THE APPLICATION OF MACHINE LEARNING TECHNIQUES TOWARDS THE OPTIMIZATION OF HIGH ENERGY PHYSICS EVENT SIMULATIONS WITHIN THE ALICE TRD AT CERN



**Christiaan Gerhardus Viljoen**

Department of Statistics, Department of Physics

Faculty of Science

University of Cape Town

This dissertation is submitted in partial fulfilment of the Degree of Master of Science

Dedicated to my mother, Elizabeth Suzanna Bloem Viljoen, who has always inspired me to follow my higher passions, despite the myriad difficulties that life makes us face; and to search fearlessly and incessantly for the deeper truths underlying our everyday world.

“A man may imagine things that are false, but he can only understand things that are true, for if the things be false, the apprehension of them is not understanding.”

—Sir Isaac Newton

Declaration

This dissertation is the result of my own work and includes nothing, which is the outcome of work done in collaboration except where specifically indicated in the text.

It has not been previously submitted, in part or whole, to any university of institution for any degree, diploma, or other qualification.

In accordance with the Department of Statistics guidelines, this thesis is does not exceed 20,000 words.

Signed:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Christiaan Gerhardus Viljoen, B.Sc., B.Sc. (Med.) (Hons.),

Cape Town

Summary / Abstract (Delete as Preferred)

The summary must be written in English and should consist of a piece of connected prose forming an abstract of the dissertation and be about 300 words in length. If at all possible, it should be accommodated on one side of A4 sized paper. It should bear the candidate's name and the exact title of the dissertation at the head of the page.

If candidates submit a soft-bound dissertation in the first instance, they will need when submitting the final hard-bound copy of the dissertation, to provide a further, loose-leaf, copy of this summary, identical to that bound into the final version, for the University Library file.

The summary will be considered by the examiners and, if the dissertation is approved, be deposited in the University Library for consultation and inter-library loan.

Acknowledgements

Firstly, I would like to thank my father, Christiaan Gerhardus Viljoen, for all the support – material, emotional and financial – he has selflessly provided to me throughout my life, and particularly towards my higher education journey. You have no idea how much appreciation I have for all the sacrifices you have made for me, and all the advice you have given me.

Secondly, I want to thank my aunt, Professor Emma Ruttkamp-Bloem, for all the mentoring she has provided to me in navigating the world of academia, and for the inspiration that her own academic career instils in me.

Thirdly, I want to thank Dr Thomas Dietel for providing me with this immense opportunity to be part of the largest scientific experiment in human history, and for the rigorous scientific guidance that he has, and continues to provide to me.

Lastly, I would like to thank my larger family, on both my father’s and mother’s side, for providing the loving and stable environment that makes any place we assemble Home.

Table of Contents

[1 Introduction 1](#_Toc1378325)

[1.1 Background 2](#_Toc1378326)

[1.2 Aims & Goals 3](#_Toc1378327)

[1.3 Summary of Work Done & Major Findings 4](#_Toc1378328)

[1.4 The Structure & Organization of this Dissertation 5](#_Toc1378329)

[2 High Energy Physics & The CERN Experiment 6](#_Toc1378330)

[2.1 A Brief History of Atomic Theory 7](#_Toc1378331)

[2.2 The Standard Model of Particle Physics 7](#_Toc1378332)

[2.2.1 Introduction 8](#_Toc1378333)

[2.2.2 The Fundamental Particles 8](#_Toc1378334)

[2.2.3 The Fundamental Forces 9](#_Toc1378335)

[2.2.4 The Higgs Boson 10](#_Toc1378336)

[2.3 The Quark Gluon Plasma 10](#_Toc1378337)

[2.4 Interactions of Particles with Matter 10](#_Toc1378338)

[2.4.1 The Bethe-Bloch Curve 11](#_Toc1378339)

[2.4.2 Transition Radiation 13](#_Toc1378340)

[2.5 The CERN Experiment 13](#_Toc1378341)

[2.5.1 Hardware 14](#_Toc1378342)

[2.5.2 Software 18](#_Toc1378343)

[3 The ALICE Collaboration, The Quark-Gluon Plasma and the Transition Radiation Detector In-Depth 22](#_Toc1378344)

[3.1 Objectives of the ALICE Experiment 24](#_Toc1378345)

[3.2 Gas Detectors 25](#_Toc1378346)

[3.3 The ALICE Detector 26](#_Toc1378347)

[3.3.1 The Transition Radiation Detector 27](#_Toc1378348)

[4 Deep Learning 29](#_Toc1378349)

[4.1 Deep Learning within the Context of Artificial Intelligence and Machine Learning 30](#_Toc1378350)

[4.2 Mathematical Background for Deep Learning 30](#_Toc1378351)

[4.2.1 Rosenblatt’s Perceptron 30](#_Toc1378352)

[4.2.2 Deep Feedforward Neural Networks 31](#_Toc1378353)

[4.3 Convolutional Neural Networks 35](#_Toc1378354)

[4.4 Variational Autoencoders 36](#_Toc1378355)

[4.5 Generative Adversarial Networks 37](#_Toc1378356)

[5 Data 38](#_Toc1378357)

[6 Methods 39](#_Toc1378358)

[6.1 Software Environment 40](#_Toc1378359)

[6.1.1 AiROOT 40](#_Toc1378360)

[6.1.2 R Statistical Software 40](#_Toc1378361)

[6.1.3 ROOTR 40](#_Toc1378362)

[6.1.4 Keras & Tensorflow 40](#_Toc1378363)

[6.1.5 Utilities 40](#_Toc1378364)

[6.1.6 Python 40](#_Toc1378365)

[6.2 Data Extraction from WLCG 42](#_Toc1378366)

[7 Results 43](#_Toc1378367)

[8 Discussion 44](#_Toc1378368)

[9 Conclusion 45](#_Toc1378369)

[10 Bibliography 47](#_Toc1378370)

[11 Appendices 49](#_Toc1378371)

[Appendix A: Plotting the Bethe-Bloch Equation 50](#_Toc1378372)

[Appendix B: Plotting Binary Cross-Entropy 57](#_Toc1378373)

List of Tables

[Table 1: The twelve fundamental fermions. 9](#_Toc1378374)

List of Figures

[Figure 1: Bethe-Bloch curve for a pion moving at relativistic speeds through silicon medium 11](#_Toc1378375)

[Figure 2: Bethe-Bloch curve for an electron moving at relativistic speeds through a silicon medium 12](#_Toc1378376)

[Figure 3: CERN facilities in geographical context (6). 14](#_Toc1378377)

[Figure 4: The LHC Proton Source, connected to the Duoplasmatron device, which strips electrons off Hydrogen molecules, to produce the beams of protons which eventually collide within the LHC (12) 15](#_Toc1378378)

[Figure 5: The CERN accelerator complex (15). 16](#_Toc1378379)

[Figure 6: Illustration of the descent towards zero, of the Binary Cross Entropy Loss Function as ŷ, or , approaches the true y. 33](#_Toc1378380)

List of Abbreviations and Acronyms

|  |  |
| --- | --- |
|  |  |
| ALICE | A Large Ion Collider Experiment |
| TRD | Transition Radiation Detector |
| CERN | European Organization for Nuclear Research |
| QGP | Quark Gluon Plasma |
| LHC | Large Hadron Collider |
| WLCG | Worldwide LHC Computing Grid |
| QCD | Quantum Chromodynamics |
| QGP | Quark-Gluon Plasma |
| ML | Machine Learning |
| Pb-Pb | Lead-Lead Collisions |
|  | Electron |
|  | Pion |
| QED | Quantum Electrodynamics |
| p | Proton |
| n | Neutron |
|  | Electron Neutrino |
|  |  |
|  |  |

List of Appendices

[Appendix A: Plotting the Bethe-Bloch Equation 50](#_Toc1378381)

[Appendix B: Plotting Binary Cross-Entropy 57](#_Toc1378382)

# Introduction

## Background

This Masters Dissertation seeks to apply cutting edge techniques in Machine Learning (ML) towards the simulation of High Energy Physics (HEP) collision events, which routinely occur at the Large Hadron Collider (LHC) as part of the ongoing fundamental research conducted by the Counsel for European Nuclear Research (CERN).

More specifically, the focus of this thesis centres around the development of Deep Generative Models which are able to produce datasets that are indistinguishable from data produced by the Transition Radiation Detector (TRD) at the ALICE (A Large Ion Collider Experiment) collaboration at CERN, during Lead-Lead (Pb-Pb) heavy ion collisions.

## Aims & Goals

## Summary of Work Done & Major Findings

## The Structure & Organization of this Dissertation

# High Energy Physics & The CERN Experiment

## A Brief History of Atomic Theory

The earliest correct model for the atom can be traced back to 400 BCE, when Democritus proposed that the entire universe consisted of fundamental particles, or “Atoms”, which cannot be divided any further.

In 1803, Dalton refined this model to state that these indivisible atoms can have distinguishing chemical and physical traits and that they combine to form chemical compounds.

Then, in 1987, JJ Thompson discovered the electron and proposed an incorrect theory for subatomic structure in which negatively charged electrons were embedded within positive charges within the atom.

Rutherford, Marsder and Geiger disproved this model in 1911, with their seminal alpha-particle scattering experiment and put forth a more accurate model for the atom, in which most of the atom consists of empty space, with a dense core of positively charged protons.

In 1913, Bohr refined this model further, indicating that electrons orbit the positively charged atomic core at distinct energy levels. While this model did explain the emission spectrum of Hydrogen, it could not explain the emission spectra of any of the other elements.

Between 1924 – 1928, De Broglie, Heisenberg and Schrödinger each separately developed a quantum paradigm, where electrons have wave-like properties and appear in much more complex orbitals. This is still the accepted theory of atomic structure today.

There have been some refinements made to the quantum theory, as new information has come to light: a neutral subatomic particle, the neutron, was discovered in 1932, which solved the puzzle of why atoms were found to be nearly twice as heavy as expected based on proton number; this discovery also disproved Dalton’s second law, which stated that all atoms of a specific element were identical, and resulted in the concept of isotopes (atoms with the same number of protons, but differing numbers of neutrons). In the same year, Cockroft and Walton split the atom for the first time, by bombarding Lithium atoms with electrons, splitting them into two Helium particles.

The 1950s brought about a new era in nuclear physics, in which particle accelerators with collision energies of a few hundreds of MeVs became affordable, along with cosmic ray and inelastic proton-scattering experiments; since this time, a whole host of subatomic elements have been discovered, many of which are unstable. The discovery of these new particles has led, over time, to the development and refinement of the modern Standard Model of Particle Physics.

## The Standard Model of Particle Physics

### Introduction

The Standard Model of Particle Physics is a framework which allows us to understand the fundamental structure and dynamics of our universe in terms of elementary particles, where all interactions between elementary particles are similarly facilitated by an exchange of particles. In summary, based on our current understanding, our entire universe consists of a very sparse array of fundamental particles once we delve into the subatomic realm (1).

At an energy scale of electron Volts (an electron Volt is a unit of energy, equivalent to the amount of work required to accelerate a single electron through a potential difference of 1 Volt), the low energy manifestation of Quantum Electrodynamics (QED) allows atoms to exist in bound states with negatively charged electrons () orbiting a positively charged nucleus consisting of positively charged protons () and electrically neutral neutrons (), based on the electrostatic attraction of these opposing electrical charges (1).

Quantum mechanics explains the emergence of unique physical properties in different elements, which arise from their exact electronic structures. Quantum Chromodynamics (QCD) is the fundamental theory of the strong interaction, which binds protons and neutrons together within the nucleus of the atom. Similarly, at this energy scale, the weak force causes nuclear β-decays of radioactive isotopes and is involved in the nuclear fusion processes that occur within stars; the nearly massless electron neutrino ) is produced during both of the abovementioned processes (1).

Therefore, almost all physical phenomena that occur under normal circumstances can be explained by the Electromagnetic-, Strong- and Weak Forces, Gravity (which is very weak, but explain the large-scale structure of the universe), and just four fundamental particles: the electron, proton, neutron and electron neutrino (1).

### The Fundamental Particles

At higher energy scales, of the order of electron Volt (or giga-electron Volt, 1 GeV), protons and neutrons are understood to be bound states of truly fundamental particles called quarks, in the following manner: protons consist of two up-quarks and a down-quark p(uud), whereas neutrons consist of two down-quarks and an up-quark n(ddu) (1).

At the lowest energy level of the standard model, the first generation of particles are then the electron, electron neutrino, the up-quark and the down-quark; these are currently considered to be truly elementary, in that they cannot be subdivided (1).

Higher energy scales, such as those achieved at modern particle accelerators, result in the second and third generation of the four elementary particles; these are heavier versions of the first generation: for example, the muon () is essentially a version of an electron which is 200 × heavier than a low energy electron, i.e. . The tau-lepton () is the third generation of the electron, and is much heavier, i.e. . These mass differences do have physical consequences, but the fundamental properties and interactions of the various generations remain identical (1).

Current experimental evidence indicates that there are no further generations than these three, and so all matter in the universe seems to be circumscribed by the following twelve fundamental fermions, reproduced from (1):

Table 1: The twelve fundamental fermions.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Leptons | | | | Quarks | | |
|  | Particle | Q | Mass/GeV | Particle | Q | Mass/GeV |
| First Generation | Electron () | -1 | 0.005 | Down (d) | -1/3 | 0.003 |
| Neutrino () | 0 | < | Up (u) | +2/3 | 0.005 |
| Second Generation | Muon () | -1 | 0.106 | Strange (s) | -1/3 | 0.1 |
| Neutrino () | 0 | < | Charm (c) | +2/3 | 1.3 |
| Third Generation | Tau () | -1 | 1.78 | Bottom (b) | -1/3 | 4.5 |
| Neutrino () | 0 | < | Top (t) | +2/3 | 174 |

While it is accepted that neutrinos are not massless, their masses are so small that they have not been precisely determined, however, the upper bounds for the estimated masses for neutrinos are around 9 orders of magnitude smaller than the other fermions (1).

The Dirac equation describes the state of each of the twelve fundamental fermions and indicates that for each fermion there is an antiparticle which has the same mass but opposite charge, which is indicated by a horizontal bar over the particle’s symbol, or a charge symbol of the opposite sign, e.g. the anti-down quark is indicated by d̅, whereas the antimuon is indicated by (1).

Interactions between particles are facilitated by the four fundamental forces, but the effect of gravity at this scale is sufficiently negligible that it can be ignored without loss of accuracy. All particles take part in weak interactions and are therefore subject to the weak force. The neutrinos are all electrically neutral and therefore are not involved in electromagnetic interactions and are, so to speak, invisible to this force. Quarks carry what is termed as “colour charge” by QCD and are therefore the only particles that feel the strong force (1).

The strong force confines quarks to confined states within hadrons and are therefore not freely observed under normal circumstances (1).

### The Fundamental Forces

Classical electromagnetism explained the electrostatic interaction between particles using a scalar potential, Newton himself that matter could interact with any other matter without the mediation of direct contact (1).

Quantum Field Theory circumvents this non-material explanation and encompasses the description of each of the fundamental forces. Electromagnetism is explained by Quantum Electrodynamics (QED), the Strong Force by Quantum Chromodynamics (QCD), the weak force by the Electroweak Theory (EWT), Gravity has not been explained by the Standard Model yet; therefore, Einstein’s General Theory of Relativity is still the best explanation of this force, but it falls within the bounds of Classical Physics. As such, the search to incorporate gravity into the Standard Model is an ongoing area of research and has resulted in exciting new theoretical research avenues such as string theory and loop quantum gravity arising (1).

Looking at electromagnetism, the interaction between charged particles occurs via the exchange of massless virtual photons, which explains momentum transfer via a particle exchange and circumventing the issue of a non-physical potential as the medium of interaction (1).

Similarly, there are virtual particles (gauge bosons) for both the Strong Force (i.e. the massless gluon) and Weak Force (i.e. and bosons, which are around 80 times heavier than the proton and the Z boson, which facilitates a weak neutral-current interaction). The gauge bosons all have spin 1, compared to the fermions whom all have spin ½ (1).

### The Higgs Boson

The Higgs Boson, whose existence was confirmed by the CMS and ATLAS collaborations at CERN in 2012, but proposed in 1964 by three separate theoretical papers, breaks rank with the other particles outlined by the standard model in that it is many orders of magnitude heavier and is a scalar particle which endows other standard model particles with mass, a property without which all particles would constantly move at the speed of light, (1).

The Higgs boson manifests as a disturbance of the Higgs field, which is non-zero in a vacuum, in contrast to the other fundamental particles which all have a vacuum expectation value of zero, i.e. (1).

In QFT, an expectation value is a real number calculated as the average over the expected values of an observable, weighted according to their respective likelihood.

On their own, all particles are massless, but interacting with the Higgs Field, which is always non-zero, the Higgs mechanism gives them their distinguishing masses (1).

## The Quark Gluon Plasma

## Interactions of Particles with Matter

In order to study subatomic particle, they need to be detected. Most particles produced during High Energy Physics Experiments are unstable and therefore decay within a specific characteristic mean lifetime . Those particles with will traverse several meters before decaying and are therefore directly detectable by particle detectors installed at the Large Hadron Collider (LHC) at CERN. Particles with shorter lifespans are usually detected indirectly, by the interaction of their decay products with detector material (1).

### The Bethe-Bloch Curve

The Bethe-Bloch equation describes the energy lost by a charged particle moving at relativistic speed through a medium, as a result of electromagnetic interactions with atomic electrons. A single charged particle with velocity , passing through a medium with atomic number and density , will lose energy as a result of ionisation of the medium, as a function the distance travelled in the medium, according to the Bethe-Bloch formula (1):

Please see Appendix A for the code used to plot Figure 1 and Figure 2, illustrating the characteristic energy loss curves for the two subatomic particles studied in this project, the pion and the electron .



Figure 1: Bethe-Bloch curve for a pion moving at relativistic speeds through silicon medium



Figure 2: Bethe-Bloch curve for an electron moving at relativistic speeds through a silicon medium

### Transition Radiation

## The CERN Experiment

At the end of 1951, a resolution was agreed upon to establish a European Council for Nuclear Research (CERN: *Conseil Européen pour la Recherche Nucléaire)* at an intergovernmental UNESCO meeting in Paris. The final draft of the CERN commission was signed by twelve nations in 1953 (2).

Today, CERN is a truly international organization, with 22 member states, who contribute to operating costs and are involved in major decision making, many countries with observer status, and even more non-member countries with co-operation agreements, including South Africa (3).

CERN’s research mandate revolves around finding answers to fundamental questions about the structure and evolution of our universe, as well as its origins; it aims to achieve these goals by providing access to its particle accelerator facilities and compute resources to international researchers, who perform research that advances the forefront of human knowledge, for the benefit of humanity as a whole. As such, CERN is politically neutral and advocates for evidence-based reasoning, knowledge transfer from fundamental research to industry and grass-roots development of future generations of scientists and engineers (4).

### Hardware

In order to fulfil its ambitious goals, CERN’s facilities, located under the Franco-Swiss border (see Figure 3 for geographical context), boasts an intricate system of particle accelerators and -detectors and a data centre with over 174,000 processor cores, 150,000 Terabytes (TB) of Disk space and over 1,000 TB of random access memory (RAM) (5); this main datacentre is connected both to its extension in Budapest, Hungary and the multi-tier Worldwide LHC Computing Grid (WLCG), all of which operates at a data transfer rate of around 10 Gigabytes/second (GiB/s).

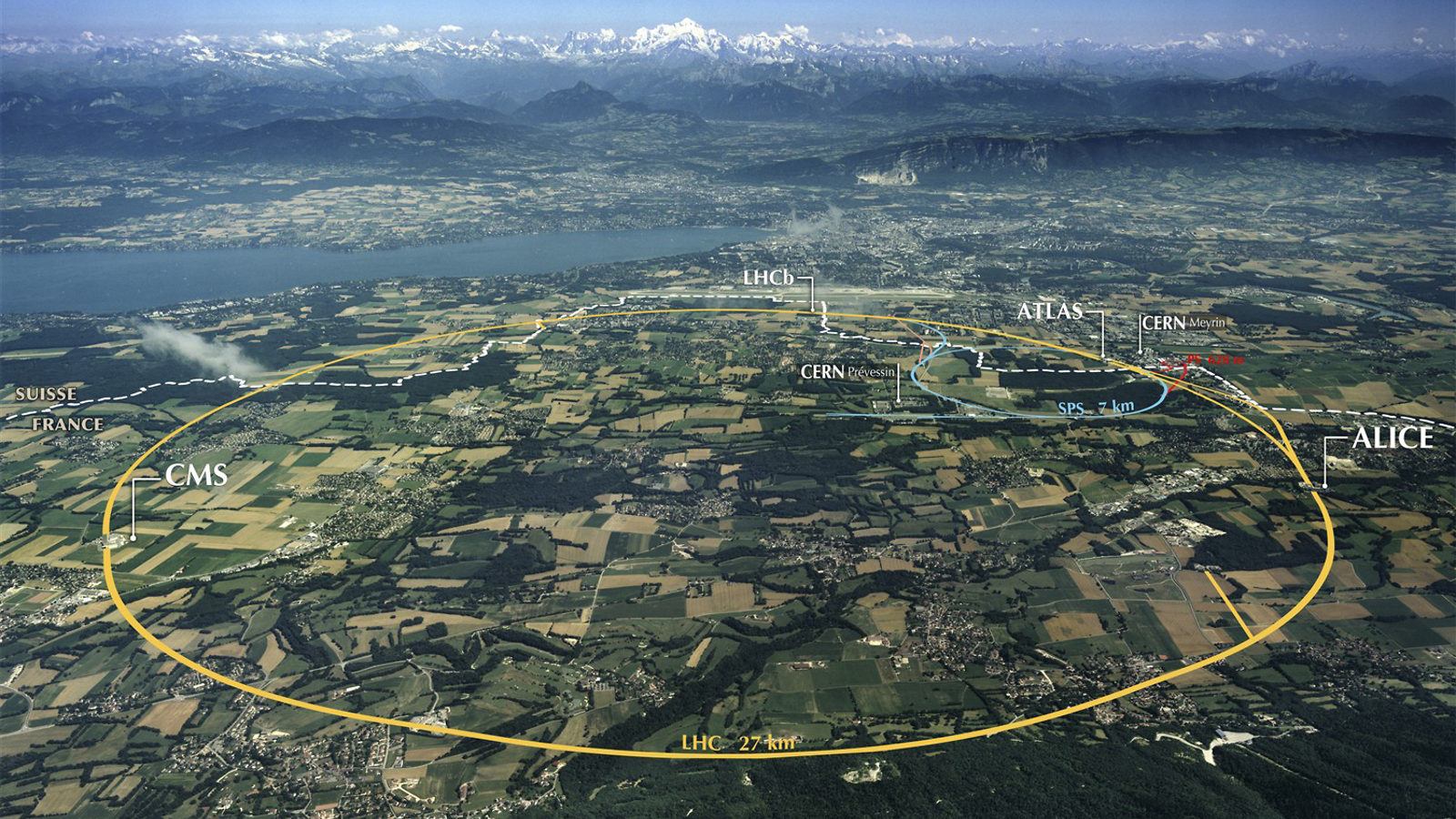


Figure 3: CERN facilities in geographical context (6).

#### Accelerators

##### The LHC Compared to Accelerators from Other Experiments World-wide

At a circumference of 27 km, the Large Hadron Collider (LHC) is currently the largest particle accelerator in the world (7). To put this into perspective, the Relativistic Heavy Ion Collider (RHIC), located at the Brookhaven National Laboratory in New York, has a circumference of 3.8 km (8), Fermilab’s Tevatron, which is no longer in operation, was 6.3 km in circumference (9) and the KEKB accelerator in Tsukuba, Japan also has a circumference of around 3 km (10).

It is also the most powerful particle accelerator in the world, with a centre of mass energy of 13 Tera-electron-Volts ( (7)), compared to RHIC, which operates at (8), the Tevatron, which reached (9) and KEKB at (11).

##### The LHC

The LHC, located 50-175 m underground, is the final step in a chain of successive accelerators feeding beams of accelerated particles into each other at increasing energies, as can be seen in Figure 5.

The LHC’s proton source is a bottle of compressed Hydrogen, which releases its contents into a Duoplasmatron device, which subsequently surrounds the molecules with an electrical field and separates it into its constituent protons and electrons (12).



Figure 4: The LHC Proton Source, connected to the Duoplasmatron device, which strips electrons off Hydrogen molecules, to produce the beams of protons which eventually collide within the LHC (12)

A linear accelerator (LinAc2) injects these protons into a booster ring (PS booster) at an energy of 50 MeV, where proton beams are accelerated up to 1.4 GeV, before being injected into the Proton Synchrotron, which accelerates them up to 25 GeV, the Super Proton Synchrotron is the final intermediate step before proton beams enter the LHC and proton beams reach an energy of 450 GeV around this accelerator beam before they begin their 20 minute acceleration around the LHC before reaching an energy of 6.5 TeV each (13).

To calculate the centre-of-mass energy at collision-time, we do:

= 13 TeV (1)

This equation is derived from the relativistic relationship between energy and momentum, where the rest energy (invariant mass of a particle) is the familiar and the kinetic energy from acceleration is . To simplify the equations, the speed of light, is set at a constant (14).

An entirely different protocol is employed to generate the lead ions used in heavy-ion collisions (pPb, PbPb) studied at ALICE. A highly pure Lead (Pb) sample is heated up to a temperature of 800°C and the resulting Pb vapour is ionized by an electron current, which manages to strip a maximum of 29 electrons from a single Pb atom. Those atoms with higher resulting charge are preferentially selected and accelerated through a carbon foil, which strips most ions to . These ions are accelerated through the Low Energy Ion Ring (LEIR) and subsequently through the PS and SPS, where it is passed through a second foil, which strips the remaining electrons and passes the fully ionized ions to the LHC, where beams of Pb-ions are accelerated up to 2.56 TeV (13), because there are many protons in a single lead ion, the collision energies reached in PbPb collisions reach a maximum of 1150 TeV (13).

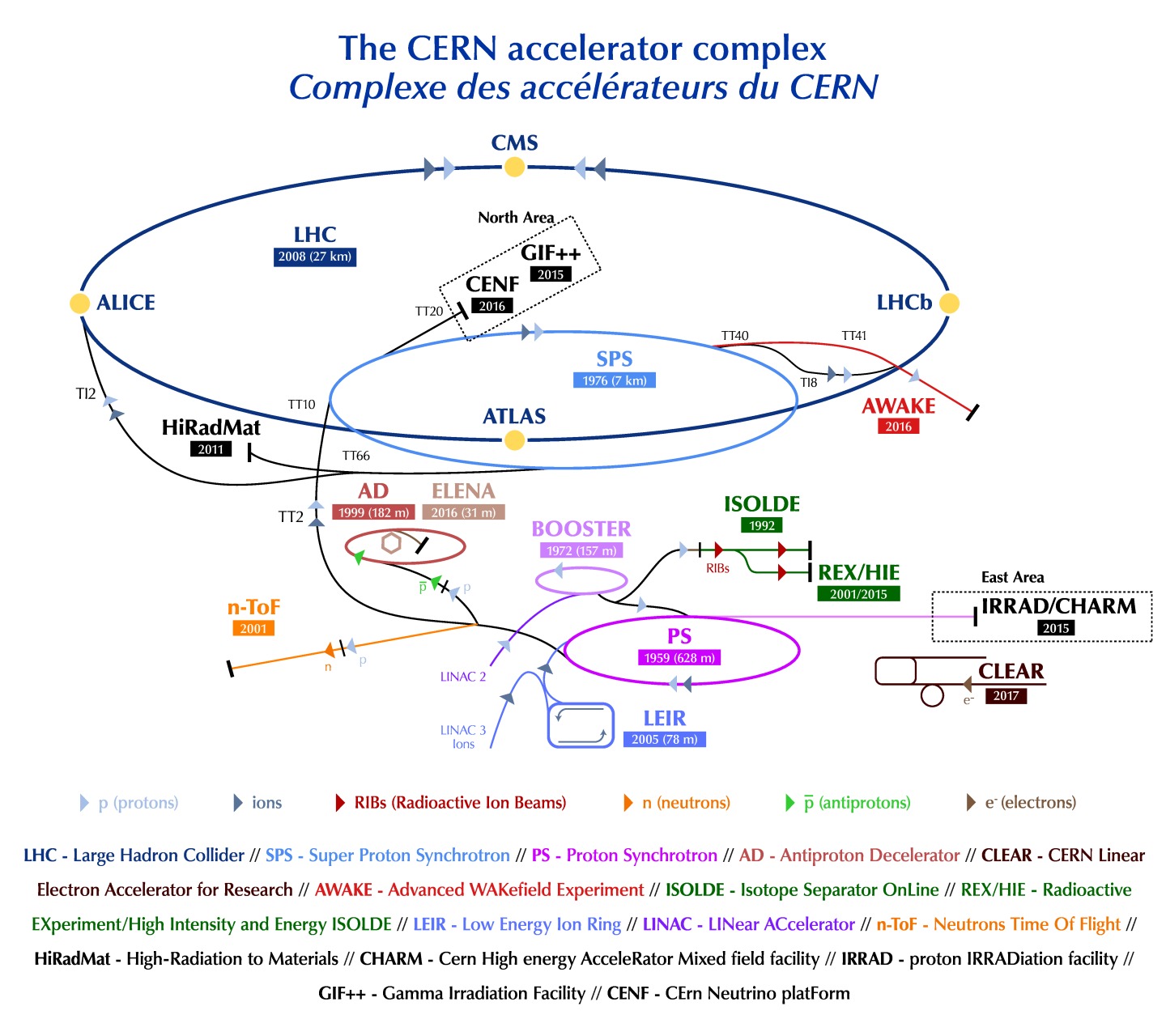


Figure 5: The CERN accelerator complex (15).

In order to achieve these high collision energies, a precise system of 1232 dipole magnets is required to keep particles in their circular orbits, with 392 quadrupole magnets employed to focus the two collision beams. The dipole magnets use niobium-titanium (NbTi) cables at a temperature of 1.9 K (-271.3°C). At these temperatures the cables conduct electricity with no resistance (i.e. they become superconducting) and allow the magnetic field to reach 8.3 Tesla (8.3 T) required to bend the beams around the circular LHC ring (13).

The beams themselves are contained within a vacuum tube emptier than outer space () and are accelerated by electromagnetic resonators and accelerating cavities to 99.9999991% of the speed of light, which means that a beam goes around the 26.659 km LHC ring around 11,000 revolutions/second, resulting in around a billion collisions per second (13).

#### The Four Main CERN Experiments and their Respective Detectors

#### The Worldwide Large Hadron Collider Computing Grid (WLCG)

### Software

#### ROOT

ROOT is an object oriented data analysis platform developed in C++ for High Energy Physics implementations; in addition to its data analysis capabilities, ROOT is also used to transform the petabytes raw data from collision events at the LHC into more compact and useful representations (16).

The basic ROOT framework provides default classes for most common use-cases and as the HEP community pushes research into new frontiers, they can use the object-oriented programming (OOP) approach followed by ROOT to make use of sub-classing and inheritance to extend existing classes. Similarly, the concept of encapsulation keeps the number of global variables to a minimum and increases the opportunity for structural reuse of code. The ROOT forums allow users of the platform to report bugs and suggest fixes and in this way contribute to the platform without being part of the official development team (16).

ROOT is freely available for download from (17) and can be installed using precompiled binaries or built from source using the GNU g++ complier on Unix platforms, such as Linux or MacOSX, Windows 10 64-bit users can make use of the Ubuntu subsystem or locally hosted Linux Virtual Machines to install and use ROOT, but native Microsoft Windows is not supported (16).

Upon installation, running the following line in a Unix terminal

> echo $ROOTSYS

will print the symbolic path to the top of the ROOT directory, e.g.

/Users/gerhard/root

Looking at the contents of this directory, $ROOTSYS/bin contains executables such as the main ROOT executable, daemons for remote ROOT file access and authentication of parallel processing capabilities, etc.

$ROOTSYS/lib contains the libraries for the C++ interpreter, image manipulation, ROOT base classes, as well as interfaces with event generators.

Additional directories exist, i.e. $ROOTSYS/tutorials which contains example .C macro files, $ROOTSYS/test which contains .cxx files and $ROOTSYS/include which contains the .h header files.

ROOT libraries are designed with minimal dependencies and as such are loaded as needed. At runtime, libCore.so (the core library) is always invoked; it is composed of the base-, container-, metadata-, OS specification- and ROOT file compression classes. Additionally, the interactive C++ interpreter library libCling.so is used by all ROOT 6 applications, it features a command line prompt with just-in-time interactive compilation to facilitate rapid application development and testing.

When building executables, libraries containing the needed classes are linked to. Extensive documentation is available online at the ROOT reference guides for ROOT 5 (18), the version of ROOT developed and used for LHC run 1 and run 2; and ROOT 6 (19), the version of ROOT developed for LHC run 3, scheduled to start in 2021 after the second long shut down period (LS2).

#### AliROOT & The Anatomy of an Analysis Task

AliROOT and AliPhysics are built on top of the base ROOT architecture to provide functionality specific to the ALICE collaboration.

C++ classes define all the code in ROOT, AliPhysics and AliROOT and enables the user to create variables (data) and functions (methods) specific to each class, as its members. A class’s variables are usually accessed via the class’s methods (20).

C++ code is split into header (.h) and implementation (.cxx) files, both having the same name as the class being defined. Header files list all the constants, functions and methods contained in a class. Implementation files use a class’s methods to set and get variables’ values in that class.

The concept of inheritance is frequently utilized to prevent unnecessary repetition of code. Child classes inherit common behaviours and attributes from base/ parent classes and define additional methods and variables that are not common to other classes deriving from the base class.

In AliROOT, all analysis tasks inherit from the base class **AliAnalysisTaskSE** (where SE stands for Single Event), which in turn is derived from the base class **AliAnalysisTask.**

All analysis tasks done in AliROOT inherit the following base methods from **AliAnalysisTaskSE**:

AliAnalysisTaskSE::AliAnalysisTaskSE();//constructor1

AliAnalysisTaskSE::AliAnalysisTaskSE(const char\*);//constructor2

AliAnalysisTaskSE::~AliAnalysisTaskSE();//destructor

AliAnalysisTaskSE::UserCreateOutputObjects();//user-defined output objects (results of physics analyses, which can be attached to output files)

AliAnalysisTaskSE::UserExec(Option\_t\*);//event loop, called for each event in the analysis: checks conditions for inclusion, accesses physics objects, fills histograms or other data containers with attributes from event

AliAnalysisTaskSE::Terminate(Option\_t\*); //deallocates memory after all steps in analysis have completed

The final element of an analysis task in AliROOT is the (.C) macro file, which creates and configures an instance of the particular C++ class.

##### The Class Header (.h)

Reproduced and modified from (20):

#ifndef AliAnalysisTaskMyTask\_H //include guard (aids in prevention of double inclusion, which may result from including parent and child classes, leading to multiple definitions for class members)

#define AliAnalysisTaskMyTask\_H //part of include guard

class AliAnalysisTaskMyTask : public AliAnalysisTaskSE //we define a class AliAnalysisTaskMyTask, which inherits from the base class AliAnalysisTaskSE

{

public:

// two class constructors, called when a new instance of the class is created

AliAnalysisTaskMyTask();

AliAnalysisTaskMyTask(const char \*name);

// class destructor, called when this instance of the class is deleted

virtual ~AliAnalysisTaskMyTask();

// called once at beginning of runtime

virtual void UserCreateOutputObjects();

// called for each event

virtual void UserExec(Option\_t\\* option);

// called at end of analysis

virtual void Terminate(Option\_t\\* option);

//class members

private:

AliAODEvent\* fAOD; //!<! pointer to a single input event

TList\* fOutputList; //!<! pointer to an output list, which holds all the output objects of the analysis

TH1F\* fHistPt; //!<! pointer to a histogram containing the transverse momentum (Pt) spectrum

//note that the !<! expression above is seen and evaluated by ROOT and is used in the generation of ROOT documentation

//ClassDef definition:

/// \cond CLASSDEF //surrounding comments for documentation generation

ClassDef(AliAnalysisTaskMyTask, 1); //this is a C pre-processor macro, used when class derives from TObject: it contains member declarations and inserts a few new members into the class, version number is incremented from 1 when definition of class changes

/// \endcond}; //surrounding comments for documentation generation

#endif //part of include guard

##### The Class Implementation (.cxx)

Reproduced and modified from (20):

//include statements for UserCreateOutputObjects:

**#include "TList.h"** *//TList class, an instance of which will contain a histogram in this example*

#include "TH1F.h" //ROOT 1-dimensional histogram class with one float per channel

//include statement for UserExec:

#include "AliAODEvent.h"

//implementation of class constructors:

AliAnalysisTaskMyTask::AliAnalysisTaskMyTask() : AliAnalysisTaskSE(),

//members of the class are initialized in the constructors with their default values, if default values are not specified, these will be filled with random values, which could lead to unexpected behaviour

fAOD{0}, fOutputList{0}, fHistPt{0}

{

// This first constructor is the ROOT IO constructor, memory should not be allocated here

}

//in the second constructor, below, the input and output objects handled by the class are defined

AliAnalysisTaskMyTask::AliAnalysisTaskMyTask(const char\* name) : AliAnalysisTaskSE(name),

fAOD{0}, fOutputList{0}, fHistPt{0}

{

//input object is a TChain

DefineInput(0, TChain::Class());

//output object is a TList

DefineOutput(1, TList::Class());

}

//implementation of the UserCreateOutputObjects class:

AliAnalysisTaskMyTask::UserCreateOutputObjects()

{

// create a new TList that OWNS its objects

fOutputList = new TList();

fOutputList->SetOwner(true);

// create a histogram:

//from ROOT’s online documentation, this is the constructor for a TH1F:

//TH1F (const char \*name, const char \*title, Int\_t nbinsx, Double\_t xlow, Double\_t xup)

//seen below, we give the histogram the pointer name defined in the header file and give the histogram plot the same title, we define the histogram itself to have 100 bins on an x-axis bounded by [0,100]

fHistPt = new TH1F("fHistPt", "fHistPt", 100, 0, 100);

//add the histogram to the output list:

fOutputList->Add(fHistPt);

// add the list to our output file

PostData(1,fOutputList); //calling PostData() notifies client tasks of the fOutPutList data container that its contents have changed

}

//UserExec: the “event loop” (operations defined here are called for each event in the analysis):

AliAnalysisTaskMyTask::UserExec(Option\_t\*)

{

// get an input event from the analysis manager and cast it as an AliAODEvent

fAOD = dynamic\_cast<AliAODEvent\*>(InputEvent());

// check if there actually is an event, and throw a fatal exception with error message if not

if(!fAOD)

::Fatal("AliAnalysisTaskMyTask::UserExec", "No AOD event found, check the event handler.");

// Loop over all the tracks in the event and fill the histogram

// get the number of tracks in the input event

int iTracks{fAOD->GetNumberOfTracks()};

// iterate through all the tracks in the event:

for(int i{0}; i < iTracks; i++) {

//get the current track, cast it as an AliAODTrack

AliAODTrack\* track = static\_cast<AliAODTrack\*>(fAOD->GetTrack(i));

//if the track variable does not exist after the above operation, continue to the next iteration of the loop

if(!track) continue;

// here we do some track selection

if(!track->TestFilterbit(128) continue;

// get the transverse momentum of the track and fill the histogram with this data

fHistPt->Fill(track->Pt());

}

// save the output list

PostData(1, fOutputList);

}

##### The AddTask macro (.C)

Reproduced and modified from (20):

//this file instantiates our class, defines its input and output, and connects it to the analysis manager

AliAnalysisTaskMyTask\* AddMyTask(TString name = "name") {

//get a pointer to the analysis manager

AliAnalysisManager \*mgr = AliAnalysisManager::GetAnalysisManager();

// resolve the name of the output file

TString fileName = AliAnalysisManager::GetCommonFileName();

fileName += ":MyTask"; // create a subfolder in this file

// create an instance of the analysis task

AliAnalysisTaskMyTask\* task = new AliAnalysisTaskMyTask(name.Data());

// add this task to the analysis manager

mgr->AddTask(task);

// connect the manager to the task’s input container

mgr->ConnectInput(task,0,mgr->GetCommonInputContainer());

// connect the manager to the task’s output container (TList)

mgr->ConnectOutput(task,1,mgr->CreateContainer("MyOutputContainer", TList::Class(), AliAnalysisManager::kOutputContainer, fileName.Data()));

// important: return a pointer to this task

return task;

}

##### Running an Analysis Task

#### MonALISA

#### AliEn

# The ALICE Collaboration, The Quark-Gluon Plasma and the Transition Radiation Detector In-Depth

## Objectives of the ALICE Experiment

## Gas Detectors

## The ALICE Detector

### The Transition Radiation Detector

# Deep Learning

## Deep Learning within the Context of Artificial Intelligence and Machine Learning

Artificial Intelligence (AI) is a branch of Computer Science concerned with getting computers to perform tasks that are characteristic of those performed by the human mind. The field of AI encompasses both hard-coded rule-based programs (known as the knowledge base approach to AI, which has largely remained ineffective), as well as Machine Learning, which is an approach to AI which aims to get computers to perform these tasks without explicitly coding the solutions for them (21).

The success of Machine Learning algorithms is largely determined by the representation of the data fed through them. Often, a large amount of an AI practitioner’s time is dedicated to engineering the right feature-set to hand to a simple machine learning algorithm (21).

In the case of machine learning for image classification, which loosely ties back to some of the aims in this project, it is not always immediately obvious as to which features will be informative to an ML algorithm. For example, feeding raw pixel values into a linear regression model should not be very effective, since images vary in terms of positional information, lighting, sharpness, rotation, etc. (21)

Representation learning is a solution to feature generation in which ML is applied, not only to map from a feature set to an output, but also towards automatically learning the most useful representation of the data; usually this representation will encompass identifying the major factors of variation which effectively explain the observed data and discarding those which are not useful to the algorithm (21).

Deep Learning is an approach to representation learning which constructs useful representations based on a combination of simpler representations. In fact, the basic unit of a neural network is the perceptron, which in itself is a very simple function, but once compiled into a Multi-layer Perceptron, the rich texture of the input data distribution can be very accurately captured, since useful features discovered in the first layers of such a neural network can be combined in various ways to create additional useful features (21). Continuing with the image classification example, an early layer of a convolutional neural network may detect edges in an image, the next layer may detect corners and shadows, and layers further down will ideally detect actual visual elements (faces, car lights, arms, etc.) (21).

## Mathematical Background for Deep Learning

### Rosenblatt’s Perceptron

The original Rosenblatt paper (22) outlining the concept of the “perceptron” aimed to develop a theory to explain: 1. How sensory information is detected by biological organisms, 2. how that information is subsequently processed and stored and 3. how mental comprehension or organismal behaviour (which he termed “*preference for a particular response*”) was driven by the first two processes.

He outlined a mathematical framework for these mechanisms, at the hand of the following constructs:

1. **S-points:** sensory units which can possess any of a number of response curves based on the signal strength of incoming information

2. **A-units:** association cells located in an “association area” , which in some of his models was preceded by a “projection area”

3. S-points are connected in specific ways to A-units and forward their stimulus response to them, in the form of an inhibitory or an excitatory impulse

4. : A threshold value assigned to each A-unit dictates whether it will fire, based on the algebraic sum of excitatory and inhibitory signals received, from either S-points or preceding A-units

5. The connections between S-points and A-units, and between A-units themselves is random, and not all elements of such a network are connected to each other

6. Response units, , receive a large number of inputs from the set, called its source-set, and have feedback mechanisms to A-units in its source set. (22)

He put forth various models for response curve summation and how these networks would learn (22), but while the mathematical constructs he proposed were oversimplifications of the complexity of biological brains, they were found to be extremely useful in training computers to emulate their capabilities.

### Deep Feedforward Neural Networks

At its most basic level, an artificial neural network (ANN) is an approximation of a mapping function , which maps from a set of input features to a response, . Feedforward neural networks have one-way information flow from input features to output, whereas recurrent neural networks have feedback connections (21).

Also called multilayer perceptrons (MLPs), deep feedforward networks are composed of an arbitrary number of nested approximating mapping functions, of the form:

The superscript of these functions, , indicates the layer index of the function in an ANN, with indicating the depth of such a neural network. It is this concept of chained functions of arbitrary depth from which the term Deep Learning is derived (22).

The process of training such a network, , to give the closest approximation to the desired output, , is an iterative process, involving passing many observations, each having the same feature set through the MLP, assessing the output, , according to an error metric, , and individually adjusting each of the mapping functions according to their contribution to the differential of the magnitude of error at the conclusion of each training step . In other words, a parameter set , pertaining to each is iteratively adjusted according to . (21).

The set of nested approximation functions outlined above are commonly referred to as hidden layers, the dimensionality of the outputs of each layer is known as its width, or as the number of neurons in that particular hidden layer (21).

In order to produce subtle derived features from the input feature set, nonlinear transformations are applied to the output of each layer in the network, which in itself is a simple linear function of the form , where is a vector of weights of the same length as the set of input features, which are essentially a set of coefficients for each in the chain of functions, and is a real-valued bias term, which is essentially an intercept term for each (21).

It is easy to see that chaining such a set of linear models without applying nonlinear transformations (denoted as ) to what are essentially an arbitrary number of linear regression functions (), one would simply arrive at another linear model (21).

A commonly used nonlinear transformation , or activation function, in modern deep learning algorithms is the rectified linear unit (the ReLU function), which is simply an affine transformation, reminiscent of the response curves envisioned in Rosenblatt’s paper, of the form (21).

Combining the concepts explained above, then gives us a representation for a single hidden layer in an ANN as follows:

And, by extension, for a neural network with three hidden layers:

We now have a vector of weights multiplied by a vector of input features, which can be the original features fed to , or the weighted outputs of previous hidden units in . Since we essentially have a vector of hidden units, we also have a vector of bias terms, and all of these hyperparameters, collectively referred to as , need to be optimized to arrive at a reasonable approximation of a theoretically optimal mapping function (21).

To achieve the optimization of , most deep learning models utilize the concept of maximum likelihood, to minimize a loss function , for example, binary cross entropy:

,

where is the model’s estimate for the probability of an observation of being of a particular class (21).

Please see Appendix B for the code used to generate Figure 6, which shows how, as approaches the true (in this binary classification example, ), the binary cross entropy loss function approaches 0.



Figure 6: Illustration of the descent towards zero, of the Binary Cross Entropy Loss Function as ŷ, or , approaches the true y.

The chain rule of calculus is employed by backpropagation to enable the derivative of the loss function to be redistributed through the network, based on the partial derivative of each hyperparameter with respect to the derivative of the loss function (21):

For k = hidden layers, , we compute the element-wise gradient on the layer’s output (before the non-linear activation function is applied):

And the gradients on the weights and the bias term:

Here, represents the weight decay penalty, where the size of the weights are constrained, in a manner inversely proportional to . A regularizer is added to the loss, where contains all the weight and bias parameters.

This gradient is then propagated to the activations of the preceding layer:

## Convolutional Neural Networks

## Variational Autoencoders

## Generative Adversarial Networks

# Data

# Methods

## Software Environment

### AiROOT

AliROOT was built locally using alidock Docker container.

AliROOT was built from source on the hep01 server at uct

### R Statistical Software

#### Packages

### ROOTR

### Keras & Tensorflow

### Utilities

#### Makefiles

all: gridfiles.md5

gridfiles.xml: query.sh

./$< > $@

gridfiles.md5: gridfiles.xml

xsltproc /alice/data/util/xml2md5.xsl $< > $@

download: $(shell cut -c 49- files.md5)

/alice/data/%:

mkdir -p $(dir $@)

alien\_cp alien:$@ file:$@

### Python

## Data Extraction from WLCG

# Results

# Discussion

# Conclusion

# Bibliography

1. *Modern Particle Physics.* **Thomson, Mark.** Cambridge, UK : Cambridge University Press, 2013. ISBN 978-1-107-03426-6.

2. **CERN.** About CERN: Who we are: Our History. *CERN.* [Online] CERN. [Cited: 26 January 2019.] https://home.cern/about/who-we-are/our-history.

3. —. CERN: Who We Are: Our Governance: Member States. *CERN.* [Online] [Cited: 26 January 2019.] https://home.cern/about/who-we-are/our-governance/member-states.

4. —. CERN: About: Who We Are: Our Mission. *CERN.* [Online] [Cited: 26 January 2019.] https://home.cern/about/who-we-are/our-mission.

5. —. Grafana IT Overview. *CERN.* [Online] [Cited: 26 January 2019.] http://monit-grafana-open.cern.ch/d/000000884/it-overview?orgId=16.

6. **Chardley, Sarah.** LHC Does a Dry-Run. *Symmetry Magazine.* [Online] 20 March 2015. [Cited: 26 January 2019.] https://www.symmetrymagazine.org/article/march-2015/the-lhc-does-a-dry-run.

7. **CERN.** CERN Resources: FAQs: Facts and Figures About the LHC. *CERN.* [Online] [Cited: 06 01 2019.] https://home.cern/resources/faqs/facts-and-figures-about-lhc.

8. **University of California Davis.** RHIC. *UC Davis Nuclear.* [Online] [Cited: 27 01 2019.] http://nuclear.ucdavis.edu/~rpicha/rhic.html.

9. **Encyclopedia Britannica.** Tevatron Particle Accelerator. *Encyclopedia Britannica.* [Online] [Cited: 27 January 2019.] https://www.britannica.com/technology/Tevatron.

10. *CIRCUMFERENCE VARIATIONS OBSERVED AT KEKB.* **Masuzawa, M, et al.** 2002. Proceedings of the 7th International Workshop on Accelerator Alignment. Vols. Spring-8.

11. **Kurokawa, Shin-Ichi and Olsen, Stephen L.** The KEK B-Factory Experiment. *Stanford University.* [Online] [Cited: 27 January 2019.] www.slac.stanford.edu/pubs/beamline/29/2/29-2-kurokawa.pdf.

12. **Taking a Closer Look at the LHC.** The LHC Proton Source. *LHC Closer.* [Online] [Cited: 27 January 2019.] https://www.lhc-closer.es/taking\_a\_closer\_look\_at\_lhc/0.proton\_source.

13. **CERN.** LHC: The Guide. *CERN.* [Online] [Cited: 2017 January 2019.] https://home.cern/resources/brochure/cern/lhc-guide.

14. **Field, Rick.** PHY2061 - Enriched Physics 2 - Relativity 4. *University of Florida Physics.* [Online] [Cited: 27 January 2019.] http://www.phys.ufl.edu/~acosta/phy2061/lectures/Relativity4.pdf.

15. **CERN.** The CERN Accelerator Complex. *CERN Document Server.* [Online] [Cited: 26 January 2019.] https://cds.cern.ch/record/2636343/files/CCC-v2018-print-v2.jpg?subformat=icon-1440.

16. —. ROOT Data Analysis Framework: User's Guide. *CERN.* [Online] May 2018. https://root.cern.ch/root/htmldoc/guides/users-guide/ROOTUsersGuideA4.pdf .

17. **[Online] https://root.cern.ch/.**

**18. ROOT 5 Reference Guide. [Online] https://root.cern/root/html534/ClassIndex.html.**

**19. ROOT 6 Reference Guide. [Online] https://root.cern/doc/v616/.**

**20. ALICE Collaboration (CERN). *ALICE Analysis Tutorial.* [Online] [Cited: 18 2 2019.] https://alice-doc.github.io/alice-analysis-tutorial.**

**21. Goodfellow, Ian, Bengio, Yoshua and Courville, Aaron. *Deep Learning.* Cambridge, Masachusetts : The MIT Press, 2016. ISBN 9780262035613.**

**22. *The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain.* Rosenblatt, F. 6, 1958, Psychological Review, Vol. 65.**

# Appendices

[Appendix A: Plotting the Bethe-Bloch Equation 50](#_Toc1378383)

[Appendix B: Plotting Binary Cross-Entropy 57](#_Toc1378384)

Appendix A: Plotting the Bethe-Bloch Equation

Create a Bethe-Bloch function:

#Planck's constant:  
h <- 6.62607004e-34  
  
#Speed of light m/s  
c <- 299792458  
  
#Fine structure constant  
alpha <- 1/137  
  
#Mass of an electron Mass/GeV  
  
m.e <- 0.005  
  
#Density n, atomic number Z, the fraction of the speed of light the particle is moving at, beta, and the particle's velocity v are specified as parameters to the equation  
  
  
dE.dx <- function(n,Z,v,beta){  
 -4 \* pi \* h^2 \* c^2 \* alpha^2 \* ((n \* Z)/(m.e \* v^2)) \* log(((2 \* beta^2 \* gamma^2 \* c^2 \* m.e)/(I.e)) - beta^2,base=exp(1))  
}  
  
#For an electron traversing a silicon detector:  
  
v <- seq(0.1\*c,c,100000)  
  
beta <- v/c  
  
#Lorentz factor  
  
gamma <- 1/(sqrt(1-(v^2/c^2)))  
  
n <- 1  
  
  
  
Z <- 14  
  
#Effective ionization potential of the material  
  
I.e <- 10 \* Z  
  
electron.y = dE.dx(n=n,Z=Z,v=v,beta=beta)  
  
require(latex2exp)

## Loading required package: latex2exp

m.e <- 273.13\*m.e  
  
pion.y = dE.dx(n=n,Z=Z,v=v,beta=beta)  
  
  
  
plot(x=beta\*gamma, y=-pion.y,type="l",main="Bethe-Bloch Curve of a Pion moving through Silicon", xlab = TeX("$\\beta\\cdot\\gamma$"),ylab=TeX("$-dE/dx$"),col="blue",cex.main=0.8)



plot(x=beta\*gamma, y=-electron.y,type="l",main="Bethe-Bloch Curve of an Electron moving through Silicon", xlab = TeX("$\\beta\\cdot\\gamma$"),ylab=TeX("$-dE/dx$"),col="red",cex.main=0.8)



v <- seq(0.8\*c,c,100000)  
  
beta <- v/c  
  
#Lorentz factor  
  
gamma <- 1/(sqrt(1-(v^2/c^2)))  
  
n <- 1  
  
m.e <- 0.005  
  
electron.y = dE.dx(n=n,Z=Z,v=v,beta=beta)  
  
m.e <- 273.13\*m.e  
  
pion.y = dE.dx(n=n,Z=Z,v=v,beta=beta)  
  
plot(x=beta\*gamma, y=-pion.y,type="l",main="Bethe-Bloch Curve of a Pion moving through Silicon \nat Speeds Upwards of 80% of the Speed of Light", xlab = TeX("$\\beta\\cdot\\gamma$"),ylab=TeX("$-dE/dx$"),col="blue",cex.main=0.8)



plot(x=beta\*gamma, y=-electron.y,type="l",main="Bethe-Bloch Curve of an Electron moving through Silicon \nat Speeds Upwards of 80% of the Speed of Light", xlab = TeX("$\\beta\\cdot\\gamma$"),ylab=TeX("$-dE/dx$"),col="red",cex.main=0.8)



v <- seq(0.9\*c,c,100000)  
  
beta <- v/c  
  
#Lorentz factor  
  
gamma <- 1/(sqrt(1-(v^2/c^2)))  
  
n <- 1  
  
m.e <- 0.005  
  
electron.y = dE.dx(n=n,Z=Z,v=v,beta=beta)  
  
m.e <- 273.13\*m.e  
  
pion.y = dE.dx(n=n,Z=Z,v=v,beta=beta)  
  
plot(x=beta\*gamma, y=-pion.y,type="l",main="Bethe-Bloch Curve of a Pion moving through Silicon \nat Speeds Upwards of 90% of the Speed of Light", xlab = TeX("$\\beta\\cdot\\gamma$"),ylab=TeX("$-dE/dx$"),col="blue",cex.main=0.8)



plot(x=beta\*gamma, y=-electron.y,type="l",main="Bethe-Bloch Curve of an Electron moving through Silicon \nat Speeds Upwards of 90% of the Speed of Light", xlab = TeX("$\\beta\\cdot\\gamma$"),ylab=TeX("$-dE/dx$"),col="red",cex.main=0.8)



Appendix B: Plotting Binary Cross-Entropy

Define a function to plot the binary cross-entropy loss function:

cross.entropy <- function(y,p){  
 -(y \* log(p,base = 10) + ((1-y)\*(1 - log(p,base=10))))  
}  
  
#if the predicted class is 1:  
  
y <- 1  
  
p <- seq(0,1,0.01)  
  
loss <- cross.entropy(y,p)  
  
require(latex2exp)

## Loading required package: latex2exp

plot(x=p,y=loss, type="b", col=rainbow(250),cex=0.5, main = TeX("J($\\theta$) = -(y log(p)-(1-log(p)))"),ylab = "Cross Entropy", xlab = TeX("$\\hat{y}$"))

