a significant boost in angular resolution (see Figure 9.6). This is possible because the hybrid method is not subject to simplifications and because it can benefit from available domain knowledge. Other applications of the generative model include likelihood scans and Markov-Chain Monte Carlos, as well as event simulation.

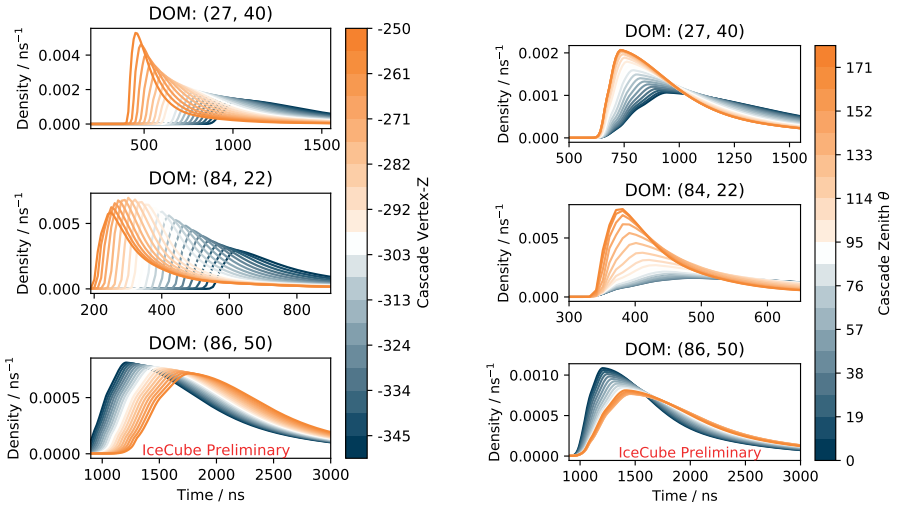


Fig. 9.10: The pulse arrival time pdf, approximated by the generative model, is shown for three different DOMs of the same event. The left panel shows the effect of modifying the z coordinate of the cascade interaction vertex, while the right panel illustrates the change due to the varying zenith angle. The figure and caption are taken from [208].

9.3 Flavor Tagging with Deep Learning LHC

Bernhard Spaan
Vukan Jevtic

Abstract: For time-dependent decay measurements of violations of $\mathcal{CP}$-symmetry in decays of neutral B mesons, it is indispensable to know whether the neutral B meson has been produced as a particle or antiparticle. The method used for this purpose is called *flavor tagging*. It allows us to infer the production state from the analysis of the other particles in the event not belonging to the neutral B meson being analyzed. At the LHC, numerous particles are produced, where most of them carry little to no information on the production state of the neutral B meson, making it extremely difficult to derive a decision on its production state. Therefore, the LHCb experiment uses a variety of algorithms, which are optimized for specific decay signatures, thereby determining the probability for the particle types at the time of production. A recent development resulted in a new type of flavor-tagging algorithm, known as *inclusive flavor tagging*. Contrary to the standard algorithms, where not only a few selected tracks are used, the inclusive flavor tagging algorithms analyze all tracks measured in the collision. This technique is also an example of the feasibility of a *Full Event Interpretation*, i.e., the

overall view and interpretation of collisions at proton-proton colliders using methods of machine learning and making use of recurrent neural networks and gated recurrent units.

9.3.1 Neutral *B* Meson Oscillations and *CP*-Violation

One of the main research focuses of the LHCb experiment is the study of the physics of neutral B_d^0 (or B^0 for short) and B_s^0 mesons. Mesons are particles that consist of two *valence quarks*. B^0 mesons contain one b quark (hence their name) and one d (down) quark and come in two *flavors* $B = (\bar{b}, d)$ and $\overline{B} = (b, \bar{d})$, where the line above the particle denotes the antiparticle state. B_s^0 mesons contain a b quark and strange quark, s, instead of d. Most often, the lifetimes of mesons carrying heavy quarks like bottom quarks are extremely short, causing them to decay at the proton-proton beam collision point right after their production. B mesons, by contrast, have a lifetime of $1.52\,\mathrm{ps}$, allowing them to fly a distance of a few millimeters up to a centimeter before they decay and thereby create a unique decay signature of a displaced decay vertex. During their comparably *long* lives, B mesons (and neutral mesons in general) oscillate between the two states $B^0 \leftrightarrow \overline{B^0}$ and $B_s^0 \leftrightarrow \overline{B_s^0}$ with a periodic, time-dependent probability. Consequently, their production flavor does not always equal their decay flavor, in which case one says that the meson has *mixed*. This time-dependent probability is given by

$$P^{\mathrm{mix}}(t) = \frac{1}{2}\left(1 - \frac{\cos(\Delta m t)}{\cosh(\Delta\Gamma t)}\right) \tag{9.10}$$

in leading order of precision. Here, Δm denotes the B oscillation frequency and $\Delta\Gamma$ is the decay width difference of two quantum superposition states of neutral B mesons B_{L} and B_{H}, whose physics will not be explored further in this text, as $\Delta\Gamma$ is of almost negligible size. For B^0 mesons, $\Delta m = 0.5065\mathrm{ps}^{-1}$ and for B_s^0 mesons, $\Delta m = 17.757\mathrm{ps}^{-1}$, hence B_s mesons oscillate around 35 times faster than B mesons. Despite the comparably slow oscillation of the B^0, it is possible at LHCb to measure up to one full oscillation with the available data.

Measuring this time-dependent oscillation opens the window to a wide range of fascinating physics phenomena, like the violation of the charge-parity symmetry. In an experimental setting this flavor oscillation is measured in terms of a time dependent asymmetry $\mathcal{A}$ as follows

$$\mathcal{A} = \frac{\Gamma(\overline{B}(t) \to \bar{f}) - \Gamma(B(t) \to f)}{\Gamma(\overline{B}(t) \to \bar{f}) + \Gamma(B(t) \to f)}. \tag{9.11}$$

Here, $\Gamma(B(t) \to f)$ denotes the rate of the decay of B mesons that are *produced* with the flavor B^0 into some unspecified final state f. Analogously, $\Gamma(\overline{B}(t) \to \bar{f})$ denotes the time dependent decay rate of mesons produced as $\overline{B}$ into the anti-final state $\bar{f}$.

At electron-electron colliders, B mesons are produced in a mechanism that is different from the one used at hadron colliders: there, $Y(4S) = (b, \bar{b})$ particles are

produced at high rates, which predominantly decay into a B and $\overline{B}$ pair. Since the daughter particle properties are fully inherited from its ancestor $Y(4S)$ particle, both B mesons are produced in a state of quantum entanglement, where the measurement of the production flavor of one of the mesons simultaneously determines the production flavor of its undecayed partner. This is why in the context of these experiments, flavor measurements are carried out in terms of B decay time differences, meaning that the first meson to decay *tag*s its partner. At hadron colliders, the b quark production mechanism does not result in the production of entangled B mesons. This hadronic production environment will be the focus of the remainder of this section.

By formulating expression (9.11) in terms of decays of mesons with some production flavor, we gain access to decays where f and $\bar{f}$ are identical. These channels are of special interest because they allow the measurement of CP-violation in the interference of the decay and oscillation processes. This expression can then be rewritten as

$$\mathcal{A} = \frac{S\sin(\Delta m t) - C\cos(\Delta m t)}{\cosh\left(\frac{1}{2}\Delta\Gamma t\right) + \mathcal{A}_{\Delta\Gamma}\sinh\left(\frac{1}{2}\Delta\Gamma t\right)}. \tag{9.12}$$

Without going into more detail, the parameters S, C, and $\mathcal{A}_{\Delta\Gamma}$, with additional input of other natural constants, can parametrize to which degree CP-symmetry is violated in the universe. CP-symmetry is the combined symmetry of charge C and parity P. If the charge were indeed a fundamental symmetry, a mirror universe in terms of charge would be in every possible way indistinguishable from our own universe; analogously, a parity-inverted (i.e., mirrored) universe would display the same phenomena as an unmirrored universe. Although these symmetries hold in classical physics, both symmetries have been shown to be violated in quantum physics. Parity violation in weak interactions (interactions mediated by the weak force) was discovered by C. Wu in 1956, and charge violation is known to be violated in weak decays. Although the combined symmetry $C \cdot P$ is a much more robust symmetry and was hypothesized to be conserved in all possible interactions, J. Cronin, V. Fitch, R. Turlay, and J. Christenson discovered CP-violation in 1964 in neutral kaon decays [123].

9.3.2 Flavor Tagging Technique

As explained in the previous section, measuring the production flavor is a crucial task to get access to CP-violation and meson oscillation parameters. Flavor tagging is the technique used at particle detectors (here, at proton-proton colliders in particular) to determine the production flavor of neutral mesons. Since the decay flavor is not always equal to the production flavor due to mixing, it cannot be deduced from the particles that the B meson decays into at the end of its individual lifetime. Instead, the production flavor is derived from information created at the time of the production itself. After the $b\bar{b}$ production process, the b quarks pair up with lighter quarks that are created from the vacuum to form a variety of B hadron types because no quark can

exist in an unbound state, i.e., in an isolated form. In Figure 9.11 such a hadronization process is illustrated in a Feynman diagram, where time passes from left to right. The b and $\bar{b}$ quarks are created as a pair in a gluon-gluon fusion following the proton-proton collision. The upper quark then interacts with the gluon field to produce additional $d\bar{d}$ and $u\bar{u}$ pairs, which form a neutral $\overline{B^0}$ meson and a charged pion, marked in red. The neutral $\overline{B^0}$ is the signal particle of interest, and all processes associated with it are

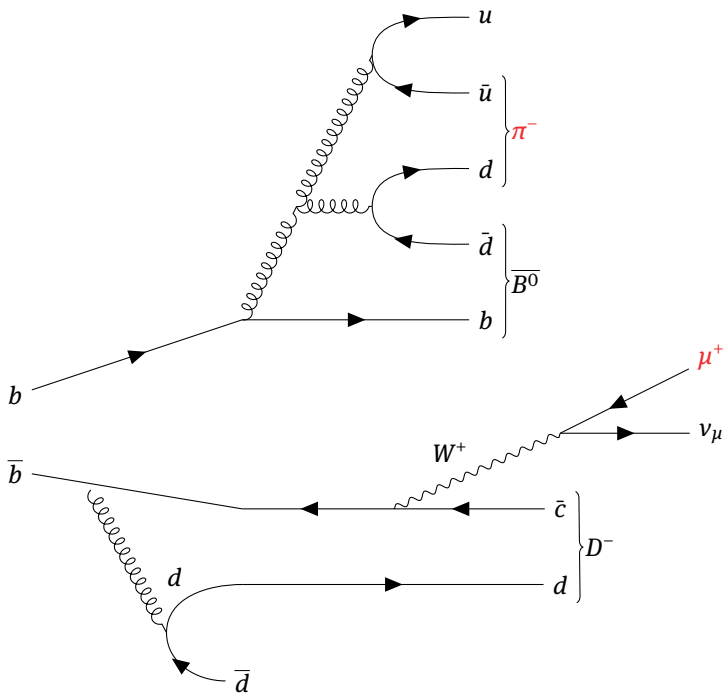


Fig. 9.11: Example of a $b\bar{b}$ hadronization. In this (incomplete) Feynman diagram, time passes from left to right. Feynman diagrams do not allow any conclusions about the spatial relations or the temporal order of any two subprocesses if they happen acausally from each other. The upper (same side) b quark hadronizes into the signal $\overline{B^0}$ meson and a (tagging) pion π^-, while the lower (opposite side) quark hadronizes differently to produce, among other particles, a charged tagging muon μ^+.

called Same-Side (SS) processes, and processes coming from the $\bar{b}$ hadronization are referred to as Opposite-Side (OS) processes. On the opposite side, the $\bar{b}$ interacts with the gluon field to form a $d\bar{d}$ pair. The created $d\bar{b}$ pair then forms an intermediate B^0 meson that undergoes a weak decay (mediated by the W boson) into a charmed meson D^- and the W decays into a muon and a neutrino.

It is now essential to remember that electromagnetic charge is conserved at every decay vertex, and therefore the charge of the SS pion is fully correlated to the flavor $\overline{B^0}$. It

is, therefore, possible to infer the production flavor by identifying these accompanying messenger particles and measuring their charge, which is precisely the idea behind *flavor tagging*.

At LHCb, processes like these happen in a hadronic environment. They contain many more tracks from the primary interaction point (PV), which are produced by the proton dismemberment and subsequent hadronizations and decays. Therefore, many tracks in a collision event may be charged and well-measured but are still unrelated to the signal meson itself. Even when the right set of tracks is found, it is impossible to unambiguously relate it to some vertex in a process like the one shown in Figure 9.11 because these interactions happen for the most part instantaneously and at a single point. Physical processes like mixing can play a role as well because if the intermediate B^0 on the opposite side would mix before it decayed as an example, it would result in the final state $D^+\mu^-\overline{\nu_\mu}$ instead of $D^-\mu^+\nu_\mu$ and suggest a wrong flavor for the signal meson. Finally, not every track can be measured and saved, and when it is measured, the uncertainty of the particle momentum, the particle type, and the measured charge is determined, resulting in some tracks being reconstructed better than others.

Taking these challenges into account, it is clear that flavor tagging cannot produce accurate measurements of the production flavor. At the same time, it is the only way to do so. Flavor misidentification of a considerable magnitude is present in most events, and the precise determination of the misidentification probability (the *mistag* probability) is instead a further optimization goal for this technique. If the misidentification probability for each flavor estimate is precisely known, it is possible to flavor tag a whole dataset correctly *on average* without distorting the underlying physics. Parameters that characterize the quality of a given tagging algorithm are explained in the following.

9.3.3 Flavor Tagging Formalism

A flavor tagging algorithm receives the reconstructed particle tracks and returns two numbers: the tagging decision d and the mistag estimate $\eta \in [0, 0.5]$. The mistag is the probability that the flavor has not been correctly determined. For events with $\eta > 0.5$, these outputs are always transformed into $\eta \mapsto 1 - \eta$ and $d \mapsto -d$. The tagging efficiency ϵ_{tag} is given by

$$\epsilon_{\text{tag}} = \frac{N_r + N_w}{Nr + N_w + N_u} = \frac{N_{\text{tagged}}}{N_{\text{total}}} \tag{9.13}$$

where N_r, N_w, N_u are the amounts of correctly tagged, incorrectly tagged and untagged events, respectively, and quantifies the fraction of tagged events. The last metric needed to quantify the quality of a flavor tagging algorithm is the flavor tagging *dilution*, $\mathcal{D} = (1 - 2\eta)$, which takes the mistag probabilities into account by assigning each event an effective weight, which is zero for $\eta = 0.5$ and 1 when $\eta = 0$. The *tagging power*, $\epsilon_{\text{tag,eff}}$, of the dataset measures the effective sample size for *CP*-violation measurementss and

is given by

$$\epsilon_{\text{tag,eff}} = \epsilon_{\text{tag}}\mathcal{D}^2 = \frac{\epsilon_{\text{tag}}}{N_{\text{tagged}}} \sum_{i,\text{tagged}} (1 - 2\eta_i)^2. \tag{9.14}$$

This performance number is used to estimate the uncertainties of measured time-dependent *CP*-asymmetries $\sigma(\mathcal{A})$ via

$$\sigma(\mathcal{A}) \propto \frac{1}{\sqrt{\epsilon_{\text{tag,eff}} N_{\text{total}}}}. \tag{9.15}$$

From this equation, it follows that the tagging power gives an estimate of the effective sample size. Imperfect flavor tagging has thus two effects, which need to be precisely known in order to measure the correct physics parameters: it introduces a *mean dilution* $\langle D \rangle$ which measures by how much the observed *CP*-asymmetry amplitude is flattened, and it introduces *uncertainties on the measured natural constants S* (which is mostly the oscillation amplitude), *C*, and $\mathcal{A}_{\Delta\Gamma}$. Standard flavor tagging algorithms have tagging powers of around 1 %, and by a combination of multiple different algorithms tagging powers of up to 4 to 6 % can be achieved, depending on the specific data set. These low effective sample sizes show how challenging flavor tagging really is.

In practice, η is obtained from the output of a multivariate classifier and does not really have the properties of an actual uncertainty in the statistical sense. Therefore, the aforementioned performance estimates are commonly not computed using the raw η distribution. Instead η is calibrated first so that the actual, true, misidentification probability $\eta^{\text{true}} = \omega$ matches the mean mistag in bins of η. To that end, a calibration polynomial $\omega(\eta)$ is optimized on a flavor-specific decay channel like $B^+ \to J/\psi K^+$, where the B flavor (B^+ or B^-) can, in fact, be determined unambiguously through its daughter particles. This is possible because charged B mesons do not oscillate, and the kaon charge can be measured with high precision.

9.3.4 Flavor Tagging Algorithms

All flavor tagging algorithms (*taggers*) at LHCb are multivariate classifiers, which are either boosted decision trees or neural networks. These networks are trained with all, or a selection of, reconstructed tracks, and the classification label is the production flavor. The taggers are trained on flavor-specific samples $B^+ \to J/\psi K^+$ or $B^+ \to J/\psi K^\star(\to K^+\pi^-)$ where the production flavor can be estimated with high precision and where *CP*-violation is not possible or is immeasurably small. In the case of the B^0 training channel, oscillation plays a role so that the true training label needs to be estimated using Equation 9.10 which is evidently imperfect due to the probabilistic nature of quantum mechanics, but it is still demonstrably possible. Due to the physics of these training channels, it is possible to train on real data as well as on Monte Carlo-generated samples. The following taggers are in use:

OSμ The OS muon tagger identifies a tagging muon on the opposite side.

OSe The OS electron tagger identifies a tagging electron on the opposite side.
OSK The OS kaon tagger identifies a tagging kaon on the opposite side.
OScharm The OS charm tagger targets decays of charmed mesons on the opposite side.
OSVtxCh The OS vertex charge tagger produces a tag using all the tracks and their charges on the opposite side.
SSπ The SS pion tagger identifies pions that hadronize alongside the signal B meson.
SSp The SS proton tagger identifies protons that hadronize alongside the signal B meson.
SSK The SS proton tagger identifies kaons that hadronize alongside the signal B_s^0 meson.

These algorithms do not cover nearly all physical processes that can be expected in $b\bar{b}$ productions, but cover the most likely and most recognizable ones. To train one of these taggers, one track needs to be selected as the tagging track for every proton-proton collision. Depending on the algorithm, quite often, a tagger cannot contribute to a tag decision because particles do not exist in the data set. The tagging track selection is usually made using simple, fixed selection cuts that exploit the conservation of momentum (SS particles usually fly in the same direction as the signal) and the decay signature of the displaced vertex to match them to the signal meson. The production flavor is then directly determined from the track charge, and the mistag is determined by a multivariate classifier.

The consequence of this approach is that tagging algorithms only use the one track for tagging that they are specialized for. If a valuable track is lost in the track selection or if really more than one track is needed to produce a solid tag decision, they fail to produce the correct result. If two taggers produce contradicting or matching results, the multivariate classifiers cannot exploit these correlations for tagging. This is why in the past years, the development of a universal kind of tagger that analyses the whole event has become the main research focus of the flavor tagging group at LHCb.

9.3.5 Inclusive Flavor Tagging

Inclusive flavor tagging is a technique where not just one but potentially all reconstructed tracks in a given proton-proton collision are analyzed to produce a tagging decision for a signal event. This creates a challenge for the choice of a multivariate classification model because, in each collision, there are different numbers of reconstructed tracks: In some collisions, as few as 30 tracks are reconstructed; in others, up to ca. 200 tracks can be found. Consequently, a classical approach of a neural network with a fixed input vector or boosted decision trees cannot be used as long as this variability of tracks is going to be tolerated. Instead, the choice was made to use an architecture where the classifier does not operate on vectors of fixed dimension but instead on sequences of

said vectors. Recurrent Neural Networks (RNN) have the property to not only feed an input vector through a set number of layers but to re-route the network output back to their input and pair it up with the next input instance (these would be vectors of particle track features in our case) until all inputs are processed. In each iteration, these networks update their "hidden" information vector, which can be interpreted in the end as the network output.

A more sophisticated type of recurrent neural network that was chosen for this task is the so-called *Gated Recurrent Unit* (GRU), which contains internal mechanisms to selectively learn to forget information that is irrelevant in each input instance and another mechanism to combine the remaining, relevant information with the next input instance to update its hidden state. The benefit of this network type is that the decay of initial information (i.e., the first input tracks) is much smaller than in a pure RNN and that the network learns to identify patterns that contain information on physical processes.

The set of 18 per-track features includes most notably:

- The reconstructed particle charge as reconstructed from the track curvature in LHCb's magnetic field.
- The track transverse momentum $p_T = \sqrt{p_x^2 + p_y^2}$, where the z axis is pointing along the proton beam. This feature is well understood to be correlated with reconstruction quality.
- Particle identification probabilities quantifying the likelihoods that a particle is either an electron, muon, kaon, proton, pion, or whether the track has been reconstructed in the absence of an actual particle.
- Angular differences in the LHCb detectors ϕ-η plane concerning the reconstructed signal B meson. Due to conservation of momentum, tracks on the SS tend to fly into the same general direction as the signal B meson. By contrast, OS tracks fly in the opposite direction in the $b\bar{b}$ collision reference frame.
- Impact parameters with respect to a) the primary interaction point of a proton-proton collision and b) the reconstructed signal B meson. The impact parameter is the closest distance of a track to a given point. This helps the network recognize tracks associated with the B meson in question and to the primary interaction point associated with the signal B production, as there can be multiple proton-proton collisions in a single event that are by nature uncorrelated to each other.

The final choice of the network architecture is shown in Figure 9.12. Before the tracks

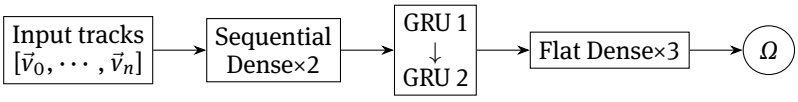


Fig. 9.12: Neural network architecture of the inclusive tagger. A set of input tracks is fed into the network and after a set of sequential and nonsequential layers, a scalar output $\Omega \in (0, 1)$ is produced.

enter the network, the track features are transformed so that all track features are standard normal distributed and then each track is further transformed with a flat neural network of depth two. The transformed sequence of tracks is then fed into a sequential GRU, which means that the hidden state is simultaneously fed into a second GRU. When the network is done with iterating over tracks, the last hidden state of the second GRU is then transformed with another 3 layers of a classical flat neural network. The last network layer produces a scalar $\Omega \in (0, 1)$ output with the help of a sigmoid activation function. This output is then directly translated into a mistag η and a tagging decision d via

$$\Omega \mapsto \begin{cases} (1 - \Omega, 1) & \text{if } \Omega > \frac{1}{2} \\ (\Omega, -1) & \text{else} \end{cases} = (\eta, d). \tag{9.16}$$

This approach differs notably from the classical approach since the tag decision is determined by the network and is not inferred purely from the charge of one carefully selected particle and its assumed production process.

The network is trained on data sets of flavor-specific decays. For example $B^{\pm} \to J/\psi K^{\pm}$, which has been prepared so that the particles, into which the B^+ meson decays, are removed because these tracks would instantly give away the initial flavor. This is undesirable since a flavor tagging algorithm should apply to B decays in general and interpret the hadronization tracks rather than applying to only one certain decay. In this example, two muon tracks coming from the J/ψ and the K^+ track are removed from each event in the training data. The training label is the production flavor which is known in this channel through the $K^{\pm}$ charge, and the network is then trained with around 1.5 million simulated events with a mean of 40 tracks in each one of them. For the validation sample, 30 % of the total statistics are allocated, and for the performance evaluation, another 10 % is reserved and does not enter the training in any way.

During the development, the choice of the activation function for the dense layers has proven to be important. This is because the widely known Rectified Linear Unit (ReLU), defined as

$$\mathrm{ReLU}(x) = \begin{cases} x & x \geq 0 \\ 0 & \text{else} \end{cases} \tag{9.17}$$

produces output distributions that exhibit a variety of features in this classification problem. These include unexpected peaks and shapes, which are of concern since such extreme differences between matter and antimatter, i.e., measurements of positive and negative charges, are not expected. During development, this triggered many discussions and investigations about possible network defects or imperfect simulation samples. This problem was eventually solved using Exponential Linear Units (ELUs), which are defined as

$$\mathrm{ELU}(x) = \begin{cases} x & x \geq 0 \\ alpha(e^x - 1) & \text{else} \end{cases} \tag{9.18}$$

ELUs have a smooth derivative, but otherwise have very similar advantages to ReLU activations. Another major point of investigation has focused on understanding shifts in the output distribution to one of the training labels. Again, this is not expected as the network is trained with very similar amounts of training labels and cannot be explained through physics. In the end, it has shown that this network architecture is so sensitive to training label asymmetries that differences in the permille level can cause tagging asymmetries of up to 15 %. Needless to say, a tagger that tags an event with a bias towards one of the flavors would be rejected. By using the *exact* same amounts of training labels for each flavor and ensuring that the training labels are perfectly balanced in each training batch, the mean observed asymmetries was reduced to a level below one percent.

Figure 9.13 shows the output distribution of the final trained network. The total output is shown in black, and the colored histograms show the output components that are made of B^- and B^+ tagged mesons, respectively. The flavor separation is clearly visible, and the output distribution is symmetric, as expected from the first principles. The performance of this network was estimated to be around $\epsilon_{\text{tag,eff}} = 10.5\,\%$ which is

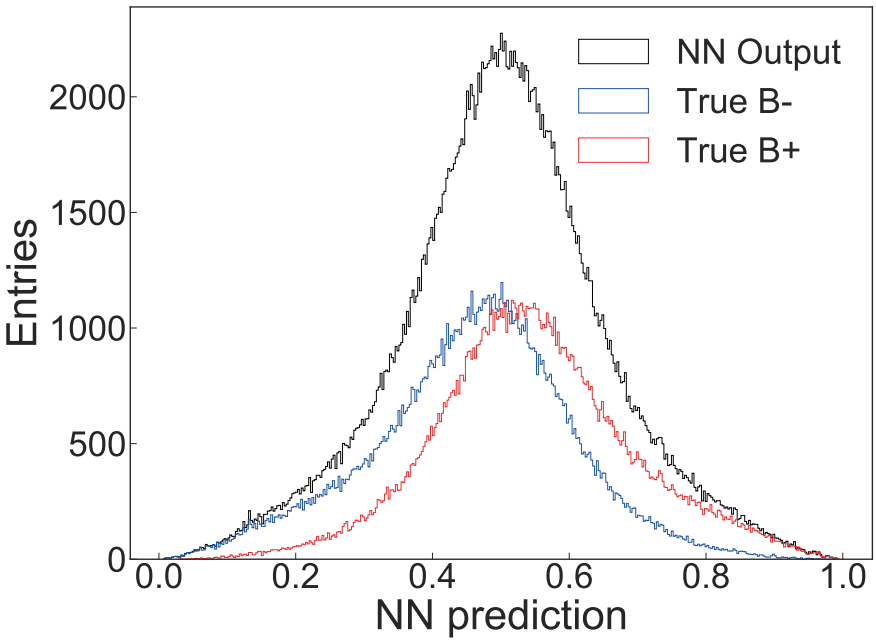


Fig. 9.13: Neural network output of a network trained on a $B^+ \to J/\psi K^+$ simulation which is applied to the test sample. The black histogram shows the total output distribution and the blue (red) histograms show the data belonging to each of the flavors B^- and B^+, respectively.

an excellent result since only $\epsilon_{\text{tag,eff}} = 6\,\%$ can be achieved with the classical method. Given that the LHC has been taking data for four years during Run 2, this relative improvement of tagging power has a comparable effect to taking data for roughly another two years. It has since been shown that this specific network performs similarly well on real data taken by the LHCb detector. Further research into applying this method to crucial channels such as $B^0 \to J/\psi K^*(892)$ and $B_s^0 \to D_s^+ \pi^-$ is ongoing. It is expected to reach similar performance improvements given that the production of associated tracks, i.e., hadronization tracks, is very similar in these channels.

9.4 A Deep Learning Analysis Pipeline for Gamma Ray Astronomy

Lukas Pfahler
Sebastian Buschjäger

Abstract: As introduced in Chapter 1, machine learning is one of the cornerstones of data analysis for modern gamma-ray astronomy. We show how to solve three important machine learning tasks in gamma-ray astronomy, namely *gamma-hadron separation*, *origin estimation*, and *energy estimation*, in a joint neural network architecture. Current solutions heavily rely on expert-designed, hand-crafted features. These features are computed in a long data-processing pipeline and are used to train separate random forest for different tasks.

By contrast, we build a single neural model that works directly on photon count data and uses convolution layers to learn suitable feature representations. The network is constructed with three prediction heads, one for each of the learning tasks, which share all convolution layers. Hence, we learn representations that are not only useful for a single task but for all three tasks. We design loss functions for each task and, more importantly, propose a novel way of combining the different loss functions to account for their different scales and behaviors.

Our experiments for the FACT telescope show that this approach outperforms hand-crafted features and random forests by a large margin on simulation data. Furthermore, we show that our approach does not only work well on simulated data but also on real cosmic events originating in the Crab Nebula, a supernova remnant.

This section is based on a prior conference paper [109]. Its long-form version was originally published in [304].

9.4.1 Introduction

To study our universe, modern astronomy observes high energy particles emitted from celestial objects in order to categorize the sources together with their key characteristics. For example, different types of supernovae remnants can be found by observing high-energy beams [115]. Large international collaborations deploy a wide variety of detector hardware including telescopes [53, 226, 303] to observe different ranges of electromagnetic beams. In high-energy gamma-ray astronomy, we have to solve three machine learning problems. The first is the distinction between gamma rays, which indicate a celestial object, and background noise mostly produced by cosmic rays from hadrons that do not allow for conclusion on a particular source. This is known as the the *gamma-hadron separation* problem. Secondly, we want to estimate the energy of the recorded particle. And third, we need to reconstruct its origin, which is particularly challenging in the mono-telescope setting of the FACT experiment.

As we have already seen in previous chapters, machine learning has been established as an effective tool for analyzing modern high-energy particle experiments and solving the above-mentioned prediction problems [88, 294]. Current approaches use a basic hardware trigger, i.e., they begin the recording of an event when sufficient energy hits the detector. Then, a complex analysis pipeline calibrates the data and extracts pre-defined features, which are used for classifying the stored event as hadron or gamma rays.

We asked ourselves whether we could replace the long pipeline with a deep learning process. Deep learning is supposed to decrease the burden of feature engineering, as the network already learns a suitable feature representation. Is this true for gamma-ray astronomy? Works in other telescope projects suggest the benefits of using deep learning [215, 291, 348]. However, deep learning is widely known to be resource-hungry, requiring not only vast amounts of training data but also GPU hours for training as well as applying models. Future monitoring facilities will be installed around the world for a round-about view of the sky[330]. They need to detect interesting gamma events *fast* so that they can notify the other telescopes, which then turn in order to record the event from their angle. In modern multi-messenger astrophysics, even different types of detectors inform each other so that the same event can be verified by different measurements. So we have to consider not only the quality of our model predictions but also take into account the *resource constraints* of executing the models.

9.4.2 Event-Tagging Pipeline

In this section, we describe the application of a multitask deep network in the data analysis pipeline of the First G-APD Cherenkov Telescope (FACT). In the training data, each event is represented as an image where each of 1440 pixels contains a photon count represented as a positive integer.

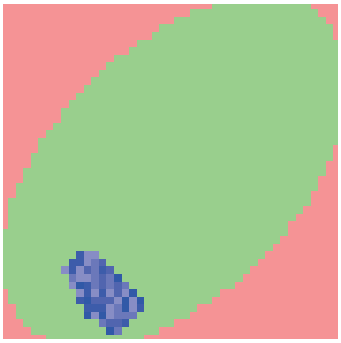


Fig. 9.14: Hexagonal data transformed to a quadratic grid. The transformation is bijective in the green (and blue) area and does not change the raw values. However, it only approximates the neighborhood relationships of the original hexagonal image. Red areas are padded with zeros.

The photon count representation is based on the single-photon extraction for FACT's SiPM sensors proposed by Mueller et al.[282] and is implemented in the `photon-stream` software library [287].

In the camera, these pixels are arranged in a hexagonal grid. We chose to embed these pixels in a regular, quadratic grid of size 45×45 where the additional pixels are padded zero values, as depicted in Figure 9.14. This allows us to use regular convolutional neural networks rather than networks specialized for hex-grids. Given the image data, the three learning tasks already introduced in Section 8.4 should be tackled, generating annotated events for further downstream analysis. These tasks, gamma-hadron separation, energy estimation, and origin estimation, will be solved by a single deep neural network without the need for manual feature engineering.

9.4.3 Multi-Task Deep Neural Networks

We design a model that solves all three learning tasks simultaneously, as also proposed by Jacquemont et al. [216]. It utilizes a shared encoder architecture that extracts features from the raw photon count images, and these features are passed into three separate prediction heads. The whole network is trained jointly on all tasks. We begin by discussing each task individually and present the corresponding prediction heads as well as loss functions. Then we present our approach for combining the losses to train with three prediction heads.

Gamma-Hadron Separation is a binary classification task. Early approaches based on manually-designed decision rules were already quite successful. Also, simple machine learning rules based on histogram features already achieve good classification performance. The random forest approaches outlined in Section 8.4.2 achieve excel-

lent classification results but are outperformed on simulation data by deep learning approaches like the one proposed by Buschjaeger et al. [109]. For application on real telescope recordings, this performance advantage on simulation data does not increase source detection performance, hinting at overfitting to peculiarities of the simulation [109], an issue that may be addressed through domain adaptation techniques [147].

For the classification task we rely on the standard cross-entropy loss function. The classification head outputs a probability $\hat{y}$ that the recorded event is a gamma event. We minimize the negative log-likelihood of the ground-truth label

$$\ell(\hat{y}, y) = -y \log \hat{y} + (1 - y) \log(1 - \hat{y}). \tag{9.19}$$

To ensure that the outputs are indeed valid probabilities, we apply the sigmoid activation $\sigma(a) = 1/(1 + \exp(-x))$ after the final affine layer.

Energy Estimation is a univariate regression task. We use an affine function to output the prediction given the last hidden representation. Instead of using an absolute error, we prefer the relative error that normalizes the difference between target and prediction by the true energy level. We use the loss function

$$\ell(\hat{y}, y) = \frac{|y - \hat{y}|}{y}. \tag{9.20}$$

and apply the exponential function as an activation function after the last layer to account for the power-law distribution of energies. This presents an alternative to minimizing the squared difference between the log-target and the prediction, another approach to dealing with the power-law proposed in Section 8.4.3.

Origin Estimation is a two-dimensional regression task. In Section 8.4.4, we have seen an approach based on the disp-method that separates the task into a 1d regression task and a classification task by assuming the event originates on the main axis of the reconstructed shower axis. However, since we no longer estimate Hillas parameters but want to rely on a fully-learned analysis pipeline parameterized by a deep network, we instead tackle the problem as a 2d-regression task and try to predict the position of the source on the camera pane. This approach uses less domain knowledge, which might be a disadvantage, but we might eliminate errors where the reconstruction of the shower axis is flawed.

Our prediction head uses an affine map to compute the coordinates given the last hidden representation. For downstream tasks like source detection, it is important that the angle θ between the predicted source position and the actual position is as small as possible. We compute this angle θ accounting for the geometry of the telescope and use it as the loss function that is minimized during training.

Multi-Head Architecture Our multitask network is structured as in Figure 9.15: We utilize a shared base encoder, i.e., a convolutional neural network that takes the input

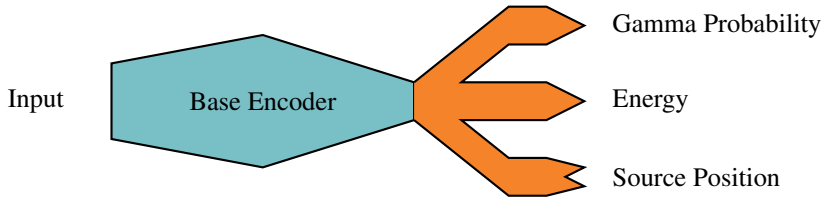


Fig. 9.15: Schematic of our Multi-Head Architecture.

images and maps them into a latent vector space. Then this vector representation is fed into three separate prediction heads, i.e., multilayer perceptrons that take the representation and compute a one- or two-dimensional output for each task.

For this study, we rely on different *EfficientNet* architectures [363] for our base encoder. The prediction heads alternate fully-connected layers of a fixed-width with batch normalization layers and ReLU activations for a specified number of layers. It computes output using a single affine layer with the desired number of outputs, as described above.

Multi-Head Loss Combining the three loss functions outlined above is not trivial because they have very different loss value ranges. This results in different magnitudes of the gradients used during the neural network training. Consequently, tasks with large gradients dominate the training. An ad-hoc solution is figuring out a set of weights and optimizing a weighted sum of the losses. This, however, is a tedious task, particularly when experimenting with different losses or transformations of the targets.

We propose using a normalization layer inspired by batch normalization that learns a normalizing constant for each loss. This constant is supposed to correspond to the inverse of a loss value easily obtained after a short amount of training, a kind of default loss. Hence we measure the loss *relative* to the default loss.

Let $l^t = (l_1^t, \ldots, l_k^t)$ be the vector of individual losses observed at the t-th weight update for $t \in \mathbb{N}_0$. We compute the following aggregates loss $\bar{l}^t$

$$w^t \leftarrow (1 - t^{-\alpha})w^{t-1} + t^{-\alpha} l^t \tag{9.21}$$

$$\bar{l}^t \leftarrow \frac{1}{k} \sum_{i=1}^{k} w_i^t l_i^t \tag{9.22}$$

where $\alpha \geq 1$ controls the speed of convergence of w to the default loss. We use $\alpha = 1.2$ in all of our experiments, but preliminary studies found that varying this hyperparameter had little effect.

Tab. 9.1: Comparison of the different *EfficientNet* model architectures used for our study regarding runtime.

Model	Number of Parameters	Fit Time in Seconds
efficientnet-b0	13 540 617	4 542.39
efficientnet-b1	16 046 253	6 694.09
efficientnet-b2	18 062 479	7 892.63
efficientnet-b3	21 877 725	10 367.49
efficientnet-b4	30 345 765	14 079.15
efficientnet-b5	42 721 037	18 886.73
efficientnet-b6	56 666 077	24 860.95
efficientnet-b7	81 234 685	33 882.70

9.4.4 Model Performance and Applications

We study two key questions: First, can CNNs replace hand-crafted features with reasonable accuracy on simulated data? Second, will these models generalize well enough to be used for real data?

Data When applying machine learning in astrophysics, it is difficult to obtain labeled data. A common approach to solve this problem is to combine Monte Carlo simulations (see Chapter 5) with a careful training of the classifier. Astrophysics has a profound understanding of particle interactions in the atmosphere. Given the energy and direction of some parent particle (gamma, proton, etc.), its interaction can be described by a probabilistic model that gives a probability for particle collisions, possibly resulting in secondary particles, which again may interact with each other. This results in a cascade of interactions that form the air shower, which can be simulated in software like the CORSIKA simulator [194]. The output is a simulated air shower, which needs to be run through a simulation of the telescope and camera device to produce realistic, raw data mimicking a shower that would have been recorded using the telescope. This allows us to simulate interesting particles (e.g., gamma) and uninteresting particles (e.g., proton) and label the resulting raw data accordingly. Furthermore, we can simulate particles of different energy levels and control the source position of the simulated gamma rays.

For our study, the simulation is run in the so-called diffuse mode that simulates gamma particles originating from positions distributed uniformly in the camera pane (see Section 8.4.1). compared with simulating only particles originating in the "Wobble"-ring, this allows to train source reconstruction models and reduces the bias toward gamma at wobble positions in gamma-hadron classifiers.

Models Architectures and Hyperparameters Our study investigates deep networks based on the *EfficientNet* architecture, which we train from scratch for our application. This family of deep networks contains seven differently sized models. Instead of the

traditional classification head, we use our multi-head architecture. We use heads of depth 3 with a fixed width of 512 neurons. In Table 9.1, we see the different resource requirements for the architectures considered in this study. The runtimes for fitting the deep network are measured on a Nvidia DGX-A100 using a single GPU.

Simulation Studies In Table 9.2, we see that the performance on holdout-simulation data is very good across all architectures. The classification performance of the gamma-hadron separation task is traditionally measured via the area under the ROC curve, and all models score roughly the same. Similarly, the origin estimation does not depend on the architecture. The energy estimation error, however, is larger for the three smallest architectures. In addition, one training run for *efficientnet-b7* failed to produce a good model for the energy estimation, resulting in the outlier mean and standard deviation score.

The performance of origin estimation is measured via the average angle θ between the true and the estimated origin. A more detailed analysis in Figure 9.16 (left) reveals that the deep-network analysis outperforms the random forest-based analysis substantially for lower energy, but is not competitive for higher energies. This angular resolution plot shows the 68 % quantile of the angles between true and estimated origin for different energy levels. A similar effect can be observed for the energy estimation. We see in Figure 9.16 (right) that for lower-energy particles, our deep network approach has a lower bias than the random forest analysis and that this effect reverses for higher energies. Here bias is defined as the median of the relative L1-error, and resolution is the difference between the 84.1 % quantile and the 15.9 % quantile of the relative L1-error, or intuitively the width of a confidence interval around the median error [294, Equation 6.5].

Two orthogonal approaches to further improve the performance come to mind: Using weighted sampling, we can increase the importance of the rarer higher-energy particles during training, with the hope of improving our results in the tail. Additionally, the combination of a random forest model with our deep network looks promising for getting a good performance across all energy levels.

Looking at the resource consumption of our models, we note that the smallest *efficientnet-b0* can still be executed efficiently and fast. On an Apple Macbook Pro 2020, using a single Intel i7 CPU core, it takes 4.48237 ± 0.3082ms to tag a single event using `onnx-runtime`[297], which is sufficiently fast for the typical arrival times in the FACT experiment without the need for specialized hardware.

OpenCrab Source Detection Now we evaluate the trained models from the last section on real-world data collected by the FACT telescope at the Observatorio del Roque de Los Muchachos (La Palma, Canary Islands, Spain). The telescope has been directed once towards a known gamma source, the Crab Nebula. Thus, a significantly higher number of gamma ray candidates originates from the direction of the known

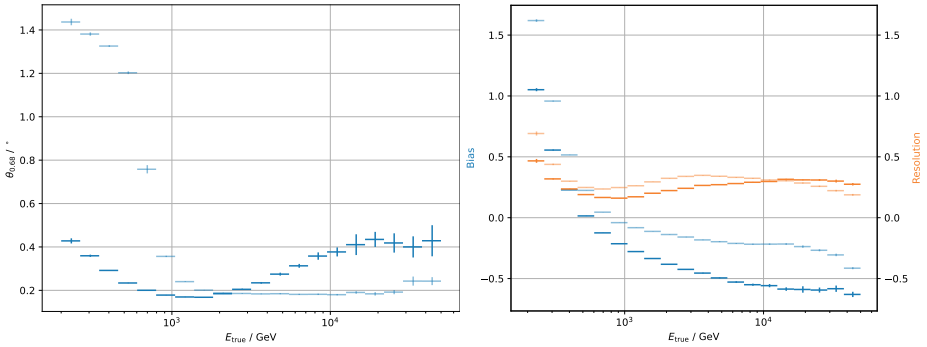


Fig. 9.16: Angular resolution of the origin estimator for different energy levels (left) and bias and resolution of the energy estimator. Transparent markers show the performance of the random-forest-based estimators.

Tab. 9.2: Performance of the three prediction heads on simulation data of the different multi-task model architectures.

Model	AUC-ROC ↑	Direction θ ↓	Energy Error ↓
efficientnet-b0	0.8939 ± 0.0020	0.3253 ± 0.0035	0.3646 ± 0.0856
efficientnet-b1	0.8972 ± 0.0026	0.3194 ± 0.0040	0.3540 ± 0.0767
efficientnet-b2	0.8968 ± 0.0038	0.3236 ± 0.0079	0.3407 ± 0.0603
efficientnet-b3	0.8953 ± 0.0040	0.3226 ± 0.0061	0.3273 ± 0.0515
efficientnet-b4	0.8898 ± 0.0066	0.3477 ± 0.0606	0.3248 ± 0.0543
efficientnet-b5	0.8883 ± 0.0083	0.3263 ± 0.0100	0.3383 ± 0.0530
efficientnet-b6	0.8929 ± 0.0035	0.3190 ± 0.0020	0.3294 ± 0.0345
efficientnet-b7	0.8897 ± 0.0041	0.3237 ± 0.0059	0.3936 ± 0.1476
efficientnet-b8	0.8876 ± 0.0087	0.3281 ± 0.0130	0.3217 ± 0.0103
SOTA Random Forest	0.7720 ± 0.0004	0.6127 ± 0.0006	0.3275 ± 0.0006

gamma-ray source than from other directions. On these recorded data, we run the full source detection pipeline of the FACT experiment and investigate the influence of the gamma-hadron separation models on the overall quality of the source detection.

The evaluation proceeds in the following steps. We take the publicly available Crab Nebula observation data [53, 84], which consist of 17.7 hours or 3 972 043 recorded events. Our multi-task model is applied to the events and estimates the probability of an event being a gamma particle and its direction of origin.

From the direction, we compute the angle between the trajectory of any incoming ray and the known direction of the Crab Nebula. The number of gamma rays can be regarded as a distribution that depends on the angle. High counts are to be expected for small angles. The distribution with respect to the direction of the Crab Nebula is called an *on-distribution* [165]. Contrasting the distribution of counts with distributions for five different positions with no known gamma sources yields the *off-distributions*. A uniform distribution of counts over angles is expected, where the majority of counted rays can be attributed to misclassified hadronic rays. We state the null hypothesis that on-distribution and off-distribution follow the same distribution or, intuitively, that there is no gamma source in the direction of the Crab Nebula. The margin by which a significance test rejects this null-hypothesis gives us a significance of detection $S_{Li\&Ma}$, reported by the number of standard deviations σ [250].

Definition 6 (Li & Ma statistic)**.** *Let N_{on}, N_{off} be the number of events in the on- and off-positions. Let $\alpha = \frac{1}{k}$ where k is the number of considered off-regions. Then we define*

$$S_{Li\&Ma}(N_{on}, N_{off}) = \left[2N_{on} \cdot \ln\left(\frac{1+\alpha}{\alpha} \cdot \frac{N_{on}}{N_{on}+N_{off}} \right) + 2N_{off} \cdot \ln\left((1+\alpha) \cdot \frac{N_{off}}{N_{on}+N_{off}} \right) \right]^{1/2}.$$

This $S_{Li\&Ma}$ is the performance metric for gamma source detection, where larger numbers are better. A more detailed discussion of this statistic is delayed until next section.

The trained classifiers output probabilities that an event is a gamma-ray. We can control the classification behavior by varying the threshold for actually predicting gamma. A large threshold yields fewer events and also fewer misclassified events because the classifier is more certain. If we set the threshold too large, we get too few total events, resulting in a small statistical significance. By contrast, if we decrease the threshold, we obtain more events but also more misclassifications. If we set the value too small, we count too many noise events, and the difference between on- and off-distribution shrinks, which also yields a low detection significance. Following common practice in gamma-ray astronomy, we chose the threshold that maximizes the *significance* of detection.

As we can see in Table 9.3, the significance of detection is large for all models that we have tested. Particularly the smaller models up to *efficientnet-b3* obtain large mean

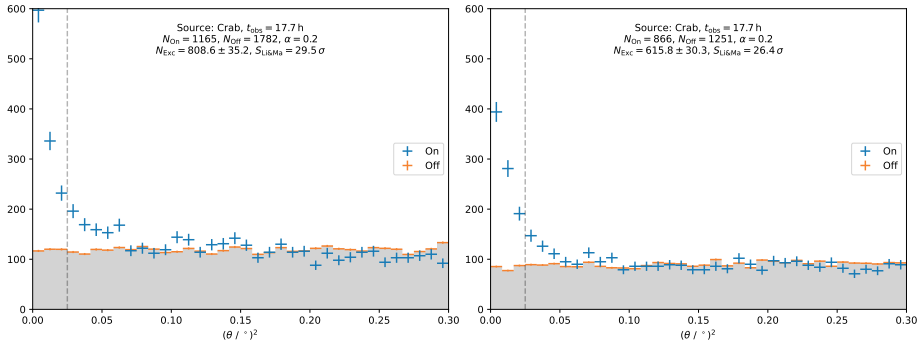


Fig. 9.17: Histogram of the frequencies of gamma predictions as a function of the squared angular distance between the trajectory of any incoming ray and a position in the sky for *efficientnet-b0* (left) and the standard analysis using random forest (right). On-events show the frequency with respect to the position of the Crab Nebula, while off-events reflect positions with no known sources. The significance of the detection test only considers angles smaller than 0.025 (left of the dashed vertical line).

Tab. 9.3: Significance of detection (Definition 6) on Crab Nebula data of the different model architectures.

Model	Mean ± Std-Dev	Min	Max
efficientnet-b0	26.85 ± 0.57	26.07	29.51
efficientnet-b1	27.28 ± 1.38	23.71	29.26
efficientnet-b2	27.02 ± 1.17	25.03	28.77
efficientnet-b3	26.73 ± 1.10	25.49	29.24
efficientnet-b4	26.80 ± 1.04	24.87	28.44
efficientnet-b5	26.49 ± 1.16	24.40	27.75
efficientnet-b6	25.64 ± 0.90	24.33	26.85
efficientnet-b7	25.64 ± 1.21	23.01	27.12
efficientnet-b8	25.66 ± 0.41	25.18	26.10
SOTA Random Forest	26.25 ± 0.15	-	-

significances of detection that seem to outperform the current state-of-the-art random forest models. However, we must note the large variance between the different runs of the examples. The standard deviation is one order of magnitude larger than for the random forest models. Unfortunately, we often observe this pattern for deep networks [94, 306]. Another drawback is that the performance measured on simulated data (Table 9.2) does not approximately predict the detection scores. For practitioners, advantages still remain, however. If we have the budget to train a number of candidate models, we can pick one that works really well. Looking at the maximum reported detection scores, we see that they exceed 29.0σ, which marks a substantial improvement on the state of the art.

We conclude that our multi-task model not only works on simulation data but also provides meaningful results on real-world source detection tasks. In Figure 9.17, we see how the *efficientnet-b0* model achieves a higher significance of detection than the current random-forest-based approach. Comparing the number of on-events and off-events, we see that it predicts more gamma events in the on-region, yielding a stronger signal of the source.

9.4.5 Discussion

Machine Learning is one of the basic building blocks of modern high-energy astroparticle experiments. One important task is the gamma-hadron separation, which aims at separating interesting gamma events from hadronic background. The FACT telescope measures Cherenkov light emitted in the earth's atmosphere when hit by gamma beams. The telescope measures at a rate of 60 events per second; better telescopes even measure orders of magnitudes faster, motivating the demand for fast processing pipelines. This is even more crucial for large telescope arrays in one location or global distribution of sites that are possibly being deployed under resource constraints and remote locations. Current approaches solve the event-tagging problem for FACT by using long processing pipelines that extract hand-crafted features, which are then used by random forests. In this section, we successfully replaced the long classification pipeline with a CNN.

10 Inverse Problems

10.1 Introduction

Tim Ruhe
Mirko Bunse
Wolfgang Rhode
Bernhard Spaan

Abstract: The reconstruction of experimentally inaccessible quantities is a common challenge in particle and astroparticle physics. This challenge arises for energy spectra of primary particles, e.g., gamma-rays or neutrinos, which are obtained only indirectly via the energy losses of secondary particles recorded in the detector. Unfortunately, the dependency between the energy of the primary particle and the energy loss of a secondary particle is governed by stochastic processes. A mathematical description of such processes is provided by the Fredholm integral equation of the first kind, which needs to be solved in order to obtain the desired energy spectra. The solution can be obtained by the utilization of deconvolution algorithms, which provide stable and accurate estimates by introducing regularization techniques to suppress their otherwise large variances. In this chapter, we describe the two deconvolution algorithms—DSEA+ (Dortmund Spectrum Estimation Algorithm) and TRUEE (Time-Dependent Regularized Unfolding for Economics and Engineering)—developed in CRC 876. While stable and accurate results can be obtained with both algorithms, the difference between the two arises from the way in which the inverse problem is solved. Within TRUEE, the inverse problem is solved in a maximum-likelihood approach, while DSEA interprets the inverse problem as a multinominal classification task. Accordingly, different fields of application arise for each algorithm.

In particle and astroparticle physics, measuring distributions of quantities that are experimentally inaccessible is a task that is both common and mathematically challenging. Inaccessible quantities of interest can only be inferred from correlated observables, an example of which is already given by the simple case of a liquid thermometer. Essentially, such a thermometer does not measure temperature directly but the height of a liquid in a narrow glass cylinder. Fortunately, a linear expression exists, which connects the height of the liquid to the temperature of the surrounding medium. For convenience, the manufacturer has already used this linear expression to ship the thermometer with a temperature scale, which allows us to directly read the temperature, although we actually read the height of the liquid.

https://doi.org/10.1515/9783110785968-010

In particle and astroparticle physics, matters are often not this fortunate. The the quantity of interest and the measured observable are not connected by a simple linear (or by a simple non-linear) equation. Instead, stochastical processes, e.g., particle interactions, are involved, which pose one of the largest challenges in measuring inaccessible quantities. Luckily, tools and algorithms exist that allow us to overcome these challenges. These tools and algorithms are referred to as *unfolding* or *deconvolution* methods. Some of these tools, in particular those developed in the context of the CRC 876, are the subject of this chapter.

This chapter is organized as follows: the remainder of this introduction provides a brief overview of the scope of a deconvolution analysis, while the mathematical background of deconvolution is provided in Section 10.2. Section 10.3 provides an overview of likelihood-based deconvolution methods, whereas a machine learning-based approach to this matter is given in Section 10.4. Deconvolution analyses are exemplified from a practical point of view via analyses from Imaging Cherenkov astronomy (Section 10.5) and neutrino astronomy (Section 10.6).

Remark (Nomenclature) Before we provide further details on the scope of deconvolution analyses, it seems appropriate to add a remark regarding the nomenclature in this book. Compared with Chapter 2, the meaning of the variables x and y are exchanged here in order to comply with the standard notation in machine learning. Within this chapter, y represents the sought-after target variable, whereas x is a measurable observable. This exchange in the meaning of the two variables is, however, purely syntactical and does not impact the validity of the statements regarding the matter of deconvolution in previous chapters.

In neutrino astronomy, as well as in many other areas of astroparticle physics, the outcome of most deconvolution analyses is an energy spectrum that is obtained for a certain type of particle and, sometimes, for a certain source of these particles. In a simplified view, these energy spectra follow power laws of the form $\frac{d\Phi}{dE} = \Phi_0 E^{-\gamma}$, where Φ_0 and γ take different values for different particles and also different values for the same particle if this particle was generated by different production mechanisms. This is, for example, the case for muon neutrinos, ν_μ, where the measured spectrum is expected to be the sum of three components. That is to say, these components are conventional atmospheric neutrinos and prompt atmospheric neutrinos, and high-energy neutrinos from astrophysical sources.

However, the scope of a deconvolution analysis is not to measure the individual Φ_0 and γ of such components. Forward-folding methods are, for example, better suited for this task. In order to do so, these methods have to assume certain flux distributions derived from theoretical models (conventional and prompt) or must simply be assumed to follow a power law (astrophysical). Theoretical considerations of this kind, although

providing valuable input to deconvolution analyses, might be incorrect with respect to certain small-scale features of the flux.

A deconvolution analysis does not have these limitations, and, even though simulated events, generated according to a certain distribution, are used to model the response function, the outcome of such an analysis is much more model-independent. By using deconvolved spectra, it was possible, for example, to confirm the observation of a flux of high-energy astrophysical neutrinos in a model-independent analysis [18]. One particular strength of the obtained deconvolution results is that they allow theorists to calibrate their theories such that their predictions agree with the experimental results. This benefit is particularly important for theoretical models that use extrapolations to predict fluxes at high energies. Deconvolution results also enable comparisons between spectra that are obtained with different experiments, although this opportunity has to be taken with a grain of salt in some cases. For neutrino telescopes, for example, the angular ranges of multiple measurements might be different, and this difference can cause deviations between their observed fluxes.

10.2 Keynote: Introduction to Inverse Problems

Michael Schmelling

Abstract: A common challenge in particle and astroparticle physics is the reconstruction of spectra distorted by the measurement process. A mathematical description of the problem is provided by the Fredholm integral equation of the first kind, which needs to be solved in order to obtain the information of interest. Here we give a general introduction, which defines the notation, describes the kind of effects one has to deal with, and quantifies them in information-theoretical terms. Algorithms that provide solutions for inverse problems by either likelihood- or classification-based approaches are then discussed in the following sections.

A central problem of any data-driven inference is the fact that a measurement almost always is only an indirect probe for the quantity of interest. This applies even to the seemingly obvious case of measuring the distance between two points with a ruler, where the actual reading is a proxy for the true thickness that is affected by the calibration of the ruler and the precision with which it can be read off.

The complexity increases dramatically when the quantity of interest is not a simple scalar quantity but a vector of discrete states or a continuum described by a probability density. Examples from the field of particle or astroparticle physics are multiplicity distributions or energy spectra of final state particles, which are created in high-energy collisions where every single measurement can be due to a vector of true values. The

inverse problem that must be solved is to infer the true distribution from a measured one when the distortion by the measurement process is known.

When a parametric model of the true distribution is given, the inverse problem can be solved by fitting those parameters to the data. This approach is referred to as "forward folding". The more demanding situation, however, is when no such model exists. Since a true continuum cannot be probed by a finite number of measurements, a more model-independent unfolding will be based on a discretized version of the density, such as a step-function that can be represented by a histogram or by an expansion into higher-order spline-functions or into a suitable basis of orthogonal polynomials.

In the following, we focus on the continuous unfolding problem and assume a linear detector response. Denoting by $\mathcal{X}$ the space of observables and by $\mathcal{Y}$ the space of the true quantities of interest, the density $f(y)$ describes the true distribution and $g(x)$ the observable density. We assume that f is a probability density function (pdf), that here the integral over $\mathcal{Y}$ is normalized. For $g(x)$ this holds only if the measurement process preserves the normalization. The measurement process is described by a response function $A(x|y)$, which for each y gives the distribution of the observable x. The relation between $g(x)$ and $f(y)$ is given by a Fredholm integral equation of the first kind

$$g(x) = \int_{\mathcal{Y}} dy\, A(x|y)\, f(y) \ . \tag{10.1}$$

When the response function $A(x|y)$ is known, e.g. from calibration measurements or accurate simulations of the detector, and given $g(x)$, then Equation 10.1 can be inverted to extract $f(y)$.

A typical experiment does not measure $g(x)$, but records a finite sample of N individual measurements $x_n \in \mathcal{X}$, $n = 1, \ldots, N$, drawn from a pdf proportional to $g(x)$. The exact density $g(x)$ is not known, but the measurements can be aggregated into an estimate $\hat{g}(x)$, which then can be used to construct an estimate $\hat{f}$ of the true pdf. Alternatively one can also directly map the measurements from the space $\mathcal{X}$ of the observations $\mathcal{Y}$ and aggregate the mappings into an estimate $\hat{f}$. In either case, the primary quantities to infer $f(y)$ are the measurements $D = \{x_n \in \mathcal{X} : 1 \leq n \leq N\}$.

10.2.1 Information

For a quantitative discussion of how the response function affects the measurement, it is helpful to have an expansion of a generic true pdf f into a complete set of orthogonal functions, where the individual terms can be associated with well defined features of f. Here we will consider functions that are defined on a finite interval, which without loss of generality can be taken as $[0, \pi]$. The true distribution can then be expanded into simple harmonic functions

$$f(y) = \sum_{k=0}^{\infty} a_k \cos(k\, y) \ , \tag{10.2}$$

where the basis functions have k equidistant zero crossings on the range $[0, \pi]$. The coefficients a_k have a simple intuitive interpretation regarding the resolution at which they probe the density f, which is qualitatively given by $\pi/(k+1)$ and closely related to the information given in a histogram with $k+1$ equal-size bins. Figure 10.1 shows those basis functions for $k = 0, \ldots, 6$.

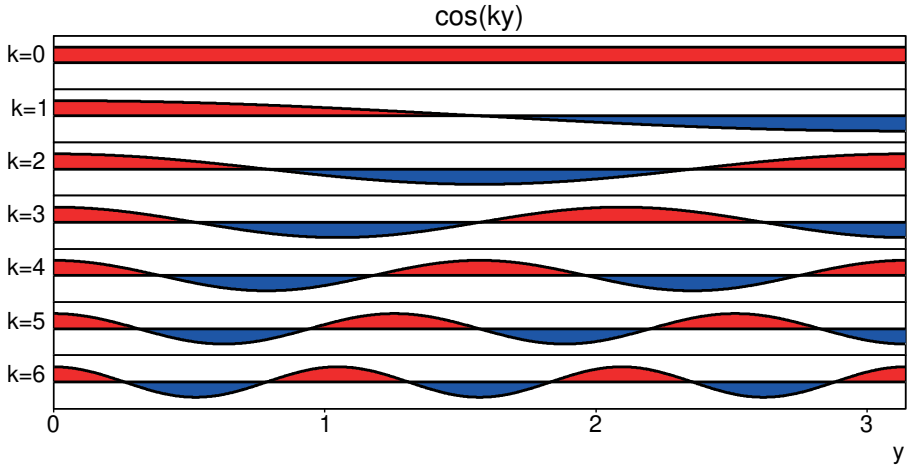


Fig. 10.1: Harmonic basis used to quantify the information about the shape of a pdf $f(y)$ defined on the range $y \in [0, \pi]$.

The knowledge about $f(y)$ then is given by the coefficients a_k, and the key question of each experiment is how well those coefficients can be estimated from a set of measurements. When taking variance as a measure for the precision of a measurement, the optimum is the Cramer-Raó limit which states that the minimum variance of an unbiased estimator of the a_k is given by the inverse of the Fisher information. The Fisher information I is proportional to the number of measurements N, with a proportionality factor that depends on the pdf ρ, which describes the distribution of the measurements x and which depends on the parameter a of interest. Given ρ and a the Fisher information per measurement is given by

$$I_\rho(a) = \left\langle \left(\frac{d \ln \rho}{da} \right)^2 \right\rangle = \int dx \, \frac{1}{\rho(x)} \left(\frac{d\rho(x)}{da} \right)^2 . \tag{10.3}$$

The information grows with the sensitivity of the pdf to the parameter of interest. For example, if one is interested in a location parameter, then a single measurement from a narrow distribution carries a larger amount of information than one from a wide distribution.

A useful variable to characterise the effect of the response function $A(x|y)$ relating a true pdf $f(y)$ to an observable one $g(x)$ are the attenuation factors ω_k defined as

$$\omega_k = \frac{I_g(a_k)}{I_f(a_k)} \ , \tag{10.4}$$

which quantify how much information about a_k is contained in a measurement drawn from g compared with a measurement drawn from the true pdf f. Since the information in a data set is proportional to the number of measurements, it follows that a set of N measurements with the actual detector has the same information about a_k as $N\omega_k$ measurements with a perfect detector. In the language of particle physics, this translates into how much the integrated luminosity[1] can be reduced to achieve a certain precision if one had a better detector.

10.2.2 The Effect of the Response Function

The ideal response function is $A(x|y) = \delta(x - y)$, where $\delta(0) = 1$ and $\delta(z) = 0 \ \forall \ z \neq 0$. Here the pdf $g(x)$ describing the measurements is equal to the truth $f(y)$ and any estimate $\hat{g}$ thus is immediately also an estimate $\hat{f}$. In general, this will not be the case, and different kinds of distortions of $g(x)$ with respect to $f(y)$ can occur, referred to as bias, efficiency losses, or migration effects. In the following, we will discuss them in turn and work out their impact in terms of the information carried by a measurement compared with a measurement with an ideal detector.

Since the Fisher information of a measurement is not only a function of the response function but also depends on the shape of $f(y)$, the general features of the problem will be discussed for the simple case of a uniform pdf $f(y) = 1/\pi$, with $y \in [0, \pi]$. The response functions considered are:

$$A(x|y) = \delta(x - b(y)) \quad \text{with} \quad b(y) = \pi \, \frac{\ln\left(2e^y - 1\right)}{\ln\left(2e^\pi - 1\right)} \tag{10.5}$$

$$A(x|y) = \epsilon(y)\delta(x - y) \quad \text{with} \quad \epsilon(y) = \frac{4y(\pi - y)}{\pi^2} \tag{10.6}$$

$$A(x|y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-y)^2/2\sigma^2} \quad \text{with} \quad \sigma = 0.2 \tag{10.7}$$

Here Equation 10.5 describes a distortion due to only a bias in the measurement; Equation 10.6, a distortion due to efficiency losses; and Equation 10.7, distortions due to a smearing of the measurements around the respective true values. The bias function $b(y)$ is constructed such that it is monotonic between $(x, y) = (0, 0)$ and $(x, y) = (\pi, \pi)$

1 The integrated luminosity is the time integral of the event rate for a given cross-section, and thus directly proportional to the size of the recorded data set.

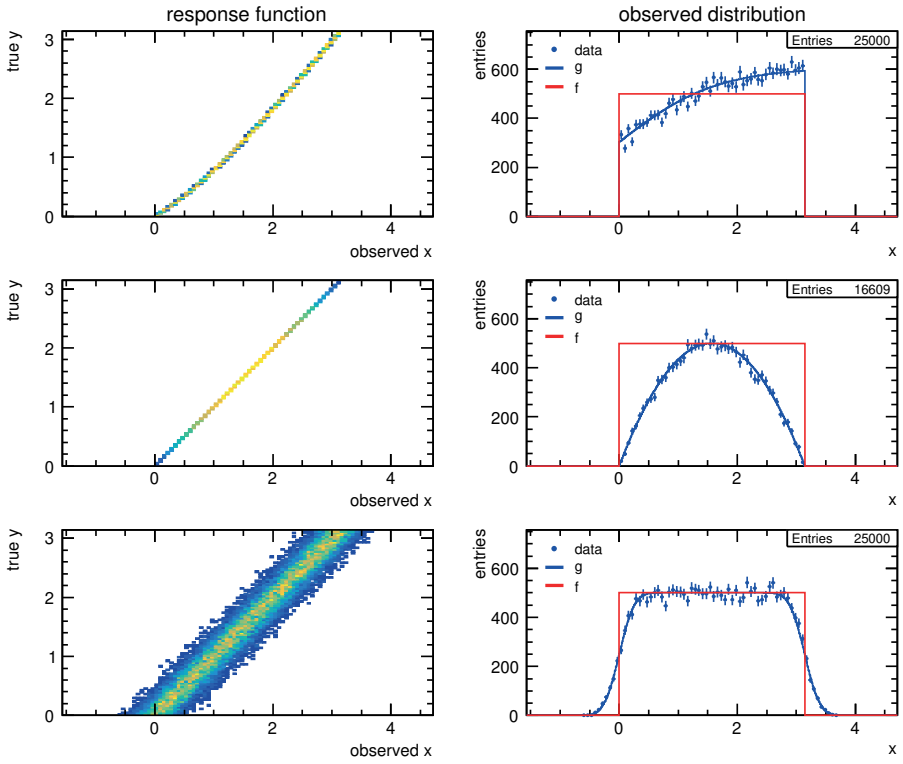


Fig. 10.2: Visualization of different effects of an imperfect detector. The left-hand column shows the different response functions; the right-hand column, the observable densities $g(x)$ compared with the assumed uniform true pdf f, and a histogram estimate of $g(x)$ corresponding to 25 000 events. The top row illustrates the case of a biased measurement; the middle, the case of efficiency losses; and the bottom, the case of finite resolution.

with finite, but different slopes at the end points. The efficiency function $\epsilon(y)$ has a simple parabolic shape with unit value in the center and zero at the boundaries. For the smearing function $A(x|y)$, a simple Gaussian is chosen. Figure 10.2 visualizes the response matrices and shows how the observable density $g(x)$ differs from the true $f(y)$ and illustrates how the measurements fluctuate around $g(x)$.

10.2.2.1 Bias

The top row of Figure 10.2 shows the effect of a biased measurement, where already a small non-linearity in the response function leads to sizeable distortions. The generic response function $A(x|y) = \delta(x - b(y))$ preserves normalization, and if the function $b(y)$ is strictly monotonic, then every measurement x can unambiguously be associated

with a true value y. For $b'(y) > 0$ and for the given example one obtains

$$g(x) = \frac{f(b^{-1}(x))}{b'(b^{-1}(x))} = \frac{\beta}{\pi(1 + e^{-x\beta})} \quad \text{with} \quad \beta = \frac{\ln(e^{2\pi} - 1)}{\pi} , \tag{10.8}$$

and calculation of the Fisher information per measurement for the expansion coefficients a_k yields

$$I_g(a_k) = \int dx \, \frac{1}{g(x)} \left(\frac{\partial g(x)}{\partial a_k} \right)^2 = \int_0^\pi dz \, \frac{1}{f(z)} \cos^2(kz) = I_f(a_k) \tag{10.9}$$

with $z = b^{-1}(x)$. This result is actually independent of any specific assumption about $f(y)$. Regarding the information content in cases where the only distortion is a bias term one finds

$$\omega_k = 1 \quad \forall \, k \, . \tag{10.10}$$

No information is lost for a one-to-one mapping between observation x and true value y.

10.2.2.2 Efficiency

Next we consider the case of no bias and perfect resolution while allowing for efficiency losses. The response function describing the measurement process in this case is $A(x|y) = \delta(x - y)\epsilon(y)$, where the function $\epsilon(y)$ is the probability that a true value y is actually recorded. Here the normalization is not conserved. Like the true pdf, the observed density is defined over $x \in [0, \pi]$ and one has

$$g(x) = f(x)\epsilon(x) \quad \text{with normalization} \quad G = \int dx \, g(x) \, . \tag{10.11}$$

The normalization G has to be taken into account when calculating the Fisher information of a measurement x about a parameter a_k. One finds

$$I_g(a_k) = \int dx \, \frac{1}{g(x)/G} \left(\frac{dg(x)/G}{da_k} \right)^2 = \frac{1}{G} \int_0^\pi dx \, \frac{\epsilon(x)}{f(x)} \cos^2(kx) \, . \tag{10.12}$$

If the efficiency is constant $\epsilon(x) = \epsilon$, then one has $G = \epsilon$ and the information per measurement is again $I_f(a_k)$. In this case, each recorded measurement contributes the same information as a measurement for a perfect detector. The efficiency function leads only to an overall reduction of the number of measurements compared with what one ideally would have recorded.

For a non-uniform efficiency function, the information content of a single measurement is affected. This is easily seen for the specific case Equation 10.6. For $k = 0$ one has $I_f(a_0) = I_g(a_0) = \pi^2$, and for $k > 0$ the explicit expressions are

$$I_f(a_k) = \frac{\pi^2}{2} \quad \text{and} \quad I_g(a_k) = \frac{\pi^2}{2} - \frac{3}{2k^2} \, . \tag{10.13}$$

One finds $\omega_0 = 1$, and

$$\omega_k = 1 - \frac{3}{k^2\pi^2} \tag{10.14}$$

for $k > 0$. There is a small loss of information in the low-order coefficients, but asymptotically each measurement contains as much information as if it had been recorded by an ideal detector. The main loss is due to the average efficiency, which results in an overall reduction in the number of usable data.

10.2.2.3 Resolution

The last effect to address is that of finite resolution. Here we will discuss only the special example Equation 10.7 of a uniform true density and a Gaussian response function, which is unbiased and conserves normalization. As before, the information of the true pdf is $I_f(a_0) = \pi^2$ and $I_f(a_k) = \pi^2/2$ for $k > 0$. What remains is to determine the information about the coefficients a_k per actual measurement x, which requires the observable density $g(x)$ and its derivative with respect to a_k. With $f(y) = 1/\pi$ over $y \in [0, \pi]$ one finds

$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} \int_0^\pi dy\, e^{-(x-y)^2/2\sigma^2} f(y) = \frac{1}{2\pi} \left(\operatorname{erf}\left(\frac{x}{\sqrt{2}\sigma} \right) - \operatorname{erf}\left(\frac{x-\pi}{\sqrt{2}\sigma} \right) \right) \tag{10.15}$$

and

$$\frac{dg(x)}{da_k} = \frac{1}{\sqrt{2\pi}\sigma} \int_0^\pi dy\, e^{-(x-y)^2/2\sigma^2} \cos(ky)\,. \tag{10.16}$$

Even for the simple case considered here, there exists no closed-form expression for the information in terms of elementary functions. However, the respective integrals can be calculated numerically, and a qualitative understanding of the case $\sigma \ll \pi$ is gained by considering the limiting cases $k \to 0$ and $k \to \infty$.

As shown in the bottom row of Figure 10.2, the requirement $\sigma \ll \pi$ leads to $g(x) \approx f(x)$, since in the limit $\sigma \to 0$ the smearing function $A(x|y)$ approached a delta function. Furthermore, for small values of k one finds that the derivatives $dg(x)/da_k$ are dominated by contributions from the interior of the interval $[0, \pi]$, and since that interval is much larger than σ one finds

$$\frac{dg(x)}{da_k} \approx \frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^\infty dy\, e^{-(x-y)^2/2\sigma^2} \cos(ky) = \cos(kx) e^{-\sigma^2 k^2/2}\,, \tag{10.17}$$

and the result for the leading coefficients becomes

$$I_g(a_k) \approx e^{-k^2\sigma^2} I_f(a_k) \quad \text{and thus} \quad \lim_{k\to 0} \omega_k = e^{-k^2\sigma^2}\,. \tag{10.18}$$

For $k \to \infty$ a different behaviour is observed. Now the rapid oscillation of the cosine function leads to a negligible contribution from the interior of $[0, \pi]$, and the derivative

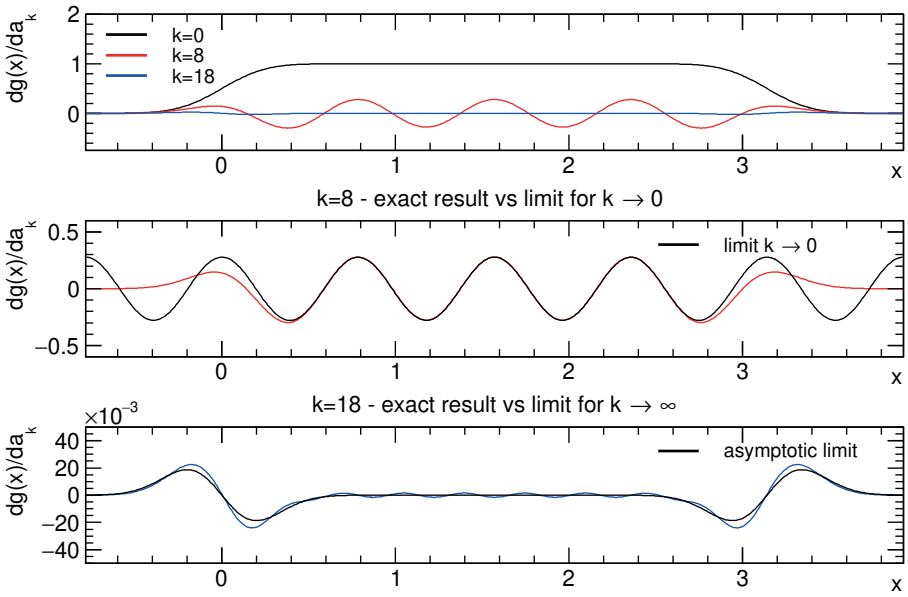


Fig. 10.3: Illustration of the derivatives $dg(x)/da_k$ for $k = 0$, 8 and 18.

$dg(x)/da_k$ as defined in Equation 10.16 is dominated by the contributions from the end points. This follows from the identity

$$\int_0^\pi dy\, f(y)\, \cos(ky) = \sum_{m=1}^{\infty} \frac{(-1)^m}{k^{2m}} \left[f^{(2m-1)}(0) - (-1)^k f^{(2m-1)}(\pi) \right] , \tag{10.19}$$

where substituting $f(y) = A(x|y)$ and keeping only the leading term $m = 1$ yields

$$\frac{dg(x)}{da_k} \approx \frac{1}{k^2\sigma^2} \frac{1}{\sqrt{2\pi}\sigma} \left((-1)^k (x-\pi)\, e^{-(x-\pi)^2/2\sigma^2} - x\, e^{-x^2/2\sigma^2} \right) . \tag{10.20}$$

This is a very interesting result, since it shows that the existence of hard limits for the pdf $f(y)$ constitutes a substantial amount of information about the higher order coefficients a_k. The rapid exponential decay of information found for $k \to 0$ is slowed down to a power law. For $\sigma \ll \pi$ the contributions from the end points are independent and the Fisher information becomes

$$I_g(a_k) \approx \frac{1}{k^4\sigma^3} \int_{-\infty}^{\infty} dz\, \frac{2z^2 e^{-z^2}}{1 - \mathrm{erf}^2(z/\sqrt{2})} \approx \frac{6.933}{k^4\sigma^3} , \tag{10.21}$$

where the factor 6.933 is the numerical value of the integral over z. It follows

$$\lim_{k\to\infty} \omega_k \approx \frac{1.405}{k^4\sigma^3} . \tag{10.22}$$

Figure 10.4 shows the attenuation factors ω_k for the Gaussian smearing of the example discussed here compared with their asymptotic expectations.

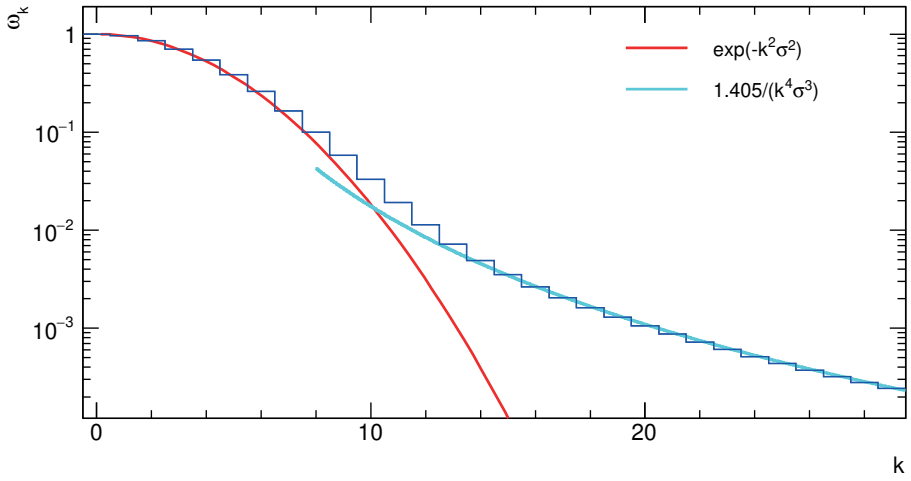


Fig. 10.4: Decay of information due to finite resolution. The leading order coefficients are exponentially damped; the higher order contributions are power-law suppressed.

10.2.2.4 Synopsis

The above considerations show how the detector response affects the information about the true distribution. A pure monotonic bias in the measurements does not entail any loss of information. Efficiency losses lead to a reduction of the number of recorded measurements, but for a sufficiently smooth efficiency function, the loss of information is marginal. The real problem for inverse problems is due to smearing effects. Here, for Gaussian resolutions with a variance σ^2, the information about the k-th order Fourier coefficient drops exponentially with $\exp(-k^2\sigma^2)$. If the true distribution is known to be bounded, the decay of information in the high-order coefficients is reduced to a power law proportional to $1/k^4$. Still, any information about the fine structure of the unknown true distribution is usually damped by many orders of magnitude and thus may not be accessible by a finite statistics measurement.

When resolution effects have to be corrected, inference about the true distribution usually requires additional information, either by, e.g., a parametric model or by another kind of a priori information such as positivity or smoothness. Regularization methods provide a framework to include such constraints in a controlled way. Even if results obtained by injecting a priori information are biased, it is always possible to ensure that the bias is small compared with the statistical uncertainties. As the above example showed, even seemingly soft constraints like the region of support of the true distribution may already contribute an amount of information that, when it comes to the fine structure of the solution, is equivalent to orders of magnitude more data.

10.2.3 Supplementary Remarks

The above discussion assumes that all elements in $D_{\mathcal{X}}$ are drawn from $g(x)$. In practical applications, one will have to deal with the case that some measurements are background from other sources that need taking care of. While certain measurements, such as the, in neutrino physics, are almost background-free, other particle or astroparticle physics measurements may be dominated by background. Here an important part of any analysis is to control the background either by "off-source" measurements where the properties of the background are probed by a data sample that contains no signal, or generally by studying data samples with different known mixtures of signal and background. This adds a new, discrete dimension to the unfolding problem.

The inverse problems discussed so far relate to a one-dimensional true distribution and a one-dimensional observed distribution. While with a suitable binning and a corresponding mapping in principle also, higher-dimensional inverse problems can be brought into this form, this may not always be practical. As a result, the response function may pick up dependencies on unknown distributions, leading to systematic uncertainties because the response function $A(x|y)$ is not precisely known.

To illustrate this, consider the case that a measurement x is affected by the values of two variables y_1 and y_2, such as an energy measurement x that depends on the true energy y_1 of the particle and the angle of inclination y_2 at which it hits the detector. In this case Equation 10.1 becomes

$$g(x) = \int_{\mathcal{Y}} dy_1\, dy_2 A(x|y_1, y_2) f(y_1, y_2)\ . \tag{10.23}$$

In order to deal with a one-dimensional problem that only involves x and y_1, one can re-write Equation 10.23 by integrating over y_2 as

$$g(x) = \int dy_1 \left(\frac{\int dy_2\, A(x|y_1, y_2) f(y_1, y_2)}{\int dy_2\, f(y_1, y_2)} \right) \cdot \int dy_2\, f(y_1, y_2) \tag{10.24}$$

or

$$g(x) = \int dy_1\, A(x|y_1) f(y_1)\ , \tag{10.25}$$

with the true energy spectrum $f(y_1)$ and the response function $A(x|y_1)$

$$f(y_1) = \int dy_2\, f(y_1, y_2) \quad \text{and} \quad A(x|y_1) = \frac{\int dy_2\, A(x|y_1, y_2) f(y_1, y_2)}{\int dy_2\, f(y_1, y_2)}\ . \tag{10.26}$$

By reducing the dimensionality of the inverse problem, the response function picks up a dependence on the unknown true density in y_1 and y_2. If the true pdf factorizes, $f(y_1, y_2) = f_1(y_1) f_2(y_2)$ one finds

$$A(x|y_1) = \int dy_2\, A(x|y_1, y_2) f_2(y_2)\ , \tag{10.27}$$

i.e. the response function depends only on the true distribution in y_2.

10.2.4 Mathematical Appendix: Integrals Over Cosine Functions

For $k > 0 \in \mathbb{N}$, the identity

$$\int_0^\pi dy\, f(y)\, \cos(ky) = \sum_{m=1}^{\infty} \frac{(-1)^m}{k^{2m}} \left[f^{(2m-1)}(0) - (-1)^k f^{(2m-1)}(\pi) \right] \tag{10.28}$$

follows from

$$\int dy\, y^n \cos(ky) = \sum_{m=0}^{n} m! \binom{n}{m} \frac{y^{n-m}}{k^{m+1}} \sin\left(ky + \frac{m\pi}{2}\right) \tag{10.29}$$

by using a Taylor expansion of $f(y)$ around $y = 0$

$$f(y) = \sum_{n=0}^{\infty} f^{(n)}(0) \frac{y^n}{n!} \,. \tag{10.30}$$

One obtains

$$\begin{aligned} \int_0^\pi dy\, f(y)\, \cos(ky) &= \sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!} \int_0^\pi dy\, y^n \cos(ky) \\ &= \sum_{n=0}^{\infty} \sum_{m=0}^{n} \frac{f^{(n)}(0)}{n!} m! \binom{n}{m} \frac{y^{n-m}}{k^{m+1}} \sin\left(ky + \frac{m\pi}{2}\right) \Bigg|_0^\pi \\ &= \sum_{m=0}^{\infty} \sin\left(ky + \frac{m\pi}{2}\right) \frac{1}{k^{m+1}} \sum_{n=m}^{\infty} f^{(n)}(0) \frac{y^{n-m}}{(n-m)!} \Bigg|_0^\pi \end{aligned} \tag{10.31}$$

where in the last step the summations over m and n were exchanged. Now the sum over n can be identified with the m-th derivative of $f(y)$,

$$\begin{aligned} \sum_{n=m}^{\infty} f^{(n)}(0) \frac{y^{n-m}}{(n-m)!} &= \frac{d^m}{dy^m} \sum_{n=m}^{\infty} f^{(n)}(0) \frac{y^n}{n!} \\ &= \frac{d^m}{dy^m} \sum_{n=0}^{\infty} f^{(n)}(0) \frac{y^n}{n!} = \frac{d^m}{dy^m} f(y) \end{aligned} \tag{10.32}$$

and the above expression simplifies to

$$\begin{aligned} &\int_0^\pi dy\, f(y)\, \cos(ky) \\ &\quad = \sum_{m=0}^{\infty} \frac{1}{k^{m+1}} \left(\sin\left(k\pi + \frac{m\pi}{2}\right) f^{(m)}(\pi) - \sin\left(\frac{m\pi}{2}\right) f^{(m)}(0) \right) \,. \end{aligned} \tag{10.33}$$

Here even m values do not contribute, and with the replacement $m \to 2m - 1$ the result Equation 10.28 follows.

10.3 Likelihood-Based Deconvolution

Mirko Bunse
Tim Ruhe
Maximilian Linhoff

Abstract: Likelihood-based deconvolution methods produce estimates that, in terms of well-defined prior assumptions, are most likely about the data-generating process. In gamma-ray astronomy, some well-motivated assumptions are that gamma particles are measured independently of each other, with some true rate that depends on the particle energy and the celestial object under study. The detector "smears" these underlying rates because the true particle energy is never measured, only estimated with uncertainty. Likelihood functions capture these assumptions and can be plugged into the Fredholm integral equation of the first kind, which models the deconvolution task. They are usually accompanied by regularization functions, which ensure that the method is robust towards noise in the data, mainly when the number of observations is small. We review existing likelihood-based deconvolution methods with a focus on our software TRUEE.

10.3.1 Introduction

Likelihood-based deconvolution methods are algorithms that maximize a likelihood function in order to produce deconvolution results. Such a likelihood function is based on a statistical model $\mathbb{P}(\vec{x} \mid \vec{\theta})$ that assigns a probability to each observation $\vec{x}$, given a parameter vector $\vec{\theta}$. Using this model, likelihood maximization approaches essentially select those parameters for which the probability of the set of all observations is maximal. In other words, we call a set of parameters "likely" if they assign a high probability to the observed data. In this case, we also refer to a good "agreement" between data and parameters.

Two crucial assumptions of the maximum likelihood method are that all $\vec{x} \in \mathcal{X}$ are *i)* statistically independent from each other, and *ii)* are identically distributed according to $\mathbb{P}(\vec{x} \mid \vec{\theta})$ with the same $\vec{\theta}$ vector. The definition of the likelihood function L reflects the intuition that a set of parameters is likely when it assigns a high probability to the random sample

$$L(\vec{\theta} \mid D) \;=\; \mathbb{P}(D \mid \vec{\theta}) \;=\; \prod_{\vec{x} \in D} \mathbb{P}(\vec{x} \mid \vec{\theta}). \tag{10.34}$$

Some likelihood functions can be maximized analytically. Often, however, we need to use numerical optimization methods to find parameters that maximize L. Unfortunately, a direct maximization of L via numerical optimization methods would introduce an

issue that can be described as a lack of "numerical stability". Namely, the many small values of $\mathbb{P}(\vec{x} \mid \vec{\theta})$ in L, which get even smaller when multiplied, are difficult to represent with typical floating-point variables. This difficulty easily leads to an overwhelming level of noise in the resulting estimates. In fact, directly maximizing L numerically would lead to many solutions that are insufficient for the purposes they are meant to solve.

Luckily, this lack of numerical stability can be circumvented by simply maximizing the logarithm of L instead of maximizing L directly. The logarithm yields values that a computer can represent more easily, but the solution of both maximization tasks is identical. As a matter of convention in numerical optimization, we further choose to *minimize* the *negative* logarithm instead of maximizing the logarithm of L. Again, the solution is identical to the solution we are actually looking for:

$$\begin{aligned} \vec{\theta}_{\max L} &= \arg\max_{\vec{\theta}} L(\vec{\theta} \mid D) \\ &= \arg\min_{\vec{\theta}} \left(-\ln L(\vec{\theta} \mid D) \right) = \arg\min_{\vec{\theta}} \left(-\sum_{\vec{x} \in D} \ln \mathbb{P}(\vec{x} \mid \vec{\theta}) \right) \end{aligned} \tag{10.35}$$

The right-hand side of Equation 10.35 defines the optimization task that many off-the-shelf optimization techniques can solve to find maximum likelihood estimates. All that remains for the general concept of likelihood maximization to solve the deconvolution problem is to define a deconvolution-specific statistical model.

We want this model to comprehend well-motivated assumptions about our data. Furthermore, the parameter values in $\vec{\theta}$ should help us understand the physical process underlying our observations. To this end, we will identify the free parameters of our model with the bins $\vec{f}_i = \vec{\theta}_i$ of the true underlying distribution. In this example, this distribution is the energy spectrum we want to estimate so that plotting $\vec{\theta}$ will result in a discrete variant of the spectrum $f(y)$ that we are after. Moreover, we will model the bins of the observed quantity as the $\vec{g}_j = \vec{x}_j$. The statistical model thus needs to relate the target bins (true energy) to the bins of the observed quantity (energy estimates or features). Maximizing the likelihood then allows us to feed the model with observations $\vec{g}$ to obtain the most likely spectrum $\vec{f}$.

The contents of $\vec{g}$ are integer counts of statistically independent events from a discretization step, which will be detailed in the next section. The important property here is that Poissonian statistics describes such a counting experiment. Each $\vec{g}_j$ is expected to follow its own Poisson distribution, which connects the true quantities in $\vec{f}$ via the response matrix A to our observed counts $\vec{g}$. This connection gives us the following likelihood function:

$$L(\vec{f} \mid \vec{g}) = \prod_{j=1}^{J} e^{-A_{\cdot j}^{\top} \vec{f}} \frac{(A_{\cdot j}^{\top} \vec{f})^{\vec{g}_j}}{\vec{g}_j!}. \tag{10.36}$$

While the Poissonian likelihood describes the problem well—for small counts—it does not yet solve the ill condition of the inverse problem. Furthermore, the optimization

of the likelihood will still yield oscillating results that i) deviate from the true $\vec{f}$ and ii) in general are *unphysical*. Energy spectra in particular are not expected to exhibit neighboring entries $\vec{f}_i, \vec{f}_{i+1}$ with large differences in between. Instead, the physical processes that are expected to play a role produce *smooth* spectra. To address this issue, an additional regularization term is added to the likelihood, that penalizes large second derivatives of $\vec{f}$:

$$r_\tau(\vec{f}) = \frac{\tau}{2} \cdot \sum_{i=2}^{I-1} \left(-\vec{f}_{i-1} + 2 \cdot \vec{f}_i - \vec{f}_{i+1} \right)^2 = \frac{\tau}{2} \cdot (C\vec{f})^2 \tag{10.37}$$

It is important to stress that additional assumptions must be made to improve the condition of the original inverse problem. Here, this assumption is that the resulting f should be *smooth*, defined by a low second-order derivative. These additional assumptions to be included in the likelihood will differ from problem to problem and from domain to domain. The method introduced above is a special form of Tikhonov regularization [366, 367] using the discrete second-order derivative to produce smooth solutions. Another common choice for C is the identity matrix, which will penalize a large norm of $\vec{f}$.

For a practical solution, it is important to consider what the values in f represent and which assumption of smoothness or small norm can be made. As introduced above, one of the effects that have to be dealt with is that of limited detection efficiency: not all events that reach a detector are recorded and not all recorded events are successfully analyzed. Smoothness assumptions in astrophysics can usually only be made on the f of the primary, physical spectrum that is not affected by these effects. This issue is discussed in more detail in Section 10.5, where the likelihood-based unfolding is applied to the FACT Open Data set.

Minimizing the following objective function is equivalent to maximizing the likelihood from Equation 10.36 with the regularization from Equation 10.37:

$$\ell(\vec{f}; \vec{g}, \tau) = -\ln L(\vec{f} \mid \vec{g}) + r_\tau(\vec{f}) \tag{10.38}$$

10.3.2 Discretization of the Observable Quantities

So far, we have not yet specified what an "observed bin" $\vec{g}_j$ is. The statistical model from Equation 10.36 is designed to handle integer counts in these bins. Consequently, each bin corresponds to one discrete state that an observed event can be in, and we count the occurrences of the states in the set of data we intend to deconvolve. The choice of the transformation from a continuous space $\mathcal{X}$ to a discrete space is ours to make. When we have defined a mapping $\mathcal{X} \to S$, where S is a finite set of disjunct states, we can map each event to one state and count the events that occur in each state. Basically, we quantize the feature space and count.

The quantization can be taken out through different methods. The most traditional approach in particle physics would be to select a single observed or reconstructed

feature dimension, preferably one that is well correlated with the target quantity $\mathcal{Y}$, and quantize only this dimension through equidistant binning. While the limitation to a single dimension ignores most of the information in the data, equidistant binning is likely to cause quantization errors. Due to these reasons, this method is mostly abandoned today.

A more effective approach is to quantize via unsupervised clustering, for instance, via k-means. This algorithm maps each point in the feature space to its nearest centroid. The number of centroids is predetermined by the user, and the centroid positions are chosen such that the reconstruction error of the training set is minimized. Therefore, k-means is a generalization of one-dimensional quantization to multiple dimensions. The k-means algorithm is able to use arbitrary numbers of features but its objective, the reconstruction error, is not necessarily aligned with the actual goal of deconvolution: to produce accurate predictions for $\mathcal{Y}$.

Therefore, we have proposed quantizing in a supervised fashion, using a simulated and labeled training set. The general idea is to interpret the target bins $\vec{f}_i$ as classes and to train a classifier that predicts the $\vec{f}_i$ from the given features. To work as a quantization method, the classification model needs to partition the feature space to make predictions. As one instance of this general proposal, we have evaluated decision trees [90] which recursively partition the feature space until a predetermined depth is achieved. We interpret the leaves of such a tree as the clusters to which each event is assigned. We count the number of events in each leaf and thereby obtain a count vector $\vec{g}$. Like in the k-means quantization, all features can be used by the tree. But unlike k-means quantization, the clusters are highly correlated with the target quantity $\mathcal{Y}$. In fact, they are designed to predict $\mathcal{Y}$, and are thus well-aligned with the goal of deconvolution.

10.3.3 Optimization

The core of our TRUEE software [272] is the RUN algorithm [85, 86], which maximizes Equation 10.38 numerically via the Newton-Raphson method [293]. Recall from Section 3.2.2 that this method evaluates a local second-order approximation $\hat{\ell}^{(k)}$ of the objective ℓ in each iteration k. The minimizer of each $\hat{\ell}^{(k)}$ is chosen as the respectively next estimate $\hat{\vec{f}}^{(k+1)}$. Taking out multiple iterations will eventually reach a local minimum of the actual objective function ℓ. Such an iterative approach is necessary because the actual function ℓ, in the case of likelihood-based deconvolution, can not be maximized analytically. Letting $\vec{f}_i = \vec{\theta}_i$, we obtain the following local model:

$$\hat{\ell}^{(k)}(\vec{f}) \;=\; \frac{1}{2}\vec{f}^{\top} H \vec{f} \;-\; \vec{f}^{\top}(H\hat{\vec{f}}^{(k)} - \vec{h}),$$

where $\vec{h} = \nabla \ell\left(\hat{\vec{f}}^{(k)}\right)$ is the gradient and $H = \nabla^2 \ell\left(\hat{\vec{f}}^{(k)}\right)$ is the Hessian of ℓ at the latest deconvolution estimate $\hat{\vec{f}}^{(k)}$.

10.3.4 Regularization with a Fixed Number of Degrees of Freedom

RUN does not fix the value of the regularization parameter τ, as other methods in machine learning, optimization, and statistics usually do. Instead, it controls the regularization strength by another parameter, n_{df}, the effective number of degrees of freedom. This parameter has a clearer interpretation, but it is still a hyperparameter that needs to be tuned just like a fixed value of τ would.

To fix n_{df}, RUN has to change τ in every iteration of the algorithm. Computing the value of τ for a given n_{df} requires an eigen-decomposition $H = U^\top V\, U$ of the Hessian matrix H. Based on this decomposition, a new matrix $C' = V^{-\frac{1}{2}} U^\top C^\top C U V^{-\frac{1}{2}}$ with the eigen-decomposition $C' = U'^\top V' U'$ is defined. By setting $\vec{f}'' = U'^\top V^{\frac{1}{2}} U^\top \vec{f}$, the regularized local model $\hat{\ell}_r^{(k)}$ can be reformulated as

$$\hat{\ell}_r^{(k)}(\vec{f}) = \frac{1}{2}\vec{f}''^\top (I + \tau \cdot V')\vec{f}'' - \vec{f}''^\top (U V^{-\frac{1}{2}} U')^\top (H\vec{f}^{(k)} - \vec{h}),$$

where the minimizer of $\hat{\ell}_r^{(k)}$ is $\hat{\vec{f}} = U V^{-\frac{1}{2}} U' \hat{\vec{f}}''$. In turn, $\hat{\vec{f}}''$ is the minimizer of the reformulated problem, being given by

$$\hat{\vec{f}}''_i = \frac{1}{1 + \tau V'_{ii}} \hat{\vec{f}}''_i$$
$$\hat{\vec{f}}'' = \left(U V^{-\frac{1}{2}} U'\right)^\top \left(H\vec{f}^{(k)} - \vec{h}\right).$$

Finally, the effective number of degrees of freedom of the minimizer $\hat{\vec{f}}$ is $n_{\mathrm{df}} = \sum_i \frac{1}{1+\tau V'_{ii}}$. In case of $\tau = 0$, each of the I components of $\hat{\vec{f}}''$ fully contributes to the solution with $n_{\mathrm{df}} = I$. Accordingly, the total contribution of $\hat{\vec{f}}''$ to $\hat{\vec{f}}$ is reduced whenever τ is greater than zero. A low value of n_{df} will smooth the deconvolution result by penalizing high curvatures.

The Newton-Raphson steps of the RUN algorithm assume a meaningful starting point $\vec{f}^{(1)}$. This first estimate is implemented as a basic least squares solution, which does not yet take the assumptions of bin-wise Poisson distributions and smoothness into account but already gives a starting point that is sufficiently close to the true solution. Here, $\vec{h}_{LSq}$ is the gradient, and H_{LSq} is the Hessian of the least-squares loss ℓ_{LSq} at the zero vector:

$$\ell_{LSq}(\vec{f}) = \frac{1}{2}(A\vec{f} - \vec{g})^2 \rightarrow \min$$

The original implementation of the RUN algorithm was written in Fortran77 and last updated in 1996. This led to a couple of compatibility issues, especially with the analysis framework ROOT [56]. Although ROOT is still extensively used in high-energy particles, say, at the LHC experiments, it has been largely replaced by designated Python packages in astroparticle physics. The TRUEE (Time-Dependent Regularized Unfolding for Economics and Engineering) is a ROOT-based C++ re-write of the RUN algorithm, which

does, however, include a couple of significant improvements. Two improvements are particularly interesting from a methodological point of view.

The first of these improvements is a verification step, in which the unfolding outcome is cross-checked via a re-weighting procedure. Simulated energy-dependent variables are re-weighted within this re-weighting scheme according to the unfolded spectrum. The obtained distributions are then compared with the experimental ones, and the cross-check is two-fold. First, an agreement between the reweighted simulated distributions and the experimental ones is expected for the variables utilized as input to the unfolding. A mismatch would hint at serious issues, concerning, say, the unfolding settings. One should, however, keep in mind that such issues are likely to also become visible via verification of the settings with Monte Carlo simulations. The second cross-check option is thus the more valuable one. Here, basically, the same comparison is carried out, but this time the experimental distributions are compared with simulated re-weighted distributions of variables *not* used for unfolding. An agreement of these distributions does, of course, not necessarily imply a successful unfolding but can be seen as a strong hint at the overall correctness of the entire procedure. This procedure is especially important in astroparticle physics, as simulated and experimentally obtained distributions do not necessarily agree (see Section 5.3.2). For details on the application of this verification procedure and practical examples, see [272], [16] and [329].

A second algorithmic improvement is a bootstrapping procedure, referred to as *pull mode*, in TRUEE. Within this bootstrapping procedure, a fixed number of events is randomly drawn from the sample of simulated events and subsequently unfolded. This is carried out n times, and the results are compared with the true number of events for each bin. This procedure verifies that correct results regarding an agreement of the unfolding result with the true underlying distribution have not been obtained by chance but are a consequence of careful optimization of the unfolding parameters. In addition, the unfolding result can be checked for a potential bias. In this case, the mean of all n unfoldings is compared with the true number of events for a particular bin. Observed disagreements indicate a bias towards an unfolding result, which possibly over- or underestimates the true bin content. Furthermore, the pull mode can be used to verify the derived statistical uncertainties, which, in addition to systematic uncertainties, impact the physical interpretation of the unfolded spectrum and should therefore neither be too large nor too small. By definition, 2/3 of all pull mode results will be contained within the statistical uncertainty, provided that it is estimated correctly. For details on the pull mode and some practical examples, see again [212] and [329].

The TRUEE package further includes several practical improvements that greatly enhance the usability of the package, including the use of up to three input variables, as well as possibilities of accounting for a background distribution and carrying out an automated acceptance correction. The possibility of visualizing the contributions of individual splines, implemented for one of the later versions of TRUEE, allows for additional cross-checks and thus also greatly enhances the reliability and the interpretability of the unfolding result.

10.3.5 Regularization with Minimum Global Correlation

As an alternative to the RUN algorithm, it is possible to minimize the likelihood objective from Equation 10.38 via off-the-shelf optimization algorithms. This approach is easy to implement but requires us to fix the regularization strength τ to a predetermined value. Usually, we want to choose τ from a set of candidate values to perform a grid search that deconvolves with all candidate values and chooses the best deconvolution result from the candidate set.

The global correlation coefficient is a criterion through which we can decide what "the best" deconvolution result in a set of candidates is. This criterion is defined as

$$\rho_{\text{global}} = \sum_{j=1}^{N_{\text{true}}} \sqrt{1 - \left((V_f)_{jj} \cdot (V_f^{-1})_{jj} \right)^{-1}}, \tag{10.39}$$

where V_f is the covariance matrix of $\vec{f}$, which we estimate as the inverse of the Hessian at the minimum of the negative log-likelihood.

We want to find the smallest global correlation, i.e., the deconvolution result where the correlation between all target bins is as small as possible. This desire stems from the fact that all target bins *should* be uncorrelated. Any apparent correlation stems merely from the fact that deconvolution methods estimate the target bins from observed bins rather than observing the target bins directly. Beyond this reasonable motivation for ρ_{global}, practitioners usually observe that results with minimum global correlation are also physically plausible.

10.4 Deconvolution as a Classification Task

Tim Ruhe
Mirko Bunse

Abstract: Given a set of distinct events, e.g., particle decays or air showers in a detector, we approach the deconvolution problem by aggregating class labels predicted for each event. This approach contrasts with likelihood-based deconvolution methods, which first aggregate the events and then produce a single estimate for the entire data set. Founding deconvolution on classification instead of likelihood functions has the advantage that the per-event information is not lost—it might be used to produce estimates dependent on other factors like time. A short digression on text analysis reveals the great flexibility of our resulting algorithm DSEA+.

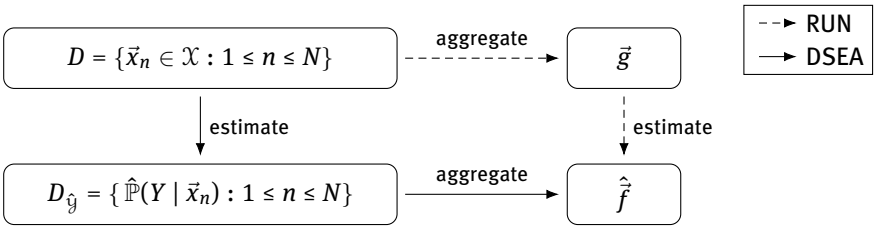


Fig. 10.5: Given a set of observations D, the target spectrum $\vec{f}$ can be estimated in two ways. RUN and other likelihood-based deconvolution methods first aggregate D and take out the estimation from there. DSEA employs a supervised machine learning model to first estimate the target quantity for each individual event. It aggregates these predictions in a second step, taking into account all uncertainties of the prediction model.

10.4.1 Introduction

So far, we have seen methods that first aggregate all observations into an observed probability density function $\vec{g}$ and then use this density to estimate the target spectrum $\vec{f}$. Having aggregated all observations into $\vec{g}$ reduces the deconvolution problem to solving a system of linear equations $\vec{g} = A\vec{f}$ that corresponds to a discrete variant of the Fredholm integral of the first kind. Starting from a set of N observations $D = \{\vec{x}_n \in \mathcal{X} : 1 \le n \le N\}$, however, we can also go a different way.

Instead of aggregating in the beginning, we can predict the target quantity, e.g., the energy of the primary particle, for each individual observation and aggregate the set $D_{\hat{\mathcal{Y}}} = \{\hat{\mathbb{P}}(Y \mid \vec{x}_n) : 1 \le n \le N\}$ of predictions in the end (see Figure 10.5). In contrast to likelihood-based methods, the contribution of individual examples to the deconvolution result is retained in $D_{\hat{\mathcal{Y}}}$. This advantage can be leveraged to study the deconvolution results as a function of other observables, like time or zenith angle.

This "late aggregation" is a key property of the *Dortmund Spectrum Estimation Algorithm* (DSEA). Moreover, DSEA supports un-discretized inputs $\vec{x}$ as long as the underlying classifier supports real-valued features. The algorithm, therefore, circumvents one layer of uncertainty that likelihood-based algorithms typically introduce through the discretization of $\vec{x}$. Last but not least, the approach can be employed in other use cases, such as political science, whenever a classifier $\hat{\mathbb{P}}(\mathcal{Y} \mid \vec{x})$ is already a part of the analysis pipeline.

To see why DSEA's way of estimating $\vec{f}$ works, consider the following marginalization, where only $\mathcal{Y}$ is discretized, but not $\mathcal{X}$.

$$\hat{\mathbb{P}}(\mathcal{Y} \equiv i) = \sum_{\vec{x} \in D} \hat{\mathbb{P}}(\mathcal{Y} \equiv i | \mathcal{X} = \vec{x}) \cdot \hat{\mathbb{P}}(\mathcal{X} = \vec{x}) \tag{10.40}$$

In DSEA, the conditional probabilities $\hat{\mathbb{P}}(\mathcal{Y} \equiv i | \mathcal{X} = \vec{x})$ are estimated by confidence values $c_h(i \mid \vec{x})$ of the underlying classifier, e.g. a random forest. In general, such value represents the confidence of the trained classifier h that the observation $\vec{x} \in D_{\text{obs}}$

actually belongs to the class i. In addition, a uniform prior is imposed on $\vec{x}$, given that $\mathcal{X}$ is a closed subset of $\mathbb{R}^d$. The outcome is the DSEA estimator, which recovers $\hat{\vec{f}}$ from confidence values:

$$\hat{\vec{f}}_i = \frac{1}{N} \sum_{n=1}^{N} c_h(i \mid \vec{x}_n) \tag{10.41}$$

10.4.2 Quantification with Classify-and-Count Methods

Classification-based deconvolution is also known as *quantification* [166, 182], a learning task for which methods quite similar to DSEA have been proposed. One of the closest connections is the Probabilistic Classify-and-Count (PCC) method, which aggregates confidence scores as in Equation 10.41. However, PCC does not reweight the training data iteratively like DSEA. Instead, it finishes after a single iteration of classifier training, prediction, and aggregation. In fact, PCC is equivalent to DSEA if and only if a single iteration is taken out.

The non-probabilistic version of Classify and Count (CC) is very similar to PCC but ignores the uncertainty of the classifier. Rather than aggregating confidence scores, which are rarely zero or one, it counts the crisp predictions of the classifier. With "crisp", we mean those predictions that assign exactly one class to each instance; in multi-class settings, these predictions are usually obtained by selecting the class label with the highest confidence score. The CC method then yields relative frequencies $\hat{\mathbb{P}}(\mathcal{Y} \equiv i) = \frac{\hat{N}_i}{N}$, where $\hat{N}_i$ is the number of events for which class i is predicted. However, ignoring the uncertainty of the classifier, which is only expressed through confidence values, produces deconvolution results with an inferior quality compared with the aggregation of confidences [333]. Therefore, DSEA is established around the PCC aggregation from Equation 10.41.

10.4.3 Accurate Estimates Through Iterative Reweighting

The confidence values returned for observations in D_{obs} are determined by the frequency of labels in D_{train}. For instance, Naive Bayes explicitly incorporates an estimate $\hat{\mathbb{P}}(\mathcal{Y} \equiv i)$ of this prior density. However, practitioners of deconvolution want the estimated target density $\hat{\vec{f}}$ to be independent of the prior density because the ultimate goal of deconvolution is to infer $\hat{\vec{f}}$ from observations, not from prior assumptions. In order to mitigate the influence of the prior density, DSEA iterates the recovery of $\vec{f}$, updating the training set density with the latest estimate of the density for D_{obs}. Such an update is performed by reweighting data points with weights $w_n \in \mathbb{R},\ 1 \leq n \leq N'$, assigned to each of the N' events in D_{train}. These weights are accounted for while training the classifier h, giving updated confidence values for the observations in D_{obs}.

The latest version of DSEA, DSEA$^+$ [105], weights each example with the ratio between the estimated probability $\hat{\vec{f}}^{(k-1)}_{i(n)}$ and the corresponding probability $\vec{f}^{\,t}_{i(n)}$ that is exhibited by the un-weighted training set. This modification accounts for non-uniformly distributed training sets and promotes fast convergence of the iterative procedure:

$$w_n^{(k)+} = \hat{\vec{f}}^{(k-1)}_{i(n)} \;/\; \vec{f}^{\,t}_{i(n)} \quad 1 \le n \le N'$$

To further speed up the convergence, DSEA$^+$ introduces scalable steps between iterations which are based on the following rationale: By iteratively updating the estimate $\hat{\vec{f}}$, a suitable approximation of the true density $\vec{f}$ is searched for in the space of all possible target densities. The vector $p^{(k)} \in \mathbb{R}^I$, also referred to as the *search direction*, depicts a step through this space. The next iterate $\hat{\vec{f}}^{(k)+}$ is computed from such a step scaled with $\alpha^{(k)} \ge 0$.

$$p^{(k)} = \hat{\vec{f}}^{(k)} - \hat{\vec{f}}^{(k-1)}, \qquad \text{where} \quad \hat{\vec{f}}^{(k)+} = \hat{\vec{f}}^{(k-1)} + \alpha^{(k)} \cdot p^{(k)}$$

Any constant step size $\alpha > 0$ and several decaying step size strategies with $\alpha^{(k)} < \alpha^{(k-1)}$ can be plugged into this general framework.

In order to find the best step size for each iteration, we propose an adaptive strategy that takes the latest estimate and the direction of search into account. This strategy computes $\alpha^{(k)}_{\text{RUN}}$ through the regularized objective function ℓ_r from the likelihood-based RUN method, see Equation 10.38. Here, unlike in RUN, ℓ_r is minimized only in the search direction determined by the DSEA estimator from Equation 10.41. The choice of $\alpha^{(k)}_{\text{RUN}}$ is parametrized by the regularization strength $\tau \ge 0$ contained in ℓ_r. Unlike RUN, we choose to fix τ instead of n_{df} when minimizing ℓ_r for simplicity.

$$\alpha^{(k)}_{\text{RUN}} = \underset{\alpha \ge 0}{\arg\min}\; \ell\left(\hat{\vec{f}}^{(k-1)} + \alpha \cdot p^{(k)}\,;\, \vec{g}, \tau\right)$$

By design, the adaptive strategy converges as soon as the likelihood can not be improved in the given search direction. Therefore, a χ^2 stopping criterion can be employed, assuming convergence if the probabilistic symmetric χ^2 distance between two subsequent iterations is below some threshold $\epsilon > 0$:

$$\chi^2(\hat{\vec{f}}^{(k)}, \hat{\vec{f}}^{(k-1)}) = 2 \cdot \sum_{i=1}^{I} (\hat{\vec{f}}_i^{(k)} - \hat{\vec{f}}_i^{(k-1)})^2 / (\hat{\vec{f}}_i^{(k)} + \hat{\vec{f}}_i^{(k-1)})$$

Usually, DSEA$^+$ will take a large first step with $\alpha > 1$ and then quickly reduce the step sizes due to its fast convergence property. It will stop after only a few iterations, as soon as the χ^2 stopping criterion is reached. This will typically result in an accurate estimate of the true target spectrum $\vec{f}$.

10.4.4 Classifier Choice

One advantage of DSEA is that any classifier which returns confidence scores can be employed, e.g., neural networks, random forests, logistic regression, Naive Bayes,

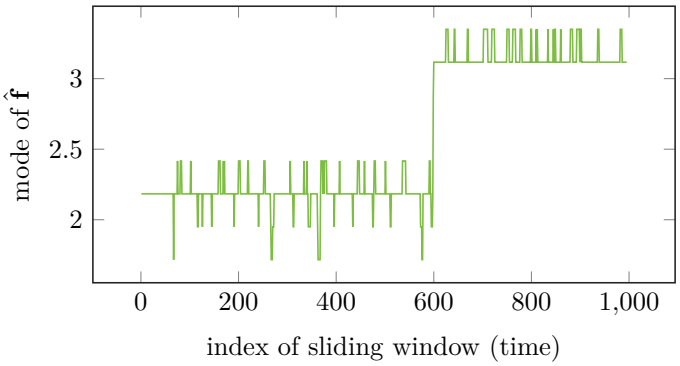


Fig. 10.6: The mode of a time-dependent deconvolution result is studied as a univariate time series. The figure clearly shows the concept shift that is artificially introduced by arranging two groups of observations.

nearest neighbors, and many others. This property is particularly beneficial if some classifier is already conventional for a given use case. Random forests, for instance, are the standard learning method in several Cherenkov telescope collaborations, and they can continue to be also used in DSEA. Moreover, the ability to plug in many different classification methods allows practitioners to easily try and test multiple alternatives for which off-the-shelf implementations are available.

Considering the marginalization from Equation 10.40, it is preferable to employ classifiers of which the confidence scores $c_h(i \mid \vec{x})$ resemble conditional probabilities $\mathbb{P}(\mathcal{Y} \equiv i \mid \mathcal{X} = \vec{x})$. In particular, we prefer *calibrated* [134] classifiers. The confidence values of such classifiers (approximately) match the long-run empirical probability $\hat{\mathbb{P}}(\mathcal{Y} \equiv i \mid \mathcal{X} = \vec{x})$, i.e.

$$\hat{\mathbb{P}}\big(Y \equiv i \;\mid\; c_h(i \mid \vec{x}) = c\big) \to c \quad \text{for } N \to \infty$$

Any classifier that is not sufficiently well calibrated can be turned into a calibrated classifier by means of *probability calibration* [396]. This additional training step uses a labeled calibration data set that is disjunct from the training set with which the actual classifier is trained. Our experience shows that probability calibration does not improve the classifier in terms of accuracy but achieves remarkable improvements in terms of the deconvolution performance.

10.4.5 Leveraging Eventwise Contributions

Since DSEA aggregates predictions instead of observations, we are free to alter the aggregation from $D_{\hat{y}}$ to $\vec{f}$ (see Figure 10.5) to account for other variables like zenith angle or time. For instance, we could study how the spectrum of a gamma-ray source

changes over time, by aggregating the event-wise contributions in $D_{\hat{y}}$ in a sliding time window.

We recognize that such a sliding window aggregation is not equal to the outcome of multiple separate, one-per-window deconvolutions, due to the reweighting in DSEA. Still, a sliding-window aggregation allows a practitioner to obtain insights about the deconvolution result in terms of additional variables in an efficient way because all event-wise contributions are readily computed. Likelihood-based methods do not have this property because their estimates are already based on aggregations; they can be run on separate windows but can not guarantee that an aggregation of their window-wise estimates is equal to an estimate of the full data set.

The feasibility of time-dependent deconvolution with DSEA$^+$ is verified in the toy study from Figure 10.6. In this study, a concept shift is generated artificially by arranging two subsets of the simulated MAGIC data, one with low energy values and one with high energy values. In the beginning, only low particle energies are observed. Suddenly, the concept changes, and only high energy values follow. This concept shift is akin to a transient event, such as a gamma-ray burst. DSEA$^+$ deconvolves both groups together, returning the contribution of each observation in the entire data set. In fact, it does not know about the two artificial groups. The returned contributions are then aggregated in each sliding window position, which moves over the time-ordered contributions. For simplicity, each reconstructed spectrum is represented only by its mode, i.e., its peak position. Figure 10.6 clearly shows that DSEA$^+$ successfully reconstructs the concept shift that was artificially generated.

Another benefit of retaining event-wise contributions is the inspection and verification of the results. If a deconvolution result does not look physically plausible, we can use these contributions to trace the error back to groups of individual events, both in D_{obs} and D_{train}. Are there specific mismatches between the simulation and the real data which hinder a successful deconvolution? Are there events that likely picture background but contribute much? Is the classifier incapable of predicting a certain group of events? We can use the eventwise contributions to trace down and debug these issues.

10.4.6 Excursion: Document Analysis in Political Science

Electoral programs allow political scientists to characterize political parties, track their positional changes over time, and identify their strategies, among other research goals [267]. The high cost of hand-labeling documents such as electoral programs motivates automated methods for text analysis. However, it is rarely the individual labels that political and social scientists are interested in. Rather, many works choose the frequency of labels in a collection to be a more appropriate representation [202]. For instance, the spectrum of political issues in an electoral program facilitates the characterization and

study of political parties. Estimating this spectrum, just like estimating energy spectra in particle and astroparticle physics, is, in fact, a deconvolution problem.

A great advantage of DSEA over likelihood-based methods when it comes to using cases like political science is that its embedded classifier is basically arbitrary. Whenever a classifier is already used, say, to classify sentences in electoral programs, applying DSEA is straightforward. Additionally, we can make use of the arbitrary aggregation already presented above to aggregate the contributions of sentences for each document in a collection. Likelihood-based methods, by contrast, struggle with the definition of the observable bins (meaningful "text bins" would need to be defined) and with the small number of sentences per document, which can lead to poor estimates. DSEA is able to leverage the full-text collection and provide its aggregated spectra for each document, using an existing classifier that is already tuned to the use case.

10.5 Deconvolution of IACT Data

Maximilian Linhoff
Mirko Bunse
Marlene Doert

Abstract: Once the initial detection of a source has been made, the main interest in gamma-ray astronomy is in the energy spectra of observed sources. After the feature extraction steps in Section 7.3 and the event-wise estimation of gamma-ray properties in Section 8.4 we now have event lists comprising just a few physical properties for each reconstructed shower. From this, we can estimate the instrument response from simulated events and then estimate the energy spectrum of the source via unfolding. In gamma-ray astronomy, there is nearly always a residual background, even after the gamma/hadron separation, that needs to be considered in the unfolding. Since we deal with event counts, regularized Poissonian likelihood approaches are most commonly employed to solve the inverse problems for imaging atmospheric Cherenkov telescopes (IACTs).

After the event property estimation in Section 8.4, our data is reduced to a few parameters for each air shower. These are the estimated energy $\hat{E}$, the estimated direction in sky-fixed coordinates right ascension ($\hat{\alpha}$), and declination ($\hat{\delta}$). For observed events, we additionally have the observation time, and for simulated events, the true energy E, true direction (α, δ), and true particle type are known. Additionally, the distribution of generated showers in the simulation is needed to compute the detection efficiency.

Strictly, using an energy estimator is not necessary for the unfolding. One could use lower-level features, such as the Hillas parameters discussed in Section 7.3, directly

as input for the unfolding. But using a direct estimate for the particle energy has some advantages:

- By combining a large number of features into a single estimate, the dimensionality of the problem is reduced while—it is hoped—keeping most of the relevant information. This reduces the dimensionality of the instrument response, and the "curse of dimensionality" decreases the amount of labeled data required to calculate it with sufficient accuracy. Together with the reduced dimension, an improved condition of the inverse problem is often the result of using one attribute with a better resolution than multiple attributes with worse resolution.
- While the low-level features are not the same between the different IACT experiments, the estimated primary energy is a universal quantity that is easily interpretable and well-defined for all experiments. This facilitates the exchange of event lists and precomputed instrument responses in a standardized data format such as GADF [138].

As in the previous sections (8.4, 9.4) on gamma-ray analysis, we provide an example of our analysis from the FACT open dataset, which is detailed in Section 6.3.

10.5.1 IACT Instrument Response Functions

An Instrument Response Function (IRF) R models the complete measurement process of an IACT, starting from primary particles, including their atmospheric interactions, emissions, detection of Cherenkov light, pre-processing, feature extraction, and event selection, until the final predictions of initial particle energies have been made. An IRF, therefore, comprises how effectively the IACT reconstructs primary particles of given properties. In the context of deconvolution, IRFs assume the role of the convolution kernel $R = A(\vec{x}|\vec{y})$ previously seen in Equation 10.1.

In gamma-ray astronomy, we model the true parameters of a primary particle, namely its true energy and sky coordinates, as the (vector-valued) target quantity of our deconvolution task, $\vec{y} = (E, \alpha, \delta)$. Given that we have estimated these parameters from telescope recordings, we model the observed quantities of our deconvolution task as $\vec{x} = (\hat{E}, \hat{\alpha}, \hat{\delta})$. The IRF, or convolution kernel, connects the two quantities, as before, so that the Fredholm equation becomes:

$$g(\hat{E}, \hat{\alpha}, \hat{\delta}) = \iiint R(\hat{E}, \hat{\alpha}, \hat{\delta}|E, \alpha, \delta) \cdot f(E, \alpha, \delta)\, \mathrm{d}E\, \mathrm{d}\alpha\, \mathrm{d}\delta + b(\hat{E}, \hat{\alpha}, \hat{\delta}). \tag{10.42}$$

Usually, R is factorized into quantity-exclusive components, assuming that all quantities are measured independently. Making this assumption consequently neglects potential correlations between the estimators. Using this factorization, we obtain

$$R(\hat{E}, \hat{\alpha}, \hat{\delta}|E, \alpha, \delta) = A_{\mathrm{eff}}(E, \alpha, \delta) \cdot M(\hat{E}|E, \alpha, \delta) \cdot P(\hat{\alpha}, \hat{\delta}|E, \alpha, \delta), \tag{10.43}$$

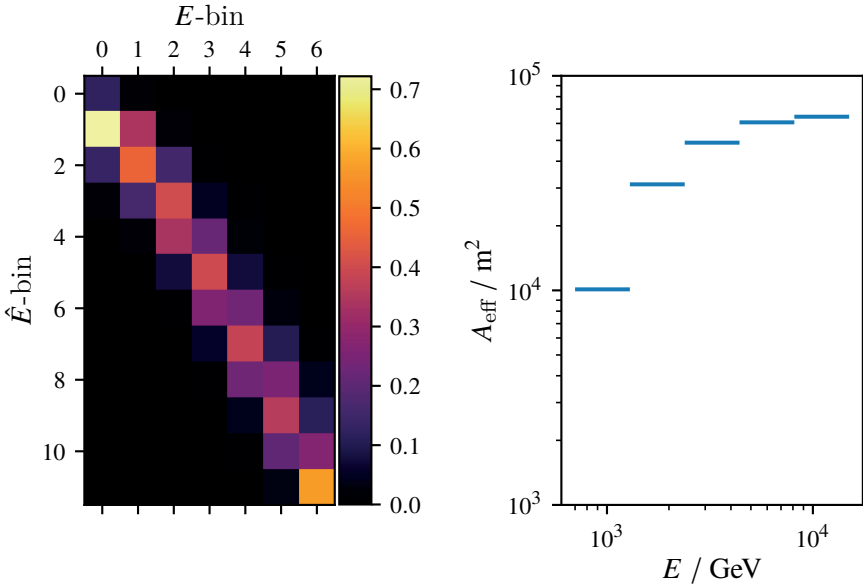


Fig. 10.7: Energy migration M and effective area A_{eff} for the FACT open data. Here, M is desired to be close to the identity matrix, which would amount to a perfect energy predictor. The effective area indicates, for each energy bin, which fraction of the actual events is seen by the telescope. Low values require a higher correction factor, which in turn require more observations to achieve a good signal-to-noise ratio in the respective energy bin.

where $A_{\text{eff}}(E)$ is the effective area that combines the energy-dependent detection efficiency with the area that the detector is able to observe. Furthermore, we have the energy migration $M(\hat{E}|E)$ and the Point Spread Function (PSF) $P(\hat{\alpha}, \hat{\delta}|\alpha, \delta)$, which characterize the error made when estimating the energy and direction of gamma rays from telescope recordings. The energy migration matrix can be computed from the confusion matrix of a classifier by a normalization that ensures the sum over the observed quantities to be the acceptance value for each bin of the true quantity.

The situation becomes dramatically simplified when known point sources of gamma rays are analyzed. The location of a point source is sufficiently characterized by a single sky coordinate, from which all gamma rays of this source originate. It, therefore, suffices to define a small region around this coordinate and to deconvolve the energy spectrum in this *on*-region. We do not need to account for the PSF in this case, because we can safely assume that the spread of direction reconstructions are sufficiently covered by the *on*-region. We can therefore drop the PSF from the above equation when analyzing a point source of gamma rays. Note, however, that the effective area must then account for the fact that all events outside of the *on*-region are discarded; falsely dropping signal events will reduce the effective area of our IACT. Without the PSF, the

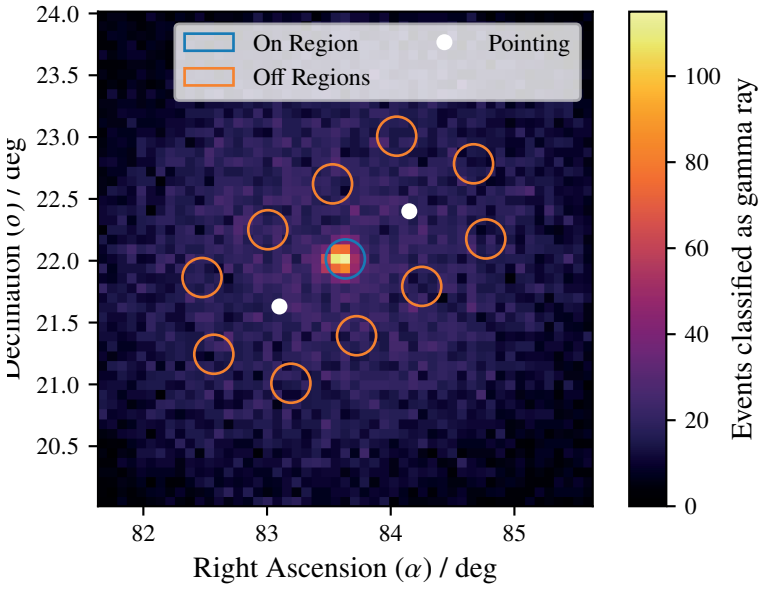


Fig. 10.8: The sky map of the estimated positions of recorded gamma-ray events. The telescope was pointed to one of two positions close to the Crab Nebula (on-region) for each observation. This allows selecting off-regions symmetrically to the on-region to estimate the remaining background (false positives).

Fredholm equation simplifies to

$$g(\hat{E}) = \int A_{\text{eff}}(E) \cdot M(\hat{E}|E) \cdot f(E)\,\mathrm{d}E + b(\hat{E}). \tag{10.44}$$

An additional benefit of analyzing point sources in an *on*-region is that we can define additional *off*-regions which the telescope can observe simultaneously. These *off*-regions allow us to estimate the background b from the same telescope recordings we use for estimating the signal, provided that no other gamma-ray source is located in any of these regions. Observing *on*- and *off*-regions simultaneously is referred to as "wobble mode operation" of the telescope. A sky map with the pointing, ON, and OFF positions is shown in Figure 10.8.

Since the effective area now only depends on the true energy, we can defer its consideration and neglect it in a first deconvolution step. Our final spectrum will be

$$\hat{f} = \hat{f}'/A_{\text{eff}}, \tag{10.45}$$

given that $\hat{f}'$ is a solution of the simplified deconvolution equation

$$g'(\hat{E}) = \int M(\hat{E}|E) \cdot f'(E)\,\mathrm{d}E + b(\hat{E}). \tag{10.46}$$

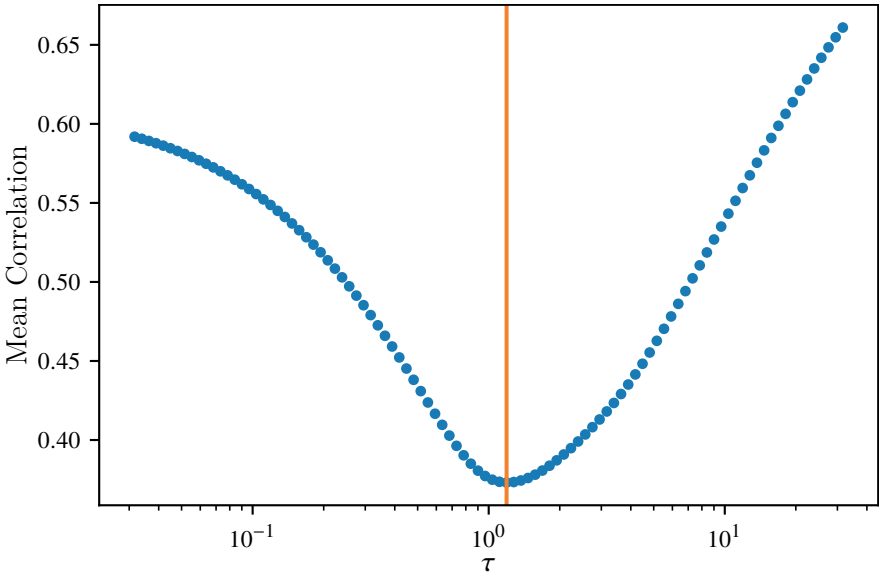


Fig. 10.9: Mean correlation of the unfolded event spectrum over the regularization strength τ. Each blue dot represents one deconvolution result. A well-acknowledged heuristic for the choice of τ is to select the value that minimizes the mean correlation, shown here by the vertical orange line.

This is often referred to as "acceptance correction". This separate application of the acceptance correction is only possible when all observations share the same IRF. Otherwise, the full IRF has to be included in the likelihood for each observation.

We transform the simplified equation above into a matrix equation by discretezing true and estimated energy with bins that are equidistant in logarithmic space:

$$\vec{g} = M \cdot \vec{f} + \vec{b}. \tag{10.47}$$

For instance, the migration matrix and effective areas that result from $N_{\text{true}} = 5$ bins in E and $N_{\text{est}} = 10$ bins in $\hat{E}$ are shown in Figure 10.7. The background $\vec{b}$ is estimated from the off positions. We only need a specialized deconvolution algorithm to solve Equation 10.46. The final spectrum is computed by plugging the deconvolution result into Equation 10.45.

10.5.2 Application of the Regularized Likelihood Deconvolution

We now solve Equation 10.46 with the regularized likelihood approach introduced in Section 10.3. Rewriting the equations to take into account the estimated background $\vec{b}$,

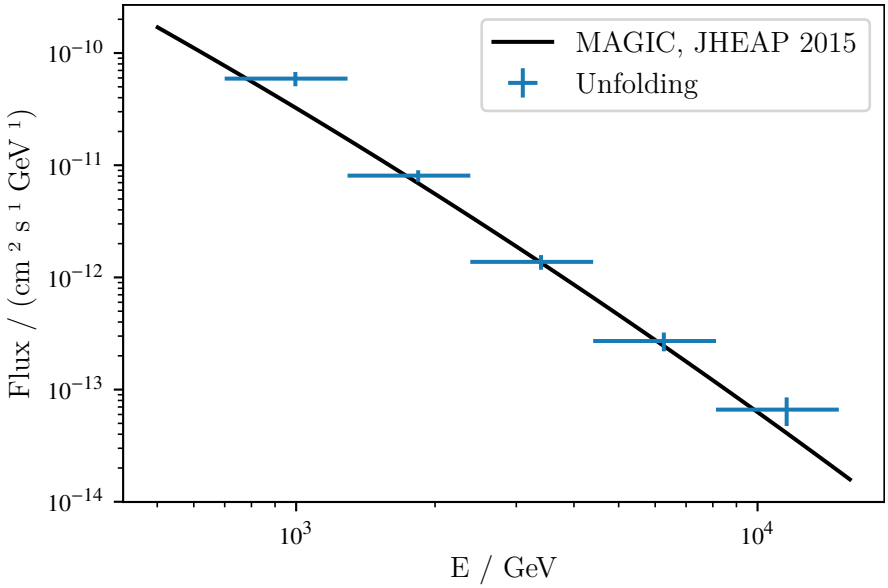


Fig. 10.10: The unfolded energy spectrum obtained with the FACT open data set (blue crosses). This result is compared with the forward-folding parametrization (black line) of the same celestial object, the Crab Nebula, obtained with data from a larger pair of telescopes named MAGIC [49].

we numerically minimize the negative log-likelihood

$$-\ln \mathcal{L}(\hat{\vec{f}}'|M, \vec{g}, \vec{b}) = \sum_{i=1}^{N_{\text{est}}} \left((\lambda_i + -g_i \cdot \ln \lambda_i\right) + r_\tau(\ln \vec{f}), \tag{10.48}$$

where

$$\lambda_i = (M \cdot \vec{f}')_i + b_i. \tag{10.49}$$

Here, r_τ is the Tikhonov regularization defined in Equation 10.37. Observe that we use the logarithm of the estimate, $\ln \vec{f}$, for the regularization because we want to penalize non-flatness only in the log-log space of the acceptance-corrected count spectrum. We expect only in this representation a steeply falling spectrum from the source.

To find the optimal regularization strength τ, we perform a grid scan that finds the smallest global correlation between the dimensions of our estimate, as detailed in Section 10.3.5. The resulting global correlation for a scan of many τ-values is shown in Figure 10.9.

The unfolding result that is obtained with the selected τ value is shown in Figure 10.10. Here, we also compare this result against a spectrum obtained with MAGIC, a pair of IACT telescopes that is considerably larger than the FACT telescope.

10.6 Deconvolution of Atmospheric Neutrino Spectra

Tim Ruhe
Karolin Hymon

Abstract: As neutrinos cannot be directly detected, their energy is assessed via their leptonic partners, e.g., muons, which emerge during neutrino interactions in the ice or the surrounding bedrock. The production of a muon of energy E_μ from a neutrino of energy E_ν, however, is a stochastic process that depends on the neutrino energy spectrum, as well as on details of the neutrino interaction. Such processes can be described via the Fredholm integral equation of the first kind, which can be solved via a likelihood-based or a machine learning-based deconvolution. This section addresses the details and challenges associated with applying the machine learning-based deconvolution algorithm `DSEA`. The utilization of DSEA is exemplified via its application in the reconstruction of muon neutrino energy spectra from data obtained with the IceCube detector in the 86-string configuration. While previous analyses were focused on the spectrum as a whole, the analysis presented here investigates the possible impacts of seasonal variations in the atmospheric temperature on atmospheric neutrino spectra.

As neutrinos cannot be directly observed, their energy spectrum cannot be directly accessed, and its reconstruction requires deconvolution techniques. Similar to gamma rays, the energy spectrum of atmospheric neutrinos is expected to follow a power law of the form $\frac{d\Phi}{dE} \propto E^{-\gamma}$, with $\gamma \approx 3.7$. There are, however, certain differences between gamma rays and neutrinos concerning the challenges and the utilized approaches in the reconstruction of the respective energy spectra. This section will first provide an overview of the differences and similarities and then provide a sample reconstruction of neutrino energy spectra, using a spectral analysis of seasonal variations in the atmospheric neutrino flux carried out using data obtained with the IceCube neutrino telescope (see Section 2.2.1.1)[210].

As analyses of seasonal variations of the neutrino flux generally utilize muon neutrinos (ν_μ), this section will only discuss challenges and techniques associated with reconstructing ν_μ energy spectra. Compared with electron and tau neutrinos, the challenges for reconstructing ν_μ-energy spectra mainly arise from the indirect measurement process and the inaccurate knowledge of the neutrino interaction vertex. Both processes are detailed below. Although the measurement process is indirect for ν_e and ν_τ as well, the knowledge about the interaction vertex and the energy resolution for these neutrino flavors are much better. This is because both neutrino flavors are observed as particle cascades where the energy is deposited almost entirely inside the detector. Concerning the reconstruction of energy spectra, the main challenges for ν_e and ν_τ arise from the detector's decreased sensitivity for these neutrino flavors. This

decreased sensitivity arises because ν_μ may also be produced outside the detector. By contrast, ν_e and ν_τ have to interact inside the instrumented volume to be detected (see Section 4.2.3).

Since ν_τ are not produced in the Earth's atmosphere, they solely contribute to the flux of high-energy cosmic neutrinos, and one expects only a handful of events per year. Within an analysis, this number is further reduced by the application of quality cuts, making the energy spectra reconstruction infeasible for this neutrino flavor. Electron neutrinos are, in fact, produced in the Earth's atmosphere but are also detected at a much lower rate compared with muon neutrinos. Two physical effects contribute here. First, the effective volume of the detector is much smaller for cascades, and second, only electron neutrinos from one specific decay channel (three-body kaon decay) can be detected with IceCube. The much more abundant ν_e from the decay of low-energy atmospheric muons are far below the energy threshold of IceCube. Studies of spectral changes due to seasonal temperature variations in the Earth's atmosphere do, however, require a resolution that is as large as possible, which requires a sufficient amount of events. The analysis discussed here, therefore, utilizes muon neutrinos to investigate spectral effects.

Rate variations of atmospheric muons and neutrinos arise from seasonal variations in the temperature of the Earth's atmosphere (cf. [173, 368, 369]). These seasonal temperature variations are expected not only to affect the overall rate of atmospheric muons and neutrinos but also to have an impact on their energy spectra, which occurs due to a temperature dependence in the critical energies of their parent mesons (pions and kaons) [173].

As stated above, ν_μ cannot be observed directly, and thus, their detection proceeds via their leptonic partner, the muon (μ), which is created in a charged-current (CC) interaction of the incident neutrino with a nucleon (see Section 4.2.3). The production of muons in CC-neutrino interactions is governed by the following equation [173]:

$$\frac{dN_\mu}{dE_\mu} = \int_{E_\mu}^{\infty} \left(\frac{dN_\nu}{dE_\nu}\right) \left(\frac{dP(E_\nu)}{dE_\mu}\right) dE_\nu. \tag{10.50}$$

In Equation 10.50 $\frac{dN_\mu}{dE_\mu}$ corresponds to the energy spectrum of the observed muons and $\frac{dN_\nu}{dE_\nu}$ is the sought-after neutrino energy spectrum. The term $\frac{dP(E_\nu)}{dE_\mu}$ contains the physics of the neutrino interaction and describes the probability of obtaining a muon of energy energy E_μ from a neutrino with energy E_ν in a CC interaction. Comparing Equation 10.50 to the Fredholm integral equation of the first kind (Equation 10.1), one finds that the reconstruction of neutrino energy spectra is indeed an inverse problem, and as such requires the use of deconvolution algorithms.

The muon energy E_μ, however, cannot be measured directly but has to be inferred from the energy deposited in the form of Cherenkov radiation inside the instrumented volume of the detector. Several E_μ-dependent observables, often referred to as energy proxies, with different levels of abstraction, can be extracted from the raw data. The

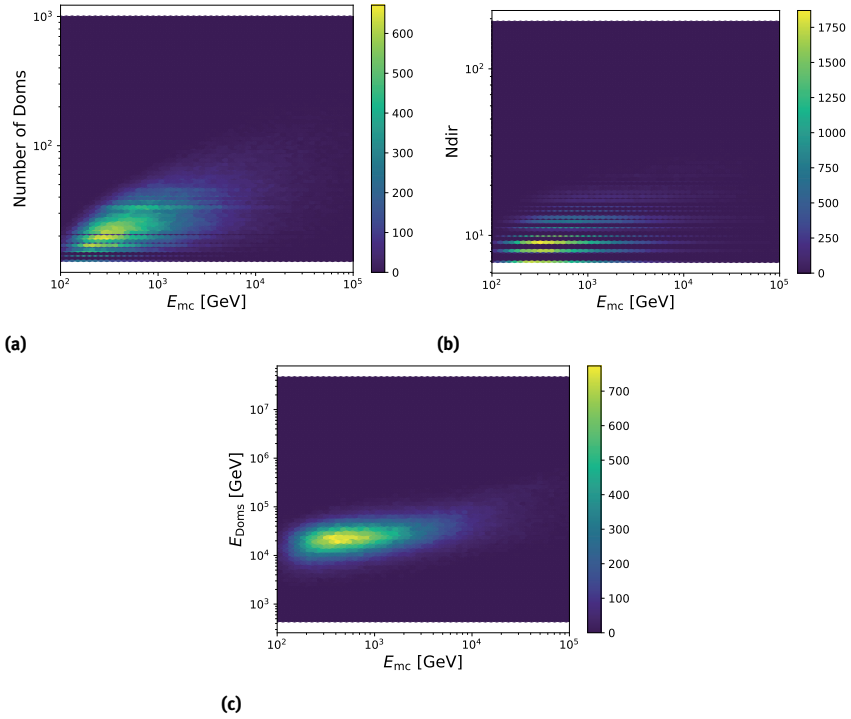


Fig. 10.11: Correlation of the true neutrino energy with the number of hit DOMs (a), the number of direct photons N_{dir} (b), and an energy estimator E_{Doms} (c). The strong smearing, which is a result of the two levels of indirection, is clearly visible. The horizontal stripes in the top panel arise from the fact that both variables are discrete. Figures from [209]

.

total number of hit optical modules and the total charge accumulated across all of the hit modules are particularly straightforward. Thus, compared with gamma rays, where the utilized variables directly approximate the gamma energy, the reconstruction of neutrino energy spectra faces an additional level of indirection, which manifests itself in a larger smearing of the energy proxies with respect to the energy of the incident neutrino. Figure 10.11 depicts the correlation of three input variables with the true neutrino energy. The correlation with the number of hit DOMs is shown in Figure 10.11a, whereas the correlation with the number of direct photons N_{dir} is depicted in Figure 10.11b. In IceCube, a photon is called *direct*, in case it can be considered to not have been subject to scattering. Unscattered photons will thus travel on a straight line from their emission point to a DOM, where they are eventually recorded. To estimate whether a photon can be considered as direct or not, the travel time t_{travel} to the respective optical module is computed with respect to the reconstructed track. Because the reconstructed track is only an estimate, however, photons are considered unscattered if they are detected

within a given window around their expected arrival time. A typical time window for direct photons is $-15\,\text{ns} \leq t_{\text{travel}} \leq 75\,\text{ns}$. Figure 10.11c depicts the correlation of the true neutrino energy with an energy estimator, generally referred to as truncated energy [32]. The impact of the two levels of indirection is directly visible in the relatively large smearing of the three variables. Horizontal lines, observed in Figures 10.11a and 10.11b, arise from the fact that both input variables are discrete. This discreteness does, however, not impact the deconvolution result.

Due to its large track length of several kilometers, the muon is not necessarily generated within the instrumented volume of the detector but in the surrounding ice or even the bedrock. Hence, the exact vertex of the neutrino interaction is in many cases inherently unknown, and the amount of energy lost by the muon on its way to the detector is unknown as well. This introduces an additional level of smearing. The occurrence of stochastic energy losses and noise caused by the readout electronics further complicate the problem. Mathematically, all sources of smearing (indirect detection, unknown point of interaction, muon propagation, and readout electronics) are captured in the kernel $A(x|y)$ in Equation 10.1. Another difference in the reconstruction of γ-ray energy spectra concerns the background level. While a certain number of background events is tolerable for gammas and handled via the additional term $b(E)$ in Equation 10.1, the purity of the utilized neutrino samples is generally well above 99 % (see Section 8.3).

As for many other deconvolution analyses, an analysis with DSEA generally commences with selecting a proper set of input variables. The choice of input variables is crucial for the accuracy of the unfolding result, and the selected input variables are required to exhibit as good as possible a correlation to the target variable. Variable selection should, however, be handled with care, as selecting n variables with the strongest correlation with the target variable will not necessarily deliver the most accurate unfolding result. The reason for this is that variables selected this way will exhibit a certain amount of correlation among each other via their mutual energy dependence. This might then lead to the use of highly redundant information encoded in different energy estimators. As the detection medium of IceCube is a natural medium, its properties are depth-dependent. The use of geometric information, e.g., the depth of an event inside the detector, can, thus, provide valuable information for spectral reconstruction [170, 171].

Input variables to the deconvolution presented here have been chosen in a trial-and-error approach by studying different variable combinations and their impact on the unfolded energy spectrum. The benefit of this is obviously not the speed but the possibility of understanding how the deconvolution algorithm deals with different variable choices. However, if the list of candidate variables grows too long, this approach becomes infeasible, and an automated variable selection is recommended instead (see Section 7.2). Although the utilization of a classification algorithm within DSEA allows for the use of an arbitrary number of input variables, three energy-correlated variables were found to be optimal for the analysis presented here. In the next step, the optimal parameter settings have to be found, as DSEA allows for choosing numerous

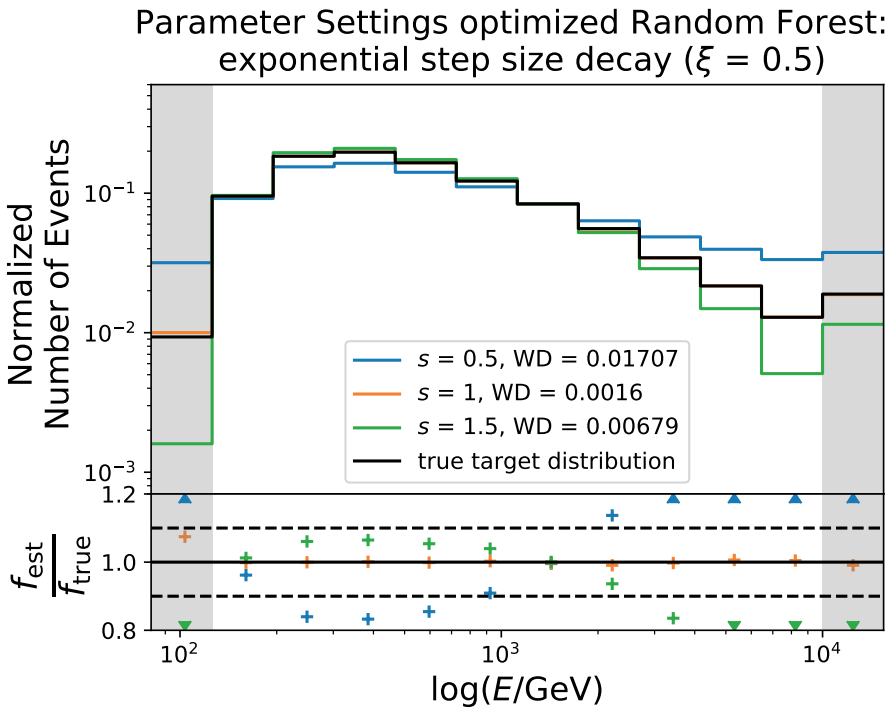


Fig. 10.12: Parameter optimization for DSEA using the normalization s as an example. WD denotes the Wasserstein distance obtained for the individual settings, whereas ξ represents the decay rate.

combinations of parameters. The parameter optimization requires a comparison to the true neutrino energy spectrum, and therefore the optimization needs to be carried out using simulated examples. In order to gain an understanding of the quality of the deconvolution in individual bins, the number of simulated events used within the optimization process should correspond to the number of events in experimental data. The agreement between the true energy spectrum and the deconvolution result can then be quantified via the use of a distance measure, e.g., the Wasserstein distance. The parameter settings were optimized for the analysis presented here in a grid search. The final settings include a Random Forest as a classifier and an exponentially decaying step size, governed by:

$$\alpha_k = s \cdot \xi^{k-1}, \tag{10.51}$$

where k represents the number of iterations, ξ is the decay rate, and s denotes the normalization of the decay rate.

The choice of the binning is—at least in principle—arbitrary, because DSEA interprets the entire deconvolution as a classification task. Classes of events are then handled by the classification algorithm, where the ordering of the classes is ignored. Ignoring the

neighborhood relations of bins in DSEA is one of the few shortcomings of the algorithm. Examples near the lower edge of bin $i+1$ might be more similar to events at the center of bin i than to events near the upper edge of bin $i+1$. This especially holds if logarithmic binning is used, which is often the case in neutrino astronomy due to the expected power-law behavior of the reconstructed spectra. So far, however, this shortcoming has not had a negative impact on the accuracy of the reconstructed spectra.

When selecting the number of bins and the respective bin edges, it is advisable to ensure that a sufficient number of events per bin is available for training and validation. The meaning of *sufficient* is somewhat vague in this context, as the required number of events largely depends on the problem at hand. If the individual classes are easily separable by the selected classification algorithm, fewer events may be chosen than for cases where the classification task is difficult. From a machine learning point of view, one should utilize as many examples as possible to ensure an accurate and robust performance of the classifier. However, this is somewhat contradicted by the physics point of view, where one would strive for as many bins as possible to ensure optimal spectrum resolution. Furthermore, annotated examples in astroparticle physics are generated in Monte Carlo simulation, which is computationally expensive.

As simulated events in astroparticle physics often extend below and above the energy range of interest for the analysis, an over-flow and an under-flow bin should be added to account for this type of event. These extra bins contain all events above and below the respective energy range of interest. If these bins were omitted, a classifier could easily overestimate the bin content of the outermost bins within the energy range of interest.

Figure 10.12 shows the optimization of the step size, using $\xi = 0.5$ as an example. The normalization s of α_k was varied in steps of 0.5, and the Wasserstein distance (WD) was computed for every s with respect to the true neutrino energy spectrum. While the true energy distribution is shown in black, spectra obtained for $s = 0.5$, $s = 1.0$ and $s = 1.5$ are depicted in blue, orange, and green, respectively and shown in the upper panel of Figure 10.12. The ratio of the estimated spectrum f_{est} to the true energy distribution f_{true} is depicted in the lower panel of Figure 10.12. The shaded areas in the highest and lowest bin highlight the over- and underflow bin, respectively. Inspecting the different settings and considering the computed Wasserstein distances, one finds that the neutrino energy spectrum is sufficiently well reproduced for $s = 1$.

In a next step, the optimized parameter settings are used in n separate applications of DSEA. The required n data sets are obtained from the original data set via bootstrapping. The obtained bin-wise distributions can then be used to verify the stability of the deconvolution process and to estimate statistical uncertainties. Due to the large number of bootstraps required for an accurate estimation of the statistical uncertainties, it is important to carefully consider the resource consumption (CPU time and memory) of each bootstrap and to ensure that each bootstrap is as resource-efficient as possible.

Often one finds that the distributions of the deconvolution results can be described by a normal distribution for all bins, indicating a stable algorithm performance. Devia-

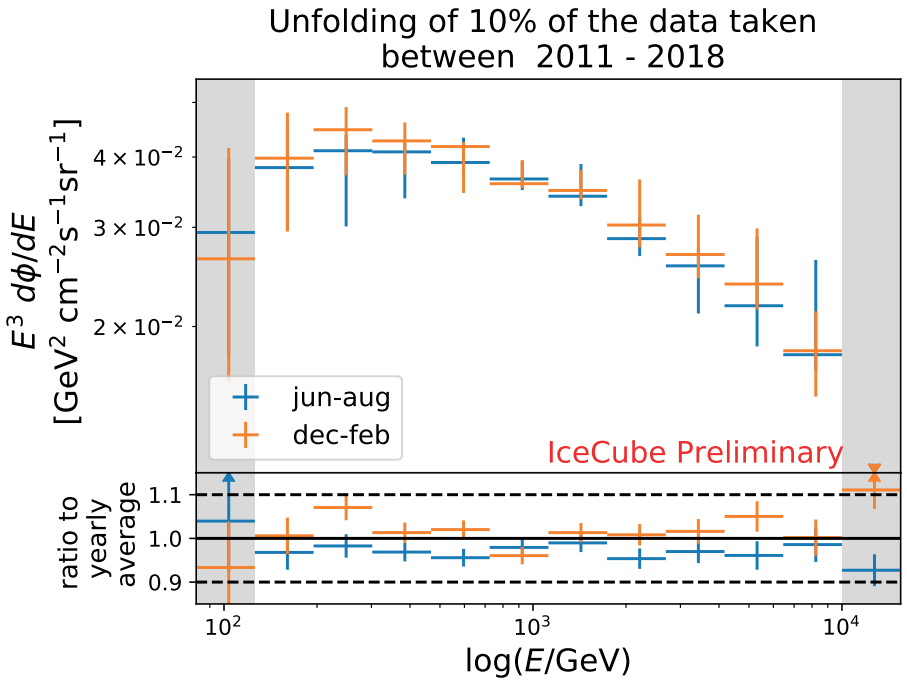


Fig. 10.13: Results of the presented analysis. The unfolding results, obtained on 10 % of the available data, are shown for summer (June to August) and winter (December to February) and compared with the yearly average. Figure taken from [210]

tions from a normal distribution or other unexpected or even unphysical properties would indicate an unstable performance, and the algorithmic settings should be revisited. A two-peak structure is a particularly straightforward example, and it indicates a non-optimal choice of the bins. This behavior can be observed in the highest energy bins, for cases where only one or two events populate these bins. The application of bootstrapping then results in these one or two events either being inside or outside the sample and might result in two or more peaks in the observed pull distribution. When all distributions—including the under- and overflow bin—indicate a stable behavior, one can fit the distributions to estimate the statistical uncertainties. After all verification steps are completed, the deconvolution can be applied to experimental data. As the deconvolution algorithm only returns the number of events per bin, this number needs to be converted into a neutrino flux. This flux is generally given in units of $1/(\mathrm{cm}^2\ \mathrm{GeV\ sr\ s})$. In a simple scenario, the deconvolution result needs to be divided by the measurement time, the solid angle covered in the measurement, and the width of the bin. The deconvolution result also needs to be divided by the so-called effective area A_{eff} to obtain a flux per area. As IceCube is a three-dimensional instrument, A_{eff}

cannot be trivially obtained from the area of the detector itself but has to be inferred from Monte Carlo simulations. Details on this procedure are given in [329] and [91].

Figure 10.13 shows the deconvolution results obtained in the analysis of seasonal variations presented here. The fluxes are depicted in the upper panel, and the results for summer (June to August) are shown in blue, whereas the results for winter (December to February) are shown in orange. As small-scale variations are often invisible to the naked eye in power-law spectra, the depicted neutrino flux is weighted by the third power of the energy. One finds that there is a certain disagreement between the two seasons, as expected due to the seasonal variations. However, a quantitative statement on the size of said disagreement can not be made due to the large uncertainties. The relatively large statistical uncertainties arise from the fact that the unfolding was carried out on only 10 % of the available data due to the IceCube blindness policy.[2] The lower panel of Figure 10.13 depicts the ratios of the obtained results to the expected yearly average. Again, a clear deviation between the two seasons and a clear deviation from the expected yearly average are observed.

The discussed analysis of seasonal variations is an example that shows how the novel interpretation of deconvolution as a classification task can be applied in practice. One finds that the approach can be utilized to even solve problems with a relatively large amount of smearing (see Figure 10.11). The application of the approach includes the selection of suitable input variables as well as the optimization of the algorithmic parameters (e.g., step size). It further includes the validation of the results and the estimation of statistical uncertainties.

2 The blindness policy requires analyzers to optimize their analyses on 10 % of the data (known as the burn sample).

Bibliography

[1] A. Aab et al. “Depth of maximum of air-shower profiles at the Pierre Auger Observatory. II. Composition implications”. In: *Physical Review D* 90.12 (2014), p. 122006 (cit. on p. 136).

[2] A. Aab et al. “Depth of Maximum of Air-Shower Profiles at the Pierre Auger Observatory: Measurements at Energies above $10^{17.8}$ eV”. In: *Physical Review D* 90.12 (2014), p. 122005 (cit. on p. 136).

[3] A. Aab et al. “Extraction of the muon signals recorded with the surface detector of the Pierre Auger Observatory using recurrent neural networks”. In: *Journal of Instrumentation* 16.07 (2021), P07016 (cit. on p. 134).

[4] A. Aab et al. “Measurement of the Fluctuations in the Number of Muons in Extensive Air Showers with the Pierre Auger Observatory”. In: *Physical Review Letters* 126.15 (2021), p. 152002 (cit. on p. 136).

[5] A. Aab et al. “Muons in Air Showers at the Pierre Auger Observatory: Mean Number in Highly Inclined Events”. In: *Physical Review D* 91.3 (2015). [Erratum: Phys.Rev.D 91, 059901 (2015)], p. 032003 (cit. on p. 137).

[6] M. Aaboud et al. “Observation of Higgs boson production in association with a top quark pair at the LHC with the ATLAS detector”. In: *Physics Letters B* 784 (2018), pp. 173–191 (cit. on p. 243).

[7] G. Aad et al. “Dijet resonance search with weak supervision using $\sqrt{s} = 13$ TeV pp collisions in the ATLAS detector”. In: *Physical Review Letters* 125.13 (2020), p. 131801 (cit. on p. 244).

[8] G. Aad et al. “Measurement of the t-channel single top-quark production cross section in pp collisions at $\sqrt{s} = 7$ TeV with the ATLAS detector”. In: *Physics Letters B* 717 (2012), pp. 330–350 (cit. on p. 243).

[9] G. Aad et al. “Search for pair production of scalar leptoquarks decaying into first- or second-generation leptons and top quarks in proton–proton collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector”. In: *European Physical Journal C* 81.4 (2021), p. 313 (cit. on p. 243).

[10] R. Aaij et al. “Measurement of CP violation in $B^0 \to D^{*\pm}D^{\mp}$ decays”. In: *Journal of High Energy Physics (JHEP)* 03 (2020), p. 147 (cit. on pp. 194, 196).

[11] R. Aaij et al. “Allen: A High-Level Trigger on GPUs for LHCb”. In: *Computing and Software for Big Science* 4 (Apr. 2020) (cit. on pp. 101, 102).

[12] R. Aaij et al. “First Measurement of Charm Production in its Fixed-Target Configuration at the LHC”. In: *Physical Review Letters* 122.13 (2019), p. 132002 (cit. on p. 142).

[13] R. Aaij et al. “First measurement of the CP-violating phase in $B_s^0 \to J/\psi(\to e^+e^-)\phi$ decays”. In: *arXiv: Physics* (2021). DOI: arXiv:2105.14738 (cit. on p. 104).

[14] R. Aaij et al. “Measurement of Antiproton Production in pHe Collisions at $\sqrt{s_{NN}} = 110$ GeV”. In: *Physical Review Letters* 121.22 (2018), p. 222001 (cit. on p. 142).

[15] R. Aaij et al. “Precision luminosity measurements at LHCb”. In: *Journal of Instrumentation* 9.12 (2014), P12005 (cit. on p. 142).

[16] M. G. Aartsen et al. “Development of a general analysis and unfolding scheme and its application to measure the energy spectrum of atmospheric neutrinos with IceCube”. In: *European Physical Journal C* 75 (Mar. 2015), p. 116. URL: https://link.springer.com/article/10.1140/epjc/s10052-015-3330-z (cit. on pp. 68, 167–169, 206, 207, 297).

[17] M. G. Aartsen et al. “Energy reconstruction methods in the IceCube neutrino telescope”. In: *Journal of Instrumentation* 9.3 (Mar. 2014), P03009. DOI: http://arxiv.org/pdf/1311.4767v3.pdforhttp://iopscience.iop.org/1748-0221/9/03/P03009/ (cit. on pp. 246, 252, 254).

https://doi.org/10.1515/9783110785968-011

[18] M. G. Aartsen et al. “Measurement of the ν_μ energy spectrum with IceCube-79”. In: *European Physical Journal C* 77 (Oct. 2017), p. 692. URL: https://link.springer.com/article/10.1140/epjc/s10052-017-5261-3 (cit. on pp. 68, 169, 206, 207, 209, 281).

[19] M. G. Aartsen et al. “Observation of High-Energy Astrophysical Neutrinos in Three Years of IceCube Data”. In: *Physical Review Letters* 113.10 (Sept. 2014), p. 101101 (cit. on p. 87).

[20] M. G. Aartsen et al. “Searches for relativistic magnetic monopoles in IceCube”. In: *European Physical Journal C* 76 (Mar. 2016), p. 133 (cit. on p. 88).

[21] M. G. Aartsen et al. “The IceCube Neutrino Observatory: instrumentation and online systems”. In: *Journal of Instrumentation* 12 (Mar. 2017), P03012 (cit. on pp. 35, 37, 85–89, 246).

[22] M. G. Aartsen et al. “The IceCube realtime alert system”. In: *Astroparticle Physics* 92 (June 2017), pp. 30–41 (cit. on pp. 88, 89).

[23] M. Aartsen et al. “Efficient propagation of systematic uncertainties from calibration to analysis with the SnowStorm method in IceCube”. In: *Journal of Cosmology and Astroparticle Physics (JCAP)* 10 (2019), p. 048 (cit. on p. 256).

[24] M. Aartsen et al. “Search for Sources of Astrophysical Neutrinos Using Seven Years of IceCube Cascade Events”. In: *The Astrophysical Journal* 886 (2019), p. 12 (cit. on p. 253).

[25] M. Aartsen et al. “South Pole glacial climate reconstruction from multi-borehole laser particulate stratigraphy”. In: *J. Glaciol.* 59.218 (2013), pp. 1117–1128 (cit. on p. 247).

[26] M. Aartsen et al. “Measurement of South Pole ice transparency with the IceCube LED calibration system”. In: *Nuclear Instruments and Methods in Physics Research A* 711 (May 2013), pp. 73–89. URL: http://www.sciencedirect.com/science/article/pii/S0168900213001460 (cit. on p. 165).

[27] M. Aartsen et al. “Search for non-relativistic magnetic monopoles with IceCube”. In: *The European Physical Journal C* 74.7 (2014), pp. 1–19 (cit. on p. 88).

[28] M. Abadi et al. “Tensorflow: Large-scale machine learning on heterogeneous distributed systems”. In: *arXiv: Computing Research Repository* (2016). DOI: arXiv:1603.04467 (cit. on p. 255).

[29] R. Abbasi et al. “A muon-track reconstruction exploiting stochastic losses for large-scale Cherenkov detectors”. In: *Journal of Instrumentation* 16.08 (2021), P08034 (cit. on p. 254).

[30] R. Abbasi et al. “Low Energy Event Reconstruction in IceCube DeepCore”. In: (Mar. 2022) (cit. on p. 254).

[31] R. Abbasi, M. Huennefeld, K. Morik, et al. “A convolutional neural network based cascade reconstruction for the IceCube Neutrino Observatory”. In: *Journal of Instrumentation* 16.07 (July 2021), P07041. DOI: https://doi.org/10.1088/1748-0221/16/07/p07041 (cit. on pp. 87, 247–253).

[32] R. Abbasi et al. “An improved method for measuring muon energy using the truncated mean of dE/dx”. In: *Nuclear Instruments and Methods in Physics Research A* 703 (Mar. 2013), pp. 190–198. URL: http://www.sciencedirect.com/science/article/pii/S0168900212014234 (cit. on pp. 166, 313).

[33] R. Abbasi et al. “IceTop: The surface component of IceCube.” In: *Nuclear Instruments and Methods in Physics Research A* 700 (2013), pp. 188–220 (cit. on p. 87).

[34] P. Abreu et al. “Constraining Lorentz Invariance Violation using the muon content of extensive air showers measured at the Pierre Auger Observatory”. In: *Proceeding of science* ICRC2021 (2021), p. 340 (cit. on p. 136).

[35] M. Ackermann et al. “Optical properties of deep glacial ice at the South Pole”. In: *Journal of Geophysical Research: Atmospheres* 111.D13 (2006) (cit. on p. 165).

[36] J. Adam et al. “Enhanced production of multi-strange hadrons in high-multiplicity proton-proton collisions”. In: *Nature Physics* 13 (2017), pp. 535–539 (cit. on pp. 134, 139, 140).

[37] S. Adrián-Martınez et al. “Letter of Intent for KM3NeT2.0”. In: *ArXiv: Physics* (Jan. 2016). DOI: arXiv:1601.07459 (cit. on p. 35).

[38] A. Aduszkiewicz et al. “Measurement of meson resonance production in π^-+ C interactions at SPS energies”. In: *European Physical Journal C* 77.9 (2017), p. 626 (cit. on p. 141).

[39] M. Ageron et al. “ANTARES: The first undersea neutrino telescope”. In: *Nuclear Instruments and Methods in Physics Research A* 656.1 (Nov. 2011), pp. 11–38 (cit. on p. 35).

[40] S. Agostinelli et al. “GEANT4: A Simulation toolkit”. In: *Nuclear Instruments and Methods in Physics Research* A506 (2003), pp. 250–303 (cit. on p. 103).

[41] A. M. Agrawal, A. Tendle, H. Sikka, S. Singh, and A. Kayid. “Investigating Learning in Deep Neural Networks using Layer-Wise Weight Change”. In: *Advances in Neural Information Processing Systems 33: Procs. of the 2020 Conference*. 2020. URL: https://www.researchgate.net/publication/353050395_Investigating_Learning_in_Deep_Neural_Networks_Using_Layer-Wise_Weight_Change (cit. on p. 75).

[42] J. A. Aguilar-Saavedra. “Top flavor-changing neutral interactions: Theoretical expectations and experimental detection”. In: *Acta Physica Polonica B* 35 (2004), pp. 2695–2710 (cit. on p. 231).

[43] F. Aharonian et al. “The Crab Nebula and Pulsar between 500 GeV and 80 TeV: Observations with the HEGRA Stereoscopic Air Cerenkov Telescopes”. In: *The Astrophysical Journal* 614.897 (2004) (cit. on p. 220).

[44] J. Ahrens et al. “Muon track reconstruction and data selection techniques in AMANDA”. In: *Nuclear Instruments and Methods in Physics Research A* 524 (May 2004), pp. 169–194 (cit. on pp. 165, 166).

[45] J.-M. Alameddine, J. Soedingrekso, A. Sandrock, M. Sackel, and W. Rhode. “PROPOSAL: A library to propagate leptons and high energy photons”. In: *Journal of Physics: Conference Series* 1690 (Dec. 2020), p. 012021 (cit. on p. 120). **SFB876-C3**

[46] J. Albrecht et al. “The Muon Puzzle in cosmic-ray induced air showers and its connection to the Large Hadron Collider”. In: (May 2021) (cit. on pp. 134, 136, 138, 139).

[47] J. Aleksić et al. “Insights into the emission of the blazar 1ES 1011+496 through unprecedented broadband observations during 2011 and 2012”. In: *ArXiv: Physics* (Mar. 2016). DOI: arXiv:1603.06776 (cit. on p. 40). **SFB876-C3**

[48] J. Aleksić et al. “The major upgrade of the MAGIC telescopes, Part II: A performance study using observations of the Crab Nebula”. In: *Astroparticle Physics* 72 (Jan. 2016), pp. 76–94 (cit. on pp. 40, 68). **SFB876-C3**

[49] J. Aleksić et al. “Measurement of the Crab Nebula spectrum over three decades in energy with the MAGIC telescopes”. In: *Journal of High Energy Astrophysics* 5 (2015), pp. 30–38 (cit. on p. 309).

[50] J. Allison, K. Amako, J. Apostolakis, H. Araujo, P. Dubois, et al. “Geant4 developments and applications”. In: *IEEE Transactions on Nuclear Science* 53 (2006), p. 270 (cit. on p. 103).

[51] A. A. Alves Jr., B. Spaan, and andere (LHCb Collaboration). “The LHCb Detector at the LHC”. In: *Journal of Instrumentation* 3 (2008), S08005 (cit. on p. 46).

[52] A. A. Alves Junior et al. “Status of the novel CORSIKA 8 air shower simulation framework”. In: *Proceedings of Science* ICRC2021 (2021), p. 284 (cit. on p. 120).

[53] H. Anderhub et al. “Design and operation of FACT - the first G-APD Cherenkov telescope”. In: *Journal of Instrumentation* 8.06 (June 2013), P06008. URL: http://iopscience.iop.org/1748-0221/8/06/P06008/ (cit. on pp. 41, 83, 187, 269, 276).

[54] B. Andersson, G. Gustafson, G. Ingelman, and T. Sjöstrand. “Parton fragmentation and string dynamics”. In: *Physics Reports* 97.2 (1983), pp. 31–145. URL: https://www.sciencedirect.com/science/article/pii/0370157383900807 (cit. on p. 103).

[55] S. Ansoldi et al. “The blazar TXS 0506+ 056 associated with a high-energyneutrino: insights into extragalactic jets and cosmic-ray acceleration”. In: *The Astrophysical Journal Letters* 863.1 (2018), p. L10 (cit. on p. 40).

[56] I. Antcheva et al. “ROOT - A C++ framework for petabyte data storage, statistical analysis and visualization”. In: *Computer Physics Communications* 180 (Dec. 2009), pp. 2499–2512 (cit. on pp. 95, 296).

[57] R. Antunes-Nobrega et al. “LHCb TDR computing technical design report”. In: (June 2005) (cit. on p. 90).

[58] D. Arpit et al. “A Closer Look at Memorization in Deep Networks”. In: *Procs. of the Int. Conference on Machine Learning 2017*. 2017. DOI: arxiv:1706.05394 (cit. on p. 74).

[59] ATLAS Collaboration. “Combination of the searches for pair-produced vector-like partners of the third-generation quarks at $\sqrt{s} = 13$ TeV with the ATLAS detector”. In: *Physical Review Letters* 121 (2018), p. 211801 (cit. on p. 242).

[60] ATLAS Collaboration. *Deep Sets based Neural Networks for Impact Parameter Flavour Tagging in ATLAS*. Tech. rep. ATL-PHYS-PUB-2020-014. 2020. URL: https://cds.cern.ch/record/2718948 (cit. on p. 243).

[61] ATLAS Collaboration. *Fast simulation of the ATLAS calorimeter system with Generative Adversarial Networks*. Tech. rep. ATL-SOFT-PUB-2020-006. 2020. URL: https://cds.cern.ch/record/2746032 (cit. on p. 244).

[62] ATLAS Collaboration. “Identification of boosted Higgs bosons decaying into b-quark pairs with the ATLAS detector at 13 TeV”. In: *The European Physical Journal C* 79 (2019), p. 836 (cit. on p. 238).

[63] ATLAS Collaboration. “Identification of high transverse momentum top quarks in pp collisions at $\sqrt{s} = 8$ TeV with the ATLAS detector”. In: *Journal of High Energy Physics* 06 (2016), p. 093 (cit. on p. 238).

[64] ATLAS Collaboration. *Identification of Jets Containing b-Hadrons with Recurrent Neural Networks at the ATLAS Experiment*. Tech. rep. ATL-PHYS-PUB-2017-003. 2017. URL: https://cds.cern.ch/record/2255226 (cit. on p. 243).

[65] ATLAS Collaboration. “Performance of top-quark and W-boson tagging with ATLAS in Run 2 of the LHC”. In: *The European Physical Journal C* 79 (2019), p. 375 (cit. on p. 238).

[66] ATLAS Collaboration. *Quark versus Gluon Jet Tagging Using Jet Images with the ATLAS Detector*. Tech. rep. ATL-PHYS-PUB-2017-017. 2017. URL: https://cds.cern.ch/record/2275641 (cit. on p. 243).

[67] ATLAS Collaboration. “Search for flavour-changing neutral currents in processes with one top quark and a photon using 81 fb^{-1} of pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS experiment”. In: *Physics Letters B* 800 (2020), p. 135082 (cit. on pp. 231–234, 236).

[68] ATLAS Collaboration. *Search for flavour-changing neutral-current couplings between the top-quark and the photon with the ATLAS detector at $\sqrt{s} = 13$ TeV*. Tech. rep. ATLAS-CONF-2022-003. 2022. URL: https://cds.cern.ch/record/2802004 (cit. on p. 243).

[69] ATLAS Collaboration. “Search for pair production of heavy vector-like quarks decaying into hadronic final states in pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector”. In: *Physical Review D* 98 (2018), p. 092005 (cit. on p. 238).

[70] ATLAS Collaboration. *Search for pair-production of vector-like quarks in pp collision events at $\sqrt{s} = 13$ TeV with at least one leptonically-decaying Z boson and a third-generation quark with the ATLAS detector*. Tech. rep. ATLAS-CONF-2021-024. 2021. URL: https://cds.cern.ch/record/2773300 (cit. on pp. 237, 240–243).

[71] ATLAS Collaboration. “The ATLAS Experiment at the CERN Large Hadron Collider”. In: *Journal of Instrumentation* 3 (2008), S08003 (cit. on p. 229).

[72] A. Avrorin et al. “BAIKAL-GVD: The New-Generation Neutrino Telescope in Lake Baikal”. In: *Bulletin of the Russian Academy of Sciences: Physics* 83.8 (2019), pp. 921–922 (cit. on p. 35).

[73] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook*. New York, NY, USA: Cambridge University Press, 2003 (cit. on p. 5).

[74] T. Back, U. Hammel, and H.-P. Schwefel. “Evolutionary computation: Comments on the history and current state”. In: *IEEE Transactions on Evolutionary Computation* 1.1 (1997), pp. 3–17 (cit. on p. 53).

[75] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli. “wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations”. In: *Advances in Neural Information Processing Systems 33: Procs. of the 2020 Conference*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Curran Associates, Inc., 2020 (cit. on p. 53).

[76] C. Barschel et al. *LHC fixed target experiments : Report from the LHC Fixed Target Working Group of the CERN Physics Beyond Colliders Forum*. Vol. 4/2020. CERN Yellow Reports: Monographs. Geneva: CERN, 2020 (cit. on p. 142).

[77] S. Baur, H. Dembinski, M. Perlin, T. Pierog, R. Ulrich, and K. Werner. “Core-corona effect in hadron collisions and muon production in air showers”. In: (Feb. 2019) (cit. on pp. 134, 137, 140).

[78] S. Bavikadi, P. R. Sutradhar, K. N. Khasawneh, A. Ganguly, and S. M. Pudukotai Dinakarrao. “A Review of In-Memory Computing Architectures for Machine Learning Applications”. In: *Procs. of the Great Lakes Symposium on VLSI 2020*. ACM, 2020, pp. 89–94. DOI: https://doi.org/10.1145/3386263.3407649 (cit. on p. 63).

[79] W. Becker and B. Aschenbach. “OSAT HRI observations of the Crab pulsar: An improved temperature upper limit for PSR 0531+21”. In: *The Lives of Neutron Stars* (1995) (cit. on p. 220).

[80] N. Beckmann, H.-P. Kriegel, and B. Schneider Ralf und Seeger. “The R*-tree: An efficient and robust access method for points and rectangles”. In: *Procs. of the ACM SIGMOD Int. Conference on Management of Data 1990*. ACM Press, 1990, pp. 322–331 (cit. on p. 155).

[81] I. Bello et al. “Revisiting ResNets: Improved Training and Scaling Strategies”. In: *arXiv: Computing Research Repository* (Mar. 2021). DOI: arXiv:2103.07579 (cit. on p. 73).

[82] K. S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. “When Is ”Nearest Neighbor” Meaningful?” In: *Procs. of the Int. Conference on Database Theory 1999*. ICDT ’99. London, UK: Springer-Verlag, 1999, pp. 217–235 (cit. on p. 179).

[83] A. Bifet et al. “MOA: A Real-Time Analytics Open Source Framework”. In: *Procs. of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2011*. Springer, 2011, pp. 617–620. DOI: 10.1007/978-3-642-23808-6_41 (cit. on p. 56).

[84] A. Biland et al. “Calibration and performance of the photon sensor response of FACT – the first G-APD Cherenkov telescope”. In: *Journal of Instrumentation* 9.10 (2014), P10012. URL: http://stacks.iop.org/1748-0221/9/i=10/a=P10012 (cit. on pp. 187, 276).

[85] V. Blobel. “An Unfolding Method for High Energy Physics Experiments”. In: *ArXiv: High Energy Physics - Experiment* (Aug. 2002). DOI: arXiv:hep-ex/0208022 (cit. on p. 295).

[86] V. Blobel. *Unfolding methods in high-energy physics experiments*. Tech. rep. CERN, 1985 (cit. on p. 295).

[87] C. Bockermann. “Mining Big Data Streams for Multiple Concepts”. PhD thesis. TU Dortmund University, 2015. URL: https://eldorado.tu-dortmund.de/handle/2003/34363 (cit. on pp. 56, 172).

[88] C. Bockermann et al. “Online Analysis of High-Volume Data Streams in Astroparticle Physics”. In: *Procs. of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2015*. Honored with the Best Paper Award

of the Industrial Track. Springer Berlin Heidelberg, 2015 (cit. on pp. 56, 113, 115, 172, 269). **SFB876-C3**

[89] S. Borkar and A. A. Chien. “The Future of Microprocessors”. In: *Communications of the ACM* 54.5 (May 2011), pp. 67–77 (cit. on pp. 120, 121).

[90] M. Börner, T. Hoinka, M. Meier, T. Menne, W. Rhode, and K. Morik. “Measurement/Simulation Mismatches and Multivariate Data Discretization in the Machine Learning Era”. In: *Procs. of the Astronomical Data Analysis Software and Systems Conference 2019*. Ed. by P. Ballester, J. Ibsen, M. Solar, and K. Shortridge. Vol. 522. ASP Conference Series. Astronomical Society of the Pacific. 2019, pp. 431–434 (cit. on pp. 130, 215, 295).

[91] M. Börner. “Bestimmung des Energiespektrums von atmosphärischen Myonneutrinos mit 3 Jahren Daten des IceCube-Detektors”. PhD thesis. TU Dortmund University, 2018 (cit. on pp. 130, 167–169, 206, 207, 209–211, 317).

[92] L. Bottou. “Online Algorithms and Stochastic Approximations”. In: *Online Learning and Neural Networks*. Cambridge, UK: Cambridge University Press, 1998 (cit. on p. 53).

[93] L. Bottou, F. E. Curtis, and J. Nocedal. “Optimization Methods for Large-Scale Machine Learning”. In: *SIAM Review* 60.2 (2018), pp. 223–311 (cit. on pp. 61, 62).

[94] X. Bouthillier et al. “Accounting for Variance in Machine Learning Benchmarks”. In: (2021). DOI: arXiv:2103.03098 (cit. on p. 278).

[95] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004 (cit. on p. 53).

[96] L. Breiman. “Bagging Predictors”. In: *Machine Learning* 24.2 (Aug. 1996), pp. 123–140. DOI: https://doi.org/10.1023/A:1018054314350 (cit. on p. 59).

[97] L. Breiman. “Random Forests”. In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32 (cit. on pp. 57, 58, 67, 130, 206).

[98] T. Bretz and D. Dorner. “MARS - CheObs goes Monte Carlo”. In: *Procs. of the Int. Cosmic Ray Conference 2009*. 2009 (cit. on p. 151).

[99] R. Bruce, M. A. Jebramcik, J. M. Jowett, T. Mertens, and M. Schaumann. “Performance and luminosity models for heavy-ion operation at the CERN Large Hadron Collider”. In: *The European Physical Journal - Plus* 136.7 (2021), p. 745 (cit. on p. 138).

[100] Bruegge, K. and Adam, J. and Ahnen, M. and Baack, D. and Balbo, M. and Bergmann, M. and Biland, A. and Bockermann, C. and Buss, J. and Blank, M. and Bretz, T. and Dmytriiev, A. and Dorner, D. and Egorov, A. and Einecke, S. and Hildebrand, D. and Hughes, G. and Linhoff, L. and Mannheim, K. and others. “Real Time Streaming Analysis of IACT Data”. In: *Procs. of the Astronomical Data Analysis Software and Systems Conference 2019*. Ed. by K. S. Marco Molinaro and F. Pasian. Vol. 521. Astronomical Society of the Pacific, 2019, p. 335. URL: http://www.aspbooks.org/a/volumes/article_details/?paper_id=39026 (cit. on p. 221). **SFB876-C3**

[101] K. Brügge. “Unmasking the gamma-ray sky: comprehensive and reproducible analysis for Cherenkov telescopes”. PhD thesis. TU Dortmund University, 2019. DOI: 10.17877/DE290R-20440 (cit. on pp. 219, 226).

[102] M. Bulinski. “OBASS – Observed Background and Artificial Signal Superposition”. MA thesis. TU Dortmund University, 2018 (cit. on p. 128).

[103] M. Bunse and K. Morik. “Active Class Selection with Uncertain Deployment Class Proportions”. In: *Procs. of the Workshop on Interactive Adaptive Learning 2021*. To appear. CEUR Workshop Proceedings, 2021 (cit. on pp. 113, 115). **SFB876-C3**

[104] M. Bunse and K. Morik. “Certification of Model Robustness in Active Class Selection”. In: *Procs. of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2021*. To appear. Springer, 2021. URL: https://2021.ecmlpkdd.org/wp-content/uploads/2021/07/sub_598.pdf (cit. on pp. 115, 116). **SFB876-C3**

[105] M. Bunse, N. Piatkowski, T. Ruhe, W. Rhode, and K. Morik. “Unification of Deconvolution Algorithms for Cherenkov Astronomy”. In: *Procs. of the Int. Conference on Data Science and Advanced Analytics 2018*. 2018. URL: https://ieeexplore.ieee.org/document/8631454 (cit. on p. 301). **SFB876-C3, SFB876-A1**

[106] M. Bunse, D. Weichert, A. Kister, and K. Morik. “Optimal Probabilistic Classification in Active Class Selection”. In: *Procs. of the Int. Conference on Data Mining 2020*. IEEE, 2020, pp. 942–947. URL: https://ieeexplore.ieee.org/abstract/document/9338430 (cit. on p. 114). **SFB876-C3**

[107] S. Buschjäger and K. Morik. “Decision Tree and Random Forest Implementations for Fast Filtering of Sensor Data”. In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 65-I.1 (Jan. 2018), pp. 209–222. DOI: https://doi.org/10.1109/TCSI.2017.2710627 (cit. on pp. 63, 69).

[108] S. Buschjäger, K. Morik, and M. Schmidt. “Summary Extraction on Data Streams in Embedded Systems”. In: *Procs. of the ECML Workshop on IoT Large Scale Learning From Data Streams 2017*. 2017. URL: http://ceur-ws.org/Vol-1958/IOTSTREAMING3.pdf (cit. on p. 182). **SFB876-A1**

[109] S. Buschjäger, L. Pfahler, J. Buss, K. Morik, and W. Rhode. “On-Site Gamma-Hadron Separation with Deep Learning on FPGAs”. In: *Procs. of the European Conference on Machine Learning and Knowledge Discovery in Databases 2020*. Springer, 2020. URL: https://link.springer.com/content/pdf/10.1007%5C%2F978-3-030-67667-4_29.pdf (cit. on pp. 268, 271). **SFB876-A1, SFB876-C3**

[110] S. Buschjäger et al. “Margin-Maximization in Binarized Neural Networks for Optimizing Bit Error Tolerance”. In: *Procs. of the Design, Automation and Test in Europe Conference 2021*. 2021. URL: https://ls12-www.cs.tu-dortmund.de/daes/media/documents/publications/downloads/2021dateyayla.pdf (cit. on pp. 63, 75, 187). **SFB876-A1**

[111] J. B. Buß. “Bad Moon Rising?” 1.0. PhD Thesis. TU Dortmund University, Dec. 2020. DOI: 10.17877/DE290R-21918 (cit. on p. 128). **SFB876-C3**

[112] A. Butter et al. “The Machine Learning landscape of top taggers”. In: *SciPost Physics* 7 (2019). Ed. by G. Kasieczka and T. Plehn, p. 014 (cit. on p. 238).

[113] M. Cakmak and A. L. Thomaz. “Designing Robot Learners That Ask Good Questions”. In: *Procs. of the Int. Conference on Human-Robot Interaction 2012*. ACM, 2012, pp. 17–24 (cit. on p. 114).

[114] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. “Deep clustering for unsupervised learning of visual features”. In: *Procs. of the European Conference on Computer Vision 2018*. Springer, 2018. DOI: https://doi.org/10.1007/987-3-030-01264-9_9 (cit. on p. 57).

[115] B. W. Carroll and D. A. Ostlie. *An introduction to modern astrophysics*. Cambridge University Press, 2017 (cit. on p. 269).

[116] L. Cazon. “Working Group Report on the Combined Analysis of Muon Density Measurements from Eight Air Shower Experiments”. In: *Proceeding of science* ICRC2019 (2020), p. 214 (cit. on p. 135).

[117] P. Chapman, J. Clinton, T. Khabaza, T. Reinartz, and R. Wirth. *The CRISP–DM Process Model*. Tech. rep. Mar. 1999 (cit. on p. 27).

[118] S. Chatterjee. “Coherent Gradients: An Approach to Understanding Generalization in Gradient Descent-based Optimizatio”. In: *Procs. International Conference on Learning Representations*. 2019 (cit. on p. 74).

[119] T. Chen and C. Guestrin. “Xgboost: A scalable tree boosting system”. In: *Procs. of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2016*. ACM. 2016, pp. 785–794 (cit. on p. 69).

[120] Cherenkov Telescope Array Consortium et al. “Science with the Cherenkov Telescope Array”. In: *arXiv: Physics* (Sept. 2017). DOI: arXiv:1709.07997 (cit. on p. 41).

[121] L. Chiapetti et al. *Definition of the Flexible Image Transport System (FITS)*. 2016 (cit. on p. 95).

[122] D. Chirkin. “Event reconstruction in IceCube based on direct event re-simulation”. In: *Procs. of the Int. Cosmic Ray Conference 2013*. 2013, p. 0581 (cit. on p. 254).

[123] J. Christenson, J. Cronin, V. Fitch, and R. Turlay. “Evidence for the 2π Decay of the K_2^0 Meson”. In: *Physical Review Letters* 13 (1964), pp. 138–140. DOI: 10.1103/PhysRevLett.13.138 (cit. on p. 260).

[124] C.-Y. Chuang, A. Torralba, and S. Jegelka. “Estimating Generalization under Distribution Shifts via Domain-Invariant Representations”. In: *Procs. of the Int. Conference on Machine Learning 2020*. PMLR, 2020. URL: http://proceedings.mlr.press/v119/chuang20a/chuang20a.pdf (cit. on p. 73).

[125] Z. Citron et al. “Report from Working Group 5: Future physics opportunities for high-density QCD at the LHC with heavy-ion and proton beams”. In: *CERN Yellow Reports: Monographs* 7 (2019). Ed. by A. Dainese, M. Mangano, A. B. Meyer, A. Nisati, G. Salam, and M. A. Vesterinen, pp. 1159–1410 (cit. on p. 138).

[126] W. F. Clocksin and C. S. Mellish. *Programming in Prolog*. Berlin: Springer, 2003 (cit. on p. 6).

[127] D. Cohn, R. Caruana, and A. McCallum. *Semi-Supervised Clustering with User Feedback*. Tech. rep. TR2003-1892. Cornell University, 2000 (cit. on p. 53).

[128] L. Collaboration. *The Boole Project*. 2020. URL: http://lhcbdoc.web.cern.ch/lhcbdoc/boole/ (cit. on p. 103).

[129] G. Cormode. “Continuous Distributed Monitoring: A Short Survey”. In: *Procs. of the Int. Workshop on Algorithms and Models for Distributed Event Processing 2011*. ACM, 2011, pp. 1–10 (cit. on p. 63).

[130] G. Cormode and S. Muthukrishnan. “An improved data stream summary: the count-min sketch and its applications”. In: *Journal of Algorithms* 55.1 (Apr. 2005), pp. 58–75 (cit. on p. 63).

[131] G. Cowan, K. Cranmer, E. Gross, and O. Vitells. “Asymptotic formulae for likelihood-based tests of new physics”. In: *The European Physical Journal C* 71 (2011). [Erratum: Eur. Phys. J. C 73 (2013) 2501], p. 1554 (cit. on p. 230).

[132] F. Croce and M. Hein. “Provable robustness against all adversarial l_p-perturbations for $p >= 1$”. In: *Procs. of the Int. Conference on Learning Representation 2020*. ICLR 2020 Conference Blind Submission, NeurIPS 2019 Workshop "Machine Learning with Guarantees". 2020. URL: https://openreview.net/forum?id=rklk_ySYPB (cit. on p. 73).

[133] N. Davidson, T. Przedzinski, and Z. Was. “PHOTOS Interface in C++: Technical and Physics Documentation”. In: (2010) (cit. on p. 103).

[134] A. P. Dawid. “The well-calibrated Bayesian”. In: *Journal of the American Statistical Association* 77.379 (1982), pp. 605–610 (cit. on p. 302).

[135] J. Dean and S. Ghemawat. “MapReduce: Simplified Data Processing on Large Clusters”. In: *Procs. of the Symposium on Operating System Design and Implementation 2004*. USENIX Association, 2004 (cit. on p. 56).

[136] J. Dean and S. Ghemawat. “MapReduce: Simplified Data Processing on Large Clusters”. In: *Communications of the ACM* 51.1 (2008), pp. 107–113 (cit. on p. 156).

[137] K. L. DeHolton. “Low energy event classification in IceCube using boosted decision trees”. In: *Journal of Instrumentation* 16.12 (2021), p. C12007 (cit. on p. 134).

[138] C. Deil et al. *Data formats for gamma-ray astronomy - version 0.2*. 2018. DOI: 10.5281/zenodo.1409831 (cit. on pp. 95, 305).

[139] C. Deil et al. “Gammapy - A prototype for the CTA science tools”. In: *Procs. of the Int. Cosmic Ray Conference 2017*. 766. Proceedings of Science, 2017 (cit. on p. 95).

[140] G. DeJong and R. Mooney. “Explanation-Based-Learning: An Alternative View”. In: *Machine Learning* 2.1 (1986), pp. 145–176 (cit. on p. 4).

[141] H. P. Dembinski et al. “Report on Tests and Measurements of Hadronic Interaction Properties with Air Showers”. In: *EPJ Web of Conferences* 210 (2019). Ed. by I. Lhenry-Yvon, J. Biteau, O. Biteau, and P. Ghia, p. 02004 (cit. on p. 135).

[142] A. P. Dempster. “A generalization of Bayesian inference”. In: *Journal of the Royal Statistical Society* 30 (1968), pp. 205–247 (cit. on p. 8).

[143] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. “Imagenet: A large-scale hierarchical image database”. In: *Procs. of the Conference on Computer Vision and Pattern Recognition 2009*. 2009 (cit. on p. 73).

[144] C. H. Q. Ding and H. Peng. “Minimum Redundancy Feature Selection from Microarray Gene Expression Data”. In: *Procs. of the IEEE Computer Society Bioinformatics Conference 2003*. Vol. 0. Los Alamitos, CA, USA: IEEE Computer Society, 2003, pp. 523–529 (cit. on pp. 59, 169).

[145] A. Djouadi and A. Lenz. “Sealing the fate of a fourth generation of fermions”. In: *Physics Letters B* 715 (2012), pp. 310–314 (cit. on p. 236).

[146] P. Domingos and G. Hulten. “Mining high-speed data streams”. In: *Procs. of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2000*. New York, NY, USA: ACM, 2000, pp. 71–80 (cit. on p. 67).

[147] R. D. Drew. “Deep Unsupervised Domain Adaptation for Gamma-Hadron Separation”. MA thesis. TU Dortmund University, 2021 (cit. on p. 271).

[148] M. Dunsch, J. Soedingrekso, A. Sandrock, M. Meier, T. Menne, and W. Rhode. “Recent improvements for the lepton propagator PROPOSAL”. In: *Computer Physics Communications* 242 (Sept. 2019), pp. 132–144. DOI: http://dx.doi.org/10.1016/j.cpc.2019.03.021 (cit. on pp. 16, 20, 136). **SFB876-C3**

[149] A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant. “A general lower bound on the number of examples needed for learning”. In: *Information and Computation* 82.3 (1989), pp. 247–261. URL: https://www.sciencedirect.com/science/article/pii/0890540189900023 (cit. on p. 66).

[150] W. Emde. “An Inference Engine for Representing Multiple Theories”. In: *Knowledge Representation and Organization in Machine Learning*. Ed. by K. Morik. auch: KIT-Report Nr. 64, TU Berlin, 1988. New York, Berlin, Tokio: Springer, 1989, pp. 148–176 (cit. on p. 7).

[151] W. Emde, C. U. Habel, and C.-R. Rollinger. “The Discovery of the Equator or Concept Driven Learning”. In: *Procs. of the Int. Joint Conference on Artificial Intelligence*. Los Altos, CA: Morgan Kaufman, 1983, pp. 455–458 (cit. on p. 7).

[152] R. Enberg, M. H. Reno, and I. Sarcevic. “Prompt neutrino fluxes from atmospheric charm”. In: *Physical Review D* 78 (Aug. 2008), p. 043005 (cit. on p. 208).

[153] R. Engel et al. “Towards A Next Generation of CORSIKA: A Framework for the Simulation of Particle Cascades in Astroparticle Physics”. In: *Computing and Software for Big Science* 3 (Dec. 2018) (cit. on p. 119).

[154] K. Eykholt et al. “Physical Adversarial Examples for Object Detectors”. In: *arXiv: Computing Research Repository* (Oct. 2018). DOI: arXiv:1807.07769 (cit. on p. 73).

[155] B. Falkenburg. “From Waves to Particle Tracks and Quantum Probabilities”. In: *From Ultra Rays to Astroparticles*. Ed. by B. Falkenburg and W. Rhode. 2012, p. 265 (cit. on p. 16).

[156] B. Falkenburg. “Pragmatic Unification, Observation and Realism in Astroparticle Physics”. In: *Journal for General Philosophy of Science* 43.2 (2012), pp. 327–345 (cit. on p. 33).

[157] B. Falkenburg and W. Rhode. *From Ultra Rays to Astroparticles*. Ed. by B. Falkenburg and W. Rhode. Dordrecht, Netherlands: Springer, 2012 (cit. on p. 32).

[158] B. C. Falkenhainer and R. S. Michalski. "Integrating Quantitative and Qualitative Discovery: The ABACUS System". In: *Machine Learning* 1 (1986), pp. 367–401 (cit. on p. 4).

[159] A. Fedynitch, R. Engel, T. K. Gaisser, F. Riehn, and T. Stanev. "Calculation of conventional and prompt lepton fluxes at very high energy". In: *EPJ Web of Conferences* 99 (2015). Ed. by D. Berge, A. de Roeck, M. Mangano, and B. Pattison, p. 08001 (cit. on p. 136).

[160] U. Feige. "A Threshold of Ln N for Approximating Set Cover". In: *Journal of the Association for Computing Machinery* 45.4 (July 1998), pp. 634–652. DOI: http://doi.acm.org/10.1145/285055.285059 (cit. on p. 180).

[161] R. C. Fernandez et al. "Liquid: Unifying Nearline and Offline Big Data Integration". In: *Procs. of the Conference on Innovative Data Systems Research 2015*. 2015 (cit. on p. 56).

[162] M. Fey, J. E. Lenssen, F. Weichert, and H. Müller. "SplineCNN: Fast Geometric Deep Learning with Continuous B-Spline Kernels". In: *Procs. of the IEEE Conference on Computer Vision and Pattern Recognition 2018*. 2018. DOI: arXiv:1711.08920 (cit. on p. 29). **SFB876-B2, SFB876-A6**

[163] *FIMI '04, Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, Brighton, UK, November 1, 2004*. Vol. 126. CEUR Workshop Proceedings, 2004. URL: http://ceur-ws.org/Vol-126/ (cit. on p. 63).

[164] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. "An analysis of approximations for maximizing submodular set functions—II". In: *Polyhedral combinatorics* (1978), pp. 73–87 (cit. on p. 180).

[165] V. P. Fomin, A. A. Stepanian, R. C. Lamb, D. A. Lewis, M. Punch, and T. C. Weekes. "New methods of atmospheric Cherenkov imaging for gamma-ray astronomy. I. The false source method". In: *Astroparticle Physics* 2.2 (1994), pp. 137–150 (cit. on p. 276).

[166] G. Forman. "Quantifying counts and costs via classification". In: *Data Mining and Knowledge Discovery* 17.2 (2008), pp. 164–206 (cit. on p. 300).

[167] S. Fort, P. Nowak, S. Jastrzebski, and S. Narayanan. "Stiffness: A New Perspective on Generalization in Neural Networks". In: *Procs. of the Int. Conference on Representation Learning 2020*. 2020. DOI: arXiv:1901.09491 (cit. on p. 74).

[168] R. J. Fox and M. W. Dimmic. "A two-sample Bayesian t-test for microarray data". In: *BMC Bioinformatics* 7.126 (2006) (cit. on p. 57).

[169] Y. Freund and R. E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting". In: *Journal of Computer and System Sciences* 55.1 (1997), pp. 119–139. URL: http://www.research.att.com/~schapire/papers/FreundSc95.ps.Z (cit. on pp. 68, 197, 198).

[170] T. Fuchs and et al. "Development of a Machine Learning Based Analysis Chain for the Measurement of Atmospheric Muon Spectra with IceCube". In: *ArXiv: Physics* (2017). DOI: arXiv:1701.04067 (cit. on pp. 169, 313).

[171] T. Fuchs. "Charmante Myonen im Eis". PhD thesis. TU Dortmund University, 2016. URL: https://eldorado.tu-dortmund.de/handle/2003/35194 (cit. on pp. 169, 313).

[172] M. Füßling et al. "Status of the array control and data acquisition system for the Cherenkov Telescope Array". In: *Procs., Software and Cyberinfrastructure for Astronomy IV*. Ed. by G. Chiozzi and J. C. Guzman. Vol. 9913. International Society for Optics and Photonics. SPIE, 2016, pp. 1220–1231 (cit. on p. 83).

[173] T. K. Gaisser, R. Engel, and E. Resconi. *Cosmic rays and particle physics*. Cambridge University Press, 2016 (cit. on p. 311).

[174] Y. Gal and Z. Ghahramani. “Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning”. In: *Procs. of the Int. Conference on Machine Learning 2016*. JMLR, 2016. URL: http://proceedings.mlr.press/v48/gal16.pdf (cit. on pp. 8, 74).

[175] A. Gazizov and M. Kowalski. “ANIS: High energy neutrino generator for neutrino telescopes”. In: *Computer Physics Communications* 172 (Nov. 2005), pp. 203–213 (cit. on p. 207).

[176] A. Ghaffari and Y. Savaria. “CNN2Gate: An Implementation of Convolutional Neural Networks Inference on FPGAs with Automated Design Space Exploration”. In: *Electronics* 9 (2020). DOI: doi:10.3390/electronics9122200 (cit. on pp. 63, 75).

[177] L. H. Gilpin, C. Testart, N. Fruchter, and J. Adebayo. “Explaining Explanations to Society”. In: *Advances in Neural Information Processing Systems 31: Procs. of the Workshop on Ethical, Social and Governance Issues in AI 2018*. 2018, pp. 1–6. DOI: arXiv:1901.06560 (cit. on p. 74).

[178] S. L. Glashow, J. Iliopoulos, and L. Maiani. “Weak Interactions with Lepton-Hadron Symmetry”. In: *Physical Review D* 2 (1970), pp. 1285–1292 (cit. on p. 231).

[179] D. Göddeke et al. “Energy efficiency vs. performance of the numerical solution of PDEs: An application study on a low-power ARM-based cluster”. In: *Journal of Computational Physics* 237 (2013), pp. 132–150. DOI: https://doi.org/10.1016/j.jcp.2012.11.031 (cit. on p. 122).

[180] P. Golonka and Z. Was. “PHOTOS Monte Carlo: A Precision tool for QED corrections in Z and W decays”. In: *European Physical Journal* C45 (2006), pp. 97–107 (cit. on p. 103).

[181] N. Golowich, A. Rakhlin, and O. Shamir. “Size-Independent Sample Complexity of Neural Networks”. In: *Procs. of the Conference On Learning Theory 2018*. Ed. by S. Bubeck, V. Perchet, and P. Rigollet. Proceedings of Machine Learning Research. PMLR. PMLR, 2018, pp. 297–299. URL: https://proceedings.mlr.press/v75/golowich18a.html (cit. on p. 74).

[182] P. González, A. Castaño, N. V. Chawla, and J. J. del Coz. “A Review on Quantification Learning”. In: *ACM Computing Surveys* 50.5 (2017), 74:1–74:40 (cit. on p. 300).

[183] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.org (cit. on pp. 58, 71, 179).

[184] M. D. Görtz et al. “Energy Efficiency of a Low Power Hardware Cluster for High Performance Computing”. In: *Procs. of INFORMATIK 2017*. Ed. by M. Eibl and M. Gaedke. Gesellschaft für Informatik, Bonn, Sept. 2017, pp. 2537–2548. URL: https://dl.gi.de/handle/20.500.12116/4032 (cit. on pp. 123, 125).

[185] A. B. Graf and S. Borer. “Normalization in support vector machines”. In: *Pattern Recognition: Procs. of the DAGM Symposium 2001*. Springer. 2001, p. 277 (cit. on p. 183).

[186] D. Guberman and R. Paoletti. “Silicon photomultipliers in Very High Energy gamma-ray astrophysics”. In: *Journal of Instrumentation* 15.05 (2020), p. C05039 (cit. on p. 83).

[187] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. “A Survey of Methods for Explaining Black Box Models”. In: *ACM Computing Surveys* 51.5 (Aug. 2018). DOI: https://doi.org/10.1145/3236009 (cit. on p. 74).

[188] C. Haack, L. Lu, and T. Yuan. “Improving the directional reconstruction of PeV hadronic cascades in IceCube”. In: *EPJ Web Conference* 207 (2019). Ed. by C. Spiering, p. 05003 (cit. on p. 254).

[189] M. Hall, E. Frank, G. Holmes, P. Pfahringer Bernhard and Reutemann, and I. H. Witten. “The WEKA Data Mining Software: An Update”. In: *SIGKDD Explorations Newsletter* 11.1 (Nov. 2009), pp. 10–18. DOI: http://doi.acm.org/10.1145/1656274.1656278 (cit. on p. 206).

[190] M. A. Hall. “Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning”. In: *Procs. of the Int. Conference on Machine Learning 2000*. Ed. by P. Langley. Morgan Kaufmann Publishers Inc., 2000, pp. 359–366 (cit. on pp. 58, 169).

[191] S. Hanneke. “The optimal sample complexity of PAC learning”. In: *Journal of Machine Learning Research* 17.1 (2016), pp. 1319–1333 (cit. on p. 65).

[192] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. 2nd. Statistics. Springer, 2009 (cit. on pp. 62, 71).

[193] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition". In: *Procs. of the IEEE Conference on Computer Vision and Pattern Recognition 2015*. 2015. DOI: https://arxiv.org/pdf/1512.03385.pdf (cit. on p. 73).

[194] D. Heck, J. Knapp, J. N. Capdevielle, G. Schatz, and T. Thouw. *CORSIKA: a Monte Carlo code to simulate extensive air showers*. Feb. 1998 (cit. on pp. 106, 118, 151, 207, 273).

[195] N. Helft. "Induction as Nonmonotonic Inference". In: *Procs. of the Int. Conference on Principles of Knowledge Representation and Reasoning 1989*. Morgan Kaufmann, 1989, pp. 149–156 (cit. on p. 6).

[196] B. Hentschel, M. S. Kester, and S. Idreos. "Column Sketches: A scan accelerator for rapid and robust predicate evaluation". In: *Procs. of the Int. Conference on Management of Data 2018*. 2018, pp. 857–872 (cit. on pp. 155, 156).

[197] R. Herbrich, N. D. Lawrence, and M. Seeger. "Fast sparse Gaussian process methods: The informative vector machine". In: *Advances in Neural Information Processing Systems 16: Procs. of the 2003 Conference*. 2003, pp. 625–632 (cit. on p. 182).

[198] G. C. Hill and K. Rawlins. "Unbiased cut selection for optimal upper limits in neutrino detectors: the model rejection potential technique". In: *Astroparticle Physics* 19.3 (June 2003), pp. 393–402 (cit. on p. 216).

[199] A. M. Hillas. "Cerenkov light images of EAS produced by primary gamma". In: *Procs. of the Int. Cosmic Ray Conference 1985*. Ed. by F. C. Jones. Vol. 3. International Cosmic Ray Conference. Aug. 1985, pp. 445–448 (cit. on p. 175).

[200] A. Hocker, J. Stelzer, F. Tegenfeldt, H. Voss, K. Voss, et al. "TMVA - Toolkit for Multivariate Data Analysis". In: *Proceeding of science* ACAT (2007), p. 040 (cit. on p. 197).

[201] A. Hoecker et al. "TMVA 4 — Toolkit for Multivariate Data Analysis with ROOT. Users Guide." In: (2009) (cit. on p. 197).

[202] D. J. Hopkins and G. King. "A method of automated nonparametric content analysis for social science". In: *American Journal of Political Science* 54.1 (2010), pp. 229–247 (cit. on p. 303).

[203] K. Hornik, M. Stinchcombe, and H. White. "Multilayer Feedforward Networks are Universal Approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366. URL: https://cognitivemedium.com/magic_paper/assets/Hornik.pdf (cit. on p. 73).

[204] I. Hossain, A. Khosravi, and S. Nahavandi. "Weighted informative inverse active class selection for motor imagery brain computer interface". In: *Procs. of the Canadian Conference on Electrical and Computer Engineering 2017*. IEEE, 2017, pp. 1–5 (cit. on p. 114).

[205] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama. "Unsupervised Discrete Representation Learning". In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer, 2019, pp. 97–119 (cit. on p. 29).

[206] X. Huang et al. "A Survey of Safety and Trustworthiness of Deep Neural Networks: Verification, Testing, Adversarial Attack and Defence, and Interpretability". In: *Computer Science Review* 37 (Aug. 2020). URL: https://www.sciencedirect.com/science/article/pii/S1574013719302527 (cit. on p. 75).

[207] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. "Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations". In: *Journal of Machine Learning Research* 18 (2018) (cit. on p. 58).

[208] M. Huennefeld. "Combining Maximum-Likelihood with Deep Learning for Event Reconstruction in IceCube". In: *Proceeding of science* ICRC2021 (2021), p. 1065 (cit. on pp. 252, 254–256, 258).

[209] K. Hymon. "Measurements of Seasonal Variations of the Unfolded Atmospheric Neutrino Energy Spectrum with IceCube". MA thesis. Ruhr University Bochum, 2020 (cit. on p. 312).

[210] K. Hymon and T. Ruhe. “Seasonal Variations of the Unfolded Atmospheric Neutrino Spectrum with IceCube”. In: *arXiv: Physics* (July 2021). DOI: arXiv:2107.09349 (cit. on pp. 310, 316).

[211] IceCube Collaboration. “Evidence for High-Energy Extraterrestrial Neutrinos at the IceCube Detector”. In: *Science* 342 (Nov. 2013) (cit. on p. 87).

[212] IceCube Collaboration and K. Morik. “Development of a General Analysis and Unfolding Scheme and its Application to Measure the Energy Spectrum of Atmospheric Neutrinos with IceCube”. In: *European Physical Journal C* (2014) (cit. on pp. 209, 297).

[213] P. Jaccard. “The distribution of the flora in the alpine zone. 1”. In: *New phytologist* 11.2 (1912), pp. 37–50 (cit. on p. 169).

[214] B. Jacob et al. “Quantization and training of neural networks for efficient integer-arithmetic-only inference”. In: *Procs. of the Conference on Computer Vision and Pattern Recognition 2018*. 2018, pp. 2704–2713 (cit. on p. 58).

[215] M. Jacquemont, T. Vuillaume, A. Benoit, C. Maurin, P. Lambert, and G. Lamanna. *First Full-Event Reconstruction from Imaging Atmospheric Cherenkov Telescope Real Data with Deep Learning*. Tech. rep. 2021. DOI: arXiv:2105.14927 (cit. on p. 269).

[216] M. Jacquemont, T. Vuillaume, A. Benoît, G. Maurin, and P. Lambert. “Single Imaging Atmospheric Cherenkov Telescope Full-Event Reconstruction with a Deep Multi-Task Learning Architecture”. In: *Procs. of the Astronomical Data Analysis Software and Systems Conference 2020*. 2020 (cit. on p. 270).

[217] N. P. Jouppi, C. Young, N. Patil, and D. Patterson. “A domain-specific architecture for deep neural networks”. In: *Communications of the ACM* 61.9 (2018), pp. 50–59 (cit. on p. 63).

[218] B. D. Jovanovic and P. S. Levy. “A Look at the Rule of Three”. In: *The American Statistician* 51.2 (1997), pp. 137–139. URL: https://www.tandfonline.com/doi/abs/10.1080/00031305.1997.10473947 (cit. on p. 181).

[219] M. Kampffmeyer, S. Løkse, F. M. Bianchi, R. Jenssen, and L. Livi. “The deep kernelized autoencoder”. In: *Applied Soft Computing* 71 (2018), pp. 816–825. URL: https://www.sciencedirect.com/science/article/pii/S1568494618304174 (cit. on p. 183).

[220] D. B. Kaplan. “Flavor at SSC energies: A New mechanism for dynamically generated fermion masses”. In: *Nuclear Physics B* 365 (1991), pp. 259–278 (cit. on p. 237).

[221] D. B. Kaplan, H. Georgi, and S. Dimopoulos. “Composite Higgs Scalars”. In: *Physics Letters B* 136 (1984), pp. 187–190 (cit. on p. 237).

[222] A.-H. Karimi, B. J. von Kügelgen, B. Schölkopf, and I. Valera. “Algorithmic recourse under imperfect causal knowledge: a probabilistic approach”. In: *Advances in Neural Information Processing Systems 33: Procs. of the 2020 Conference*. 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/02a3c7fb3f489288ae6942498498db20-Abstract.html (cit. on p. 8).

[223] T. Kato, S. Omachi, and H. Aso. “Asymmetric Gaussian and Its Application to Pattern Recognition”. In: *Lecture Notes in Computer Science* (2002), pp. 405–413. DOI: https://doi.org/10.1007/3-540-70659-3_42 (cit. on p. 254).

[224] M. Kearns and U. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994 (cit. on p. 66).

[225] V. Khachatryan, I. Bediaga, B. Spaan, others (CMS, and L. Collaborations). “Observation of the rare $B_s^0 \to \mu^+\mu^-$ decay from the combined analysis of CMS and LHCb data”. In: *Nature* 522 (2015), pp. 68–72 (cit. on p. 153).

[226] D. Kieda, V. Collaboration, et al. “Status of the VERITAS ground based GeV/TeV Gamma-Ray Observatory”. In: *Bulletin of the American Astronomical Society*. Vol. 36. 2004, p. 910 (cit. on p. 269).

[227] D. P. Kingma and J. L. Ba. “Adam: A Method for Stochastic Optimization”. In: *Procs. of the Int. Conference on Learning Representation 2015*. 2015 (cit. on p. 61).

[228] S. Kleene. “Finite axiomatizability of theories in the predicate calculus using additional predicate symbols. – Two Papers on the Predicate Calculus”. In: *Memoirs of the American Mathematical Society* 10 (1952) (cit. on p. 6).

[229] J.-H. Koehne et al. “PROPOSAL: A tool for propagation of charged leptons”. In: *Computer Physics Communications* 184.9 (Sept. 2013), pp. 2070–2090 (cit. on pp. 16, 20, 136). **SFB876-C3**

[230] R. Kohavi and G. H. John. “Wrappers for feature subset selection”. In: *Artificial Intelligence* 97.1-2 (1997), pp. 273–324 (cit. on p. 57).

[231] H. Kotthaus, L. Schönberger, A. Lang, J.-J. Chen, and P. Marwedel. “Can Flexible Multi-Core Scheduling Help to Execute Machine Learning Algorithms Resource-Efficiently?” In: *Procs. of the Int. Workshop on Software and Compilers for Embedded Systems 2019*. SCOPES ’19. ACM, 2019, pp. 59–62. DOI: https://dl.acm.org/doi/10.1145/3323439.3323986 (cit. on p. 63). **SFB876-A3, SFB876-C5**

[232] D. Kottke et al. “Probabilistic Active Learning for Active Class Selection”. In: *Advances in Neural Information Processing Systems 29: Procs. of the Workshop on the Future of Interactive Learning Machines 2016*. 2016 (cit. on p. 114).

[233] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems 25: Procs. of the 2012 Conference*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Curran Associates, Inc., 2012, pp. 1097–1105 (cit. on pp. 72, 247).

[234] M. J. Kruger. *Building a Parallella Board Cluster*. Bachelor Thesis, Rhodes University, Grahamstown, South Africa. Nov. 2015 (cit. on p. 122).

[235] L. I. Kuncheva. “A stability index for feature selection”. In: *Procs. of the Int. Conference on Artificial Intelligence and Applications 2007*. Ed. by V. Devedzic. IASTED/ACTA Press, 2007, pp. 421–427 (cit. on p. 169).

[236] P. Laforgue, S. Clémençon, and F. d’Alché-Buc. “Autoencoding any data through kernel autoencoders”. In: *Procs. of the Int. Conference on Artificial Intelligence and Statistics 2019*. PMLR. 2019, pp. 1061–1069 (cit. on p. 183).

[237] S. Lakshminarasimhan et al. “ISABELA-QA: query-driven analytics with ISABELA-compressed extreme-scale scientific data”. In: *Procs. of Int. Conference for High Performance Computing, Networking, Storage and Analysis 2011*. 2011, pp. 1–11 (cit. on p. 156).

[238] C. A. Lang, B. Bhattacharjee, T. Malkemus, S. Padmanabhan, and K. Wong. “Increasing buffer-locality for multiple relational table scans through grouping and throttling”. In: *Procs. of the IEEE Int. Conference on Data Engineering 2007*. IEEE. 2007, pp. 1136–1145 (cit. on p. 156).

[239] D. Lange. “The EvtGen particle decay simulation package”. In: *Nuclear Instruments and Methods in Physics Research* A462 (2001), pp. 152–155 (cit. on pp. 103, 104).

[240] Y. LeCun et al. “Handwritten Digit Recognition with a Back-Propagation Network”. In: *Advances in Neural Information Processing Systems 2: Procs. of the 1990 Conference*. Morgan-Kaufmann, 1990, pp. 396–404. URL: http://papers.nips.cc/paper/293-handwritten-digit-recognition-with-a-back-propagation-network.pdf (cit. on p. 247).

[241] R. W. Lessard, J. H. Buckley, V. Connaughton, and S. Le Bohec. “A new analysis method for reconstructing the arrival direction of TeV gamma rays using a single imaging atmospheric Cherenkov telescope”. In: *Astroparticle Physics* 15.1 (2001), pp. 1–18. DOI: 10.1016/s0927-6505(00)00133-x (cit. on p. 224).

[242] F. Lettich et al. “Parallel Traversal of Large Ensembles of Decision Trees”. In: *IEEE Transactions on Parallel and Distributed Systems* 30.9 (2019), pp. 2075–2089. DOI: https://doi.org/10.1109/TPDS.2018.2860982 (cit. on p. 69).

[243] LHCb Collaboration. “Addendum to the LHCb online system technical design report”. In: (Nov. 2005) (cit. on p. 80).
[244] LHCb Collaboration. “For LHCb Talks”. General Photo. June 2012. URL: https://cds.cern.ch/record/1463546 (cit. on p. 44).
[245] LHCb Collaboration. *LHCb Trigger and Online Upgrade Technical Design Report*. May 2014. URL: https://cds.cern.ch/record/1701361/files/LHCB-TDR-016.pdf (cit. on p. 81).
[246] LHCb Collaboration, CERN. “A Comparison of CPU and GPU Implementations for the LHCb Experiment Run 3 Trigger”. In: *Computing and Software for Big Science* 6.1 (2022), p. 1. DOI: arXiv:2105.04031 (cit. on p. 98).
[247] LHCb Collaboration, CERN. *Upgrade Software and Computing*. Tech. rep. Geneva: CERN, Mar. 2018. URL: https://cds.cern.ch/record/2310827 (cit. on p. 92).
[248] LHCb Collaboration, CERN (Meyrin). *Computing Model of the Upgrade LHCb experiment*. Tech. rep. Geneva: CERN, May 2018. URL: https://cds.cern.ch/record/2319756 (cit. on p. 82).
[249] F.-F. Li, R. Fergus, and P. Perona. “One-shot learning of object categories”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.4 (2006), pp. 594–611. URL: http://vision.stanford.edu/documents/Fei-FeiFergusPerona2006.pdf (cit. on p. 57).
[250] T.-P. Li and Y.-Q. Ma. “Analysis methods for results in gamma-ray astronomy”. In: *The Astrophysical Journal* 272 (1983) (cit. on pp. 227, 276).
[251] J. Z. Liu, Z. Lin, S. Padhy, D. Tran, T. Bedrax-Weiss, and B. Lakshminarayanan. “Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness”. In: *arXiv: Computing Research Repository* (Oct. 2020). DOI: arXiv:2006.10108 (cit. on p. 74).
[252] Q. Liu et al. “Hello ADIOS: the challenges and lessons of developing leadership class I/O frameworks”. In: *Concurrency and Computation: Practice and Experience* 26.7 (2014), pp. 1453–1473 (cit. on p. 156).
[253] S. P. Lloyd. “Least squares quantization in PCM”. In: *IEEE Transactions on Information Theory* 28.2 (Mar. 1982), pp. 129–137 (cit. on p. 185).
[254] R. Lomasky, C. E. Brodley, M. Aernecke, D. Walt, and M. Friedl. “Active Class Selection”. In: *Procs. of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2007*. 2007, pp. 640–647 (cit. on pp. 113, 114).
[255] M. Honda et al. “New calculation of the atmospheric neutrino flux in a three-dimensional scheme”. In: *Physical Review D* 70 (Aug. 2004), p. 043008 (cit. on p. 208).
[256] W. Maass. “Bounds on the computational power and learning complexity of analog neural nets”. In: *Procs. of the ACM Symposium on the Theory of Computing 1993*. 1993, pp. 335–344 (cit. on p. 66).
[257] L. van der Maaten. “Learning a Parametric Embedding by Preserving Local Structure.” In: *Journal of Machine Learning Research - Proceedings Track* 5 (Jan. 2009), pp. 384–391 (cit. on p. 188).
[258] G. Maier et al. “Performance of the Cherenkov Telescope Array”. In: *Procs. of the Int. Cosmic Ray Conference 2017*. 2017 (cit. on p. 68).
[259] G. S. Manku and R. Motwani. “Approximate frequency counts over data streams”. In: *Procs. of the Int. Conference on Very Large Data Bases 2002*. VLDB Endowment, 2002, pp. 346–357 (cit. on p. 56).
[260] P. Marwedel. *Embedded System Design - Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things*. 4th ed. Springer, 2021. URL: https://link.springer.com/book/10.1007/978-3-030-60910-8 (cit. on p. 63). **SFB876-A3**
[261] N. Marz and J. Warren. *Big Data – Principles and Best Practices of Scalable Realtime Data Systems*. Manning Publications, 2015 (cit. on p. 56).
[262] J. Matthews. “A Heitler model of extensive air showers”. In: *Astroparticle Physics* 22 (2005), pp. 387–397 (cit. on pp. 136, 137).

[263] M. May. “Effizente Bildverarbeitung hexagonaler Strukturen mittels Deep Convolutional Neural Networks”. MA thesis. 2018 (cit. on p. 188).

[264] M. Meier. “Search for Astrophysical Tau Neutrinos using 7.5 years of IceCube Data”. PhD thesis. TU Dortmund University, 2019 (cit. on pp. 213–217). **SFB876-C3**

[265] H. Menjo et al. “Status and Prospects of the LHCf and RHICf experiments”. In: *Procs. of Int. Cosmic Ray Conference 2021*. Vol. 395. 2021, p. 301 (cit. on p. 141).

[266] P. Mercado, F. Tudisco, and M. Hein. “Generalized Matrix Means for Semi-Supervised Learning with Multilayer Graphs”. In: *Advances in Neural Information Processing Systems 32: Procs. of the 2019 Conference*. Vol. 32. Vancouver, Canada, 2019. URL: https://uni-tuebingen.de/securedl/sdl-eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE2MTYzNjk1ODEsImV4cCI6MTYxNjQ1OTU3NywidXNlciI6MCwiZ3JvdXBzIjpbMCwtMV0sImZpbGUiOiJ
VzeaDxQgwYIMOvZslNQCND1AsX3MRue-vmeC8j8fQ_M/2019_GeneralizedMatrixMeansforSemi-SupervisedLearningwithMultilayerGraphs.pdf (cit. on p. 53).

[267] N. Merz, S. Regel, and J. Lewandowski. “The Manifesto Corpus: A new resource for research on political parties and quantitative text analysis”. In: *Research & Politics* 3.2 (2016) (cit. on p. 303).

[268] C. Meurer, J. Bluemer, R. Engel, A. Haungs, and M. Roth. “Muon production in extensive air showers and its relation to hadronic interactions”. In: *Czechoslovak Journal of Physics* 56 (2006). Ed. by J. Ridky, M. Bohacova, D. Nosek, J. Rames, R. Smida, and P. Travnicek, A211 (cit. on p. 141).

[269] I. Mierswa. “Non-Convex and Multi-Objective Optimization in Data Mining”. PhD thesis. TU Dortmund University, 2008 (cit. on p. 53).

[270] I. Mierswa and K. Morik. “Automatic Feature Extraction for Classifying Audio Data”. In: *Machine Learning Journal* 58 (2005), pp. 127–149 (cit. on pp. 29, 58).

[271] I. Mierswa and M. Wurst. “Efficient Feature Construction by Meta Learning – Guiding the Search in Meta Hypothesis Space”. In: *Procs. of the Int. Conference on Machine Learning, Workshop on Meta Learning 2005*. 2005 (cit. on p. 67).

[272] N. Milke, M. Doert, S. Klepser, D. Mazin, V. Blobel, and W. Rhode. “Solving inverse problems with the unfolding program TRUEE: Examples in astroparticle physics”. In: *Nuclear Instruments and Methods in Physics Research A* 697 (Jan. 2013), pp. 133–147 (cit. on pp. 295, 297). **SFB876-C3**

[273] T. Mitchell. “Introduction to machine learning”. In: *Machine Learning* 7 (1997), pp. 2–5 (cit. on p. 65).

[274] T. M. Mitchell. *Machine Learning*. New York: McGraw Hill, 1997 (cit. on pp. 58, 66).

[275] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Mueller. “Layer-wise relevance propagation: an overview”. In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer, 2019, pp. 193–209 (cit. on p. 75).

[276] K. Morik. “Sloppy Modeling”. In: *Knowledge Representation and Organization in Machine Learning*. Ed. by K. Morik. Berlin, New York: Springer Verlag, 1989, pp. 107–134 (cit. on p. 27).

[277] K. Morik. “The Representation Race - Preprocessing for Handling Time Phenomena”. In: *Procs. of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2000*. Ed. by R. L. de Mántaras and E. Plaza. Vol. 1810. Lecture Notes in Artificial Intelligence. Berlin, Heidelberg, New York: Springer Verlag Berlin, 2000, pp. 4–19. URL: http://www.springerlink.com/content/n51nhla4f82ygk4t/?p=19f2b4ad2ab849db84e3794d32036c89%5C&pi=0 (cit. on p. 57).

[278] K. Morik, C. Bockermann, and S. Buschjäger. “Big Data Science”. In: 32.1 (Dec. 2017), pp. 27–36. DOI: https://doi.org/10.1007/s13218-017-0522-8 (cit. on p. 56). **SFB876-A1, SFB876-C3**

[279] K. Morik and J.-U. Kietz. “Constructive Induction of Background Knowledge”. In: *Procs. of the IJCAI-Workshop on Evaluating and Changing Representations in Machine Learning* (1991) (cit. on p. 6).

[280] K. Morik and C.-R. Rollinger. “The Real Estate Agent - Modeling Users by Uncertain Reasoning”. In: *The AI Magazine* 6.2 (1985), pp. 44–52. URL: http://www.aaai.org/ojs/index.php/aimagazine/article/view/479 (cit. on pp. 8, 34).

[281] K. Morik, S. Wrobel, J.-U. Kietz, and W. Emde. *Knowledge Acquisition and Machine Learning - Theory, Methods, and Applications*. London: Academic Press, 1993 (cit. on p. 7).

[282] S. A. Mueller et al. “Single photon extraction for FACT’s SiPMs allows for novel IACT event representation”. In: *Proceedings of Science*. 2017 (cit. on pp. 187, 270). **SFB876-C3**

[283] S. Muggleton. “Inductive Logic Programming: Derivations, Successes and Shortcommings”. In: *Procs. of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 1993*. Ed. by P. Brazdil. 1993 (cit. on p. 6).

[284] K. P. Murphy. *Machine Learning: a Probabilistic Perspective*. MIT Press, 2013 (cit. on p. 62).

[285] B. Nachman, P. Nef, A. Schwartzman, M. Swiatlowski, and C. Wanotayaroj. “Jets from Jets: Re-clustering as a tool for large radius jet reconstruction and grooming at the LHC”. In: *Journal of High Energy Physics* 02 (2015), p. 075 (cit. on p. 239).

[286] B. K. Natarajan. *Machine Learning. A Theoretical Approach*. San Mateo, CA: Morgan Kaufmann, 1991 (cit. on p. 65).

[287] D. Neise, L. Tani, M. Nöthe, M. Bunse, K. Brügge, and J. Buß. *photon-stream*. 2021. URL: https://github.com/fact-project/photon_stream (cit. on p. 270).

[288] I. Newton. *The Principia: mathematical principles of natural philosophy. 3rd ed. Quoted after: The Principia: A New Translation by I. B. Cohen and A. Whitman*. Berkeley, California: University of California Press, 1999 (cit. on p. 2).

[289] L. Nickel. “Stereo Reconstruction for the Early Days of CTA”. MA thesis. TU Dortmund University, 2020 (cit. on pp. 224, 226).

[290] S.-H. Nienhuys-Cheng and R. de Wolf. “Foundations in Logic Programming”. In: Number 1228 in LNAI Series. 1997 (cit. on p. 6).

[291] D. Nieto et al. “Studying deep convolutional neural networks with hexagonal lattices for imaging atmospheric Cherenkov telescope event reconstruction”. In: *Proceedings of Science* (2019), pp. 1–7. DOI: arXiv:1912.09898 (cit. on p. 269).

[292] Nigro, C. et al. “Towards open and reproducible multi-instrument analysis in gamma-ray astronomy”. In: *A&A* 625 (2019), A10. DOI: https://doi.org/10.1051/0004-6361/201834938 (cit. on p. 150). **SFB876-C3**

[293] J. Nocedal and S. J. Wright. *Numerical Optimization*. Second. Springer Series in Operations Research and Financial Engineering. Springer, 2006. URL: https://books.google.de/books?id=epc5fX0lqRIC (cit. on pp. 53, 61, 62, 295).

[294] M. Nöthe. “Monitoring the high energy universe – Open, reproducible, machine learning based analysis for the First G-APD Cherenkov Telescope”. PhD Thesis. TU Dortmund University, 2020. DOI: http://dx.doi.org/10.17877/DE290R-21143 (cit. on pp. 68, 219, 222, 269, 274). **SFB876-C3**

[295] M. Nöthe, K. Brügge, and J. Buß. *fact-project/aict-tools: v0.27.1 – 2021-05-05*. May 2021. DOI: https://doi.org/10.5281/zenodo.4738501 (cit. on p. 221).

[296] M. Nöthe, K. Kosack, L. Nickel, and M. Peresano. “Prototype Open Event Reconstruction Pipeline for the Cherenkov Telescope Array”. In: *Procs. of the Int. Cosmic Ray Conference 2021*. Vol. 395. 744. Proceedings of Science, 2021. DOI: 10.22323/1.395.0744 (cit. on pp. 95, 177). **SFB876-C3**

[297] *onnx-runtime*. 2021. URL: https://onnxruntime.ai/ (cit. on p. 274).

[298] Z. Ou, B. Pang, Y. Deng, J. K. Nurminen, A. Ylä-Jääski, and P. Hui. "Energy- and Cost-Efficiency Analysis of ARM-Based Clusters". In: *Procs. of the IEEE/ACM Int. Symposium on Cluster, Cloud and Grid Computing 2012*. IEEE Computer Society, 2012, pp. 115–123. DOI: https://doi.org/10.1109/CCGrid.2012.84 (cit. on p. 122).

[299] T. D. Parsons and J. L. Reinebold. "Adaptive virtual environments for neuropsychological assessment in serious games". In: *Transactions on Consumer Electronics* 58.2 (2012), pp. 197–204 (cit. on p. 114).

[300] A. Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32: Procs. of the 2019 Conference*. 2019 (cit. on p. 188).

[301] J. Pearl and D. Mackenzie. *The book of why: The new science of cause and effect*. Basic Books, 2018 (cit. on p. 8).

[302] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on pp. 206, 221).

[303] D. Petry. "The MAGIC Telescope-prospects for GRB research". In: *Astronomy and Astrophysics Supplement Series* 138.3 (1999), pp. 601–602 (cit. on p. 269).

[304] L. Pfahler. "Some Representation Learning Tasks and the Inspection of their Models". PhD thesis. TU Dortmund University, 2022 (cit. on p. 268). **SFB876-A1**

[305] L. Pfahler and K. Morik. "Explaining Deep Learning Representations by Tracing the Training Process". In: *arXiv: Computing Research Repository* (Sept. 2021). DOI: arXiv:2109.05880 (cit. on p. 75).

[306] H. V. Pham, J. Wang, T. Lutellier, J. Rosenthal, and L. Tan. "Problems and Opportunities in Training Deep Learning Software Systems : An Analysis of Variance". In: *Procs. of the IEEE/ACM Int. Conference on Automated Software Engineering 2020*. URL: https://www.cs.purdue.edu/homes/lintan/publications/variance-ase20.pdf (cit. on p. 278).

[307] N. Piatkowski. "Hyper-Parameter-Free Generative Modelling with Deep Boltzmann Trees". In: *Procs. of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2019*. Springer, 2019 (cit. on p. 29).

[308] N. Piatkowski, S. Lee, and K. Morik. "Integer undirected graphical models for resource-constrained systems". In: *Neurocomputing* 173.1 (Jan. 2016), pp. 9–23. URL: http://www.sciencedirect.com/science/article/pii/S0925231215010449 (cit. on p. 58). **SFB876-A1, SFB876-C1**

[309] N. Piatkowski and K. Morik. "Parallel Inference on Structured Data with CRFs on GPUs". In: *Procs. of the Int. Workshop at ECML PKDD on Collective Learning and Inference on Structured Data 2011*. Athens, Greece, 2011 (cit. on p. 63). **SFB876-A1**

[310] N. Piatkowski and F. Schnitzler. "Compressible Reparametrization of Time-Variant Linear Dynamical Systems". In: *Solving Large Scale Learning Tasks. Challenges and Algorithms - Essays Dedicated to Katharina Morik on the Occasion of Her 60th Birthday*. 2016, pp. 234–250. DOI: http://dx.doi.org/10.1007/978-3-319-41706-6_12 (cit. on p. 55). **SFB876-A1**

[311] T. Pierog, I. Karpenko, J. M. Katzy, E. Yatsenko, and K. Werner. "EPOS LHC: Test of collective hadronization with data measured at the CERN Large Hadron Collider". In: *Physical Review C* 92.3 (2015), p. 034906 (cit. on p. 137).

[312] T. Pierog, S. Baur, H. Dembinski, M. Perlin, R. Ulrich, and K. Werner. "When heavy ions meet cosmic rays: potential impact of QGP formation on the muon puzzle". In: *Procs. of the Int. Cosmic Ray Conference 2021*. Vol. 395. 2021, p. 469 (cit. on p. 134).

[313] M. Pivk and F. R. Le Diberder. "SPlot: A Statistical tool to unfold data distributions". In: *Nuclear Instruments and Methods in Physics Research Section A* 555 (2005), pp. 356–369 (cit. on pp. 197, 199).

[314] M. Plum. “Cosmic ray composition study using machine learning at the IceCube Neutrino Observatory”. In: *Proceeding of science* ICRC2019 (2020), p. 394 (cit. on p. 134).

[315] K. R. Popper. *The Logic of Scientific Discovery*. London, England: Routledge, 1934 (cit. on p. 4).

[316] R. R. Prado. “Recent results from the cosmic ray program of the NA61/SHINE experiment”. In: *EPJ Web of Conferences* 208 (2019). Ed. by B. Pattison, Y. Itow, T. Sako, and H. Menjo, p. 05006 (cit. on p. 141).

[317] R. R. Prado. “Measurements of Hadron Production in Pion-Carbon Interactions with NA61/SHINE at the CERN SPS”. In: *Proceeding of science* ICRC2017 (2018), p. 315 (cit. on p. 141).

[318] G. Punzi. “Sensitivity of searches for new signals and its optimization”. In: *eConf: Electronic Conference Proceedings Archive* C030908 (2003), MODT002 (cit. on p. 201).

[319] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Machine Learning. San Mateo, CA: Morgan Kaufmann, 1993 (cit. on pp. 66, 69).

[320] S. Rajamoney, G. DeJong, and B. Faltings. “Towards A Model of Conceptual Knowledge Acquisition Through Directed Experimentation”. In: *Procs. of the Int. Joint Conference on Artificial Intelligence 1985*. Los Altos, CA: Morgan Kaufman, 1985, pp. 688–690 (cit. on p. 4).

[321] RapidMiner, Inc. *RapidMiner*. Mar. 2017. URL: https://rapidminer.com/ (cit. on p. 206).

[322] C. Rasmussen and C. Williams. *Gaussian processes for machine learning*. The MIT Press, 2006, p. 95. URL: http://gaussianprocess.org/ (cit. on p. 182).

[323] A. L. Read. “Presentation of search results: The CL_s technique”. In: *Journal of Physics G* 28 (2002), pp. 2693–2704 (cit. on p. 230).

[324] F. Riehn, R. Engel, A. Fedynitch, T. K. Gaisser, and T. Stanev. “Hadronic interaction model Sibyll 2.3d and extensive air showers”. In: *Physical Review D* 102.6 (2020), p. 063002 (cit. on p. 142).

[325] J. Rissanen. “A universal prior for integers and estimation by minimum description length”. In: *The Annals of statistics* (1983), pp. 416–431 (cit. on p. 55).

[326] J. Rissanen. “Modeling by Shortest Data Description”. In: *Automatica* 14 (1978), pp. 465–471 (cit. on p. 55).

[327] L. Rosasco, E. De, V. A. Caponnetto, M. Piana, and A. Verri. “Are loss functions all the same”. In: *Neural Computation* 15 (2004), p. 2004 (cit. on p. 54).

[328] S. Rötner. “Deep Learning on Raw Telescope Data”. MA thesis. 2017 (cit. on p. 188).

[329] T. Ruhe. “Data Mining on the Rocks – A Measurement of the Atmospheric Neutrino Flux using IceCube in the 59-string Configuration and a Novel Data Mining Based Approach to Unfolding”. PhD thesis. TU Dortmund University, 2013 (cit. on pp. 167, 168, 170, 171, 208, 297, 317).

[330] T. Ruhe, D. Elsässer, W. Rhode, M. Nöthe, and K. Brügge. “Cherenkov Telescope Ring - An Idea for World Wide Monitoring of the VHE Sky”. In: *EPJ Web of Conferences* 207 (Jan. 2019), p. 03002 (cit. on p. 269). **SFB876-C3**

[331] T. Ruhe, K. Morik, and W. Rhode. “Application of RapidMiner in Neutrino Astronomy”. In: *RapidMiner: Data Mining Use Cases and Business Analytics Applications*. Ed. by M. Hofmann and R. Klinkenberg. CRCPress Book, 2013. Chap. Feature Selection and Classification in Astroparticle Physics and in Medical Domains (cit. on pp. 57, 68).

[332] T. Ruhe, K. Morik, and B. Schowe. “Data Mining on Ice”. In: *Procs. of the Workshop on Astro-statistics and Data Mining in Large Astronomical Databases*. Ed. by L. Sarro, C. Bailer-Jones, L. Eyer, W. O’Mullane, and J. de Ridder. 2011 (cit. on p. 167). **SFB876-C3**

[333] T. Ruhe, M. Schmitz, T. Voigt, and M. Wornowizki. “DSEA: A Data Mining Approach to Unfolding”. In: *Procs. of the Int. Cosmic Ray Conference 2013*. 2013 (cit. on p. 300). **SFB876-C3**

[334] "The calibration of the first Large-Sized Telescope of the Cherenkov Telescope Array". In: *Procs. of the Int. Cosmic Ray Conference 2019*. Ed. by S. Sakurai, D. Depaoli, R. López-Coto, et al. Vol. 358. 780. 2019. DOI: 10.22323/1.358.0780 (cit. on p. 173).

[335] R. Salakhutdinov and G. Hinton. "Deep Boltzmann machines". In: *Procs. of the Int. Conference on Artificial Intelligence and Statistics 2009*. 2009, pp. 448–455 (cit. on p. 29).

[336] D. Salinas et al. "DeepAR: Probabilistic forecasting with autoregressive recurrent networks". In: *International Journal of Forecasting* 36.3 (2020), pp. 1181–1191. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301888 (cit. on p. 257).

[337] W. Samek and K.-R. Müller, eds. *Towards Explainable Artificial Intelligence*. Springer, 2019 (cit. on p. 75).

[338] A. Sandrock and W. Rhode. "Coulomb corrections to the bremsstrahlung and electron pair production cross section of high-energy muons on extended nuclei". In: (July 2018) (cit. on p. 136).

[339] A. Sandrock, S. R. Kelner, and W. Rhode. "Radiative corrections to the average bremsstrahlung energy loss of high-energy muons". In: *Physics Letters B* 776 (2018), pp. 350–354 (cit. on p. 136).

[340] T. Scheffer. "Email Answering Assistance by Semi-Supervised Text Classification". In: *Intelligent Data Analysis* 8.5 (2004), pp. 481–493 (cit. on p. 53).

[341] F. Scheriau. "Data-Mining in der Astroteilchenphysik". PhD thesis. TU Dortmund University, 2014 (cit. on pp. 167, 168, 206, 210).

[342] M. Schmitz. "Neuartige Boten aus dem All". PhD thesis. TU Dortmund University, 2014 (cit. on pp. 167, 168).

[343] B. Schölkopf and A. J. Smola. *Learning with Kernels – Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive computation and machine learning. Cambridge MA: MIT Press, 2002 (cit. on p. 23).

[344] B. Schowe and K. Morik. "Fast-Ensembles of Minimum Redundancy Feature Selection". In: *Ensembles in Machine Learning Applications*. Ed. by O. Okun, G. Valentini, and M. Re. Studies in Computational Intelligence. Springer, 2011, pp. 75–95 (cit. on pp. 59, 167, 169).

[345] M. Seeger. *Greedy forward selection in the informative vector machine*. Tech. rep. Technical report, University of California at Berkeley, 2004 (cit. on p. 182).

[346] B. Settles. *Active Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2012 (cit. on p. 113).

[347] G. Shafer and J. Pearl. "Introduction: Uncertain Reasoning". In: *Readings in Uncertain Reasoning*. Ed. by G. Shafer and J. Pearl. Morgan Kaufmann, 1990. Chap. I, pp. 1–6 (cit. on p. 8).

[348] I. Shilon et al. "Application of Deep Learning methods to analysis of Imaging Atmospheric Cherenkov Telescopes data". In: *arXiv: Physics* (2018). URL: arXiv:1803.10698 (cit. on p. 269).

[349] H. Simon. "An almost optimal PAC algorithm". In: *Procs. of the Conference on Learning Theory 2015*. 2015 (cit. on p. 65).

[350] K. Simonyan and A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *arXiv: Computing Reseach Repository* 1409 (2014). DOI: arXiv:1409.1556 (cit. on pp. 73, 188).

[351] A. M. Sirunyan et al. "Measurement of the average very forward energy as a function of the track multiplicity at central pseudorapidities in proton-proton collisions at $\sqrt{s} = 13$ TeV". In: *The European Physical Journal C* 79.11 (2019), p. 893 (cit. on p. 140).

[352] J. Sitarek, M. Gaug, D. Mazin, R. Paoletti, and D. Tescaro. "Analysis techniques and performance of the Domino Ring Sampler version 4 based readout for the MAGIC telescopes". In:

Nuclear Instruments and Methods in Physics Research Section A 723 (2013), pp. 109–120 (cit. on p. 173).

[353] T. Sjöstrand, S. Ask, J. R. Christiansen, R. Corke, N. Desai, et al. "An Introduction to PYTHIA 8.2". In: *Computer Physics Communications* 191 (2015), pp. 159–177 (cit. on p. 103).

[354] T. Sjöstrand, S. Mrenna, and P. Z. Skands. "PYTHIA 6.4 Physics and Manual". In: *Journal of High Energy Physics* 0605 (2006), p. 026. DOI: https://doi.org/10.1088/1126-6708/2006/05/026 (cit. on p. 103).

[355] J. Soedingrekso, A. Sandrock, and W. Rhode. *The effect of improved high-energy muon cross-sections*. 2019 (cit. on p. 20).

[356] D. Soldin. "Update on the Combined Analysis of Muon Measurements from Nine Air Shower Experiments". In: *Procs. of the Int. Cosmic Ray Conference 2021*. Vol. 395. 2021, p. 349 (cit. on p. 135).

[357] I. Stahl. "The appropriateness of predicate invention as bias shift operation in ILP." In: *Machine Learning Journal* 20 (1995), pp. 95–117. DOI: https://doi.org/10.1007/BF00993476 (cit. on p. 6).

[358] W. Stammer, P. Schramowski, and K. Kersting. "Right for the Right Concept: Revising Neuro-Symbolic Concepts by Interacting With Their Explanations". In: *Procs. of the Conference on Computer Vision and Pattern Recognition 2021*. Computer Vision Foundation IEEE, 2021. DOI: https://dblp.org/rec/conf/cvpr/2021.bib. URL: https://openaccess.thecvf.com/CVPR2021 (cit. on p. 10).

[359] H. Stevens. "SciFi meets GPU, Tracking performance and GPU trigger studies for the SciFi Tracker". PhD thesis. TU Dortmund University, 2021. DOI: 10.17877/DE290R-22182 (cit. on p. 100).

[360] K. Stockinger and K. Wu. "Bitmap indices for data warehouses". In: *Data Warehouses and OLAP: Concepts, Architectures and Solutions*. IGI Global, 2007, pp. 157–178 (cit. on p. 155).

[361] K. Stockinger, K. Wu, and A. Shoshani. "Strategies for processing ad hoc queries on large data warehouses". In: *Procs. of the ACM Int. Workshop on Data Warehousing and OLAP 2002*. 2002 (cit. on p. 155).

[362] C. Szegedy et al. "Intriguing properties of neural networks". In: *Procs. of the Int. Conference on Learning Representations 2014*. 2014. DOI: arXiv:1312.6199 (cit. on p. 73).

[363] M. Tan and Q. V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *Procs. of the Int. Conference on Machine Learning 2019*. 2019. DOI: arXiv:1905.11946 (cit. on pp. 73, 272).

[364] M. Telgarsky. "Benefits of depth in neural networks". In: *Procs. of the Conference on Learning Theory 2016*. Ed. by V. Feldman, A. Rakhlin, and O. Shamir. Vol. 49. Proceedings of Machine Learning Research. Columbia University, New York, New York, USA: PMLR, June 2016, pp. 1517–1539. URL: https://proceedings.mlr.press/v49/telgarsky16.html (cit. on p. 74).

[365] "Teraelectronvolt emission from the γ-ray burst GRB190114C". In: *Nature* 575.7783 (2019), pp. 455–458 (cit. on p. 41).

[366] A. N. Tikhonov. "On the solution of ill-posed problems and the method of regularization". In: *Doklady Akademii Nauk*. Vol. 151. 3. 1963, pp. 501–504 (cit. on p. 294).

[367] A. N. Tikhonov, A. Goncharsky, V. Stepanov, and A. G. Yagola. *Numerical methods for the solution of ill-posed problems*. Springer Science & Business Media, 1995 (cit. on p. 294).

[368] S. Tilav, P. Desiati, T. Kuwabara, F. Rocco Dominick and Rothmaier, M. Simmons, and H. a. o. Wissing. "Atmospheric Variations as observed by IceCube". In: *Procs. of the Int. Cosmic Ray Conference 2009*. 2009 (cit. on p. 311).

[369] S. Tilav, T. K. Gaisser, D. Soldin, and P. Desiati. "Seasonal variation of atmospheric muons in IceCube". In: *arXiv: Physics* (Sept. 2019). DOI: arXiv:1909.01406 (cit. on p. 311).

[370] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. “Large Margin Methods for Structured and Interdependent Output Variables”. In: *Journal of Machine Learning Research* 6 (2005), pp. 1453–1484 (cit. on p. 53).

[371] G. Tsoumakas, I. Partalas, and I. Vlahavas. “An ensemble pruning primer”. In: *Applications of supervised and unsupervised ensemble methods*. Springer, 2009 (cit. on p. 69).

[372] V. G. Tusher, R. Tibshirani, and G. Chu. “Significance analysis of microarrays applied to the ionizing radiation response.” In: *Proceedings of the National Academy of Sciences of the United States of America* 98.9 (Apr. 2001), pp. 5116–5121. DOI: http://dx.doi.org/10.1073/pnas.091062498 (cit. on p. 57).

[373] R. Ulrich, R. Engel, and M. Unger. “Hadronic Multiparticle Production at Ultra-High Energies and Extensive Air Showers”. In: *Physical Review D* 83 (2011), p. 054026 (cit. on pp. 137, 138).

[374] M. Unger. “New Results from the Cosmic-Ray Program of the NA61/SHINE facility at the CERN SPS”. In: *Proceeding of science* ICRC2019 (2020), p. 446 (cit. on p. 141).

[375] L. G. Valiant. “A Theory of the Learnable”. In: *Communications of the ACM* 27.11 (1984), pp. 1134–1142 (cit. on p. 64).

[376] L. Van Der Maaten and G. Hinton. “Visualizing data using t-SNE”. In: *Journal of Machine Learning Research* (2008) (cit. on p. 184).

[377] J. Van Santen. “Neutrino interactions in icecube above 1 tev constraints on atmospheric charmed-meson production and investigation of the astrophysical neutrino flux with 2 years of icecube data taken 2010–2012”. PhD thesis. The University of Wisconsin-Madison, 2014 (cit. on p. 216).

[378] V. Vapnik and A. Sterin. “On structural risk minimization or overall risk in a problem of pattern recognition”. In: *Automation and Remote Control* 10.3 (1977), pp. 1495–1503 (cit. on p. 55).

[379] V. N. Vapnik and A. Y. Chervonenkis. “On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities”. In: *Theory of Probability and its Applications* 16.2 (1971), pp. 264–280. URL: http://link.aip.org/link/?TPR/16/264/1 (cit. on pp. 55, 65).

[380] V. N. Vapnik. *The Nature of Statistical Learning Theory (Information Science and Statistics)*. 2nd. New York: Springer, 1999 (cit. on pp. 23, 65).

[381] V. N. Vapnik. *Statistical Learning Theory*. Chichester, GB: Wiley-Interscience, Sept. 1998. URL: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20%5C&path=ASIN/0471030031 (cit. on pp. 55, 57, 58).

[382] M. Vasileiou. “Strangeness production with ALICE at the LHC”. In: *Physica Scripta* 95.6 (2020), p. 064007 (cit. on p. 139).

[383] P. Veres et al. “Observation of inverse Compton emission from a longγ-ray burst”. In: *Nature* 575.7783 (2019), pp. 459–463 (cit. on p. 41).

[384] J. Vreeken, M. van Leeuwen, and A. Siebes. “Krimp: mining itemsets that compress.” In: *Data Mining and Knowledge Discovery* 23 (2011), pp. 169–214 (cit. on p. 55).

[385] W. Wagner. “Design and Realisation of a new AMANDA Data Acquisition System with Transient Waveform Recorders”. PhD thesis. TU Dortmund University, Oct. 2004 (cit. on p. 33).

[386] R. Weber, H.-J. Schek, and S. Blott. “A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces”. In: *VLDB – International Conference on Very Large Data Bases* 98 (1998), pp. 194–205 (cit. on pp. 155, 157).

[387] D. Williams, D. Xu, and P. Zarzhitsky. “Detecting Tau Neutrinos in IceCube with Double Pulses”. In: *Proceeding of science (ICRC2013)* 643 (2013) (cit. on p. 214).

[388] D. Wolpert and W. Macready. “No Free Lunch Theorems for Optimisation”. In: *IEEE Transactions on Evolutionary Computation* 1 (1997), pp. 67–82 (cit. on p. 62).

[389] D. H. Wolpert. “The Lack of A Priori Distinctions Between Learning Algorithms”. In: *Neural Computation* 8.7 (Oct. 1996), pp. 1341–1390. DOI: https://doi.org/10.1162/neco.1996.8.7.1341 (cit. on p. 62).

[390] S. Wrobel. *Concept Formation and Knowledge Revision*. Dordrecht: Kluwer Academic Publishers, 1994 (cit. on p. 7).

[391] D. Wu, B. J. Lance, and T. D. Parsons. “Collaborative filtering for brain-computer interaction using transfer learning and active class selection”. In: *PloS one* 8.2 (2013) (cit. on p. 114).

[392] K. Wu et al. “FastBit: interactively searching massive data”. In: *Journal of Physics: Conference Series* 180.1 (2009) (cit. on p. 155).

[393] T. Wu, H. Shyng, J. Chou, B. Dong, and K. Wu. “Indexing blocks to reduce space and time requirements for searching large data files”. In: *Procs. of the IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing 2016*. IEEE. 2016, pp. 398–402 (cit. on p. 156).

[394] J. Xie, R. Girshick, and A. Farhadi. “Unsupervised Deep Embedding for Clustering Analysis”. In: *Procs. of the Int. Conference on Machine Learning 2016*. Vol. 1. 2016, pp. 740–749. DOI: arXiv:1511.06335 (cit. on p. 184).

[395] R. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec. “Hierarchical Graph Representation Learning with Differentiable Pooling”. In: *Advances in Neural Information Processing Systems 31: Procs. of the 2018 Conference*. 2018 (cit. on p. 29). **SFB876-A6**

[396] B. Zadrozny and C. Elkan. “Transforming classifier scores into accurate multiclass probability estimates”. In: *Procs. of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2002*. ACM, 2002, pp. 694–699 (cit. on p. 302).

[397] M. Zukowski, S. Héman, N. Nes, and P. Boncz. “Cooperative scans: dynamic bandwidth sharing in a DBMS”. In: *Procs. of the Int. Conference on Very Large Data Bases 2007*. 2007, pp. 723–734 (cit. on p. 156).

[398] M. Zukowski, S. Héman, N. Nes, and P. A. Boncz. “Super-Scalar RAM-CPU Cache Compression”. In: *Procs. of the IEEE Int. Conference on Data Engineering 2006*. Atlanta, GA, USA, Apr. 2006, pp. 59–70 (cit. on p. 159).

Index

https://doi.org/10.1515/9783110785968-012

List of Contributors

Editors

Morik, Katharina, Prof. Dr. TU Dortmund University, Department of Computer Science, Head of the chair for Artificial Intelligence & Speaker of the Collaborative Research Center 876 on Providing Information by Resource-Constrained Data Analysis, katharina.morik@tu-dortmund.de

Rhode, Wolfgang, Prof. Dr. Dr. TU Dortmund University, Department of Physics, Head of the group for Astroparticle Physics, Hon.-Professor Ruhr University of Bochum, CRC 876, wolfgang.rhode@tu-dortmund.de

Contributors

Alameddine, Jean-Marco, M. Sc. TU Dortmund University, Department of Physics, Astroparticle Physics Group, CRC 876, jean-marco.alameddine@tu-dortmund.de

Albrecht, Johannes, Prof. Dr. TU Dortmund University, Department of Physics, Professorship for Experimental Flavour Physics (LHCb), CRC 876, johannes.albrecht@tu-dortmund.de

Baack, Dominik, M. Sc. TU Dortmund University, Department of Computer Science, Artificial Intelligence Group and Department of Physics, Astroparticle Physics Group, CRC 876, dominik.baack@tu-dortmund.de

Bunse, Mirko, M. Sc. TU Dortmund University, Department of Computer Science, Artificial Intelligence Group, CRC 876, mirko.bunse@cs.tu-dortmund.de

Buschjäger, Sebastian, M. Sc. TU Dortmund University, Department of Computer Science, Artificial Intelligence Group, CRC 876, sebastian.buschjaeger@tu-dortmund.de

Dembinski, Hans, Dr. TU Dortmund University, Department of Physics, Particle Physics Group (LHCb), hans.dembinski@tu-dortmund.de

Doert, Marlene, Dr. TU Dortmund University, Department of Physics, Astroparticle Physics Group, CRC 876, marlene.doert@tu-dortmund.de

Elsässer, Dominik, PD Dr. TU Dortmund University, Department of Physics, Astroparticle Physics Group, dominik.elsaesser@tu-dortmund.de

Erdmann, Johannes, Prof. Dr. RWTH Aachen University, III. Physikalisches Institut A, johannes.erdmann@physik.rwth-aachen.de

Fattorini, Alicia, M. Sc. TU Dortmund University, Department of Physics, Astroparticle Physics Group, CRC 876, alicia.fattorini@tu-dortmund.de

Hünnefeld, Mirco, M. Sc. TU Dortmund University, Department of Physics, Astroparticle Physics Group, CRC 876, mirco.huennefeld@tu-dortmund.de

https://doi.org/10.1515/9783110785968-013

Hymon, Karolin, M. Sc. TU Dortmund University, Department of Physics, Astroparticle Physics Group, CRC 876, karolin.hymon@tu-dortmund.de

Jevtic, Vukan, M. Sc. TU Dortmund University, Department of Physics, Particle Physics Group (LHCb), CRC 876, vukan.jevtic@tu-dortmund.de

Lindemann, Thomas, M. Sc. TU Dortmund University, Department of Computer Science, Databases and Information Systems Group, CRC 876, thomas.lindemann@cs.tu-dortmund.de

Linhoff, Lena, Dr. TU Dortmund University, Department of Physics, Astroparticle Physics Group, lena.linhoff@tu-dortmund.de

Linhoff, Maximilian, Dr. TU Dortmund University, Department of Physics, Astroparticle Physics Group, CRC 876, maximilian.linhoff@tu-dortmund.de

Meier, Gerwin, M. Sc. TU Dortmund University, Department of Physics, Particle Physics Group (LHCb), CRC 876, gerwin.meier@tu-dortmund.de

Meier, Maximilian, M. Sc. TU Dortmund University, Department of Physics, Astroparticle Physics Group, CRC 876, mmeier@icecube.wisc.edu

Nickel, Lukas, M. Sc. TU Dortmund University, Department of Physics, Astroparticle Physics Group, CRC 876, lukas.nickel@tu-dortmund.de

Pfahler, Lukas, M. Sc. TU Dortmund University, Department of Computer Science, Artificial Intelligence Group, CRC 876, lukas.pfahler@tu-dortmund.de

Ruhe, Tim, Dr. TU Dortmund University, Astroparticle Physics Group, Department of Physics, Astroparticle Physics Group, CRC 876, tim.ruhe@tu-dortmund.de

Schellenberg, Margarete, Dr. TU Dortmund University, Department of Physics, Particle Physics Group (LHCb), CRC 876, margarete.schellenberg@tu-dortmund.de

Schmelling, Michael, aplProf. Dr. TU Dortmund University and MPI for Nuclear Physics Heidelberg, Particle Physics (LHCb), michael.schmelling@mpi-hd.mpg.de

Soedingrekso, Jan, Dr. TU Dortmund University, Department of Physics, Astroparticle Physics Group, CRC 876, jan.soedingrekso@tu-dortmund.de

Stevens, Holger, Dr. TU Dortmund University, Department of Physics, Particle Physics Group (LHCb), CRC 876, holger.stevens@tu-dortmund.de

Spaan, Bernhard, Prof. Dr. † TU Dortmund University, Department of Physics, Head of the group for Particle Physics (LHCb), CRC 876

Teubner, Jens, Prof. Dr. TU Dortmund University, Department of Computer Science, Databases and Information Systems Group, CRC 876, jens.teubner@cs.tu-dortmund.de

Technical Editors

Becker, Andreas, Dr. TU Dortmund University, Department of Computer Science, CRC 876, andreas3.becker@tu-dortmund.de

Buß, Jens, Dr. TU Dortmund University, Department of Computer Science, Managing Director of the Collaborative Research Center 876, jens.buss@tu-dortmund.de

Acknowledgment

Part of the work on this book is the result of research of the Collaborative Research Center 876 "Providing Information by Resource-Constrained Analysis", which was funded from 2011–2022 by the Deutsche Forschungsgemeinschaft (DFG) under DFG project number 124020371, see: https://gepris.dfg.de/gepris/projekt/124020371?language=en.