DEEP LEARNING TECHNIQUES APPLIED T0 PARTICLE IDENTIFICATION AND HIGH ENERGY PHYSICS EVENT SIMULATIONS

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.

ABSTRACT

This Masters Dissertation outlines the application of deep learning methods on raw data from the Transition Radiation Detector at CERN as well as on simulated data from the Monte Carlo Event Generator Geant4, in order to achieve the following goals:

i. Classification Part I: Particle identification; distinguishing between electrons and pions

To this end, various feedforward neural networks, convolutional neural networks, as well as recurrent neural networks were built using Keras with a TensorFlow back-end, resulting in an ultimate pion efficiency of $\varepsilon_\pi = 1.2\%$ in the $P \leq 2\ GeV$ range, $\varepsilon_\pi = 1.14\%$ in the $2\ GeV < P \leq 3\ GeV$ and $\varepsilon_\pi = 1.51\%$ in the $3\ GeV < P \leq 4\ GeV$ range, all at electron efficiency $\varepsilon_e \approx\ 90\%$, using an incrementally trained convolutional neural network, which was fed training data from particles in increasing momentum ranges sequentially, during separate training runs, by saving weights obtained from the previous momentum ranges.

Raw data was extracted from the Worldwide LHC Computing grid using the ROOT data analysis framework, a C++ based platform maintained by physicists at CERN. R and Python were used interchangeably during various stages of data exploration, processing, analysis and model-building.

ii. Classification Part II: Distinguishing real data from data generated by Geant4

This stage of the project focused on employing convolutional neural networks towards distinguishing real data from simulated data. Data was simulated using Geant4, a Monte Carlo toolkit which simulates the passage of particles through matter. ROOT was used to reconstruct the simulated data to deliver it in a similar format to that given by raw data after processing. A balanced accuracy score of 91.5% (with Sensitivity = 0.8575 and Specificity = 0.9725) was achieved, using a 2D Convolutional Neural Network.

iii. Deep Generative Modeling: Prototyping Variational Autoencoders and various types of Generative Adversarial Networks towards TRD data generation

Various deep generative models were built to take as input raw TRD data and produce simulated observations which are likely under the training data distribution. While results were not as accurate as simulations from Geant4, it is nonetheless worthwhile to see how they performed on this problem.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Aims and Motivation

This Masters Dissertation seeks to apply cutting edge techniques in Machine Learning (ML) towards:

- Particle identification of electrons and pions, from raw signal data produced by these particles as they traverse the Transition Radiation Detector (TRD), using convolutional neural networks
- Distinguishing between real data and data simulated by the Geant4 Monte Carlo simulation environment
- Prototyping of variational autoencoders and Generative Adversarial Networks towards the simulation of High Energy Physics (HEP) collision events

The motivation for each of these elements is as follows:

- Accurate particle identification (in particular, electron samples that are as pure as possible) allows physicists at the ALICE (A Large Ion Collider Experiment) experiment to study the properties of the Quark Gluon Plasma (QGP), the primordial state of matter in the early universe. Since this deconfined state of matter rehadronizes quite soon after forming, it cannot be studied directly, but only via its decay products, of which the electron is one. To this end, having an electron sample which is as pure as possible is desirable and being able to accurately reject pions from the electron sample, whilst keeping as many as possible actual electrons in the sample under investigation, is a major concern

- Being able to distinguish Monte Carlo simulations from real data, could be indicative that Monte Carlo simulations, used for calibration and calculations of detector response functions, etc. are not accurate enough and that they could potentially be tuned via various parameter settings in future studies to increase their accuracy
- Using deep generative models such as Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs), instead of Monte Carlo simulations, could be a desirable future course of action, since these simulations are extremely fast compared to Geant4 simulations; but their practical use is contingent on whether they provide comparable accuracy to Geant4 simulations, as well as their customizability, e.g. is it possible to specify which particle, and at what momentum you want to simulate?

A variety of different software packages were utilized during the course of this project, including ROOT for data extraction, Geant4 for event simulation, Python and R for statistical analysis and Keras with a Tensorflow back-end for deep learning implementations.

1.2 Summary of Work Done & Major Findings

The lowest pion efficiencies that were obtained per momentum bin are as follows: $\varepsilon_{\pi} = 1.2\%$ in the $P \leq 2\ GeV$ range, $\varepsilon_{\pi} = 1.14\%$ in the $2\ GeV < P \leq 3\ GeV$ and $\varepsilon_{\pi} = 1.51\%$ in the $3\ GeV < P \leq 4\ GeV$ range, all at electron efficiency $\varepsilon_{e} \approx\ 90\%$. These results were obtained using an incrementally trained convolutional neural network as described in **Error! Reference source not found.**.

The highest balanced accuracy in distinguishing Geant4 simulated data from true raw data was 91.5%. This was also achieved using a convolutional network, discussed in **Error! Reference source not found.**.

In terms of Deep Generative Models, Variational Autoencoders gave results that look quite realistic to the human eye, but which could be easily distinguished from real data using a convolutional neural network.

Several variations on the Generative Adversarial Network concept was tested out, each of which performed in vastly different ways, but none of these were found to generate samples that looked as realistic as those obtained from Variational Autoencoders or which could compete with Geant4 simulations.

2 HIGH ENERGY PHYSICS & CERN

2.1 The Standard Model of Particle Physics

2.1.1 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 10^0 eV, the low energy manifestation of Quantum Electrodynamics (QED) allows atoms to exist in bound states with negatively charged electrons (e^-) orbiting a positively charged nucleus consisting of positively charged protons (p) and electrically neutral neutrons (n), 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 (v_e) 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 particles: the electron, proton, neutron and electron neutrino [1].

2.1.2 The Fundamental Particles

At higher energy scales, 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 (μ^-) is essentially a version of an electron which is 200 × heavier, i.e. $m_\mu \approx 200\, m_e$. The tau-lepton (τ^-) is the third generation of the electron, and is much heavier, i.e. $m_\tau \approx 3500\, m_e$. These mass differences do have physical consequences, but the fundamental properties and interactions of the various generations remain identical [1].

There hasn't been any evidence of further generations than these three, and so – according to current understanding – 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 (e^-)	-1	0.005	Down (d)	-1/3	0.003
	Neutrino (v_e)	0	$< 10^{-9}$	Up (u)	+2/3	0.005
Second Generation	Muon (μ^-)	-1	0.106	Strange (s)	-1/3	0.1
	Neutrino (v_μ)	0	$< 10^{-9}$	Charm (c)	+2/3	1.3
Third Generation	Tau (τ^-)	-1	1.78	Bottom (b)	-1/3	4.5
	Neutrino (v_τ)	0	$< 10^{-9}$	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 $\bar{\mathrm{d}}$, whereas the antimuon is indicated by μ^+ [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 bound states within hadrons and quarks are therefore not freely observed under normal circumstances [1].

2.1.3 The Fundamental Forces

Historically, Newton stated that matter could interact with any other matter without the mediation of direct contact and classical electromagnetism explained the electrostatic interaction between particles using fields [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 circumvents 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. W^+ and W^- 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].

2.1.4 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 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, c [1].

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

2.2 The Quark Gluon Plasma (QGP)

2.2.1 Introduction to QGP

As mentioned above in 2.1.2, quarks and gluons are confined by the Strong Force to remain within the bound states of colour-neutral hadrons (e.g. protons and neutrons) and are therefore never found freely in nature. However, the currently held view of the early universe, predicted by the standard model and supported by over three decades of High Energy Physics experiments and lattice QCD simulations, is that directly subsequent to the Big Bang, the universe was composed of a deconfined state of matter, known as the Quark-Gluon Plasma (QGP) [2].

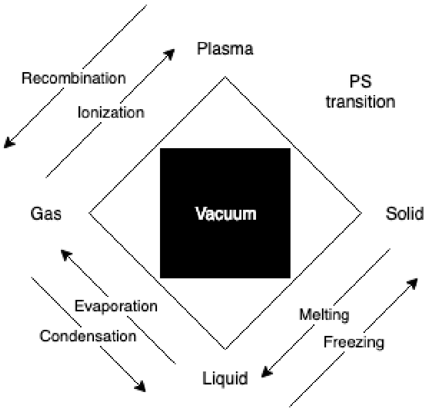


Figure 1: Simplified diagram of classical states of matter and transitions between them, with the Vacuum added as a fifth element, providing the space in which matter exists [3], reproduced and modified from [4].

Statistical mechanics understands matter as a system in thermal equilibrium. Global observables, such as net charge, temperature and energy density define the average properties of such a system. As these global observables take on different values, radically different average properties can be held by the system, manifesting as different states of matter bounded by phase boundaries, which matter traverses via phase transitions [4], see Figure 1 for an illustration of this process.

If nucleons (protons and neutrons) were truly fundamental, i.e. if they were not bound states of smaller composite elements (quarks and gluons), a density limit of matter would be reached, when compressing it under ever higher pressure conditions. If, however, nucleons were truly composite states, increasing density would eventually cause their boundaries to overlap and nuclear matter would transition from a stable state of colour-neutral three-quark or quark-antiquark hadronic matter to a state of deconfinement, consisting mainly of unbound quarks [4].

Hadrons all have the same characteristic radius of around 1 fm; it has been found experimentally that increasing density (through compression or heating), results in the formation of clusters where there are more quarks within such a hadronic volume than logical partitioning into colour neutral hadrons allows for, thus leading to colour-deconfinement [4].

In Figure 2, a simplified phase diagram of hadronic matter is depicted. Within the hadronic phase, there is a baryonic density/temperature boundary where transitions between mesons (colour-neutral quark-antiquark systems) and nucleons (colour-neutral three-quark systems) occur, (not shown in this diagram). The existence of diquarks as localised bound states within the QGP medium allows for yet another state of matter, the colour superconductor, discussion of which is outside of the scope of this dissertation.

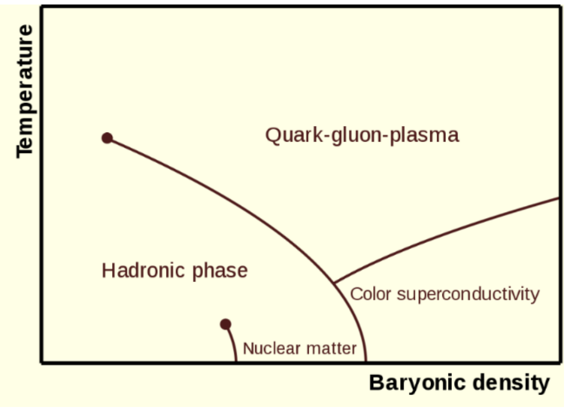


Figure 2: Phase diagram of hadronic matter [5]

2.2.2 QGP, the Big Bang and the Micro Bang

It is estimated that, during the Plank epoch, which lasted until $t < 10^{-43}s$ after the Big Bang, the prevailing temperature was T $\simeq$ 10^{19} GeV, a temperature so high that the principles of general relativity do not apply, and which cannot be understood with present-day physical theory [6].

Shortly after the Planck epoch, following a short exponential inflation phase, quarks and gluons propagated freely in an early deconfined space-time QGP expansion phase of the Universe, down to a temperature of T $\simeq$ 150 MeV, a phenomenon thought to be caused by a change in the vacuum properties of the extremely hot early Universe [2].

To understand how matter was formed in the early Universe, heavy ion collisions, such as the Pb-Pb collisions performed at ALICE, result in a miniscule space-time domain of QGP (which one can refer to as a ‘micro bang’), in which local quark-gluon deconfinement occurs. The subsequent hadronization process, where protons, neutrons and other subatomic particles are formed, leaves traces in the ALICE detector material, giving physicists an indication of how matter arose as the early Universe rapidly cooled down [2].

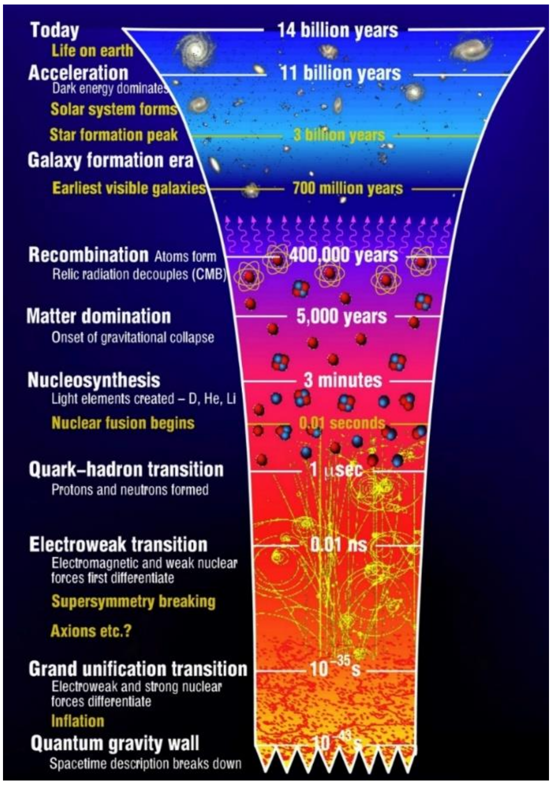


Figure 3: The evolution of the Universe, from the Big Bang to Modern Day [7]

Since the QGP cannot be detected directly, it is studied via its decay products. Accurately distinguishing between electrons and pions is an important step in this process and as such is the motivation for the particle identification phase of this Masters project.

2.3 CERN

At the end of 1951, a resolution was agreed upon to establish a European Organisation 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 [8].

Today, CERN is a truly international organization, with 23 member states ,who contribute to operating costs and are involved in major decision making, a few countries with associate member status or observer status, and non-member countries with co-operation agreements, including South Africa [9].

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 [10].

2.3.1 The Large Hadron Collider

The Large Hadron Collider (LHC), located under the Franco-Swiss border (see Figure 4 for geographical context), boasts an intricate system of particle accelerators and -detectors. The LHC is currently the largest and most powerful particle accelerator in the world, with a circumference of ~27km and a centre of mass energy of $E_{CM} = 13\ TeV$ [11].

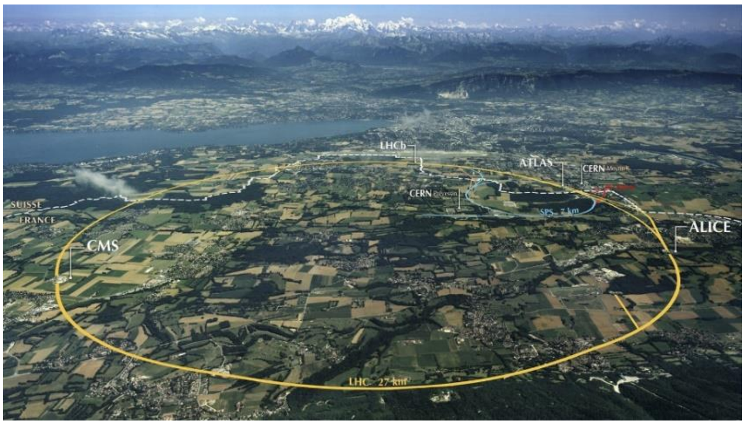


Figure 4: CERN facilities in geographical context [12].

2.3.1.1.1 The LHC

Located 50-175 m underground, the LHC 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 6.

The LHC's proton source is a bottle of compressed Hydrogen, which releases its contents into a Duoplasmatron device, which subsequently surrounds the H_2 molecules with an electrical field and separates it into its constituent protons and electrons [13]. A simplified diagram depicting this process can be seen in Figure 5.

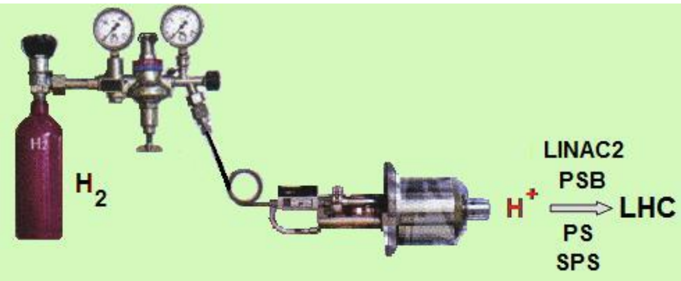


Figure 5: The LHC Proton Source, connected to the Duoplasmatron device, which strips electrons off Hydrogen molecules, to produce the beams of protons which eventually collide within the LHC [13]

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 (PS), which accelerates them up to 25 GeV, the Super Proton Synchrotron (SPS) 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 [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 Pb^{54+}. 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 Pb^{82+} ions to the LHC, where beams of Pb-ions are accelerated up to 2.56 TeV per nucleon [14]; because there are many protons in a single lead ion, the collision energies reached in PbPb collisions reach a maximum of 1150 TeV [14].

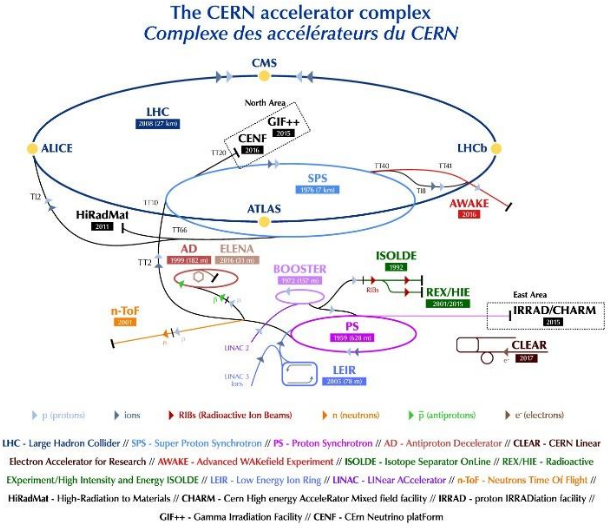

The CERN accelerator complex
Complexe des accélérateurs du CERN
CMS
LHC
2008 (27 km)
North Area
GIF++
2015
CENF
2016
ALICE
LHCb
TT20
TT40
TT41
SPS
1976 (7 km)
TI8
TI2
TT10
ATLAS
AWAKE
2016
HiRadMat
2011
TT66
AD
1999 (182 m)
ELENA
2016 (31 m)
TT2
BOOSTER
1972 (157 m)
ISOLDE
1992
RIBs
REX/HIE
2001/2015
East Area
IRRAD/CHARM
2015
n-ToF
2001
n
p
p̄
PS
1959 (628 m)
LINAC 2
CLEAR
2017
e⁻
LINAC 3
Ions
LEIR
2005 (78 m)
p (protons)
ions
RIBs (Radioactive Ion Beams)
n (neutrons)
p̄ (antiprotons)
e⁻ (electrons)
LHC - Large Hadron Collider // SPS - Super Proton Synchrotron // PS - Proton Synchrotron // AD - Antiproton Decelerator // CLEAR - CERN Linear Electron Accelerator for Research // AWAKE - Advanced WAKefield Experiment // ISOLDE - Isotope Separator OnLine // REX/HIE - Radioactive EXperiment/High Intensity and Energy ISOLDE // LEIR - Low Energy Ion Ring // LINAC - LINear ACcelerator // n-ToF - Neutrons Time Of Flight // HiRadMat - High-Radiation to Materials // CHARM - Cern High energy AcceleRator Mixed field facility // IRRAD - proton IRRADiation facility // GIF++ - Gamma Irradiation Facility // CENF - CErn Neutrino platForm

Figure 6: 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 become superconducting and allow the magnetic field to reach the 8.3 T required to bend the beams around the circular LHC ring [14].

The beams themselves are contained within a beam pipe emptier than outer space ($P_{vac} = 10^{-13} atm$) 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 an average bunch crossing frequency of 30 MHz and around a billion collisions per second [14].

2.3.1.2 The CERN Experiments

Collisions at the LHC result in a multitude of particles being produced. Observing the produced particles from different perspectives produces evidence relevant to different research streams; as such, there are several collaborations at CERN which use detectors with differing attributes to study specific areas within the broad area of fundamental subatomic Physics [16].

ATLAS and CMS investigate a very broad range of particle physics. Their independent design specifications allow any new discoveries at one of these detectors, such as the discovery of the Higgs' Boson in 2012, to be corroborated by the other [16]. Other research avenues pursued at these experiments include the search for additional dimensions as well as the constituent elements of dark matter. The ATLAS detector is the largest particle detector ever built, weighing 7000 tonnes with dimensions 46m × 25m × 25m [17].

ALICE and LHCb are the other two main experiments at CERN and are tasked with the discovery of specific physical phenomena [16]. ALICE focuses on the extreme energy densities present during heavy ion collisions, which leads to the production of the Quark Gluon Plasma, a newly discovered phase of matter thought to have been dominant in the early universe, directly subsequent to the big bang [18]. LHCb investigates subtle distinguishing nuances in the matter-antimatter dichotomy, as evidenced by attributes of the beauty quark [19]. In addition, there are several smaller experiments hosted at the LHC as well.

2.4 The ALICE Detector & the Transition Radiation Detector

2.4.1 The ALICE Detector System

Colliding heavy ions, such as the Pb-Pb collisions conducted at the LHC and studied at ALICE, offers the most ideal experimental conditions currently achievable for the reproduction of the primordial QGP matter [20]. A transition from ordinary matter to a state of deconfinement occurs at a critical temperature $T_c \approx 2 \times 10^{12} K$, which is around 100,000 times hotter than the core temperature of our sun [20].

The QGP cannot be probed directly, but is studied via particles produced during the hadronization process that occurs as the QGP cools down and quarks and gluons recombine in various ways; the ordinary-matter particles produced in this process interact with various detector elements and leave traces in the detector material that are generally recorded via electronic signals [20].

The scale of the ALICE detector system is illustrated in Figure 7. The detector weighs 10,000 tonnes and has spatial dimensions 26m × 16m × 16m [18].

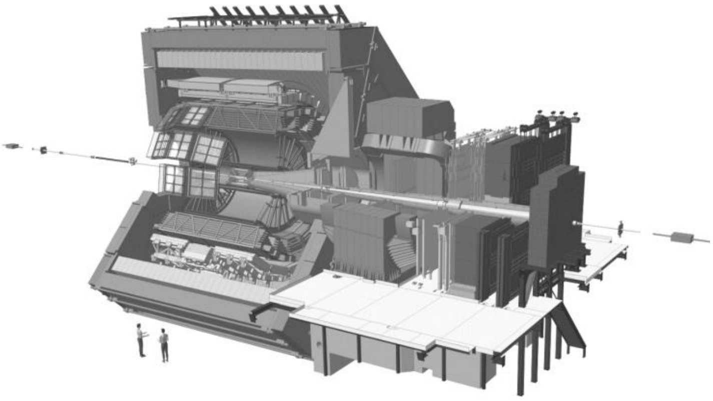


Figure 7: The ALICE detector system [21]

A uniform magnetic field is applied over the detector, to allow particles to propagate in curved paths through the detector geometry, with the extent of curvature of the particle's track through the detector being inversely correlated to the particle's momentum; additionally, the sign of charge of a particle can also be deduced from its track curvature [20].

The ALICE detector has a total of 18 stacked subdetectors involved in specific particle tracking tasks, these are broadly divided into: Tracking Systems, situated closest to the collision area, which make use of digital track-reconstruction of particle-detector interaction traces to indicate the path of a particle; these are followed by Electromagnetic and Hadronic Calorimeters, through which particle cascades are generated as particles enter and are absorbed by the calirometric material, with the magnitude of a particle's energy deposition acting as the signal in these subdetectors; all of which is surrounded by the Muon System in the outermost layer, which detects muons, which interact very weakly with matter and therefore generally travel much further through the detector system [20].

High momentum resolution is obtained in all the detector elements over the high multiplicity densities (number of particles produced per unit volume) present in heavy ion collisions [22]. In addition to heavy ion collisions, lighter ion- as well as proton-nucleus and proton-proton collisions are also performed at ALICE, and this entire momentum range can be accurately measured by the ALICE detector [22].

2.4.1.1 The Transition Radiation Detector

At particle momenta above 1 GeV/c, the pion rejection strategy for electron identification employed in the TPC is no longer sufficient. The TRD's main goal is to expand the range of the ALICE Collaboration's Physics objectives by providing accurate electron identification capabilities at these high momenta, by supplementing its own data with data obtained from the ITS and TPC; as well as the operation of event triggers that determine whether data from a specific collision should be kept, based on measurements such as collision centrality, amongst others. As an added benefit, the TRD informs the ALICE central barrel's calibration, and the data it produces is used extensively during track reconstruction and particle identification [23].

2.4.1.1.1 TRD Design Synopsis

Pseudorapidity coverage in the TRD is similar to the other detector elements in the central barrel, i.e. $|\eta| \leq 0.9$. The space between the TOF and TPC detectors is filled by the six layers of the TRD, which are subdivided in azimuthal angle into 18 sectors, with an additional segmentation into 5 sectors occurring along the z-axis. So, in total, we have $18 \times 5 \times 6 = 540$ individual detector elements in the TRD [23] at a radial distance of 2.9 – 3.7 m from the beam axis [24].

Each individual detector element consists of the following broad components: 1) a radiator (4.8 cm thick), 2) a 0.7 cm multiwire proportional readout chamber, and 3) front-end electronics to convert from an amplified particle energy-deposition signal to a digital signal, which is eventually stored if deemed interesting by the multi-tiered TRD trigger system [23].

2.4.1.1.2 TRD Measurement Mechanism

2.4.1.1.2.1 Interactions of Particles with Matter

In order to study subatomic particles, they need to be detected. Most particles produced during High Energy Physics Experiments are unstable and therefore decay within a specific characteristic mean lifetime τ. Those particles with $\tau > 10^{-10}s$ will traverse several meters before decaying and are therefore directly detectable by particle detectors such as those 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].

2.4.1.1.2.2 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 $v = \beta c$, passing through a medium with atomic number Z and density n, 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 (Equation 1) [1]:

$$\frac{dE}{dx} \approx -4\pi\hbar^2c^2\alpha^2\frac{nZ}{m_e v^2}\left\{ln\left[\frac{2\beta^2\gamma^2c^2m_e}{I_e}\right] - \beta^2\right\}$$

Equation 1

In Equation 1, I_e is the effective ionisation potential of the medium. While the $\frac{1}{v^2}$ term explains the high energy loss for low energy particles, for high energy particles studied in Modern Particle Physics, where $v \approx c, \frac{dE}{dX}$ depends logarithmically on $(\beta\gamma)^2$, which is defined by Equation 2. This explains the relativistic rise seen in Figure 8, which illustrates the characteristic energy loss curves for various subatomic particles as measured by the TPC, including the two subatomic particles studied in this project, the pion π and the electron e^-.

$$\beta\gamma = \frac{v/c}{\sqrt{1-(\frac{v}{c})^2}} = \frac{p}{mc}$$

Equation 2

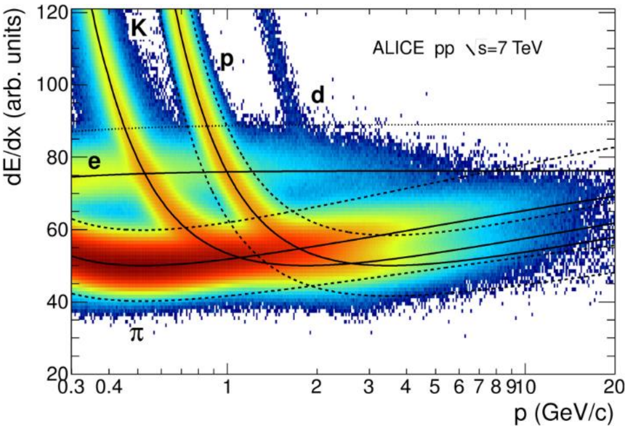


Figure 8: Bethe-Bloch curve for various subatomic particles as measured by the ALICE TPC at $\sqrt{s} = 7TeV$

2.4.1.1.2.3 Transition Radiation

Transition radiation is radiation emitted by a charged particle as it traverses the boundary between two mediums with different optical properties, no significant energy loss occurs in this process, but the resultant radiation is an important aid in detecting charged particles in HEP experiments [25].

For relativistic particles, the photons emitted in this process extends into the X-ray domain and are highly forward-peaked compared to the direction the particle is moving in; transition radiation yield is increased by stacking multiple radiative boundaries in gas detectors, such as the Transition Radiation Detector (TRD) at ALICE, and placing high atomic number (high-Z) gases within subsequent chambers to absorb the emitted X-ray photons [26].

The drift time of gas particles within the MWPC provides fine-grained positional information about where the particle tracklet passed through the radiator. The detected signal takes the form of charged gas molecules (ionized via interaction with transition radiation photons and amplified through a chain of interactions between gas molecules), finally being absorbed by a negatively charged wire (anode), this process is shown in Figure 9.

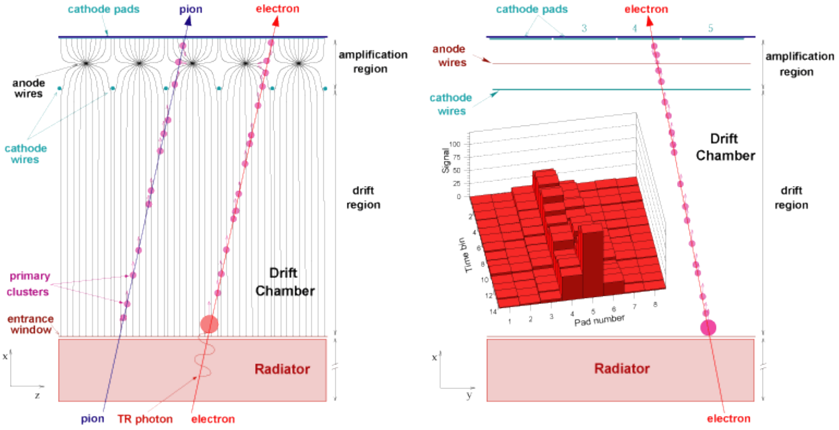


Figure 9: A schematic representation of the components in an MWPC module

One of the main aims of this thesis is distinguishing electrons from pions. This is facilitated by the fact that electrons and pions have different characteristic energy loss curves, and particularly at low momenta, electrons have a higher relative energy loss, as well as the fact that electrons emit transition radiation and pions don’t.

2.4.2 HEP Software

2.4.2.1 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 of raw data from collision events at the LHC into more compact and useful representations [27].

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 [27].

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 [28], the version of ROOT developed and used for LHC run 1 and run 2; and ROOT 6 [29], the version of ROOT developed for LHC run 3, scheduled to start in 2021 after the second long shut down period (LS2).

2.4.2.2 AliROOT

It is a common concept for each experiment at CERN to build software specific to their needs on top of the base ROOT architecture; as such, AliROOT and AliPhysics are built on top of ROOT 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 [30].

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.

2.4.2.3 O^2 Software for Run 3

LHC run 3, scheduled to start in 2020, will require some upgrades to the ALICE detector to accommodate the much higher interaction rate that is being planned for, in order to more precisely measure attributes of heavy flavour hadrons, low mass di-leptons and low-momentum quarkonia. Since these physics probes have a very low signal-to-background ratio, a continuous readout process could result in upwards of 1TB/s of data being generated by the ALICE detector.

This will result in unique challenges, which will need to be met by an upgraded software framework for run 3 and run 4, known as O^2 (The Online-Offline Software Framework), which is currently being developed.

2.4.2.4 Geant4

Geant4 is a C++ toolkit for simulating how particles traverse through matter. Comprehensive and accurate simulations of particle detectors, using platforms like Geant4, is extremely important, since it provides a theoretical reference against which data can be compared. Should there be any statistically significant discrepancies between simulations and data, it could indicate that phenomena occurred which are not explicable by the Standard Model of Particle Physics and could in rare circumstances lead to the discovery of new fundamental principles of nature [31].

Simulation software typically rests on four key components:

1. Event Generation
2. Detector Simulation
3. Reconstruction

4. Analysis

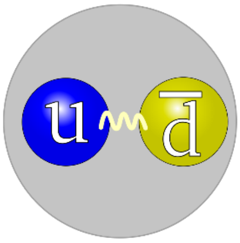
In a typical High energy Physics Simulation set-up, Geant4 is often used as the Detector Simulation component, tied to an event generator such at Pythia or HIJING, with ROOT used for Reconstruction and Analysis. As such, Geant4 has well-defined interfaces to the other components in the simulation set-up [32].

The following aspects of simulation are implemented in Geant4: materials and geometry of the detector system, fundamental particles and their transition through the detector and external electromagnetic fields, how the detector responds to these processes to generate data and storing said data for downstream analysis [32].

3 Particle Identification

3.1 Introduction

3.1.1 Electrons vs Pions: Physical Characteristics

Characteristic	$\boldsymbol{e}$	$\boldsymbol{\pi}$
Charge varieties	e^-, e^+	π^0, π^+, π^-
Substructure	• No known substructure • Fundamental Particle • Point-like	• Colour-neutral  **Figure 10: Pion substructure [33]**

3.1.2 Particle Identification in the TRD

At momenta $P > 1\ GeV/c$, the TRD provides electron identification via the measurement of transition radiation. At these momenta, pion rejection (achieved in the TPC via specific energy loss as per characteristic Bethe-Bloch dE/dx curves for pions vs. electrons) is no longer possible. The time evolution of signals generated in the TRD is an important factor in distinguishing between electrons and pions. The electron identification capability is also used to trigger at level 1 [24].

Electron identification and triggering as mentioned above enables the in-depth study of physical phenomena such as jets, the semi-leptonic decay of heavy-flavour hadrons and the di-electron mass spectra of heavy quarkonia; in turn, these phenomena act as probes to study the Quark Gluon Plasma [24].

The TRD signal originally induced on the segmented cathode plane is captured and processed by a preamplifier-shaper circuit, this processed signal is then digitized by a 10 MHz ADC to take samples of the time-evolution of the signal at defined 100 ns intervals [24].

Figure 11 shows the time evolution of the abovementioned signal at p = 2 GeV, for both electrons and pions. The initial peak seen in earlier time-bins on the graph originates from the amplification region of the detector and the plateau that follows is caused by particles moving through the 3 cm drift region in the detector [24].

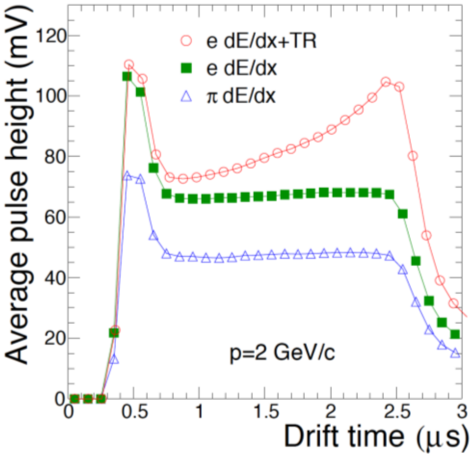


Figure 11: Time evolution of the TRD signal, measured as pulse height vs drift time for electrons and pions (both at p = 2GeV) [24].

Also evident from Figure 11 is that, in this momentum region, the pulse height of electrons is much higher than that for pions, because electrons have higher characteristic energy loss (dE/dx) in this region [24].

An average of one transition radiation photon in the X-ray domain will be emitted by an electron traveling at a highly relativistic speed (above $\gamma \sim 800$), since it will cross many dielectric boundaries in the radiator portion of a detector element, the absorption of this type of photon is evidenced by an additional peak at later times in Figure 11, since it will be absorbed preferentially close to the radiator, adding its signal to the ionization energy of the track [24].

3.1.2.1 Methods used in Particle Identification

Currently, the following methods are employed for particle identification based on TRD data:

1. Truncated mean of the signal
2. One- and two-dimensional likelihood estimations
3. Neural Networks

3.1.2.1.1 Truncated Mean

The truncated mean signal is the combined signal of Transition Radiation + Specific Ionization Energy; this method focusses on classifying electrons vs pions based on their expected energy loss as per the Bethe Bloch curve shown in Figure 12.

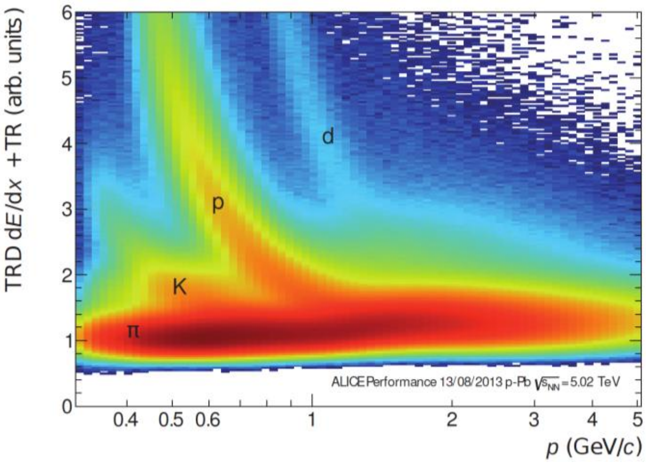


Figure 12: Truncated mean signal (dE/dx + TR) for various charged particles as measured for p-Pb collisions at $\sqrt{\mathbf{5.02}}\ \boldsymbol{TeV}$. This method allows for particle identification of light particles and hadrons [24].

3.1.2.1.2 One-dimensional Likelihood (LQ1D)

One dimensional likelihood estimation is performed on the total integrated charge left by a particle in a single chamber in the TRD (i.e. a single tracklet). Figure 13 shows that electrons have on average a higher charge deposit, because they experience higher characteristic energy loss in this momentum range, as well as the fact that they emit Transition Radiation and pions don't [24].

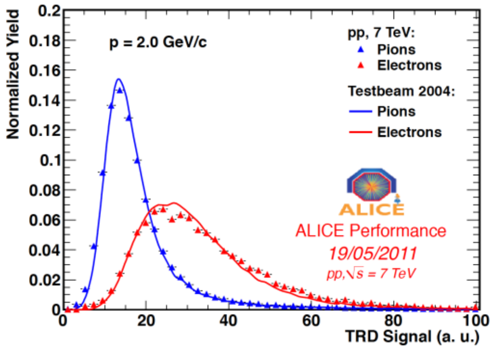


Figure 13: Normalised distribution of charge deposition for electrons and pions in a single TRD chamber [24].

The reference distributions allow maximum likelihood estimations to be carried out on each particle traversing the TRD, i.e. the likelihood of it being a muon, pion, kaon or an electron. Pions are rejected based on momentum-dependent cuts based on the likelihood for electrons, taking into account an electron efficiency score calculated using a clean reference sample of electrons arising from photon conversion [24].

3.1.2.1.3 Two-dimensional Likelihood (LQ2D)

Two-dimensional likelihood takes the temporal evolution of the signal (Figure 11) into account by splitting the signal into two time-bins and summing the charge in each bin and calculating the likelihood based on pure pion- and electron samples from collision data [24].

3.1.2.1.4 Neural Networks

A neural network was trained using a similar approach as LQ2D, but instead of splitting and summing over two time-bins, the input feature-set to the neural network was obtained by splitting into seven time-bins and summing the charge over each bin, respectively [24].

3.1.2.2 Particle Identification Accuracy

To calculate the accuracy of the abovementioned methods, clean reference samples were used. The separating power of these approaches are often expressed as pion efficiency (the fraction of pions incorrectly classified as electrons, i.e. the false positive rate or fallout rate) at a specific electron efficiency (the fraction of electrons correctly identified, i.e. the true positive rate or sensitivity) [24].

Error! Reference source not found. shows the obtained pion efficiency for the methods discussed above, a s a function of electron efficiency, it is clear from this plot that the misidentification of pions as electrons (False Positive Rate) is reduced substantially by the LQ2D and Neural Network techniques, compared to truncated mean- and LQ1D methods, and that the temporal evolution of the signal is therefore a highly informative feature for particle identification [24].

It is important to note that pion suppression (the inverse of pion efficiency) is hampered when a particle passes through fewer than the available six layers of the TRD, and that electron efficiency is sometimes sacrificed during analysis to obtain a more pure sample [24].

Figure 14 shows how pion efficiency depends on momentum for the four methods under discussion, data is plotted for samples where electron efficiency of 90% is obtained. LQ1D and LQ2D are quite accurate at low momenta where the emission of transition radiation commences, but their separating power decreases at higher momenta as transition radiation production saturates and pions deposit more energy, making it harder to tell them apart. The truncated mean method performs poorly at high momenta, since transition radiation with its attendant high charge deposition is more likely to be removed during the truncation procedure [24].

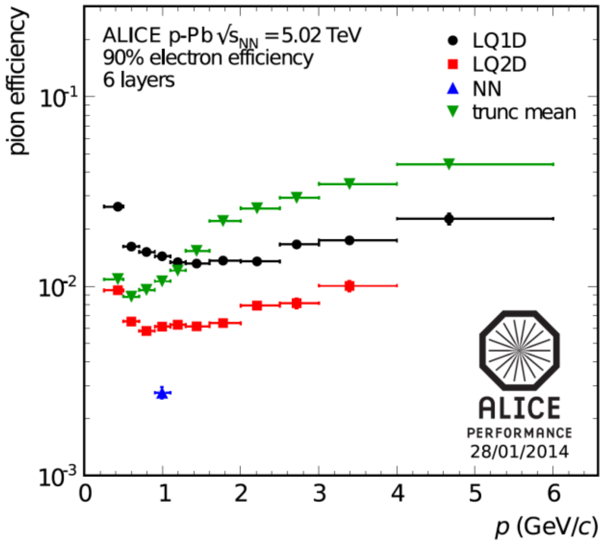


Figure 14: Momentum dependence of pion efficiency for various methods (where electron efficiency is at 90%)

3.1.3 Data Extraction

Using the Hep01 cluster in the Physics Department at UCT, an AliRoot macro was used to extract data from the WLCG.

TRD Analog to Digital (ADC) digits were extracted and filtered for p-Pb runs during 2016, by redirecting the C++ standard out to a text file.

Jobs were submitted onto the WLCG and monitored using http://alimonitor.cern.ch/. Upon completion, data was extracted back onto Hep01 using the aliensh environment.

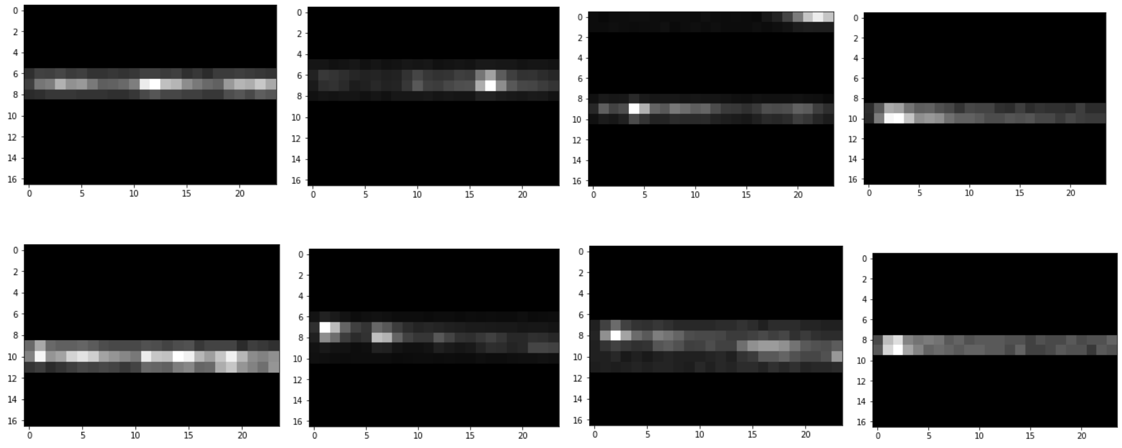
Data was backed up in a semi-private GitLab repository, accessible by CERN members, at https://gitlab.cern.ch/cviljoen/msc-thesis-data.

3.1.4 Data Structure

An example of the data obtained for a single track can be viewed at https://github.com/PsycheShaman/MSc-thesis/blob/master/NEW/example_pythonDict.txt. This data structure consists of a header section with meta-information about the track, as well as the raw TRD digits.

Below are some examples of single tracklets (a tracklet refers to the signal a particle induced in a single layer of the TRD, whereas a track refers to 6 tracklets produced when a particle crosses all 6 layer of the TRD).

In the images below, the signal for 17 pads in the TRD layer were added (along the rows of the image), centred around the expected position of the tracklet. The columns in the images below represent the charge deposited during a specific time bin within the pad, giving an indication of the time-evolution of the signal.



3.1.5 Data Exploration

When read into a single list data structure, the full dataset amounts to ~19.7GiB.

While data for 1 565 438 tracks were extracted, only 7 735 493 tracklets of the expected 6 layers × 1 565 438 tracks = 9 392 628 tracklets were obtained. This is mainly the result of detector elements in the TRD being switched off or not working. Missing data of this type manifests as either an empty list at that layer in the python dictionary, or as a NULL value.

There is also a second type of missing data: 1 098 636 tracklets returned images, but these images carried no information to assist in particle identification. Every pixel in this type of image was equal to 0.

This number of tracklets with empty arrays, resulted in an additional 14.5% of all pion tracklets and 12.6% of all electron tracklets being removed from the dataset used for training and testing.

Technically, excluding this data also affects the true electron efficiencies reported in this thesis, but this data does not add any additional information, other than the insight that pions result in a slightly higher proportion of empty images compared to electrons.

3.1.5.1.1 Total Number of Tracklets per Particle ID

Figure 15 illustrates the extreme class imbalance in this dataset, if not accounted for, such a distorted class distribution can result in unwanted results when training Machine Learning models, a problem which is addressed in various ways in Chapter **Error! Reference source not found.**.

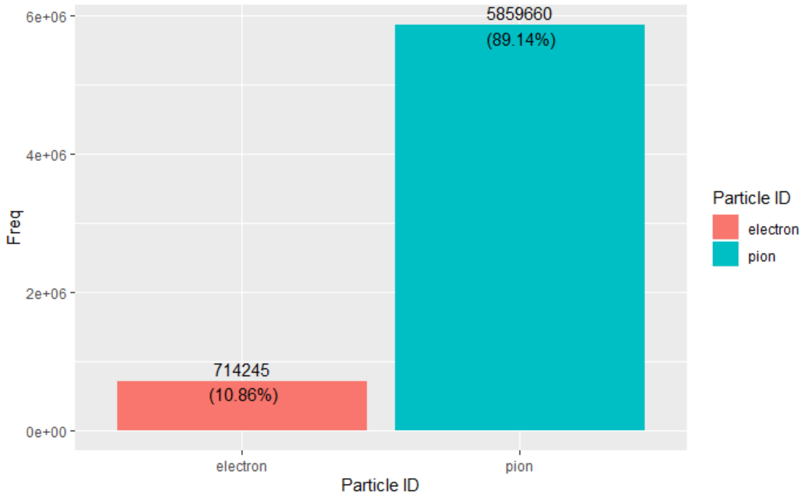


Figure 15: Number of Particles, per Particle ID, across all runs

3.1.5.1.2 Momentum bin counts: number of tracklets per Particle ID

From Figure 16, one can also see how this class distribution differs for particles of different momenta. Particularly, there is a larger proportion of electrons in the lower two momentum bins, i.e. $P \leq 2\ GeV$ and $2\ GeV < P \leq 3\ GeV$. This only partly explains the increased performance in this momentum range (which will be discussed), since electrons are easier distinguishable in this momentum range, according to its characteristic energy loss (Bethe-Bloch), discussed next in 3.1.5.1.3.

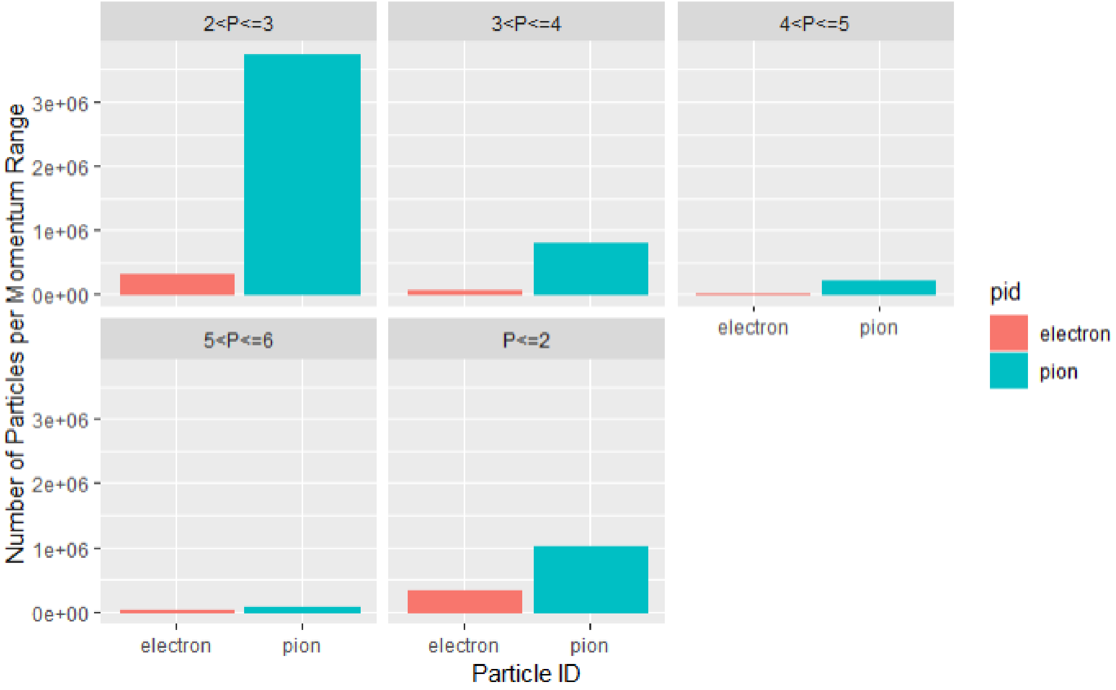


Figure 16: Number of Particles (electrons and pions) in each of a set of defined momentum bins

3.1.5.1.3 Characteristic Energy Loss Curves (Bethe-Bloch)

From Figure 17, the expected increased energy loss of electrons relative to pions, in the low GeV range is apparent. It should be noted that a cut was made on momentum, to keep only tracklets in the $P \leq 6GeV$ range.

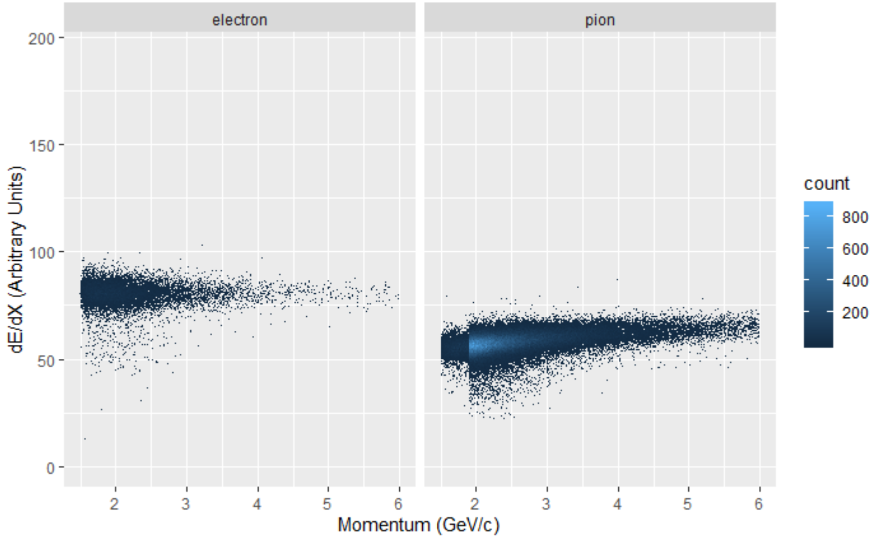


Figure 17: Energy Loss per Unit Path Length as a function of Momentum, for Electrons and Pions

3.1.5.1.4 Average Pulse Height

Figure 18 shows the average pulse height as a function of time, for electrons vs pions, across the entire momentum range; while not as distinct as in Figure 8, which was restricted to tracklets in the 2 GeV range, the characteristic Transition Radiation (TR) signal can be seen for electrons in the later timebins of the plot. The average pulse height for electrons is also higher than that for pions, across all timebins, this fact, in conjunction with the TR signature were the motivation for feeding some particle identification neural networks with the timebins sums of signal arrays, but there are significant fluctuations around this average (as can be seen in Figure 19 and Figure 20), which makes the task much less straight forward than Figure 18 would suggest.

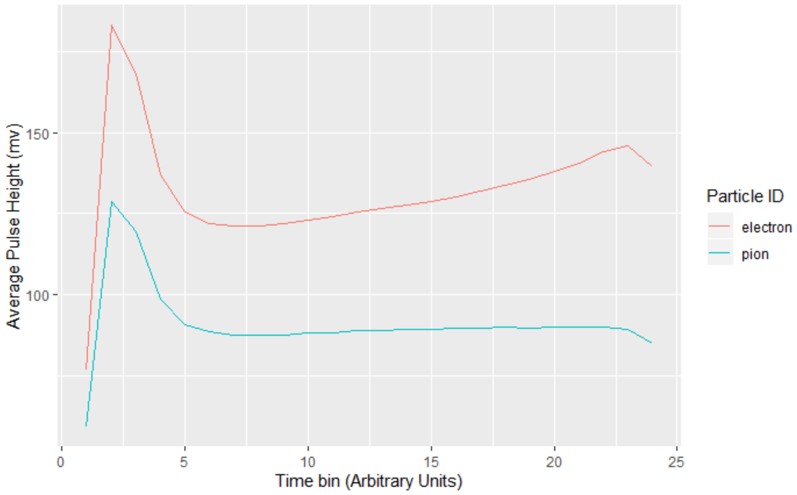


Figure 18: Time Evolution of the Average Pulse Height Signal, per Particle ID (for tracklets from the entire momentum range)

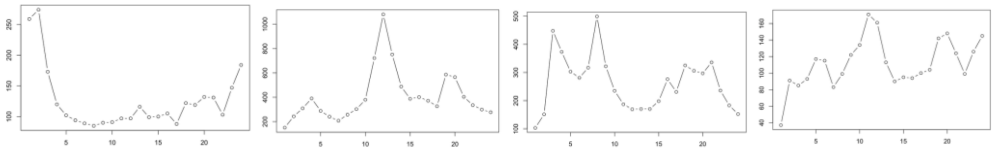


Figure 19: Pulse height as a function of time for 4 randomly sampled electrons

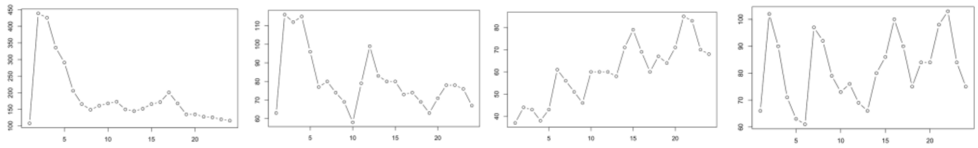


Figure 20: Pulse height as a function of time for 4 randomly sampled pions

3.2 Theory: Artificial Neural Networks

At its most basic level, an artificial neural network (ANN) is an approximation of a mapping function f_a, which maps from a set of input features x_i ; $i = \{1,2, \dots , n\}$ to a response, y. Feedforward neural networks have one-way information flow from input features to output, whereas recurrent neural networks have feedback connections [33].

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

$$f\left(x_{i,..,n}\right) = f_a^m(f_a^{\cdots}(f_a^2(f_a^1(x_{i,\dots,n}))))$$

Equation 3

The superscript of these functions, $f^{\cdot}$, indicates the layer index of the function in an ANN, with m 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 [34].

The process of training such a network, f, to give the closest approximation to the desired output, y, is an iterative process, involving passing many observations, each having the same feature set $x_{i,\ldots,n}$ through the MLP, assessing the output, $\hat{y}$, according to an error metric, E, and individually adjusting each of the mapping functions $f_a^{j,\ldots,m}$ according to their contribution to the differential of the magnitude of error at the conclusion of each training step k. In other words, a parameter set θ, pertaining to each f_a^j is iteratively adjusted according to $\frac{\partial E_k}{\partial f_a^j}$. [33].

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 [33].

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 $w^T x + b$, where w^T is a vector of weights of the same length as the set of input features, which are essentially a set of coefficients for each f_a in the chain of functions, and b is a real-valued bias term, which is essentially an intercept term for each f_a [33].

It is easy to see that chaining such a set of linear models without applying nonlinear transformations (denoted as $\phi(f_a(x))$) to what are essentially an arbitrary number of linear regression functions ($y = \beta_1 x_1 + \beta_1 x_1 + \beta_1 x_1 + c$), one would simply arrive at another linear model [33]. Non-linear transformations applied over $w^T x + b$ allow deep learning models to more accurately model the multidimensional feature space of the data distribution. Various nonlinear transformations (more commonly known as activation functions) can be viewed on the Keras website at [35].

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

$$h = \phi(W^T x + b)$$

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

$$h^{(1)} = \phi^{(1)}(W^{(1)T} x + b^{(1)})$$

$$h^{(2)} = \phi^{(2)}(W^{(2)T} h^{(1)} + b^{(2)})$$

$$h^{(3)} = \phi^{(3)}(W^{(3)T} h^{(2)} + b^{(3)})$$

We now have a vector of weights multiplied by a vector of input features, which can be the original features fed to h_1, or the weighted outputs of previous hidden units in $h_{2,\ldots,n}$. 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 θ, need to be optimized to arrive at a reasonable approximation of a theoretically optimal mapping function $f^*(x) = y$ [33].

To achieve the optimization of θ, most deep learning models utilize the concept of maximum likelihood, to minimize a loss function $J(\theta)$.

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 [33]:

$$g \leftarrow \nabla_{\hat{y}} J = \nabla_{\hat{y}} L(\hat{y}, y)$$

In this manner all weights and biases in the ANN are repeatedly adjusted, proportionately to their contribution to the loss function at that iteration, until a (hopefully global) minimum is achieved [33].

3.2.1.1 Optimization

The essential optimization objective in deep learning is to find the optimal set of hyperparameters θ to minimize the objective function $J(\theta)$ [33].

Adaptive learning rates, utilization of the second derivative of the loss function during training and various parameter initialization- and other advanced strategies can be employed to make the training/ optimization process more effective [33].

3.2.1.1.1.1 Adam

Originating as an acronym for “adaptive moments”, the Adam algorithm is generally touted as an optimization strategy robust to various settings of hyperparameters. Adam combines features of momentum and RMSProp, by using momentum to estimate the first moment of the gradient and by applying bias corrections to both the first and second order moments of the gradient [33].

3.2.1.2 Loss Functions

3.2.1.2.1 Binary Cross-entropy

3.2.1.2.2 Focal Loss

Focal Loss was proposed by [37] as a loss function which down-weights the importance of well-classified examples, effectively making training examples that are more difficult to classify contribute more to the overall loss. This is useful in cases, such as this project, where one class dominates the class distribution and therefore becomes favoured during prediction, since favouring the dominant class results in a low overall loss.

Focal loss is not a default loss function in Keras, but has been implemented as a custom loss function by [38] for Python. The author subsequently adapted this for use in Keras with R [39], since no equivalent custom focal loss implementation could be found online for R.

Figure 21 shows how the Focal loss function decreases steeply as classification probability approaches the true class level. This loss function allowed for building models on a dataset with imbalanced classes, without having to resort to down-sampling or up-sampling and therefore made it possible to use a much larger training dataset, the combination of these two factors probably played the biggest part in the very low pion efficiencies obtained with this approach, during the final stage of model building.

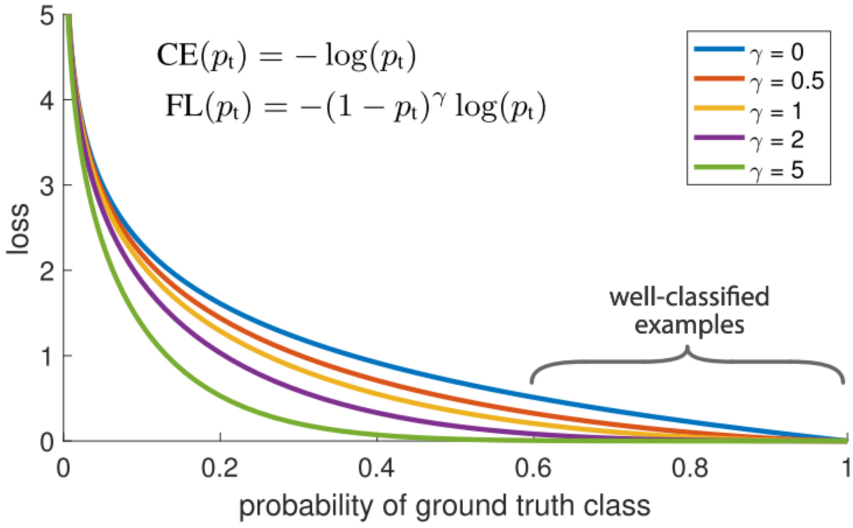


Figure 21: Focal Loss where true class is 1.

3.2.2 Regularization and Optimization for Deep Learning

3.2.2.1 Regularization

Regularization strategies are often employed in Deep Learning to reduce test error; by potentially sacrificing accuracy on training set predictions; effective regularization reduces overfitting of the model to features only present in the training data, and therefore increases accuracy on unseen data [33].

Regularization strategies can be achieved by, for example, constraining parameter values by adding penalty terms to an objective function or by explicitly constraining parameters. Carefully designed regularization processes can improve performance on test data by encoding prior domain knowledge, making an undetermined problem determined, or by simplifying the model so that it generalizes better [33].

3.2.2.1.1 Dropout

Dropout is a computationally inexpensive alternative regularization method, which consists of training the entire ensemble of subnetworks which can be achieved by setting the output of a subset of hidden units to zero, thus approximating model averaging methods [33].

Practically, dropout is achieved by a combination of mini-batch training and binary mask generation during each minibatch training round. The binary mask is of the same dimensions as the input- and hidden- units and each element in the mask is multiplied by its corresponding neuron, effectively pruning the neural network by setting the output of a random subset of neurons to zero [33].

The probability of sampling a 1 at each unit of the mask is a hyperparameter set before training. Each unit in the mask is sampled independently [33].

3.3 Convolutional Neural Networks

3.3.1 The Kernel Concept and Motivation for CNNs

Convolutional Neural Networks (CNNs) are an extension of deep learning models, highly successful in processing data with a grid-like topology, e.g. images. At least one linear mathematical operation, called a convolution, is applied in CNNs, usually in addition to the general matrix multiplication performed in traditional feedforward neural networks [33].

An example of a simple 2D convolution (multiplying a 3×4 matrix by a 2×2 kernel) is shown below (adapted from [33]).

$$\begin{matrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{matrix} \quad * \quad \begin{matrix} w & x \\ y & z \end{matrix}$$

$$=$$

$$\begin{matrix} aw + bx + ey + fz & bw + cx + fy + gz & cw + dx + gy + hz \\ ew + fx + iy + jz & fw + gx + jy + kz & gw + hx + ky + lz \end{matrix}$$

There are three major mechanisms that improve the accuracy of ML algorithms that motivate the implementation of convolutions in a deep learning architecture, namely parameter sharing, equivariant transformations and sparse interactions [33]. These will be discussed below.

Sparse interactions occur in CNNs because of kernels that are smaller than the input matrix, which means that every input unit does not have a connection to every output unit (as is the case in fully connected traditional ANNs), this sparsity of weights allows for the detection of meaningful small-scale features, such as edges, which are combined downstream (via indirect interactions of neurons in preceding layers) into progressively larger features, such as textures, shapes and actual visual elements, such as faces. Reducing the number of weights in this manner also leads to an increase in the efficiency of the neural network, since fewer operations are required per layer and fewer weights need to be stored and adjusted [33].

Parameter sharing allow certain parameters to be used by more than one function in a CNN, unlike traditional neural networks, which use each weight in a neural network in just one operation when the network's output is calculated. In a CNN, each element of the kernel is multiplied by every element of the input matrix (where dimension differences do not allow for this, edges may be padded with zero-valued matrix elements to enable it). The weights of the kernel function are learnt and applied uniformly, i.e. they are not relearned at each position of the input matrix, again this has benefits with regards to computational efficiency [33].

Equivariance to translation is a phenomenon which results from parameter sharing and means that the output of a convolutional layer changes in the same way that its input changes, i.e. $f(x)$ is said to be equivariant to a function g if $g\big(f(x)\big) = f(g(x))$. In a convolution operation, the function g translates (shifts) the input matrix in some way, but since the convolution operation is equivariant to the function g, it does not matter at which (x,y) coordinates a feature occurs in the input matrix, since it will still result in the same output after the convolution operation has been applied [33].

3.3.2 Pooling

CNN layers are generally composed of three operations:

1. The appropriate amount of convolution operations, as introduced above, are applied in parallel over the input matrix
2. A non-linear activation function is applied to the output of each convolution operation performed in step one
3. A pooling operation introduces an additional final modification to the layer output

The pooling function in step 3 above, performs a statistical summary over a window of outputs within a defined range, which could be, for example, the L_2-norm, mean or maximum over the series of rectangular ranges thus defined [33].

Pooling serves the purpose of insuring invariance to local translation, where the presence of a feature matters more than its location. In some cases, the specific orientation and location of a feature does matter though. Pooling over separate convolutions that are independently parameterized can allow the ANN to learn which translations it should be invariant to which translations it shouldn't be invariant to [33].

For computational efficiency, downsampling of the convolution function can be implemented by skipping over some positions in the kernel, specified by a parameter called stride [33].

Figure 22 illustrates how implementing a convolution with stride = 2, i.e. only sampling every second pixel for convolution, is mathematically equivalent to performing downampling after a convolution applied to all pixels (i.e. stride = 1), followed by downsampling [33].

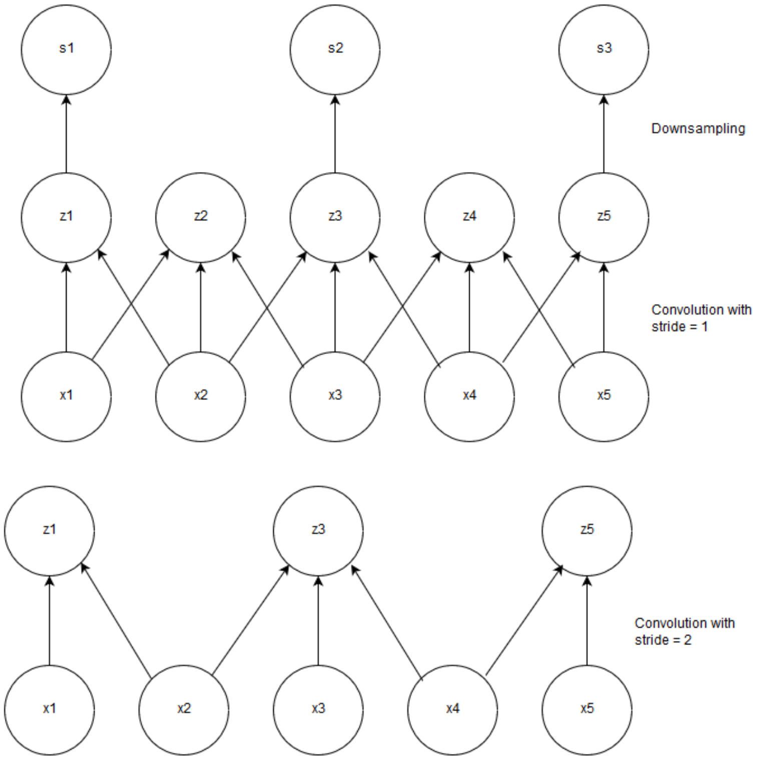


Figure 22: Illustration of mathematical equivalence of implementing a convolution with unit stride followed by downsampling to implementing a convolution with stride = 2.

Zero-padding is often applied to the input vector in order to prevent it from shrinking by one pixel less than the applied kernel width, i.e. for an input image of width m and kernel width k, the output of the convolution with no zero-padding will be m-k+1, a situation which would enforce smaller networks and smaller subsequent kernels if not accounted for, which in turn would limit the capacity of the network to find useful representations of the data [33].

Convolutions applied with no padding of the input image are known as valid convolutions, where pixels in the output of a convolution are a function of the same amount of pixels in the input, and the kernel can only be applied to positions on the image where the kernel is contained by the image [33].

When just enough zero-padding is applied to the input image to ensure that the output will be of the same dimensions, the convolution is known as a same convolution [33]. Although same convolutions do not limit the size of the network and allow one to build neural networks of arbitrary depth, they still result in pixels close to the edges of the image having less connections to the output image and therefore that their influence on the network as a whole will be reduced [33].

3.4 Statistical Tests

3.4.1 Hypotheses

Statistical tests are mathematical constructs designed to enable a researcher to make a measurable statement concerning to what extent observed data agrees with probabilistic predictions made about it in the form of a hypothesis [36].

When performing a statistical test, a null hypothesis, denoted as H_0, is put forth, as well as one or more alternative hypotheses, $(H_1, H_2, \ldots)$.

Given a dataset of n measurements of a random variable $x = x_1, \ldots, x_n$, a set of hypotheses H_0, H_1 are proposed, each specifying a joint probability density function (p.d.f.), i.e. $f(x|H_0), f(x|H_1), \ldots$

In order to assess how well the observed data agrees with any given hypothesis, a test statistic $t(x)$, which is a function of the observed data, is constructed.

A specific p.d.f. for the test statistic, t, is implied by each of the hypotheses, i.e. $g(t|H_0), g(t|H_1), \ldots$

While the test statistic can be a multidimensional vector $t = t_1, t_2, \ldots, t_m$ (in principle, even the original vector of observed data points $x = x_1, x_2, \ldots, x_n$ can be used), constructing a test statistic of lower dimension (where m < n) reduces the amount of data being assessed, without losing discriminative power.

If a scalar function $t(x)$ is used as the test statistic, a p.d.f. $g(t|H_0)$ is given which t will conform to when H_0 is true, similarly t will conform to a different p.d.f. $g(t|H_1)$ when H_1 is true. Figure 23 illustrates how setting a threshold value for the test statistic, i.e. t_{cut}, results in rejection of the null hypothesis when $t > t_{cut}$.

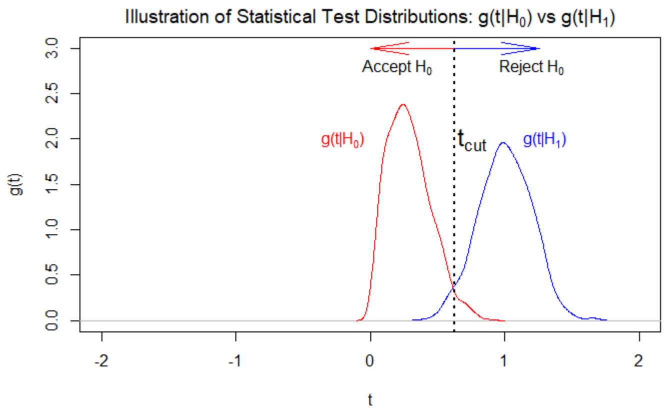


Figure 23: An illustration of rejection or acceptance of the null hypothesis, under the assumed distributions of H_0 and H_1, when t falls in the critical region $t > t_{cut}$

The support for various hypotheses under the observed data distribution is framed in terms of acceptance or rejection of the null hypothesis by defining a critical region for the test statistic, beyond which the null hypothesis is rejected; i.e. when the observed value of t lies within the critical region, we reject H_0. Conversely, when t lies within the complement of the critical region, it is said to be within the acceptance region, which will result in the researcher accepting H_0.

3.4.2 Significance Level and Power

The critical region for rejection of the null hypothesis is defined by a cut-off point, such that the probability of t being observed there is defined by a value α, called the significance level of the test.

In the example shown in Figure 23, a critical region is defined by a value: t_{cut}, which defines the lower decision boundary for rejecting the null hypothesis.

The significance level defined as such is given by

$$\alpha = \int_{t_{cut}}^{\infty} g(t|H_0)dt$$

H_0 would not be rejected when $t < t_{cut}$, and there is a probability of α of rejecting H_0 when H_0 is in fact true (called an error of the first kind), as well as a probability of accepting H_0 when H_1 was actually true. The probability of making an error of the second kind is given by

$$\beta = \int_{-\infty}^{t_{cut}} g(t|H_1)dt$$

$1 - \beta$ is called the power of the statistical test to discriminate against H_1.

3.4.3 Statistical Tests for Particle Selection

In the case of electron-pion particle identification dealt with in this dissertation, we consider the class "electron" as signal and "pion" as background. As such, we define $H_0 = e,\ \ H_1 = \pi$, and by extension, we treat the output of the final hidden unit in the neural network as a test statistic in its own right, lying either within a p.d.f. $g(t|H_0)$ when it is an electron or $g(t|H_1)$ when it is a pion. In order to accept or reject H_0, we define a critical region t_{cut}. When $t \geq t_{cut}$, we classify the particle as an electron.

When looking at the probability of classifying a specific particle as a given type, we define the selection efficiencies, i.e. the electron efficiency ε_e and pion efficiency ε_π as follows:

$$\varepsilon_e = \int_{-\infty}^{t_{cut}} g(t|e)dt = 1 - \alpha$$

$$\varepsilon_\pi = \int_{-\infty}^{t_{cut}} g(t|\pi)dt = \beta$$

This cut-off point can be chosen so as to accept as many electrons as possible, but the price paid for high electron efficiency is a large amount of pion contamination in the electron sample.

Based on the probability of a particle being an electron obtained from each of the 6 detector layers in the TRD, we use a Bayesian approach outlined in the formula below:

$$P(elec) = \frac{\prod_{j=1}^{6} P_j(elec)}{\sum_{k\in e,\pi} \prod_{j=1}^{6} P_j(k)}$$

Here, $P_j(elec)$ is the probability of the track being an electron obtained from layer j.

3.5 Implementation

3.5.1 Methods

Various Machine Learning strategies were employed in the task of particle identification. Deep learning models were built in Keras, with a Tensorflow backend, using the SLURM-managed UCT HPC Cluster extensively to build multiple models simultaneously.

Non-Deep Learning Methods were implemented locally, using H2O.ai.

3.5.2 Results

Three sets of results will be presented:

The most successful particle identification strategy on uncalibrated raw digits will be discussed in detail in Section 3.5.2.1. Next, a summary of other models that were built to this end will be presented, at the hand of Figure 25 for all 2D Convolutional Neural Networks built, and as a text summary in Section 3.5.2.2 for all models built. Please note that models shown in Section 3.5.2.2 were trained on down-sampled data, incorporating all clean tracks (i.e. 6 tracklets obtained) for electrons and an equal number of pions. This data was normalised as follows:

$$x = {}^{x}/_{\max(x)}$$

3.5.2.1 Most successful approach

The most successful pion rejection and electron acceptance results were obtained by incrementally training a 2D Convolutional Neural Network, using Focal Loss as the loss function to be optimised, Adam as the optimizer…

- The full dataset was used during this stage.
- All tracklets with no signal, i.e. images where all the pixel values were zero, were removed.
- Data was not normalised or standardised.
- Data was not down-sampled or up-sampled to account for class imbalances.
- Data was split into the following momentum bins:
  - $P \leq 2\ GeV$, $2\ GeV < P \leq 3\ GeV$ and $3\ GeV < P \leq 4\ GeV$
  - Results in the $4\ GeV < P \leq 5\ GeV$ and $5\ GeV < P \leq 6\ GeV$ were much worse and are not included here

- A Convolutional Neural Network was trained incrementally, per momentum bin, by saving the weights configuration after training on the previous momentum bin
- Results are summarised in Figure 24.

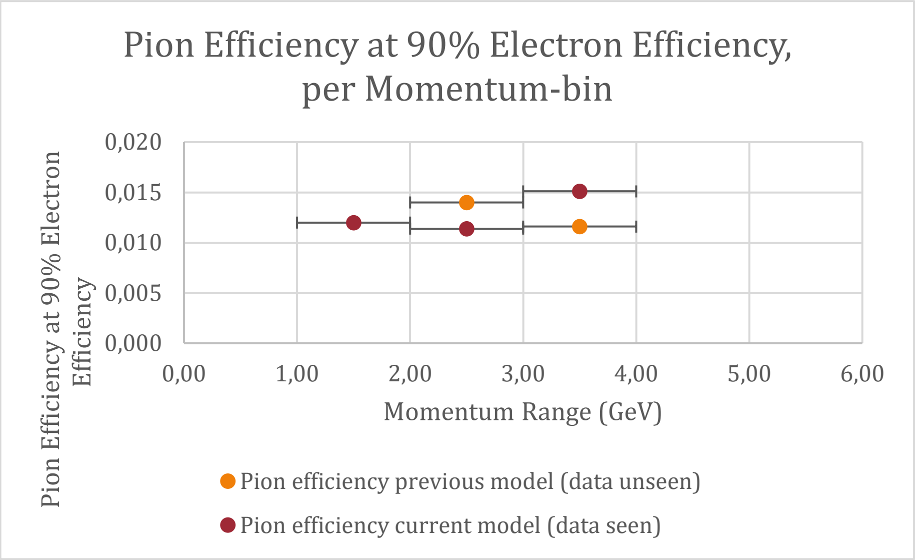


Figure 24: Summary of incrementally trained 2D Convolutional Neural Network: Red marks indicate pion efficiency at 90% electron efficiency, when the incrementally trained model was evaluated on data from the momentum-bin the model was last trained on. Orange marks indicate the results when testing the model on data from the next momentum bin (before training on data in that bin). i.e. after training the model on tracklets in the $P \leq 2GeV$ range, the model was first evaluated on this range (first red mark), then tested on the $2GeV < P \leq 3GeV$ range (first orange mark), then trained and evaluated further.

3.5.2.2 Summary of Other Results

3.5.2.2.1 2D Convolutional Neural Networks

$\varepsilon_{\pi} = 2.2\%$ at electron efficiency $\varepsilon_{e^-} = 90\%$

3.5.2.2.2 1D Convolutional Neural Networks

$\varepsilon_{\pi} = 6.55\%$ at electron efficiency $\varepsilon_{e^-} = 90\%$

3.5.2.2.3 Fully Connected Feedforward Neural Networks

$\varepsilon_{\pi} = 14.86\%$ at electron efficiency $\varepsilon_{e^-} = 89.99\%$

3.5.2.2.4 LSTM Neural Networks

$\varepsilon_{\pi} = 5.3\%$ at electron efficiency $\varepsilon_{e^-} = 90\%$

3.5.2.2.5 Non-Deep Learning (Tree Based) Models

3.5.2.2.5.1 Random Forests

$\varepsilon_{\pi} = 5.8\%$ at electron efficiency $\varepsilon_{e} = 90\%$

3.5.2.2.5.2 Gradient Boosting Machines

$\varepsilon_\pi = 6.59\%$ at electron efficiency $\varepsilon_e = 89.99\%$

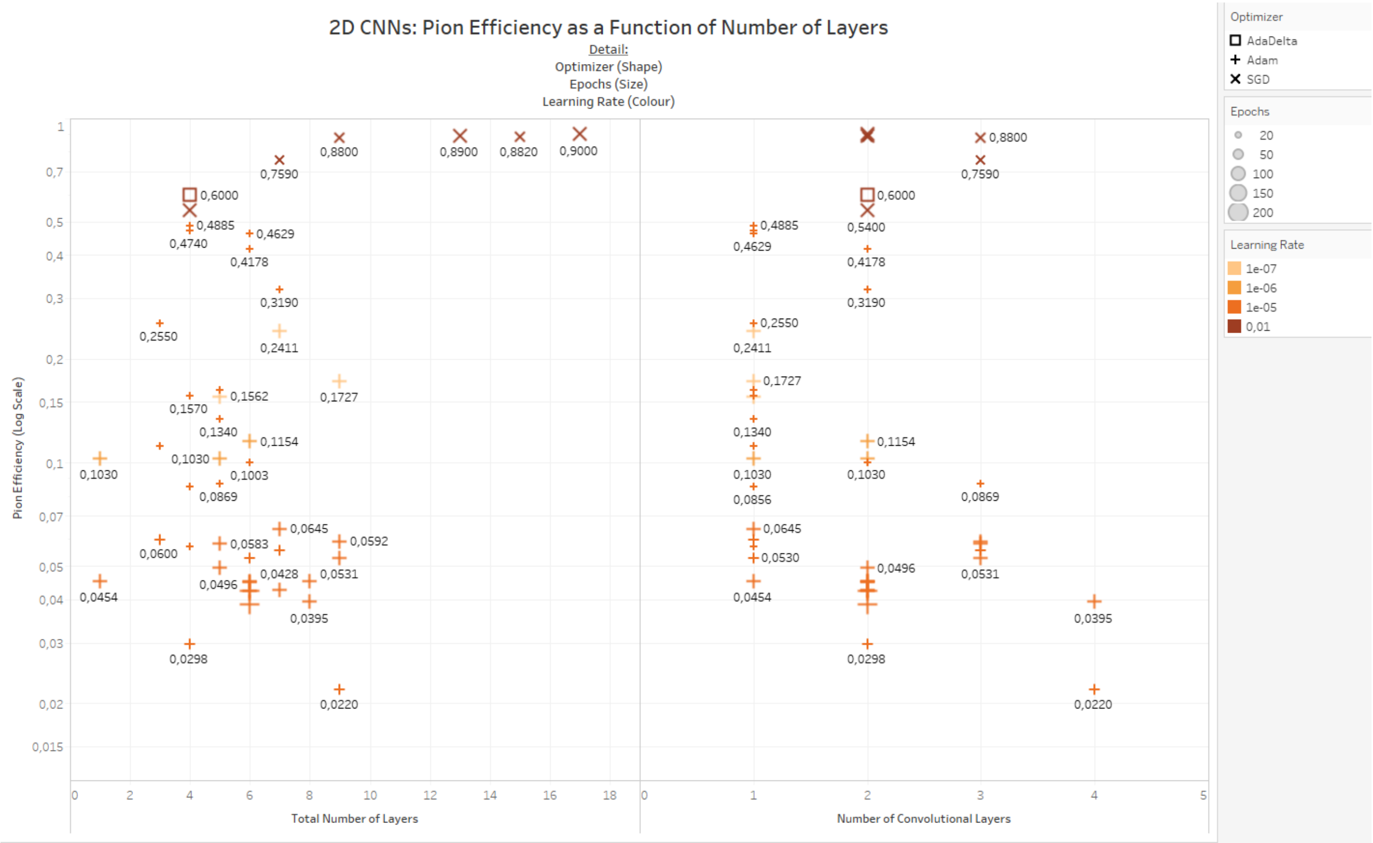


Figure 25: 2D Convolutional Networks are compared in terms of number of total layers (dense and convolutional combined, LHS) and number of convolutional layers (RHS). Learning rate is shown on a colour gradient as indicated by the legend. Number of epochs trained (mark size) and optimization algorithm (mark shape) are also indicated. Pion efficiency is plotted on the y-axis (logarithmic scale). *Learning rate seems to be the most important distinguishing element in achieving low pion efficiency (some of the best-performing models used a learning rate of $\alpha = 10^{-5}/\alpha = 10^{-6}$, a very high learning rate ($\alpha = 0.01$) results in poorly performing models, whereas a very low learning rate ($\alpha = 10^{-7}$) does not converge within a feasible number of epochs.) Using more than one convolutional layer is also a seemingly more successful strategy than only using one layer, but no outright statements about architecture can be made. The use of a lower learning rate and the Adam optimizer are entangled variables (models trained with low learning rates were all optimised using Adam, but common practice suggests that Adam is the more robust algorithm to use.*

4 HIGH ENERGY PHYSICS EVENT SIMULATIONS

4.1 Monte Carlo Simulations: Geant4

4.1.1 Theory

4.1.2 Implementation

In order to prove that Geant4 simulations might not be as accurate as assumed to be, a simulation was run, set to generate pions from the following LHC run: 2016/LHC16q/000265343. A convolutional neural network was able to distinguish simulated pions from real pions obtained during that run to a high degree of accuracy and therefore motivated the Deep Generative Modelling Section of this thesis.

4.1.2.1 Geant4 Configuration and Simulation

Geant4 simulations were configured using https://github.com/PsycheShaman/trdpid/blob/master/sim/Config.C , simulations were run as per the following shell script: https://github.com/PsycheShaman/trdpid/blob/master/sim/runtest.sh which calls upon the simulation script https://github.com/PsycheShaman/trdpid/blob/master/sim/sim.C the reconstruction script https://github.com/PsycheShaman/trdpid/blob/master/sim/rec.C and the analysis script https://github.com/PsycheShaman/trdpid/blob/master/sim/ana.C in sequence in order to create Monte Carlo simulations in a similar format to raw data analysed during particle identification.

4.1.2.2 Distinguishing Geant4-simulated Pions from Real Pions

The task of distinguishing simulated from real data was performed using a 2D convolutional neural network, with architecture shown in Figure 27.

Distinguishing Geant4 simulations from real data proved to be a much easier task than distinguishing real electrons from real pions, as depicted in the training graphs in Figure 26.

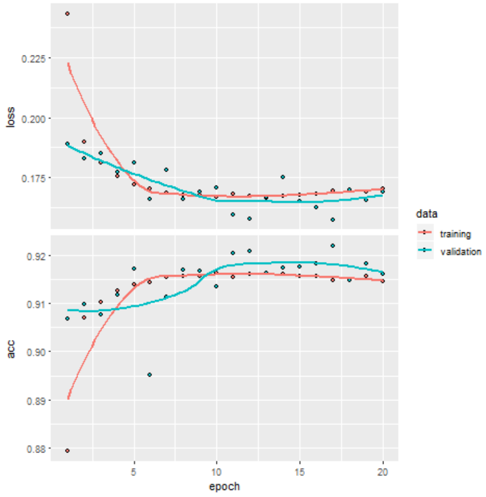


Figure 26: Training loss and accuracy curves for training and validation data

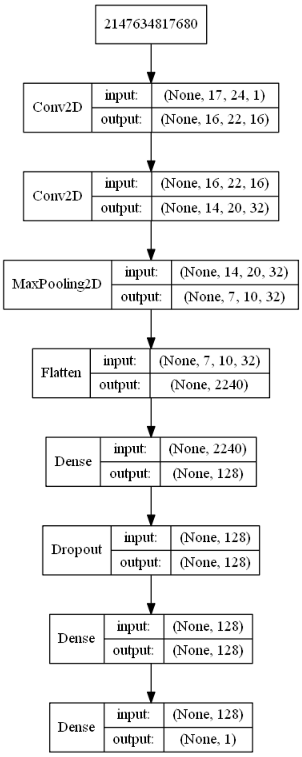


Figure 27: Model architecture for distinguishing real from Geant4-simulated data

Table 2 shows the obtained confusion matrix for the following model architecture:

Table 2: Confusion Matrix for distinguishing between Geant4 vs Real Data

Prediction/Actual	π_{geant}	π_{real}
π_{geant}	42 553	681
π_{real}	7 069	24 058

4.2 Deep Generative Models

Generative models are concerned with modelling potentially high-dimensional distributions. Dependencies between various random variables in the multidimensional distribution can also be captured during this modelling process [40].

Generative models are concerned with generating data that is similar to seen data, but not exactly the same, i.e. our training examples X are distributed according to some unknown distribution $P_{gt}(X)$ and we want to model a distribution P which is as similar as possible to P_{gt} and therefore allows us to generate new examples X by sampling from P [40].

Neural networks can be utilised as function approximators towards constructing a modelled distribution P as outlined above [40].

4.2.1 Background: Latent Variable Models

When there are complex dependencies between the dimensions of the data, generative models become very hard to train. Latent variables are samples drawn from specific latent distributions constructed during training, before the generative process commences, i.e. the model first chooses what it is going to simulate before it starts simulating [40].

In order to deduce that a generative model is representative, one needs to find that for each datapoint X in χ, there are one or more latent variable settings which result in the model generating something sufficiently similar to X [40].

A vector of latent variables z, are sampled from a high dimensional latent space Z, according to a probability density function (p.d.f.): $P(z)$ defined over Z. A group of deterministic functions $f(z;\theta)$ parameterized by a vector θ in some space Θ, with $f\colon Z \times \Theta \to \chi$. While f is deterministic, z is randomly sampled and θ is fixed, which makes $f(z;\theta)$ a random variable in the space χ. θ needs to be optimized so that sampling z from $P(z)$ will result in a high probability of $f(z;\theta)$ outputting data similar to the training data X [40].

More formally, we want to maximize the probability of each X, according to:

$$P(X) = \int P(X|z;\theta\ )P(z)dz$$

$f(z;\theta)$ has been changed to a distribution $P(X|z;\theta\ )$ in the expression above, in order to show explicitly that X depends on z. Maximum Likelihood underpins the notion that if X is likely to be reproduced, generated examples that are highly similar to X are also likely to be produced, and dissimilar examples are unlikely [40].

VAEs often model the output distribution as a Gaussian, $P(X|z;\theta\ ) = N(X|f(z;\theta),\sigma^2 * I)$, i.e. the distribution has mean $f(z;\theta)$ and covariance equal to some scalar σ multiplied by the identity matrix I, with σ being a tuneable hyperparameter [40].

A VAE will in general not produce examples identical to any X, especially not during early training, but under the Gaussian assumption, $P(X)$ can be increased via gradient descent by making $f(z;\theta)$ approach X given some z [40].

4.2.2 Variational Autoencoders

Variational Autoencoders (VAEs) aim to maximize $P(X) = \int P(X|z;\theta\ )P(z)dz$ by defining latent variables z and integrating over z. Choosing the latent variables z are not trivial, since z is generally not just defined by the label of the example that needs to be generated, but by other features specific to the example [40], in our case, z would not just be electron or pion, but additional dimensions such as the particle’s momentum, angle, etc. Generally, a researcher would not explicitly specify what the dimensions of z specify, nor how the dimensions of z depend on one another [40].

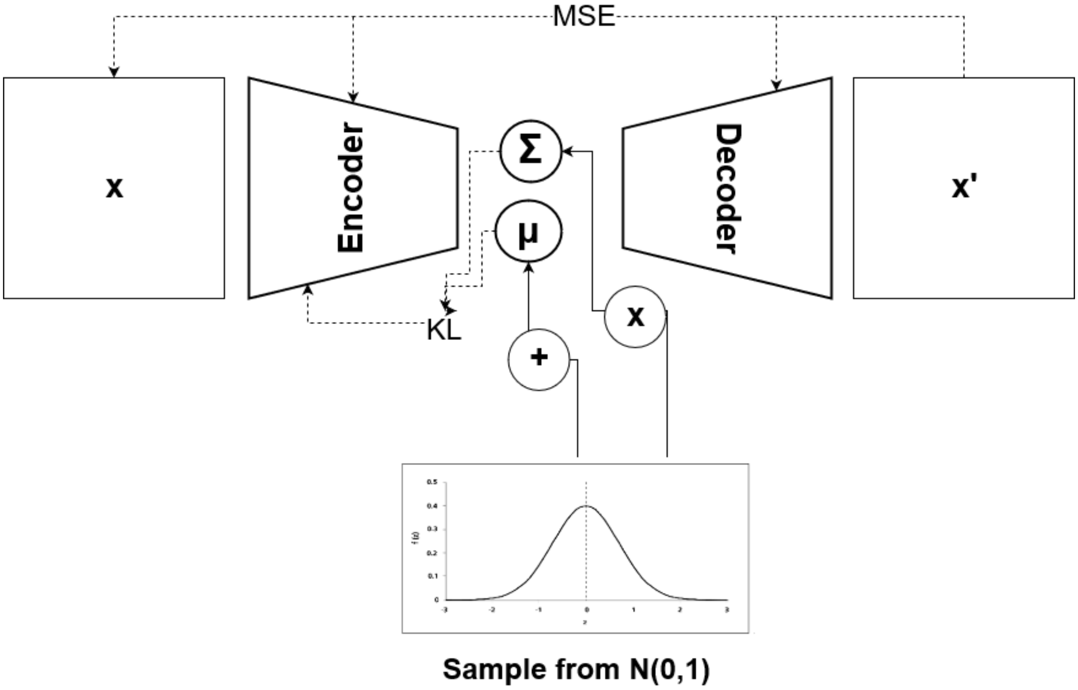


Figure 28: Simplified diagram of a Variational Autoencoder

In VAEs, z is drawn from a distribution $N(0, I)$, where I is the identity matrix, since any distribution in d dimensions can be generated by sampling from d normally distributed variables and mapping them through a function with high enough capacity to generate X. When $f(z; \theta)$ is a neural network then the initial layers will be involved in generating z while the later layers will be concerned with mapping z to X; $P(X)$ will be maximized by finding a computable formula for it, taking its gradient at each epoch and optimizing it using stochastic gradient ascent [40].

$P(X)$ can be computed approximately by sampling z values repeatedly $z = \{z_1, z_2, \dots, z_n\}$ and computing $P(X) \approx \frac{1}{n}\sum_i P(X|z_i)$, in high dimensional spaces, n might have to be very large before $P(X)$ can be accurately approximated [40].

For most z, $P(X|z)$ will be close to zero, but in order for the VAE to be useful, we need to sample z values that are likely to have resulted in X and sample only from that subset, a new function $Q(z|X)$ is needed to take an existing X value and calculate a distribution of z values that could have realistically resulted in X being generated; this narrows the universe of z values down from the larger universe of all $z's$ likely under the prior $P(z)$ [40].

How $E_{Z\sim Q}P(X|z)$ and $P(X)$ are related is one of the basic tenets upon which variational Bayesian methods are built. The Kullback-Leibler divergence ($\mathcal{D}$) between $P(z|X)$ and $Q(z)$ for an arbitrary Q which does not necessarily have to depend on X, is given by:

$$\mathcal{D}[Q(z)||P(z|X)] = E_{Z\sim Q}[log\, Q(z) - log\, P(z|X)]$$

$P(X)$ and $P(X|z)$ can be added to this equation by applying Bayes rule:

$$\mathcal{D}[Q(z)||P(z|X)] = E_{Z\sim Q}[log\, Q(z) - log\, P(z|X) - log\, P(z)] + log\, P(X)$$

Since $log\, P(X)$ does not depend on z, it appears outside the expectation. Rearrangement of this formula, negation and contraction of part of $E_{z\sim Q}$ into a KL-divergence term gives us:

$$log\, P(X) - \mathcal{D}[Q(z)||P(z|X)] = E_{Z\sim Q}[log\, P(X|z)] - \mathcal{D}[Q(z)||P(z)]$$

In the above equation, X is fixed and Q can be any distribution, regardless of whether it accurately maps X to $z's$ that could have produced X, but in our case we are interested in accurately inferring $P(X)$ and therefore we want to find a Q which does depend on X and which also keeps $\mathcal{D}[Q(z)||P(z|X)]$ as small as possible:

$$log\, P(X) - \mathcal{D}[Q(z|X)||P(z|X)] = E_{Z\sim Q}[log\, P(X|z)] - \mathcal{D}[Q(z|X)||P(z)]$$

The formula above is the central formula of the VAE, the left hand side is what needs to be maximized: $P(X)$, penalized by $-D[Q(z|X)||P(z|X)]$, which will be minimized if Q is a high capacity distribution which produces z values that are likely to reproduce X, the right hand side is differentiable and can therefore be optimized using gradient descent.

When looking at the above equation, the right hand side takes the form of an autoencoder, where Q encodes X into latent variables z and P decodes these latent variables to reconstruct X.

On the left side of the equation, $log\, P(X)$ is being maximized while $\mathcal{D}[Q(z|X)||P(z|X)]$ is being minimized. While $P(z|X)$ is not analytically solvable and simply describes z values likely to reproduce X, the second term in the KL-divergence on the left is forcing $Q(z|X)$ to be as similar as possible to $P(z|X)$, and under a model with sufficient capacity $Q(z|X)$ should be able to be exactly the same as $P(z|X)$, which will result in $\mathcal{D}$ being zero and the direct minimization of $log\, P(X)$, in addition $P(z|X)$ is no longer intractable since $Q(z|X)$ can be used to solve for it.

In order to minimize the right hand side of the above equation via gradient descent, $Q(z|X)$ will usually take the form:

$$Q(z|X) = N(z|\mu(X;\vartheta), \Sigma(X;\vartheta))$$

Where μ and Σ are deterministic functions with learnt parameters ϑ; in practice μ and Σ are learnt via neural networks and Σ is constrained to a diagonal matrix format. $\mathcal{D}[Q(z|X)||P(z)]$ therefore becomes a KL-divergence between two multivariate Gaussians, computed in closed form as:

$$\mathcal{D}[N(\mu_0, \Sigma_0)||N(\mu_1, \Sigma_1)] = \frac{1}{2}(tr(\Sigma_1^{-1}\Sigma_0) + (\mu_1 - \mu_0)^{\intercal}\Sigma^{-1}(\mu_1 - \mu_0) - k + log(\frac{det\Sigma_1}{det\Sigma_0})$$

With k indicating the number of dimensions of the distribution; this can be simplified to become:

$$\mathcal{D}[N(\mu(X), \Sigma(X))||N(0, I)] = \frac{1}{2}(tr\big(\Sigma(X)\big) + \big(\mu(X)\big)^{\intercal}\big(\mu(X)\big) - k - log\, det\, (\Sigma(X)))$$

The other term on the right hand side of the equation, $E_{Z\sim Q}[log\, P(X|z)]$, can be estimated by taking a sample from z and calculating $P(X|z)$ for that single sample to approximate $E_{Z\sim Q}[log\, P(X|z)]$.

Since we are doing stochastic gradient descent over different X values from our dataset D, we want to perform gradient descent on the following formula:

$$E_{X\sim D}[\log P(X) - \mathcal{D}[Q(z|X)||P(z|X)]] = E_{X\sim D}[E_{Z\sim Q}[\log P(X|z)] - \mathcal{D}[Q(z|X)||P(z)]]$$

By sampling a single value of X and a single value of z, we can compute the gradient of $\log P(X|z) - \mathcal{D}[Q(z|X)||P(z)]$, which when averaged over multiple samples, converges to the full equation to be optimized.

The issue here is that $E_{Z\sim Q}[\log P(X|z)]$ does not only depend on the parameters of P, but also those of Q, but this is not accounted for in the above equation. For VAEs to work properly, Q needs to be driven to produce $z's$ from X that are likely to be reliably decoded by P.

Figure 29 illustrates how this proxy formula can be used by averaging over multiple samples to get to the expected outcome, but since there is a sampling procedure embedded within the neural network, gradient descent cannot be performed on it.

Figure 30, on the other hand, shows how a “reparameterization trick” removes he sampling procedure from the neural network proper and treats it as an input layer. Since we have $\mu(X)$ and $\Sigma(X)$, we can sample ϵ from $N(0, I)$ and compute z from ϵ as follows: $z = \mu(X) + \Sigma^{\frac{1}{2}}(X) * \epsilon$.

As a result, the gradient of the following equation will actually be taken:

$$E_{X\sim D}\left[E_{\epsilon\sim N(0,I)}\left[\log P\left(X\middle|z = \mu(X) + \Sigma^{\frac{1}{2}}(X) * \epsilon\right)\right] - \mathcal{D}[Q(z|X)||P(z)]\right]$$

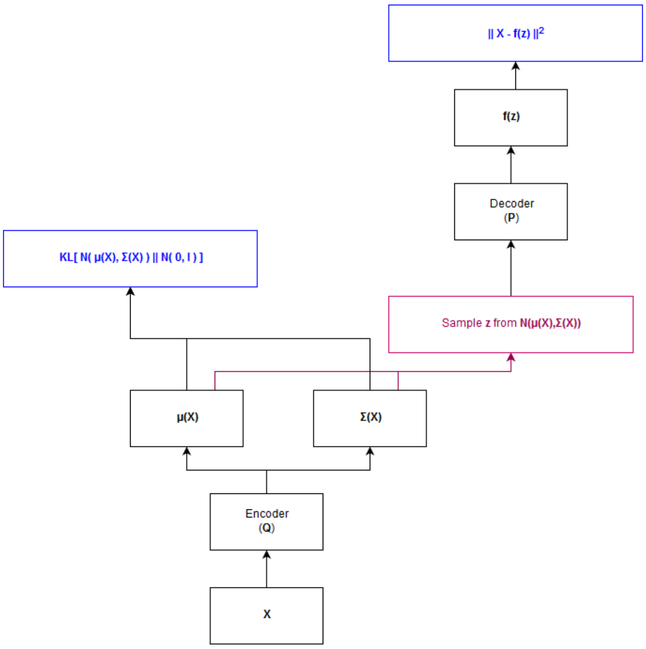

|| X - f(z) ||²
f(z)
Decoder
(P)
KL[N(μ(X), Σ(X)) || N(0, I)]
Sample z from N(μ(X),Σ(X))
μ(X)
Σ(X)
Encoder
(Q)
X

Figure 29: Training-time VAE

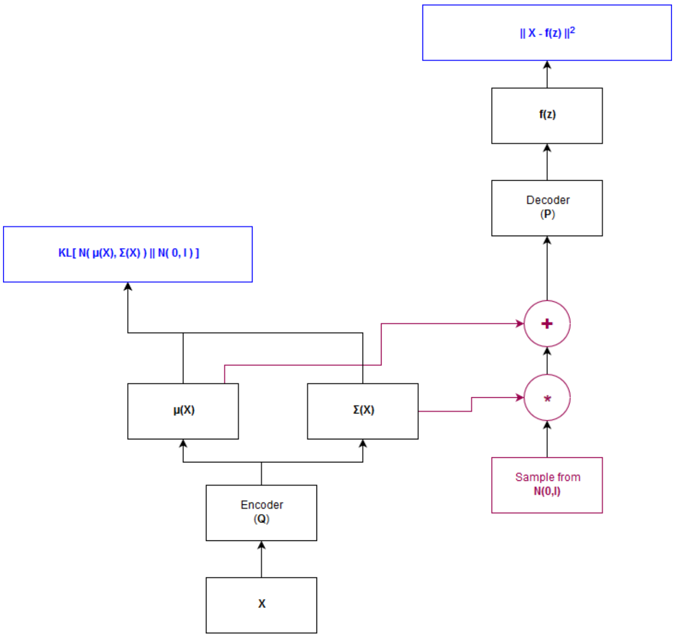


Figure 30: Training-time VAE with reparameterization trick to enable backpropagation

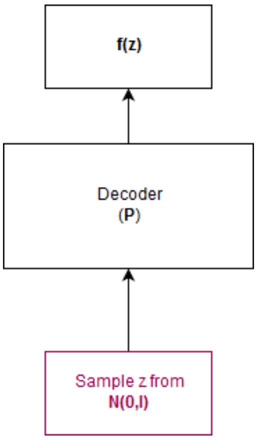


Figure 31: Testing time VAE

Once the model is ready to be tested, values from $z \sim N(0, I)$ are sampled and fed to the decoder; the encoder, along with the attendant reparameterization trick used during training are thrown away.

4.2.3 Generative Adversarial Networks

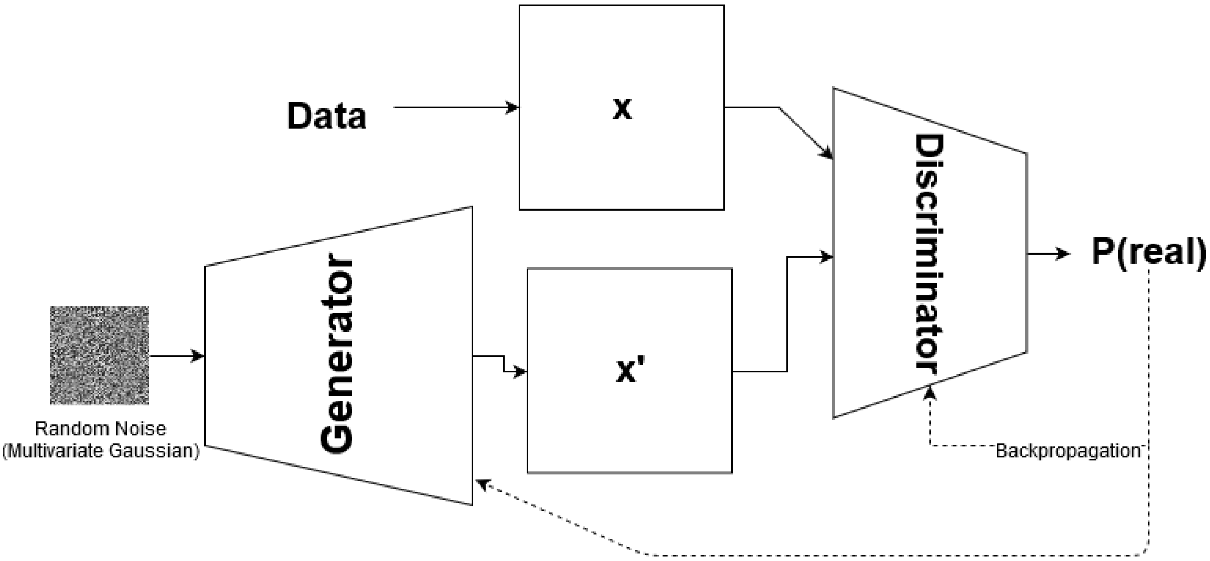


Figure 32: Simplified Diagram of a Generative Adversarial Network

Generative Adversarial Networks (GANs) are a deep learning framework which pits two neural networks against each other in an adversarial mini-max game: the generative model G is trained to the point where it accurately captures the distribution of the training data, and the discriminative network D takes the output of G and estimates the probability of whether G's output originated from the actual data distribution or from a model distribution [41].

The mini-max game can be expressed mathematically as:

$$\min_{G} \max_{D} V(D,G) = E_{x\sim p_{data}(x)}[log\, D(x)] + E_{z\sim p_z(z)}[log\left(1 - D\big(G(z)\big)\right)]$$

Essentially, the objective is to maximize the probability of D assigning the correct label to samples from G, i.e. is a given observation from the "data"- or "model" distribution, while training G to minimize $log\left(1 - D\big(G(z)\big)\right)$, i.e. we want G to produce samples that are hard to discriminate from samples from the true data distribution.

This is done by sampling from a random noise vector z, with a defined prior $p_z(z)$ and learning a transformation from the noise vector to a distribution which is highly similar (preferably identical) to the true data distribution; in practice, this transforming function is the generative network $G(z, \theta_g)$, with θ_g being the parameters of a deep neural network which maps z to data space.

In practice, the training algorithm will alternately optimize D for k steps and G for a single step, which allows D to remain close to its optimum if G does not change too rapidly, this also allows for the algorithm to run computationally more efficiently and prevents overfitting. During the early stages of training, it will be quite easy for D to discriminate between data and model samples, since G will still be learning to output more realistic samples, therefore G's objective function $log\left(1 - D\big(G(z)\big)\right)$ will saturate, so an alternative objective function $log\ D\big(G(z)\big)$ is maximized in practice by G, which does not change the dynamics of D and G much but allows for gradients that are sufficiently large to perform useful stochastic gradient descent.

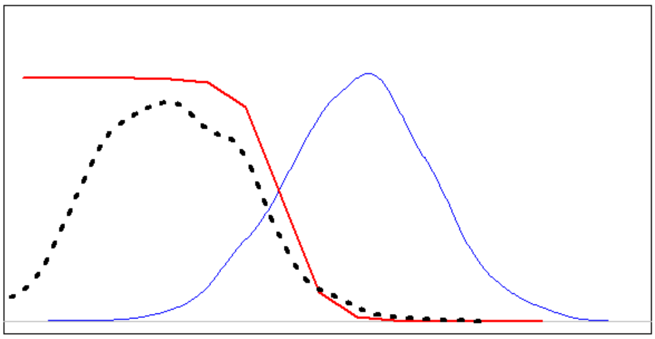


Figure 33: Gan Densities during training, close to convergence, P(x) is shown in black, G(z) in blue and D(G(z)) in red

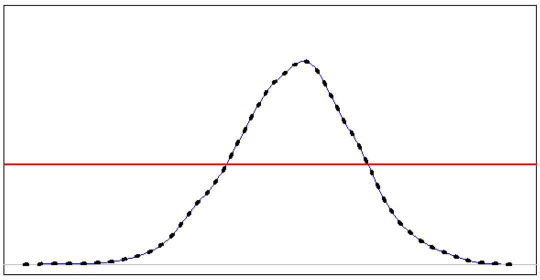


Figure 34: Gan Densities during training, once the Algorithm has converged, G(z) matches P(x) perfectly and D(G(z)) outputs 0.5 everywhere

4.2.4 Adversarial Autoencoders

Adversarial Autoencoders match the aggregated posterior of the latent space vector from an autoencoder $q(z) = \int_x q(z|x)p_d(x)dx$ with an arbitrary prior distribution $p(z)$, a process which results in meaningful samples being generated from any sample from any part of the prior space. The decoder function learns a function to map from the imposed prior distribution to the data distribution. In this set-up, the generator of the GAN also acts as the encoder function of the autoencoder, a process which assists the generator in fooling the discriminator of the GAN into misclassifying simulated data as real data [44].

4.3 Deep Generative Models Towards Event Simulation

4.3.1 Variational Autoencoders

Although various VAEs were prototyped, the following architecture produced the best results:

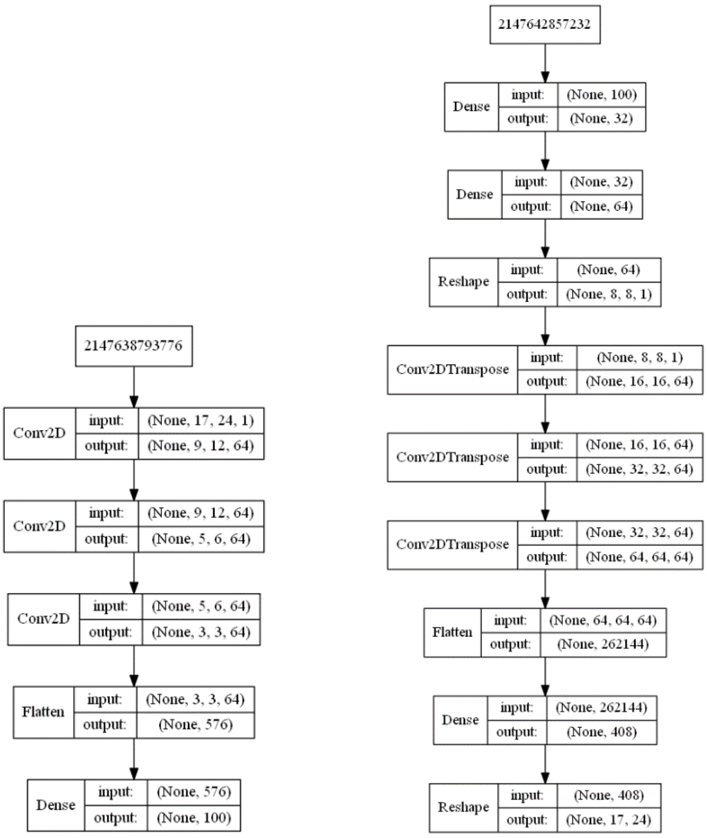


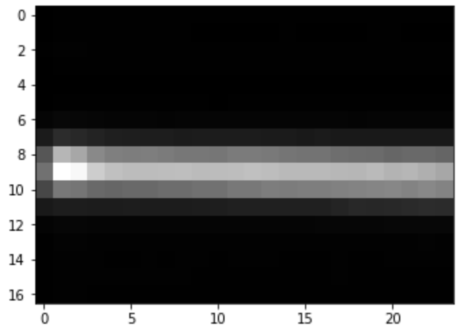
Figure 35: Encoder (left) and Decoder (right)

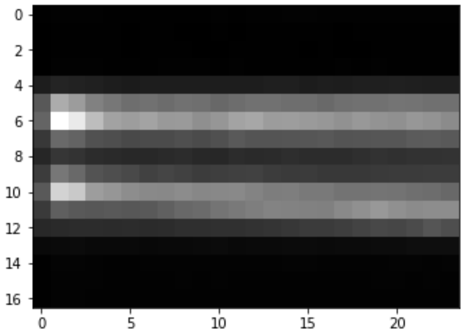
Encoder returns the μ for each of the 100 latent dimensions, $\sum$, which is calculated as $\mu \times 0.5$; and z, which is the result of multiplying a random normal vector ε with e^{Σ} and adding μ, i.e.

$$z = \left(\varepsilon \times e^{\Sigma}\right) + \mu$$

The input to the decoder is a sampled z vector as defined above.

Below are four examples of simulated tracklet image data, produced by the VAE as explained above.





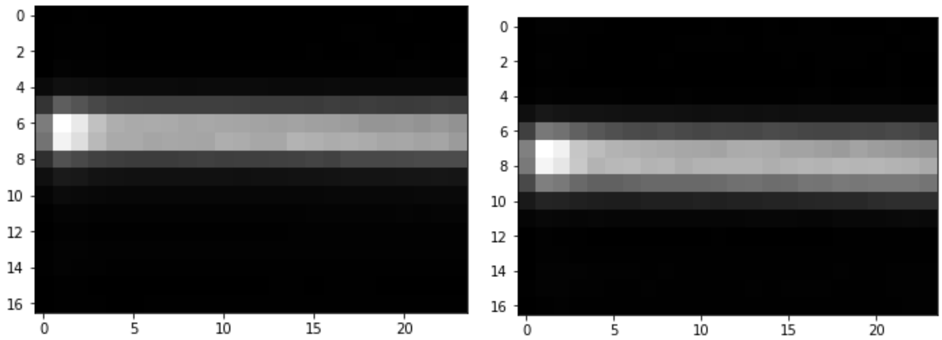


Figure 36: Four examples of simulated data created using a Variational Autoencoder

4.3.1.1.1 Deep Learning Towards Distinguishing Variational Autoencoder Data from Real Data

While these results look quite believable at first glance, it was quite easy to distinguish 100 000 real data samples vs 100 000 samples simulated with VAE using a CNN to 100% accuracy, as can be seen in Figure 37.

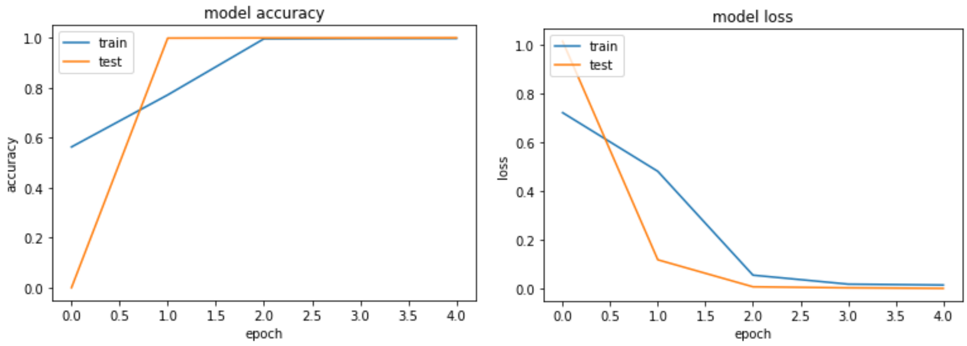
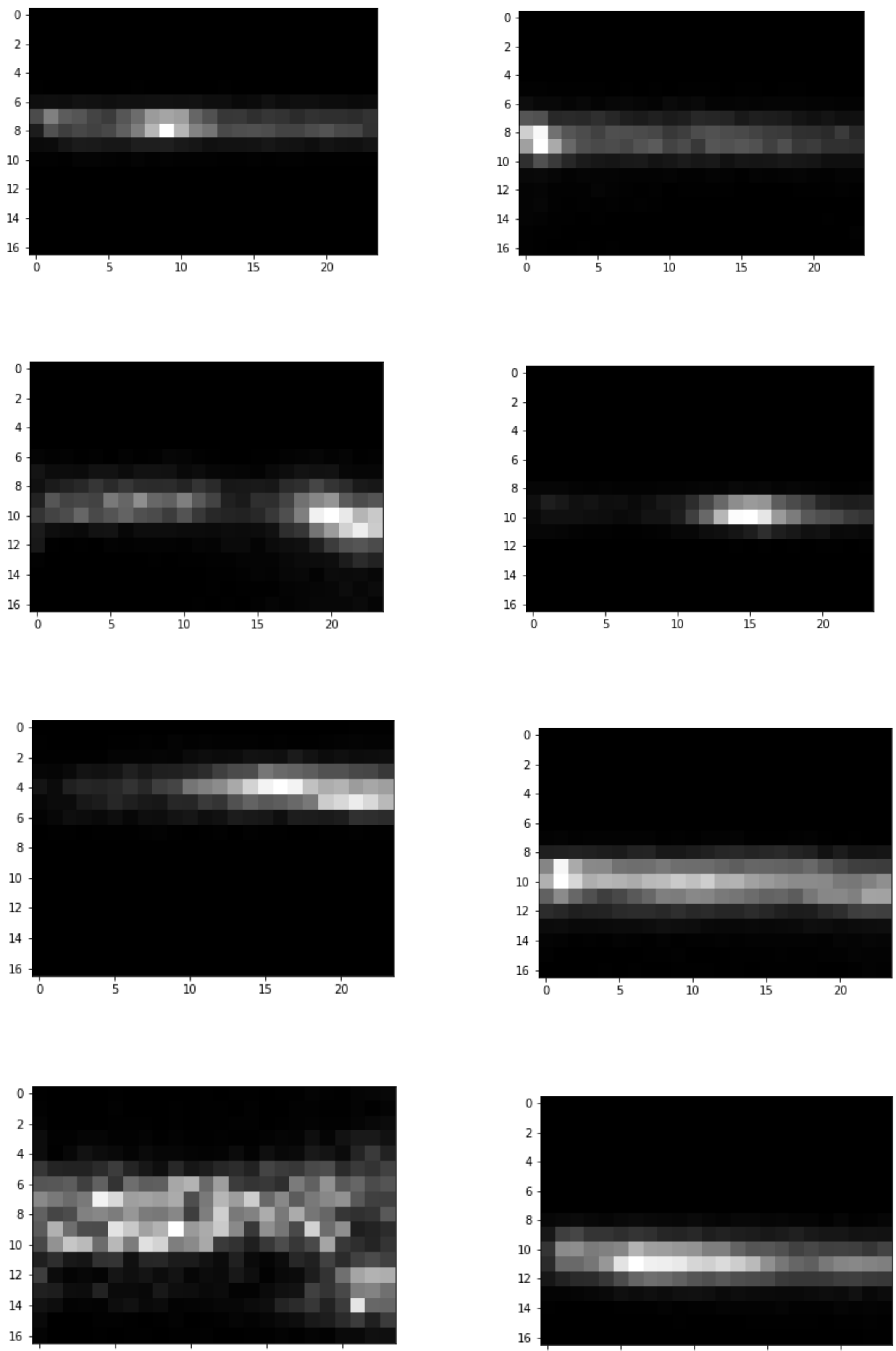


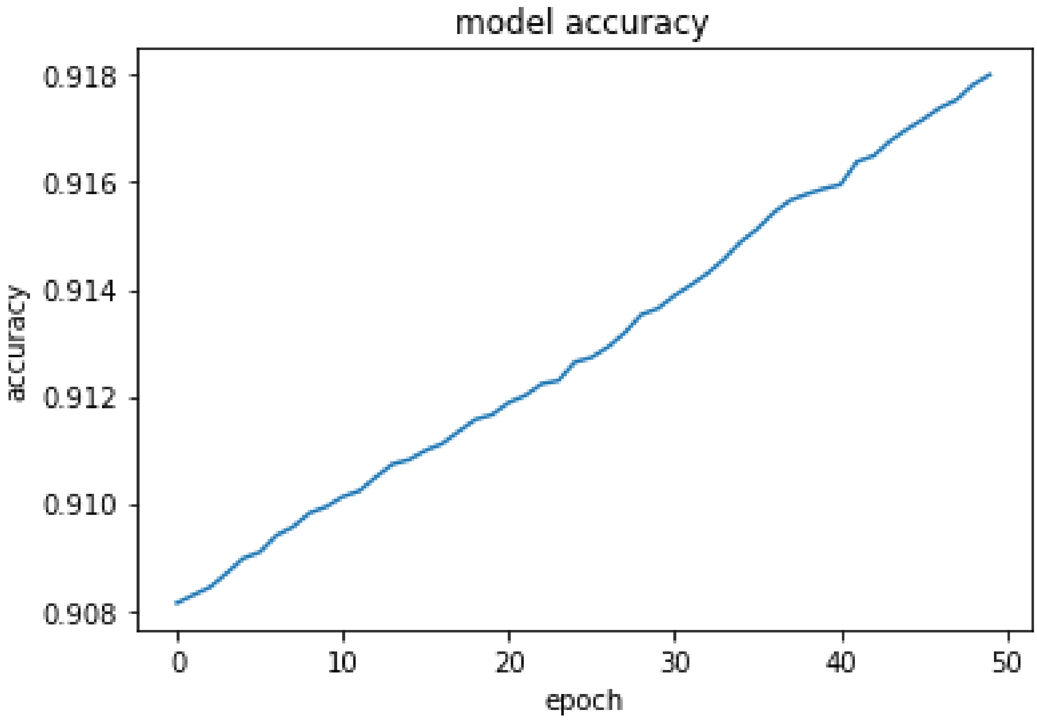
Figure 37: Training accuracy and loss curves for training vs validation data

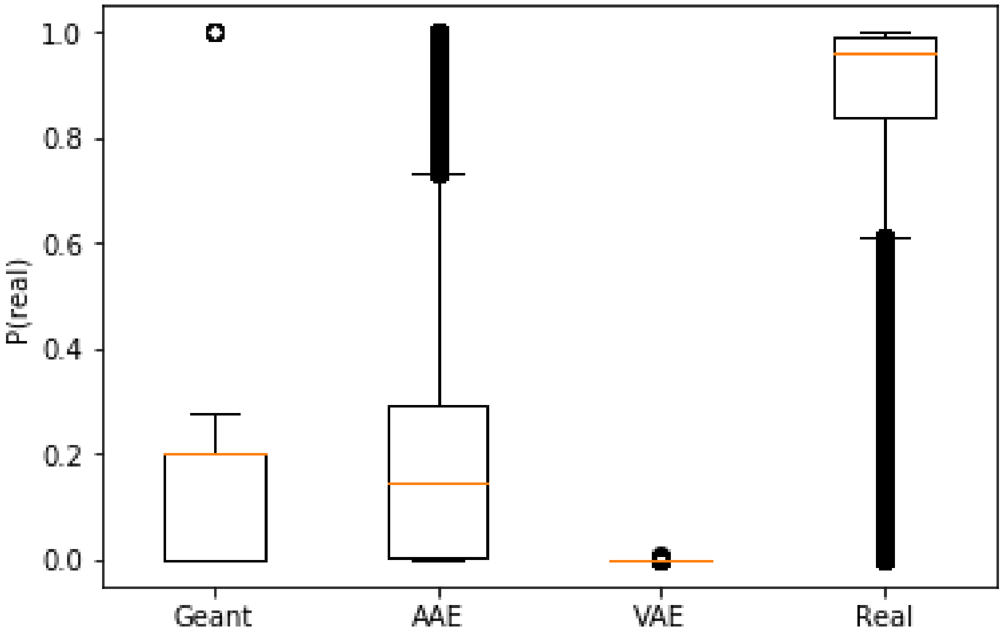
4.3.2 Generative Adversarial Networks

4.3.3 Adversarial Autoencoders

- $n_{latent} = 4$
- Discriminator optimizer: SGD with learning rate $\alpha = 0.00003$
- Generator optimizer: Adam with learning rate $\alpha = 0.00001$ and parameter $\beta_1 = 0.5$
- Label smoothing:
  - Positive labels: smoothed to be in the range 0.9-1.1
  - Negative labels: smoothed to be in the range 0-0.1
- Epochs = 400 000
- Batch size=32


0 2 4 6 8 10 12 14 16
0 5 10 15 20
0 2 4 6 8 10 12 14 16
0 5 10 15 20
0 2 4 6 8 10 12 14 16
0 5 10 15 20
0 2 4 6 8 10 12 14 16
0 5 10 15 20
0 2 4 6 8 10 12 14 16
0 5 10 15 20
0 2 4 6 8 10 12 14 16
0 5 10 15 20
0 2 4 6 8 10 12 14 16
0 5 10 15 20
0 2 4 6 8 10 12 14 16
0 5 10 15 20


model accuracy
accuracy
0.918
0.916
0.914
0.912
0.910
0.908
0
10
20
30
40
50
epoch


P(real)
1.0
0.8
0.6
0.4
0.2
0.0
Geant
AAE
VAE
Real

5 DISCUSSION AND CONCLUSIONS

5.1 Discussion

5.2 Conclusions

6 BIBLIOGRAPHY

[1] M. Thomson, “Modern Particle Physics,” 2013.

[2] J. Rafelski, “Connecting QGP-Heavy Ion Physics to the Early Universe,” in *Nuclear Physics B Proceedings Supplement*, 2013.

[3] C. G. Viljoen. [Online]. Available: https://www.draw.io/?lightbox=1&highlight=0000ff&edit=_blank&layers=1&nav=1#G1X-ZGzxO_b4zo_74rY9Z9zaikz-Il6R8o.

[4] H. Satz, “The Quark-Gluon Plasma∗ A Short Introduction,” in *6th International Conference on Physics and Astrophysics of Quark Gluon Plasma*, 2011.

[5] “QCD Phase Diagram SVG,” [Online]. Available: https://commons.wikimedia.org/wiki/File:QCDphasediagram.svg. [Accessed 18 2 2019].

[6] “Week 3: Thermal History of the Universe,” [Online]. Available: www.astro.caltech.edu/~george/ay127/kamionkowski-earlyuniverse-notes.pdf. [Accessed 20 February 2019].

[7] The Steven Hawking Center for Theoretical Cosmology, [Online]. Available: http://www.ctc.cam.ac.uk/images/contentpics/outreach/cp_universe_chronology_large.jpg.

[8] CERN, “About CERN: Who we are: Our History,” CERN, [Online]. Available: https://home.cern/about/who-we-are/our-history. [Accessed 26 January 2019].

[9] CERN, “CERN: Who We Are: Our Governance: Member States,” [Online]. Available: https://home.cern/about/who-we-are/our-governance/member-states. [Accessed 26 January 2019].

[10] CERN, “CERN: About: Who We Are: Our Mission,” [Online]. Available: https://home.cern/about/who-we-are/our-mission. [Accessed 26 January 2019].

[11] CERN, “CERN Resources: FAQs: Facts and Figures About the LHC,” [Online]. Available: https://home.cern/resources/faqs/facts-and-figures-about-lhc. [Accessed 06 01 2019].

[12] S. Chardley, “LHC Does a Dry-Run,” 20 March 2015. [Online]. Available: https://www.symmetrymagazine.org/article/march-2015/the-lhc-does-a-dry-run. [Accessed 26 January 2019].

[13] Taking a Closer Look at the LHC, “The LHC Proton Source,” [Online]. Available: https://www.lhc-closer.es/taking_a_closer_look_at_lhc/0.proton_source. [Accessed 27 January 2019].

[14] CERN, “LHC: The Guide,” [Online]. Available: https://home.cern/resources/brochure/cern/lhc-guide. [Accessed 2017 January 2019].

[15] CERN, “The CERN Accelerator Complex,” [Online]. Available: https://cds.cern.ch/record/2636343/files/CCC-v2018-print-v2.jpg?subformat=icon-1440. [Accessed 26 January 2019].

[16] CERN, “LHC Experiments,” [Online]. Available: https://home.cern/science/experiments. [Accessed 21 February 2019].

[17] CERN, “ATLAS Experiment,” [Online]. Available: https://home.cern/science/experiments/atlas. [Accessed 21 February 2019].

[18] CERN, “ALICE Experiment,” [Online]. Available: https://home.cern/science/experiments/alice. [Accessed 21 Fenruary 2019].

[19] CERN, “LHCb Experiment,” [Online]. Available: https://home.cern/science/experiments/lhcb. [Accessed 21 February 2019].

[20] ALICE, “ALICE Homepage,” [Online]. Available: http://alice.web.cern.ch/. [Accessed 21 February 2019].

[21] CERN, [Online]. Available: https://cds.cern.ch/record/2302924. [Accessed 21 February 2019].

[22] The ALICE Collaboration, The ALICE Experiment at the CERN LHC, INSTITUTE OF PHYSICS PUBLISHING AND SISSA, 2008.

[23] The ALICE Collaboration, The Technical Design Report of the Transition Radiation Detector, Geneva: CERN, 2001.

[24] Y. Pachmayer, “Particle Identification with the ALICE Transition Radiation Detector,” 2014.

[25] Particle Data Group, The Review of Particle Physics, 2018.

[26] ALICE Collaboration, The ALICE Transition Radiation Detector: construction, operation, and performance, CERN, 2017.

[27] CERN, “ROOT Data Analysis Framework: User's Guide,” May 2018. [Online]. Available: https://root.cern.ch/root/htmldoc/guides/users-guide/ROOTUsersGuideA4.pdf .

[28] “ROOT 5 Reference Guide,” [Online]. Available: https://root.cern/root/html534/ClassIndex.html.

[29] “ROOT 6 Reference Guide,” [Online]. Available: https://root.cern/doc/v616/.

[30] ALICE Collaboration (CERN), [Online]. Available: https://alice-doc.github.io/alice-analysis-tutorial. [Accessed 18 2 2019].

[31] CERN, “High Energy Physics Simulations,” [Online]. Available: http://lhcathome.web.cern.ch/projects/test4theory/high-energy-physics-simulations. [Accessed 26 July 2019].

[32] S. Agostinelli, J. Allison, J. Apostolakis and P. Arce, “Geant4 - a simulation toolkit,” *Nuclear Instruments and Methods in Physics Research,* vol. A 506, pp. 250-303, 2003.

[33] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, Cambridge, Masachusetts: The MIT Press, 2016.

[34] F. Rosenblatt, “The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain,” *Psychological Review,* vol. 65, no. 6, 1958.

[35] keras.io, “Available Activations,” [Online]. Available: https://keras.io/activations/#available-activations. [Accessed 19 July 2019].

[36] G. Cowan, Statistical Data Analysis, Oxford: Oxford University Press, 1998.

[37] T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, “Focal Loss for Dense Object Detection,” *Computer Vision and Pattern Recognition,* 2018.

[38] U. Griffo, “Umberto Griffo - Focal Loss Keras,” [Online]. Available: https://github.com/umbertogriffo/focal-loss-keras.

[39] C. Viljoen, “Psyche Shaman - Custom Focal Loss Keras R,” [Online]. Available: https://gist.github.com/PsycheShaman/ea39081d9f549ac410a3a8ea942a072b.

[40] C. Doersch, “Tutorial on Variational Autoencoders,” ResearchGate, 2016.

[41] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, “Generative Adversarial Nets,” 2014.

[42] [Online]. Available: https://commons.wikimedia.org/wiki/File:Standard_Model_Feynman_Diagram_Vertices.png. [Accessed 2 March 2019].

[43] CERN, “Grafana IT Overview,” [Online]. Available: http://monit-grafana-open.cern.ch/d/000000884/it-overview?orgId=16. [Accessed 26 January 2019].

[44] University of California Davis, “RHIC,” [Online]. Available: http://nuclear.ucdavis.edu/~rpicha/rhic.html. [Accessed 27 01 2019].

[45] Encyclopedia Britannica, “Tevatron Particle Accelerator,” [Online]. Available: https://www.britannica.com/technology/Tevatron. [Accessed 27 January 2019].

[46] M. Masuzawa, H. Koiso, K. Oide, R. Sugahara and et al., “CIRCUMFERENCE VARIATIONS OBSERVED AT KEKB,” in *Proceedings of the 7th International Workshop on Accelerator Alignment*, 2002.

[47] S.-I. Kurokawa and S. L. Olsen, “The KEK B-Factory Experiment,” [Online]. Available: www.slac.stanford.edu/pubs/beamline/29/2/29-2-kurokawa.pdf. [Accessed 27 January 2019].

[48] R. Field, “PHY2061 - Enriched Physics 2 - Relativity 4,” [Online]. Available: http://www.phys.ufl.edu/~acosta/phy2061/lectures/Relativity4.pdf. [Accessed 27 January 2019].

[49] CERN, “LHCf Experiment,” [Online]. Available: https://home.cern/science/experiments/lhcf. [Accessed 21 Febraury 2019].

[50] CERN, “MoEDAL Experiment,” [Online]. Available: https://home.cern/science/experiments/moedal. [Accessed 21 Febrary 2019].

[51] [Online]. Available: https://root.cern.ch/.

[52] CERN, “Physics Vectors,” [Online]. Available: https://root.cern.ch/root/htmldoc/guides/users-guide/PhysicsVectors.html. [Accessed 23 February 2019].

[53] “Spherical Coordinates,” [Online]. Available: https://mathinsight.org/spherical_coordinates. [Accessed 23 February 2019].

[54] C. Viljoen. [Online]. Available: https://www.draw.io/?lightbox=1&highlight=0000ff&edit=_blank&layers=1&nav=1#G1Zbad00aGA6OXlYpDzl8ggBK2kRM08xzO.

[55] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation,* vol. 9, no. 8, pp. 1735-1780, 1997.

[56] A. Odena, C. Olah and J. Shlens, “Conditional Image Synthesis with Auxiliary Classifier GANs,” in *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[57] A. Makhzani, I. Goodfellow, B. Frey, J. Shlens and N. Jaitly, “Adversarial Autoencoders,” 2016.

[58] J. Donahue, T. Darrell and P. Krahenbuhl, “Adversarial Feature Learning,” 2017.

[59] L. Metz, A. Radford and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Genarative Adversarial Networks,” in *ICLR 2016*, 2016.

[60] X. Mao, Q. Li, Xi, Haoran, R. Lao, Z. Wang and S. P. Smolley, “Least Squares Generative Adversarial Networks,” 2017.

[61] S. Weinberg, The First Three Minutes, Cambridge, Masachusettes: Fontana Paperbacks, 1976.

[62] D. Robinson. [Online]. Available: http://donrmath.net/CalcIII/unit1/lesson7/u1l7.html. [Accessed 23 February 2019].

[63] “NN SVG,” [Online]. Available: http://alexlenail.me/NN-SVG/AlexNet.html. [Accessed 21 April 2019].

[64] [Online]. Available: h2o.ai.

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.

Computations were performed using facilities provided by the University of Cape Town's ICTS High Performance Computing team: hpc.uct.ac.za

Travel to CERN was paid for by iThemba Labs via the SA-CERN agreement