Application of Machine Learning for Energy Reconstruction in the ATLAS Liquid Argon Calorimeter

by

Lucas A. Polson
B.Sc., University of Victoria, 2019

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Physics and Astronomy

Application of Machine Learning for Energy Reconstruction
in the ATLAS Liquid Argon Calorimeter

by

Lucas A. Polson
B.Sc., University of Victoria, 2019

Supervisory Committee

Dr. M. Lefebvre, Supervisor
(Department of Physics and Astronomy)

Dr. R. Kowalewski, Departmental Member
(Department of Physics and Astronomy)

Supervisory Committee

Dr. M. Lefebvre, Supervisor
(Department of Physics and Astronomy)

Dr. R. Kowalewski, Departmental Member
(Department of Physics and Astronomy)

ABSTRACT

The beam intensity of the Large Hadron Collider will be significantly increased during the Phase-II long shut down of 2024-2026. Signal processing techniques that are used to extract the energy of detected particles in ATLAS will suffer a significant loss in performance under these conditions. This study compares the presently used optimal filter technique to alternative machine learning algorithms for signal processing. The machine learning algorithms are shown to outperform the optimal filter in many relevant metrics for energy extraction. This thesis also explores the implementation of machine learning algorithms on ATLAS hardware.

Contents

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

Michel Lefebvre For the countless treasured and invaluable hours of instruction, discussion, and mentorship as a supervisor and friend.

The ATLAS collaboration and in particular those who provided careful, thoughtful, and patient assistance during various email conversations.

My family and friends For their infinite support and love throughout my journey thus far.

Chapter 1

Introduction

The purpose of this thesis is to assess the strengths and weaknesses of machine learning as a signal processing tool for the measurement of energy in the ATLAS liquid argon calorimeter. In particular, machine learning will be compared to traditional techniques currently implemented on the detector. Future upgrades to the Large Hadron Collider (LHC) will significantly increase the rate of proton-proton collisions, up to three times more than the maximum instantaneous luminosity achieved so far, and this will strain the current ATLAS detector readout. The current signal processing tools used for energy reconstruction are shown to suffer a performance loss under such conditions. Therefore, it is crucial to examine alternative techniques for signal processing in the future conditions of the LHC.

An overview of the LHC and ATLAS is given in Chapter 2. The hadronic endcap liquid argon calorimeter (HEC) subsystem of ATLAS, which constitutes the primary study of this thesis, is introduced. In Chapter 3, a derivation and analysis of the presently used optimal filter signal processing technique is examined. In particular, its loss of performance in expected future LHC conditions is shown. Chapter 4 focuses on machine learning techniques for energy reconstruction. Models are trained using a novel loss function developed in this thesis and outperform the optimal filter in many areas of the detector. Finally, in Chapter 5, implementation of machine learning techniques on detector electronics is analyzed.

Chapter 2

The ATLAS Experiment

2.1 The Large Hadron Collider

The Large Hadron Collider (LHC) [1] is the largest and most powerful particle accelerator in the world. It consists of a 27-kilometre ring of superconducting magnets with various accelerating structures. Its main purpose is to produce energetic proton-proton collisions; this is accomplished by counter-rotating two beams of protons and steering them to collide at designated interaction points.

The LHC is the final stage of the CERN accelerator complex, shown in Figure 2.1. Before reaching the LHC, protons are first accelerated by the LINAC2, BOOSTER, Proton Synchrotron, and the Super Proton Synchroton respectively; at this stage, the protons have an energy of 0.45 TeV. The protons are subsequently injected into the LHC and accelerated such that two counter-rotating beams have an energy of 6.5 TeV each. This corresponds to a center-of-mass energy of 13 TeV. Furthermore, the LHC is equipped with eight 400 MHz radio-frequency cavities which keep protons constrained to packets or "bunches". Each bunch can contain up to 10^{11} protons and counter-rotating packets collide at a rate of 40 million times per second, or every 25 ns. After a certain amount of time, the proton beams are expelled from the LHC and the process of injecting, accelerating, colliding, and expelling new protons is repeated.

The beams collide at four different points around the LHC ring corresponding to four different experiments. Two experiments, ATLAS [2] and CMS [3], discovered the Higgs boson and are now primarily used to make precise measurements of Standard

Model parameters and to search for physics beyond the Standard Model. The ALICE experiment [4] examines lead-lead nuclei collisions to study quark-gluon plasma and the LHCb experiment [5] seeks to measure charge parity violation in the production and interaction of bottom quarks.

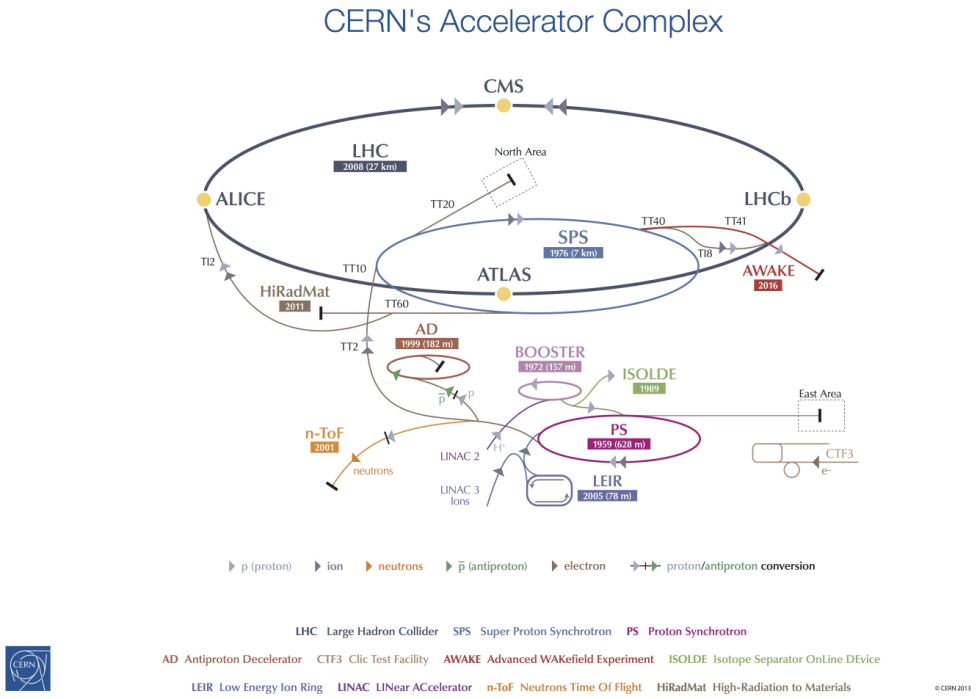


Figure 2.1: The CERN accelerator complex [6].

The intensity of the LHC beam is quantified by the instantaneous luminosity [7], given by equation 2.1.

$$\mathcal{L} = \frac{f N_1 N_2}{4\pi\sigma_x\sigma_y} \tag{2.1}$$

where $f = 40$ MHz is the frequency at which proton bunches cross, N_1 and N_2 are the number of protons in the colliding bunches, and σ_x and σ_y are the x and y components of the RMS of the spacial distribution of particle bunches. Another quantity of interest is the expected number of interactions per bunch crossing, μ, given by:

$$\mu = \sigma_{tot} \cdot t_{bc} \cdot \mathcal{L} \tag{2.2}$$

where σ_{tot} is the total proton-proton cross section of interaction and $t_{bc} = 1/f = 25$ ns is the time between bunch crossings, referred to in this thesis as a bunch crossing. The time between proton acceleration and the dumping of proton beams is known as a fill; during this period, $\mathcal{L}$ and μ generally decrease. The number of collisions per bunch crossing is distributed according to a Poisson distribution with mean μ. Figure 2.2 shows the peak luminosity in the LHC on particular days in 2017.

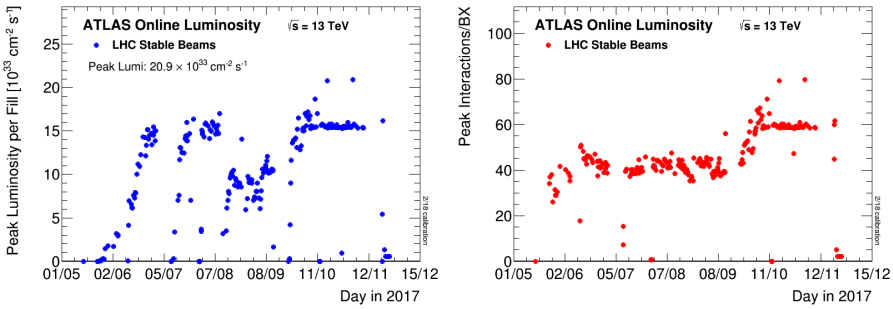


Figure 2.2: Peak Lumionisty $\mathcal{L}$ (left) and peak interactions per bunch crossing μ (right) on particular days in 2017 [8].

The LHC is periodically upgraded to increase the beam luminosity; a higher luminosity yields an increased rate of particle detection. The high-luminosity LHC [9], expected to be in operation by the end of 2027, may have a peak luminosity of $\mathcal{L} = 7.5 \times 10^{34}$ cm^{-2}s^{-1}: greater than three times the maximum luminosity seen in Figure 2.2. The ATLAS trigger system, used for classifying and saving important events, needs to be upgraded correspondingly as LHC luminosity increases.

2.2 The ATLAS Detector

A representation of the ATLAS detector is shown in Figure 2.3.

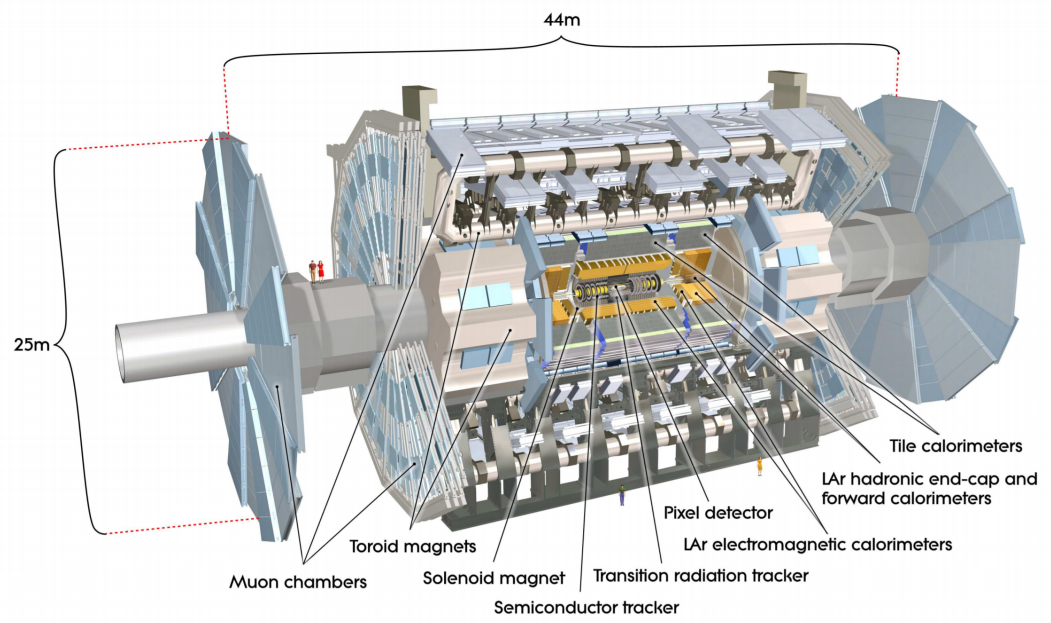


Figure 2.3: Schematic of the ATLAS detector [10]

The detector was built by the ATLAS collaboration, formed in 1992. Construction of detector components began at various institutions around the world; these parts were later shipped to Geneva for the final build. The detector was assembled in 2008; after collecting data during 2010-2012, the collaboration succeeded in measuring signatures of the Higgs Boson particle [11], the only particle in the standard model that had not yet been observed by previous experiments.

The right-handed coordinate system of ATLAS is defined as follows: $\hat{x}$ points towards the center of the LHC ring and $\hat{y}$ points upwards, resulting in $\hat{z}$ along the beam pipe. The azimuthal angle ϕ goes from $-\pi$ to π and is measured relative to $\hat{x}$ in the xy-plane. The polar angle θ ranges from 0 to π and is measured relative to the $\hat{z}$ direction. The polar angle is used to construct a quantity known as the pseudorapidity η, the primary angular measurement in ATLAS:

$$\eta = -\ln\left[\tan\left(\frac{\theta}{2}\right)\right] = \frac{1}{2}\ln\left(\frac{|\boldsymbol{p}| + p_z}{|\boldsymbol{p}| - p_z}\right) \tag{2.3}$$

where p_z is the z-component of some 3-momentum $\boldsymbol{p}$. A diagram comparing η and θ is shown in Figure 2.4. Highly relativistic particles (i.e. particles moving close to the speed of light) have energy $E \approx |\boldsymbol{p}|$. Differences in pseudorapidity $\Delta\eta$ are invariant

under Lorentz boosts in the z direction. η is often referred to instead of θ because

1. Particle production $dN/d\eta$ is approximately constant as a function of η; this can be used to determine optimal positioning and sizing for individual detector cells. For a given $\Delta\eta$ cell size, cells at smaller η (perpendicular to the beam axis) will be large compared to cells at larger η so that they both detect a similar rate of particles.

2. Measurements of pseudorapidity differences $\Delta\eta$ between two detected particles is independent of the longitudinal ($\hat{z}$) Lorentz boost of the parton collision that produced them. This is especially useful in the LHC where colliding partons have a priori unknown longitudinal momenta.

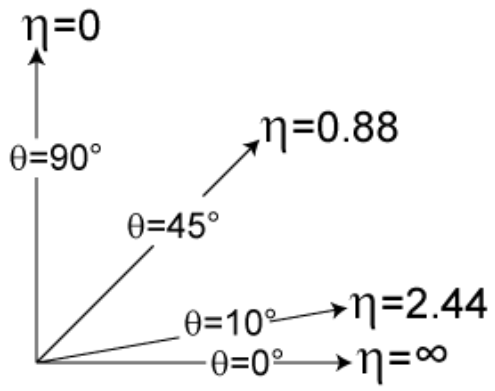


Figure 2.4: Pseudorapidity η shown on a polar plot.

The LAr calorimeter [12] shown in Figure 2.5 is a subdetector of ATLAS and consists of the LAr electromagnetic calorimeter, the LAr hadronic end cap calorimeter, and the LAr forward calorimeter. Each component requires a cryostat maintained at approximately -183 °C to sustain argon in liquid form. The electromagnetic calorimeter is specifically designed to measure the energy of $e^{\pm}$ and photons by containing and sampling produced electromagnetic showers and covers the range $|\eta| < 3.2$. The hadronic end cap calorimeter (HEC) is designed to measure the energy of hadrons by sampling the produced hadronic and electromagnetic showers, and covers $|\eta| > 1.5$. The forward calorimeter is designed to measure the energy of both electromagnetic and hadronic showers and covers up to $|\eta| = 4.9$.

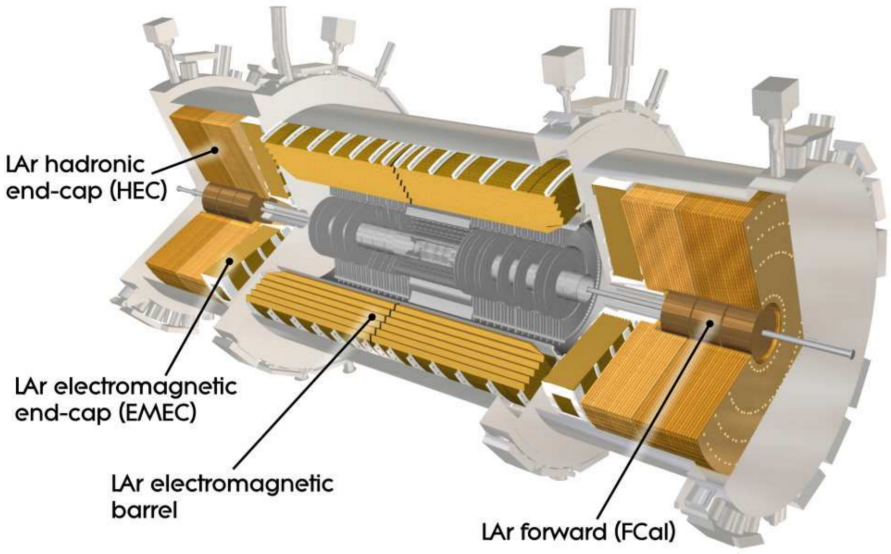


Figure 2.5: The different subsystems of the LAr calorimeter [12]

Outside of the LAr calorimeter is the scintillator tile calormeter [13], also used for hadronic calorimetry. Scintillators are materials that emit photons when struck by ionizing particles. They provide a suitable means for particle detection when combined with photomultipliers: devices that output a current proportional to the number of detected photons. Together, the LAr calorimeter and the scintillator tile calorimeter are referred to as the calorimeter. The calorimeter is surrounded by the muon spectrometer [14], where muons are bent by a toroidal magnetic field and their momenta are measured using three layers of high-precision tracking chambers.

This thesis primarily studies the HEC subsystem of the LAr calorimeter. The HEC consists of many individual detector cells; a diagram is shown in Figure 2.6.

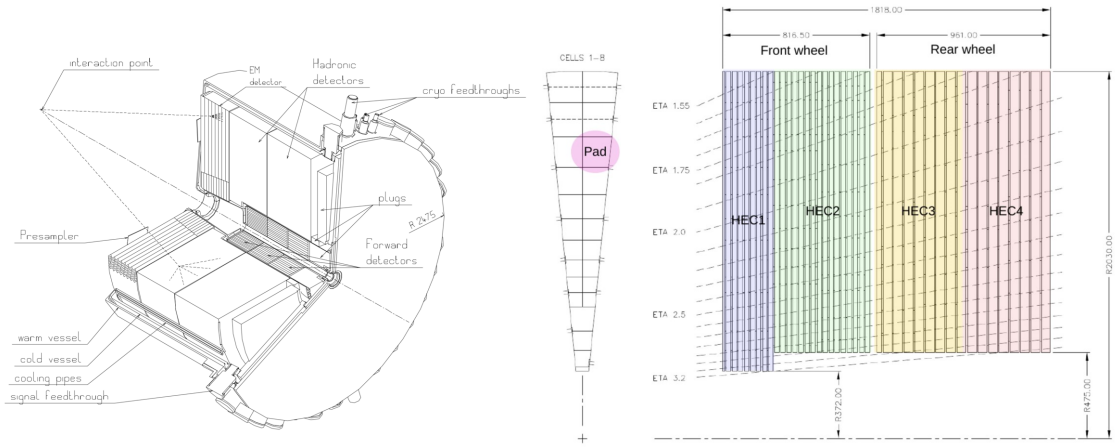


Figure 2.6: (Left) End-cap cryostat containing the EM, HEC, and Forward calorimeters. (Middle) $R - \phi$ view of one "slice" of cells in the HEC. (Right) $R - z$ view of the cells in the HEC; shown are all HEC subsystems. All measurements are in millimeters. [15]

The primary components used in HEC detection cells are grounded copper absorbers and liquid argon; the structure is shown in Figure 2.7. Particles impinging on the copper plates induce particle showers: a cascade of secondary particles as the result of a high-energy collision with dense matter. Charged particles from the showers ionize the LAr in the gaps between the copper plates. A high voltage is applied across the gap such that liberated electrons drift and produce a measureable current. To prevent a baseline shift, this primary current is passed through electronics that transform incident triangular pulses into bipolar pulse shapes known as ionization currents. The ionization currents are used to determine the energy of the original particle that interacted with the copper plate. This is further analyzed in Chapter 3.

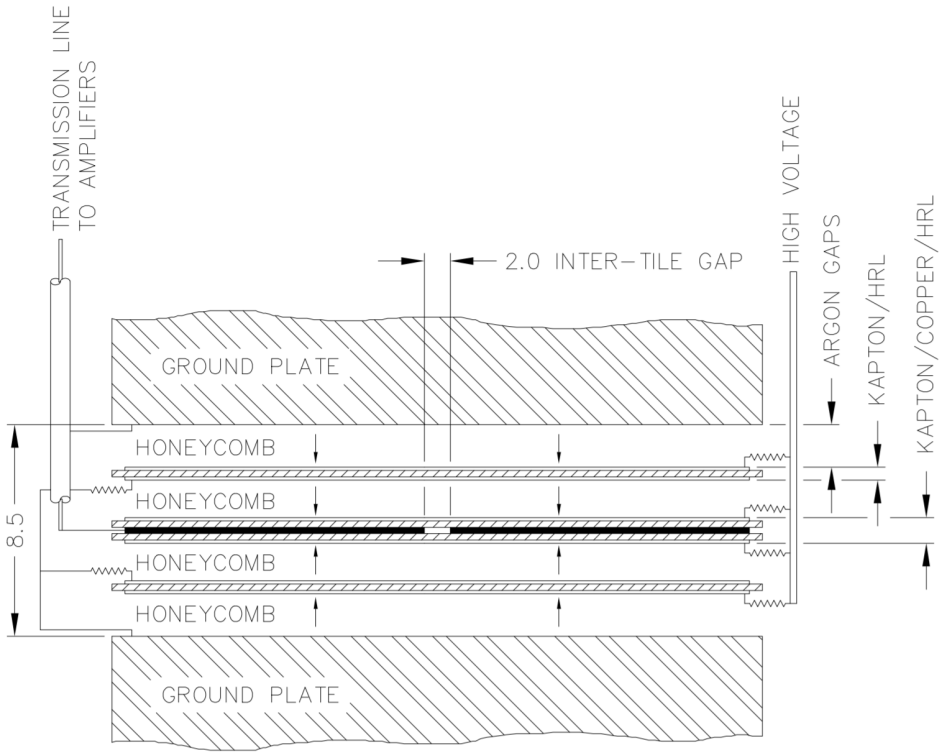


Figure 2.7: Structure of individual detector cells in the HEC calorimeter. Measurements in mm. The diagram is shown in the $z - \phi$ plane of ATLAS. [15]

There are two planned upgrades of the ATLAS detector [16]. The Phase-I upgrade is ongoing as of the writing of this thesis. The primary purpose of the Phase-I upgrade is to deploy a new digital trigger to handle the increased luminosity expected in the HL-LHC. The Phase-II upgrade is scheduled for deployment in 2024-2026. During this period, all readout electronics responsible for signal processing and online energy reconstruction will be replaced. The Phase-II upgrade may be a suitable time for transitioning to the machine learning techniques outlined in Chapter 4. This is discussed thoroughly in Chapter 5.

Chapter 3

Energy Reconstruction

For a given cell in the LAr calorimeter, energy is deposited as a function of time: this can be summarized in the notation $E(t)$. This energy is not directly measurable, but it can be converted into a measurable electrical signal time series $X(t)$ through an appropriate detection mechanism. Signal processing techniques are then used to obtain an estimate of $E(t)$, defined as $\hat{E}(t)$, given $X(t)$: this is known as energy reconstruction. The validity of the reconstruction can be tested through simulation. A time series $E(t)$ can be generated to simulate conditions in the detector. With a model of the detector electronics, $E(t)$ can be transformed into $X(t)$. The signal processing techniques used to obtain $\hat{E}(t)$ from $X(t)$ can then be tested, since $E(t)$ and $\hat{E}(t)$ can be compared. This chapter examines how $X(t)$ is generated from $E(t)$ in the LAr electronics chain, how $E(t)$ can be simulated, and how the optimal filter (OF) can be used to construct $\hat{E}(t)$ from $X(t)$.

3.1 Data Generation

This section outlines the physical process by which energy is measured in the actual experiment and how energy and current data can be simulated.

Given uniform ionization in the argon gap of Figure 2.7, the ionization current caused by an incident particle is the triangular pulse

$$I(t) = I_0 \left(1 - \frac{t}{\tau_{dr}}\right) \qquad 0 < t < \tau_{dr} \tag{3.1}$$

where $I_0 = Q_0/\tau_{\mathrm{dr}} = Ne/\tau_{\mathrm{dr}}$ [17]. N is number of free electrons, e is electron charge,

and $\tau_{\rm dr}$ is known as the drift time (with a typical average of 440 ns in the HEC). N can also be expressed as $N = Ef_{\rm samp}/W$ where E is total energy that the shower deposits in the absorber and active material, $f_{\rm samp}$ is the sampling fraction of the detector, and $W = 23.6$ eV is the ionization energy of LAr. In addition, the correction $Q_0 \rightarrow Q_0 \frac{\ln((1+C/U))}{C/U}$ is used to account for recombination of ions and electrons [18], where $C = 0.80$ kV/cm and $U \approx 10$ kV/cm. This results in a correction of 3.8% to Q_0. In summary

$$I_0(E) = \left(\frac{\ln((1 + C/U))}{C/U} \frac{f_{\rm samp} e}{W \tau_{\rm dr}} \right) E \equiv \gamma E \tag{3.2}$$

so current and energy are linearly proportional. In this thesis, for simplicity, the units $\gamma = 1$ are selected so that $I_0(E) = E$.

The derivative of the ionization current is

$$\frac{d}{dt} I(t) = E \left(\delta(t) - \frac{1}{\tau_{dr}} \right) \tag{3.3}$$

The current in equation 3.1 is passed through the HEC electronic chain for shaping; the electronic chain is shown in Figure 3.1. The corresponding electronic chain response to a step function for the HEC is given by equation 3.4, with D_k and τ_k related to the electronics parameters:

$$h(t) = \sum_{k=1}^{14} D_k e^{-t/\tau_k} \tag{3.4}$$

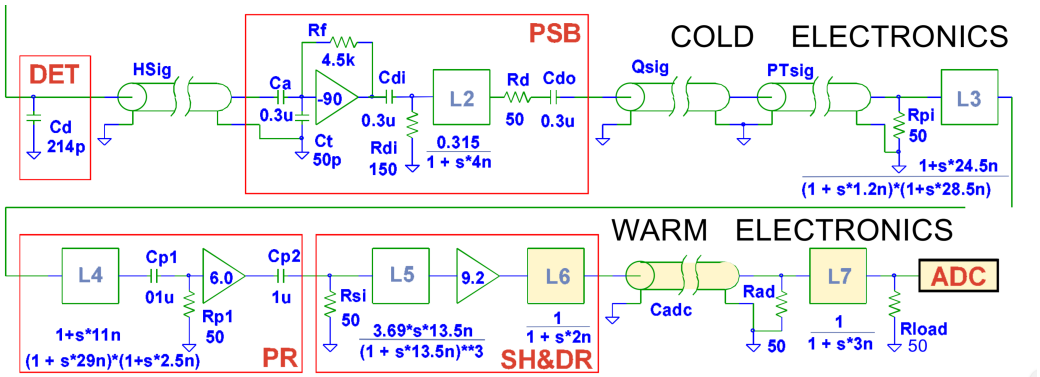


Figure 3.1: Schematic diagram of the HEC electronics chain. The corresponding electronic response to a step function is given by equation 3.4. [17]

The resulting current from an event with energy E deposited in a cell is the convolution of Equations 3.3 and 3.4. In addition, the pulse shape can be normalized as $D_k \to D'_k = D_k/\zeta$ where ζ is a constant such that the front lobe has a maximum value of E. This is what is known as the **ionization current**.

$$I_c(t;E) \equiv E \sum_{k=1}^{14} D'_k \left(\delta(t) - \frac{1}{\tau_{dr}}\right) \oplus e^{-t/\tau_k} \tag{3.5}$$

where $\oplus$ represents a convolution. It is also useful to define the energy-independent quantity known as the **normalized ionization current**; $g(t) = I_c(t;E)/E$:

$$g(t) \equiv \sum_{k=1}^{14} D'_k \left(\delta(t) - \frac{1}{\tau_{dr}}\right) \oplus e^{-t/\tau_k} \tag{3.6}$$

It follows that for any injected energy, $I_c(t;E) = Eg(t)$. A sample injected energy and normalized ionization current are shown in Figure 3.2.

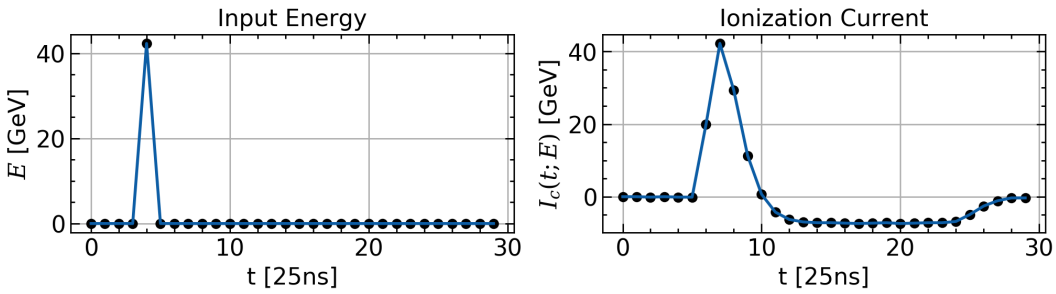


Figure 3.2: Ionization current (right plot; from equation 3.5) corresponding to a single injected energy event in a calorimeter cell (left plot). Blue lines are linear interpolation.

In the LAr calorimeter, the ionization current is measured and used to determine the energy. The optimal filter is one such technique presently used in ATLAS for energy reconstruction. A possible alternative technique discussed in Chapter 4 is the convolutional neural network.

Note that the ionization current in Figure 3.2 is the signal response from a single event with energy E deposited in a cell. In practice, energy is deposited every bunch crossing which can be represented by the discrete time series $E(t_i)$. The true measured signal is the sum of all these responses:

$$X(t_i) \equiv \sum_{j=-\infty}^{\infty} I_c(t_i - t_j; E(t_j)) \tag{3.7}$$

A sample energy time series $E(t_i)$ and the corresponding signal output of Equation 3.7 is shown in Figure 3.3.

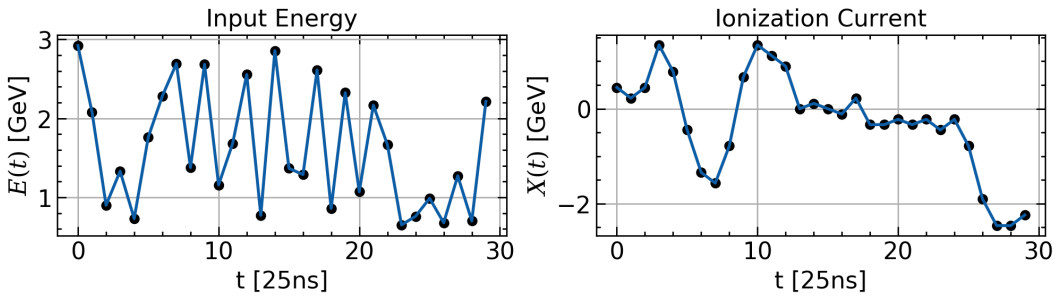


Figure 3.3: Full signal response from continuous energy in the calorimeter. Blue lines are linear interpolation.

Equation 3.7 provides the means to obtain $X(t)$ from $E(t)$. Discussed now is a procedure that can be used to simulate the time series $E(t)$. At a time t, the energy E is simulated using

$$E(t_i) = E^{(s)}(t_i) + E^{(n)}(t_i) \tag{3.8}$$

where $E^{(s)}(t_i)$ is the energy from a signal event at discrete time t_i and $E^{(n)}(t_i)$ is energy from a pileup event at time t_i. Signal events correspond to simulated physics events of interest, whereas pileup events correspond to background energy measurements inherent to a particular calorimeter cell. They are generated as follows:

1. $E^{(n)}(t_i)$: At each bunch crossing t_i, random energies $E_1 ... E_N$ are selected where N is Poisson distributed around μ. The total energy $E^{(n)}$ at the given bunch crossing is $\sum_j E_j$. Each E_j is distributed according to the probability density function (pdf) $f_E(E) = q\delta(E) + (1-q)p_k(E)$ where q is the probability that no energy is deposited, and $p_k(E)$ is a probability density function dependent on the region k in the detector. Figure 3.5 provides a sample.

2. $E^{(s)}(t_i)$: Each injected energy is distributed according to Figure 3.4 and the time between successive energy injections is a random variable distributed according to a uniform distribution between 30 and 50 bunch crossings. Figure 3.5 provides a sample.

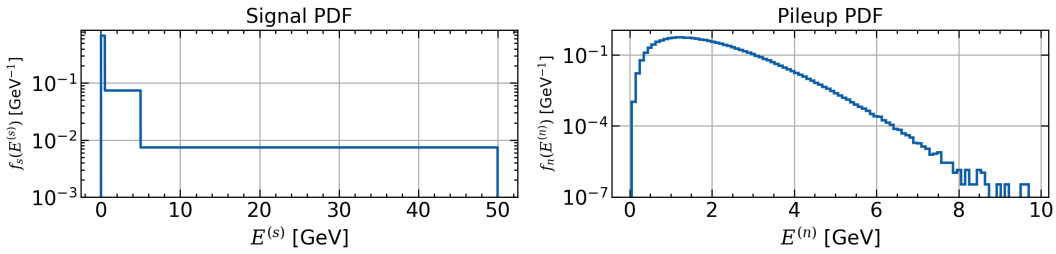


Figure 3.4: Probability density functions of $E^{(s)}$ and $E^{(n)}$ during bunch crossings corresponding to injection. $E^{(n)}$ is injected at every bunch crossing but $E^{(s)}$ is injected every 30 and 50 bunch crossings according to a uniform distribution.

Note that there will be a different ionization current from signal events and pileup events and these add linearly; while $E(t_i) \to X(t_i)$, it is also true that $E^{(s)}(t_i) \to S(t_i)$ and $E^{(n)}(t_i) \to n(t_i)$, where $X(t_i) = S(t_i) + n(t_i)$. This is true because the Laplace transform is a linear operator. This is summarized and displayed in Figure 3.5.

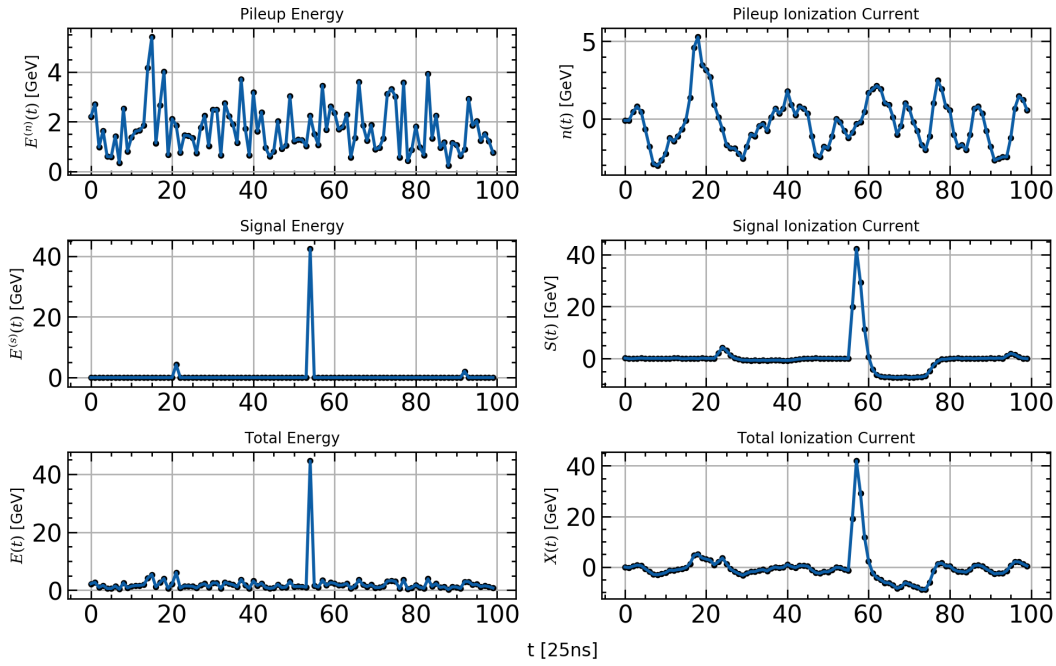


Figure 3.5: Signal, pileup, and total energy (left) and signal, pileup, and total ionization current (right). Blue lines represent linear interpolation.

Ionization currents can be obtained from $E(t_i)$ using the ATLAS Readout Elec-

tronics Upgrade Simulation (AREUS) software package [19]. AREUS is a fast, flexible, and fully-featured simulation tool of the full trigger-readout chain for the Phase I upgrade and beyond. It is also used to obtain the necessary parameters and predictions of the optimal filter estimator, discussed in section 3.2.

The generation of simulated data can be summarized as follows:

1. Generate $E^{(s)}(t_i)$ and $E^{(n)}(t_i)$ for 30 million consecutive time points where $\Delta t =$ 25 ns.

2. Pass $E^{(s)}(t_i)$ and $E^{(n)}(t_i)$ into AREUS, which computes $E(t_i)$ and the corresponding ionization current $X(t_i)$.

3.2 The Optimal Filter Technique

The optimal filter is an energy reconstruction technique whereby a filter is applied to the ionization current time series $X(t_i)$ to produce the energy time series $E(t_i)$. In this section, the process by which the filter coefficients are derived is discussed.

The following derivation is strongly based on the work of Cleland & Stern [20]. The notation $a_i \equiv a(t_i)$ is used here for all time series and similarly for other variables. Suppose that consecutive bipolar pulse shapes from signal events are non-overlapping. Mathematically, this means that signal events are spaced sufficiently far apart such that at any time, S_i depends only on a single energy deposition $E_j^{(s)}$ where $j = f(i) \leq i$. $f(i)$ should resemble a staircase. Expressed in terms of the normalized ionization current (Equation 3.6) and τ_j, a deviation from the assumed crossing time,

$$S_i = E_j^{(s)} g(t_{i-j} - \tau_j) \tag{3.9}$$

By Taylor expansion, assuming τ_j is small, the following is obtained:

$$X_i = E_j^{(s)} g(t_{i-j} - \tau_j) + n(t_i) = E_j^{(s)} g_{i-j} - E_j^{(s)} \tau_j g'_{i-j} + n_i \tag{3.10}$$

where the last part of the equation defines a simplified notation. The following quantities can then be defined:

$$u_j = \sum_{i=0}^{N} a_i X_{i+j} \qquad v_j = \sum_{i=0}^{N} b_i X_{i+j} \tag{3.11}$$

where N is the filter depth of the optimal filter. Requiring the expectation of u_j to be $E_j^{(s)}$ and the expectation of v_j to be $E_j^{(s)}\tau_j$ yields the following conditions

$$\begin{aligned} E_j^{(s)} &= \langle u_j \rangle = \textstyle\sum_i \left(E_j^{(s)} a_i g_i - E_j^{(s)} \tau_j a_i g_i' + a_i \langle n_{i+j} \rangle \right) \\ E_j^{(s)} \tau_j &= \langle v_j \rangle = \textstyle\sum_i \left(E_j^{(s)} b_i g_i - E_j^{(s)} \tau_j b_i g_i' + b_i \langle n_{i+j} \rangle \right) \end{aligned} \tag{3.12}$$

Furthermore, since $\langle n_i \rangle \approx 0$,

$$\begin{aligned} \textstyle\sum_i a_i g_i &= 1 \qquad & \textstyle\sum_i a_i g_i' &= 0 \\ \textstyle\sum_i b_i g_i &= 0 \qquad & \textstyle\sum_i b_i g_i' &= -1 \end{aligned} \tag{3.13}$$

The variances of u_j and v_j are given by

$$\begin{aligned} \mathrm{Var}(u_j) &= \textstyle\sum_{il} a_i a_l \langle n_i n_l \rangle \\ \mathrm{Var}(v_j) &= \textstyle\sum_{il} b_i b_l \langle n_i n_l \rangle \end{aligned} \tag{3.14}$$

The optimal values of a_i and b_i are found by minimizing Equation 3.14 while satisfying Equation 3.13. This problem, which can be solved using the method of Lagrange multipliers, requires knowledge of the noise autocorrelation function $\langle n_i n_j \rangle$. As such, preliminary data $n(t_i)$ are required to configure the filter.

Once optimized, the estimator for the injected energy of signals becomes

$$\hat{E}^{(s)}(t_j) = \sum_{i=0}^{N} a_i X_{i+j} \tag{3.15}$$

The goal of this thesis is to estimate E, which can be accomplished by adding $\left\langle \hat{E}^{(n)} \right\rangle$, producing

$$\boxed{\hat{E}(t_j) = \sum_{i=0}^{N} a_i X_{i+j} + \left\langle \hat{E}^{(n)} \right\rangle} \tag{3.16}$$

In Chapter 4, this formula will be compared to the estimator obtained through machine learning.

The following provides justification for why $\langle n_i \rangle \approx 0$. Pileup noise is generated at each bunch crossing and passed through the electronics. If $E^{(n)}(t_i)$ is the pileup at the i^{th} bunch crossing then the corresponding signal output $n(t_i)$ from the electronics is

$$n_i = \sum_{j=-\infty}^{j=\infty} E^{(n)}(t_i) g(t_i - t_j) \tag{3.17}$$

Noting $\left\langle E^{(n)}(t_i) \right\rangle = \left\langle E^{(n)} \right\rangle$ is constant, this results in the following

$$\begin{aligned} \langle n_i \rangle &= \sum_{j=-\infty}^{j=\infty} \left\langle E^{(n)}(t_i) \right\rangle g(t_i - t_j) \\ &= \left\langle E^{(n)} \right\rangle \sum_{j=-\infty}^{j=\infty} g(t_i - t_j) \\ &= \left\langle E^{(n)} \right\rangle \sum_{j=-\infty}^{j=\infty} g(t_j) \end{aligned}$$

In the continuous limit, the sum equals zero since the normalized ionization current is bipolar and has an area of zero.

3.3 Optimal Filter Predictions

Using Equation 3.16, the optimal filter produces estimates $\hat{E}$ of the true energy E at all times and for each cell in the calorimeter. This process can be simulated using AREUS, which is able to solve for the coefficients a_i and simulate energy predictions, provided:

1. N (the number of a_i coefficients) is specified. This is known as the filter depth of the optimal filter. The current filter depth used in ATLAS is $N = 5$.

2. The normalized ionization current g defined by Equation 3.6 is known. This can be obtained given the electronic transfer function.

3. The noise autocorrelation function $\langle n_i n_j \rangle$ is known; the number of time points used to estimate the autocorrelation from a time series n_t is defined as T.

4. Ionization current data, defined by Equation 3.10, are provided. In this study, for each separate configuration of η, μ, N, and T, 30 million bunch crossings of data are used to generate predictions.

With this information, the Lagrange multiplier problem defined by Equations 3.13 and 3.14 can be solved and energy predictions $\hat{E}$ can be made. Residuals are defined as $\hat{E} - E$ and their distribution can be examined to determine the strength of a model's predictability. This study only examines residuals when $E^{(s)} > 0$. Sample predictions are shown in Figure 3.6.

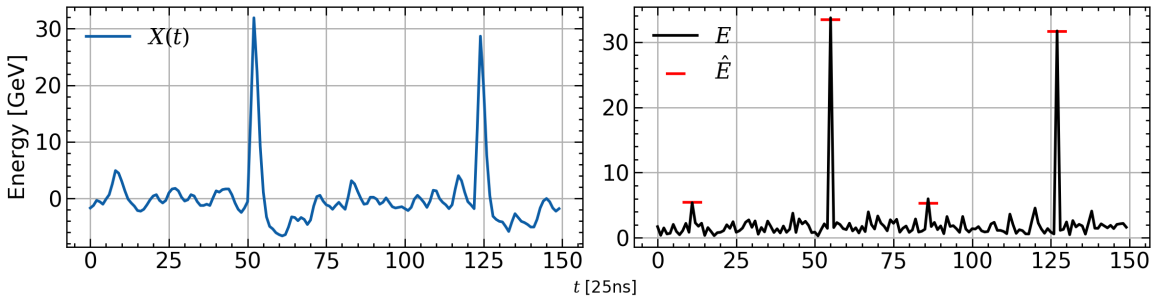


Figure 3.6: Left: Ionization current defined by Equation 3.10. Right: Input ionization time series defined by Equation 3.8 and predictions obtained by equation 3.16 at times where $E^{(s)} > 0$.

The root-mean-squared-error (RMSE), defined by Equation 3.18, is a descriptive statistic that measures the similarity between all predicted energies $\hat{E}$ and the corresponding actual energies E.

$$\text{RMSE}(\hat{E}, E) = \sqrt{\left\langle (\hat{E} - E)^2 \right\rangle} \tag{3.18}$$

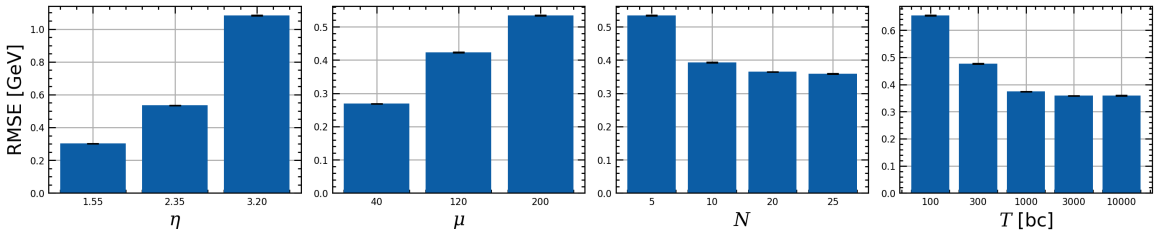
In Figure 3.7, the RMSE is shown:

1. as a function of η, for $\mu = 200$, $N = 5$, and $T = 3000$. The increase in the RMSE with η is due to the fact that cells at large η detect more background particles, and hence have more noise.

2. as a function of μ, for $N = 5$, $T = 3000$, and $\eta = 2.35$. The increase in the RMSE with μ is a direct consequence of the simulation of $E^{(n)}(t)$: a larger value of μ results in more pileup noise. This is further explored in Figure 3.8.

3. as a function of N, for $\mu = 200$, $T = 3000$, and $\eta = 2.35$. More optimal filter coefficients a_i enable greater optimization of Equation 3.14 and thus a smaller RMSE between predicted and simulated energy. The improvement in RMSE decreases as more coefficients are used; an increase in filter depth from $N = 5$ to $N = 10$ results in a significant reduction (percent difference $\approx 30\%$) but a much smaller reduction is seen in going from $N = 10$ to $N = 20$ (percent difference $\approx 7.3\%$).

4. as a function of T, for $\mu = 200$, $N = 25$, and $\eta = 2.35$. Optimization of Equation 3.14 requires precise knowledge of $\langle n_i n_j \rangle$; hence sufficient number of noise data n_t is required. However, once $T \approx 3000$, there is no statistically significant improvement in the RMSE by increasing T.



η	RMSE [GeV]
1.55	0.302 ± 0.001
2.35	0.534 ± 0.001
3.20	1.083 ± 0.003

μ	RMSE [GeV]
40	0.269 ± 0.001
120	0.423 ± 0.001
200	0.534 ± 0.001

N	RMSE [GeV]
5	0.534 ± 0.001
10	0.393 ± 0.001
20	0.365 ± 0.001
25	0.359 ± 0.001

T [bc]	RMSE [GeV]
100	0.655 ± 0.002
300	0.477 ± 0.001
1000	0.375 ± 0.001
3000	0.359 ± 0.001
10000	0.359 ± 0.001

Figure 3.7: RMSE of optimal filter predictions vs η, μ, N, and T.

It is also useful to examine the complete distributions of $\hat{E} - E$. Figure 3.8 shows the distribution of residuals $\hat{E} - E$ for $N = 5$, $T = 3000$, and $\eta = 2.35$ at different values of μ. The distributions, including the tails, are Gaussian.

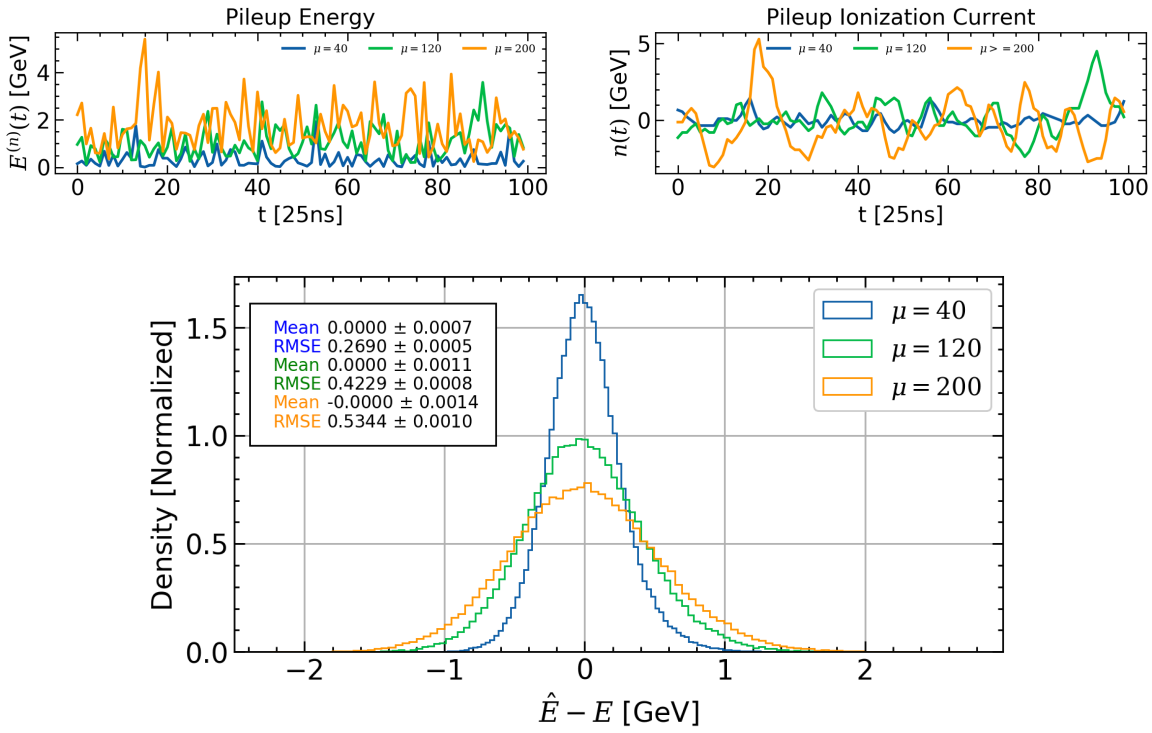


Figure 3.8: Top: Pileup energy and pileup ionization current for different values of $\langle\mu\rangle$ in the LHC. Bottom: Histograms of $\hat{E} - E$ for optimal filter when $E^{(s)} > 0$; mean and RMSE shown are in units of GeV.

While the RMSE of $\hat{E} - E$ increases as μ increases, the mean of the residuals always remains consistent with zero. This result, which follows directly from the condition in Equation 3.13, displays the effectiveness of the optimal filter as a non-biased estimator of energy predictions. Typically, however, it is not sufficient for a model to be non-biased across all predictions. A model should additionally make non-biased predictions within distinct energy regions. In Figure 3.9, residuals $\hat{E} - E$ are shown in 7 distinct subplots where each subplot corresponds to a specific range of $E^{(s)}$ values. As required, the optimal filter is able to make non-biased predictions across distinct energy intervals, even as μ increases.

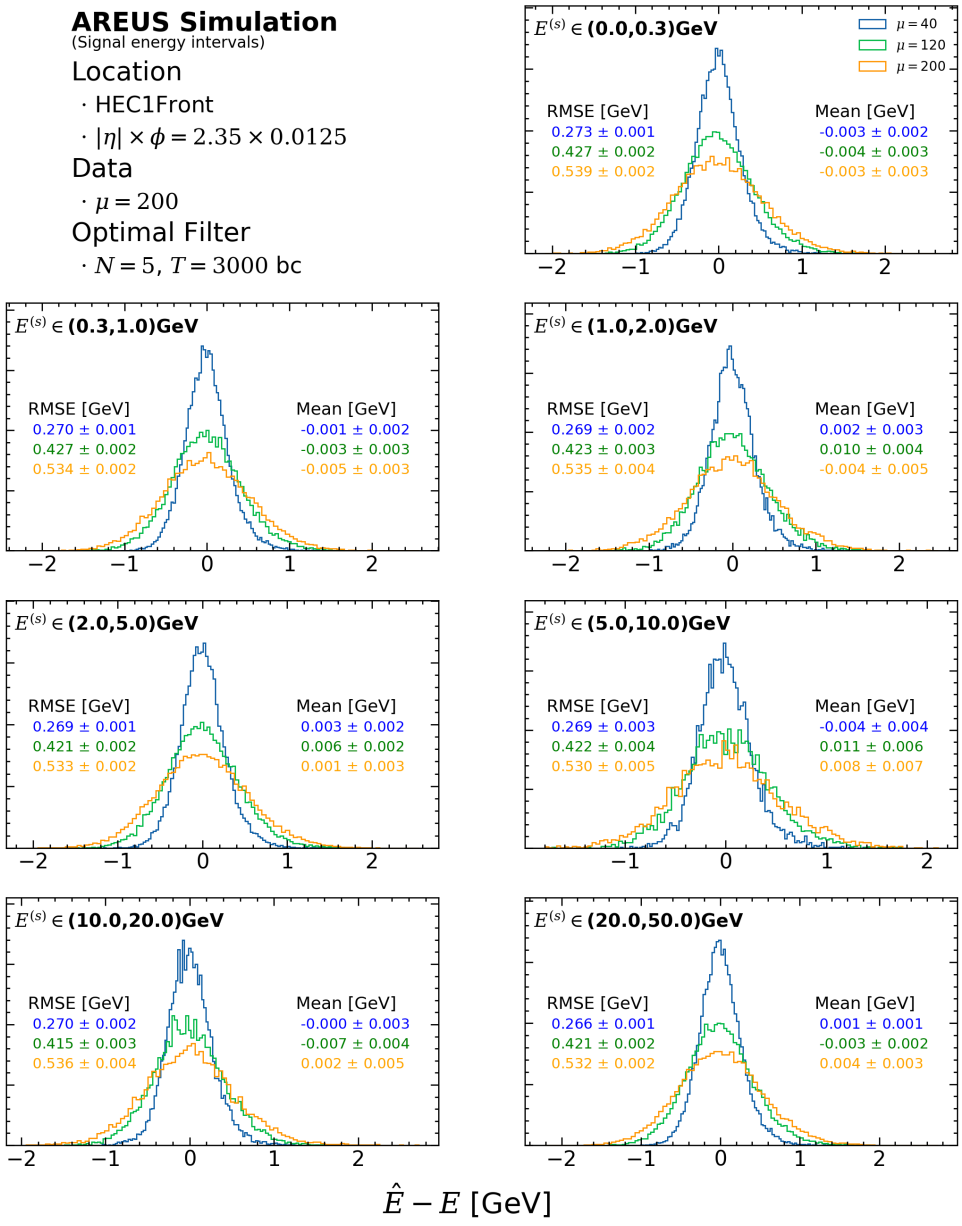


Figure 3.9: Predictions $\hat{E}$ vs. energies E; each subplot contains a histogram of residuals for different $E^{(s)}$ regions. Residuals are shown for the optimal filter trained on three different data sets: $\mu = 40$ (blue), $\mu = 120$ (green), and $\mu = 200$ (orange). The simulated data correspond to the cell located in the HEC1 subsystem at $\eta = 2.35$ and $\phi = 0.0125$ with $N = 5$ and $T = 3000$. Mean and RMSE shown on plot are in units of GeV.

A major deficiency of the optimal filter estimator is the significant increase in the

variance of residuals $\hat{E} - E$ as the luminosity of the LHC and, correspondingly, μ increase. One method of reducing the variance is by increasing the filter depth N, as shown in Figure 3.7. The purpose of the subsequent chapters of this thesis is to develop and examine an alternative estimator for E such that residuals $\hat{E} - E$ have a low RMSE at high pileup μ.

Chapter 4

Machine Learning and Energy Reconstruction

In this thesis, the convolutional neural network was the main machine learning tool explored for the purpose of energy reconstruction. This was primarily due to the model versatility, as described in the WaveNet paper [21], the ability to train many models relatively quickly given access to a powerful graphics card, and the ease of implementing the model architecture on future ATLAS hardware.

4.1 The Convolutional Neural Network

A 1 dimensional, causal, convolutional neural network (CNN) consists of a sequence of mappings

$$(X_t = x_{t0}^{(0)}) \to x_{tj}^{(1)} \to ... \to x_{tj}^{(L-1)} \to (x_{t0}^{(L)} = \hat{y}) \tag{4.1}$$

$$x_{tj}^{(i+1)} = \sum_{m=0}^{f^{(i)}-1} \sum_{n=0}^{T^{(i)}-1} R^{(i)} \left(A_{jmn}^{(i)} x_{n'm}^{(i)} + b_j^{(i)} \right) \tag{4.2}$$

where

1. i represents the layer index, and $f^{(i)}$ is the number of feature maps (or dimensionality) of the time series in layer i. The first and last layers have $f^{(i)} = 1$, for example, corresponding to a univariate time series. L is the number of layers in the network.

2. j, which can take on the indices 0 to $f^{(i)} - 1$ in layer i, represents the feature map (or filter) index. t represents the time index for the time series.

3. $A^{(i)}_{jmn}$ is the weight matrix and $b^{(i)}_j$ is the bias term for feature map j in layer i. These parameters are modified during the training procedure.

4. $n' = t - d^{(i)}n$ where $d^{(i)}$ is the dilation rate of layer i.

5. $T^{(i)}$ is the size of the filter in layer i. It is also typically referred to as the kernel size.

6. $R^{(i)}$ is the activation function for layer i of the neural network. This thesis uses the activation function $R^{(i)}(x) = 0$ if $x < 0$ and $R^{(i)}(x) = x$ if $x \geq 0$. This is known as the *relu* activation function.

For energy reconstruction applications, there is only one feature map in the first and final layer since the input time series is 1 dimensional and the output time series is 1 dimensional. The intermediate layers may have many feature maps. A diagram of a sample network is shown in Figure 4.1.

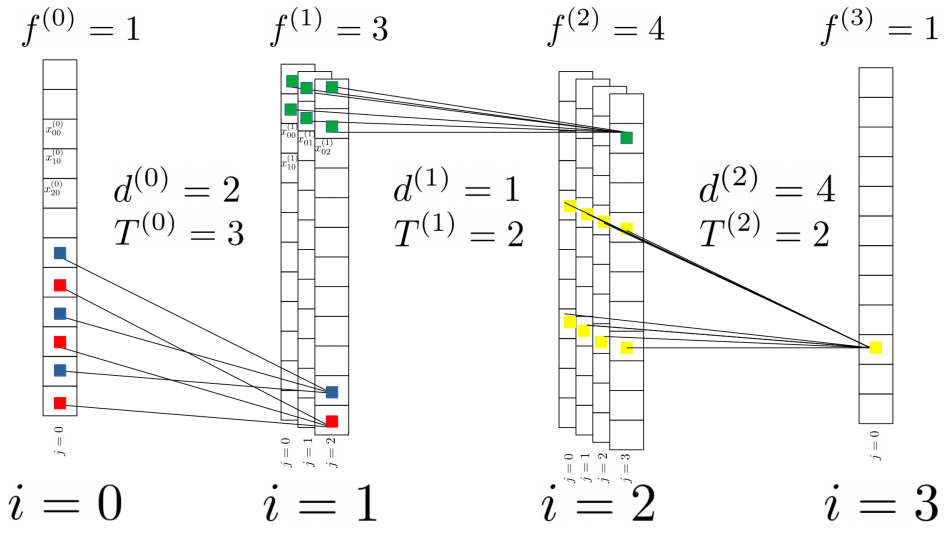


Figure 4.1: A visualization of a sample 1 dimensional, causal, convolutional neural network. The lines connecting the layers represent the transformation $R^{(i)}\left(A^{(i)}_{jmn}x^{(i)}_{n'm} + b^{(i)}_j\right)$. Each color corresponds to the input and output of a corresponding transformation.

A crucial feature of a causal neural network is that a prediction y_t depends only on points $X_{t'}$ where $t' \leq t$. It is possible, however, to break causality and let y_t depend on points $X_{t'}$ where $t' \leq t + T_0$ and T_0 is the number of future time points. This is done by realigning X_t and y_t such that X_t is shifted to the left by T_0 time points. In practice, this means that predictions are delayed: a prediction for y_t occurs precisely at $t + T_0$.

Another important quantity of the causal neural network is the filter depth: the number of points $X_{t'}$ used to make a prediction at y_t. This quantity is analogous to the filter depth N in equation 3.16. For a typical[1] causal neural network, this is given by

$$F_N = 1 + \sum_i (T^{(i)} - 1) \cdot d^{(i)} \tag{4.3}$$

The network can be concisely summarized in the following function:

$$\hat{y}_t = \chi(X_t; \boldsymbol{\theta}, \boldsymbol{H}) \tag{4.4}$$

where $\hat{y}_t$ is an estimator for y_t. $\boldsymbol{H}$ includes the values $f^{(i)}$, $T^{(i)}$, $R^{(i)}$, T_0, and the number of layers i. These are known as **hyperparameters** and they do not change when the neural network is trained. $\boldsymbol{\theta}$ includes $A^{(i)}_{jmn}$ and $b^{(i)}_j$: these are known as **parameters** or **weights**. The parameters are initially randomized using a scheme based on the activation function $R^{(i)}$ of the network [22]. Afterwards, in what is known as the training phase, some variation of Gradient Descent is used to update and optimize the network weights. Gradient Descent works as follows: first a loss function is defined, such as the mean squared error:

$$L_0(\boldsymbol{\theta}) = \frac{1}{N} \sum_{t=1}^{N} \left(\hat{y}_t(\boldsymbol{\theta}) - y_t\right)^2 \tag{4.5}$$

The gradient of this loss function with respect to $\boldsymbol{\theta}$ can be computed and an iterative procedure can be used to update the network weights to decrease the loss function:

[1] It is assumed here that a prediction y_t is made using a sequence of points $X_{t-a}, X_{t-a+1}, ...X_t$ for some integer a. While this is not necessarily satisfied for certain choices of $T^{(i)}$ and $d^{(i)}$, these cases are rare and unpractical. For example, a single layered network with $T^{(1)} = 2$ and $d^{(1)} = 2$ will have predictions y_t made using X_t and X_{t-2}; this sequence does not include X_{t-1} and thus Equation 4.3 does not apply.

$$\boldsymbol{\theta} \to \boldsymbol{\theta} - \beta \nabla_{\boldsymbol{\theta}} L_0(\boldsymbol{\theta}) \tag{4.6}$$

β is known as the learning rate. After many iterations a local minima of $L(\boldsymbol{\theta})$ can be reached in $\boldsymbol{\theta}$ space. A visual example of gradient descent where $\boldsymbol{\theta}$ is two-dimensional is shown in Figure 4.2. Models in this paper are trained using Nadam optimization [23], a slight modification of traditional Gradient Descent.

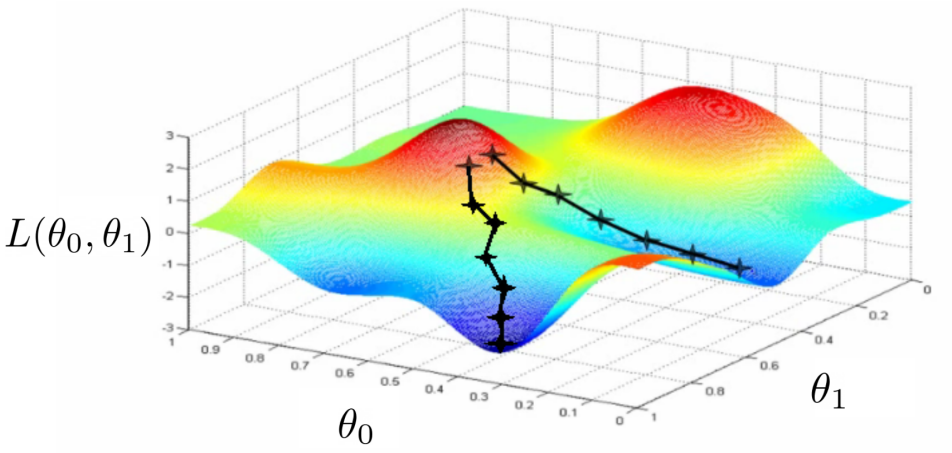


Figure 4.2: Visualization of gradient descent where $\boldsymbol{\theta}$ is two dimensional. The two initial points on the red peak, although close together, eventually reach different minima.

In a traditional machine learning procedure, the data $\{X_t, y_t\}$ is split into a training and a test set, where the test set usually includes 20-30% of the data. The model is trained using the training set, and the predictions are evaluated on the test set. To properly assess over-fitting in machine learning without utilizing the test set, the entire training set is further separated into a training and validation set. Gradient descent only uses the training data set; afterwards, the loss function computed using the training and validation data is compared. This procedure is summarized in Figure 4.3. Over-fitting occurs when the loss function evaluated on the training set at the end of training is much smaller than on the validation data set.

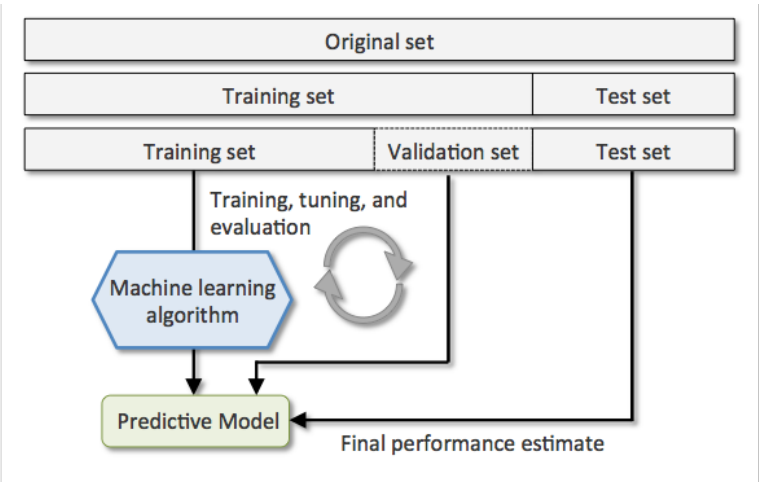


Figure 4.3: A traditional machine learning procedure.

Two more important machine learning terms are **batches** and **epochs**. For large data sets, it is atypical to use the entire training set $\{X_t, y_t\}$ to compute a loss function (such as Equation 4.5) and perform a gradient descent step (Equation 4.6). Instead, the entire training set is split up into many distinct subsets (or batches) and gradient descent steps are performed on each batch consecutively. The weights are updated sequentially. After an iteration has been applied for each batch, the process repeats, starting with the initial batch. Each full pass of the data is defined as an epoch. This is summarized in Figure 4.4. The learning rate β in Equation 4.6 is often an exponentially decreasing function of the epoch: the process of optimization where β depends on epoch is known as learning rate scheduling.

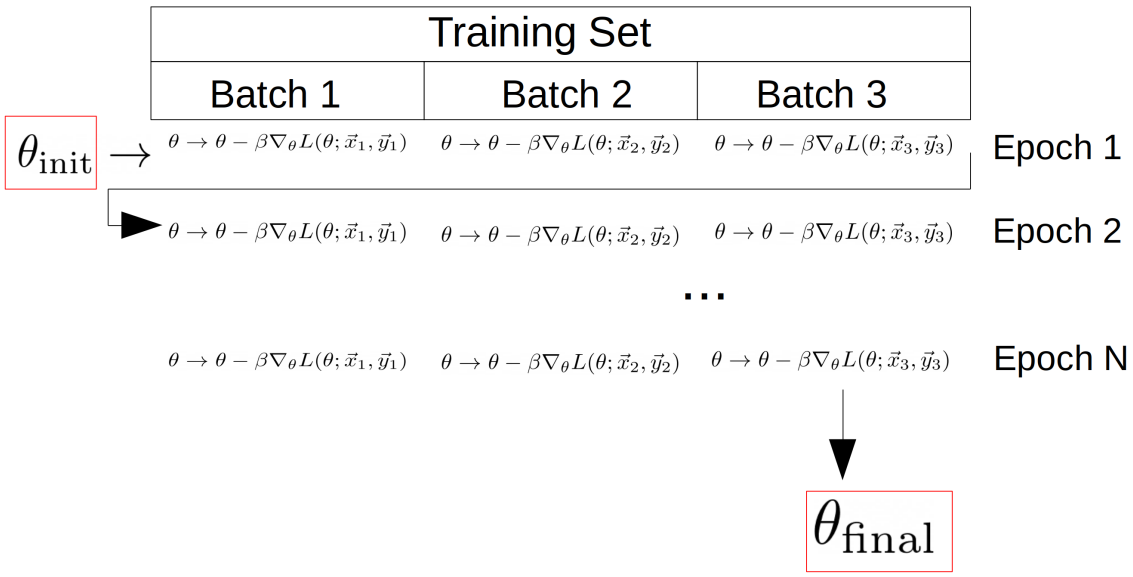


Figure 4.4: The gradient descent procedure with reference to batches and epochs. In this example, the full training set is split into three batches. When the loss function is computed for a certain batch, it is computed using only the data in that batch.

The software package used in this thesis for developing CNNs is TensorFlow [24], an open-source machine learning library developed by google. Models are trained using Compute Canada's WestGrid system. The Cedar cluster possesses powerful graphics processing units (GPUs) that significantly speed up the optimization procedure of equation 4.6.

In this thesis, $y_t = E(t)$ and $X_t = X(t)$ (specified by Equation 3.7). Specifically, for each calorimeter cell in this chapter, the full length of X_t is 30 million data points; this is split into a training set consisting of 80% of the time series and a validation set consisting of 20% of the time series. During training, 80 batches of 300000 bunch crossings are used during gradient descent. All results are typically shown for the validation set.

Layers k	Dilations $d^{(i)}$	Filters $f^{(i)}$	K Size $T^{(i)}$	Activation $R^{(i)}$	FBCs T_0
3	1,1,2	2	7	Relu	3

Table 4.1: Parameters used in basic neural network: identical parameters are used in each layer i. In addition, a final layer with a single filter ($f = 1$) and a kernel size of 3 ($T = 3$) are used to ensure the network output is a 1 dimensional time series. The filter depth, given by Equation 4.3, is $F_N = 27$.

4.2 A Traditional Approach

The purpose of the following sections of this chapter is to compare the energy reconstruction of two models:

1. The Optimal Filter (OF) given by Equation 3.16 with coefficients obtained using the Lagrange multiplier technique on Equations 3.13 and 3.14;

2. The Convolutional Neural Network (CNN) given by Equation 4.1 with weights obtained using Equation 4.6.

In this section, we examine a traditional CNN trained using the MSE loss function of Equation 4.5. Before analyzing the output of the model, the training procedure is first examined in depth.

Since the purpose of the model is to make predictions when signal is injected, only times t where $E^{(s)}(t) > 0$ are included in the loss function. Data are from the HEC1 cell located at $\eta = 2.35$, with $\mu = 200$. To examine variability in the training procedure, 20 identical models are trained with hyperparameters given in Table 4.1. Each model is trained for 500 epochs with the Nadam variation of gradient descent [23] and the learning rate scheduling function

$$\beta(x) = \begin{cases} 10^{-2} & 0 \leq x < 20 \\ 10^{-3} & 20 \leq x < 150 \\ 10^{-4} & 150 \leq x < 500 \end{cases} \tag{4.7}$$

where x is the current epoch number (starting at 0). The time required to train each of the 20 models was 220.1 ± 1.2 s.

In Figure 4.5, the training and validation loss for a single trial are shown. Both appear to nearly converge after training for 500 epochs. The two losses differ by less

than 0.1% at the end of training; this suggests that overfitting is unlikely for this model configuration and data set. This is rigorously confirmed in Figure 4.6, where all 20 trials are taken into account. The absence of overfitting likely arises from the large and homogenous data set used.

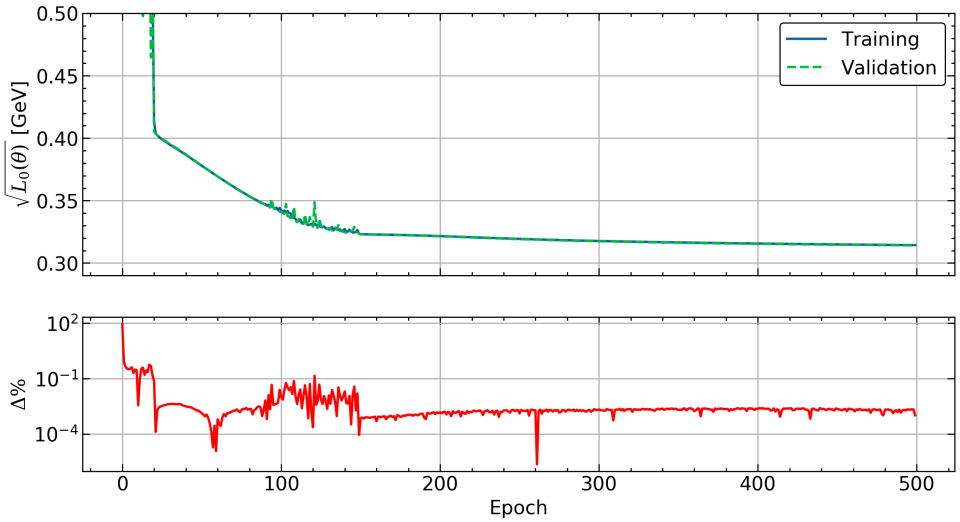


Figure 4.5: Optimization of loss function during gradient descent process. $\Delta\%$ is defined as $100\% \cdot (T_L - V_L)/T_L$ where T_L is the training loss (blue) and V_L is the validation loss (green).

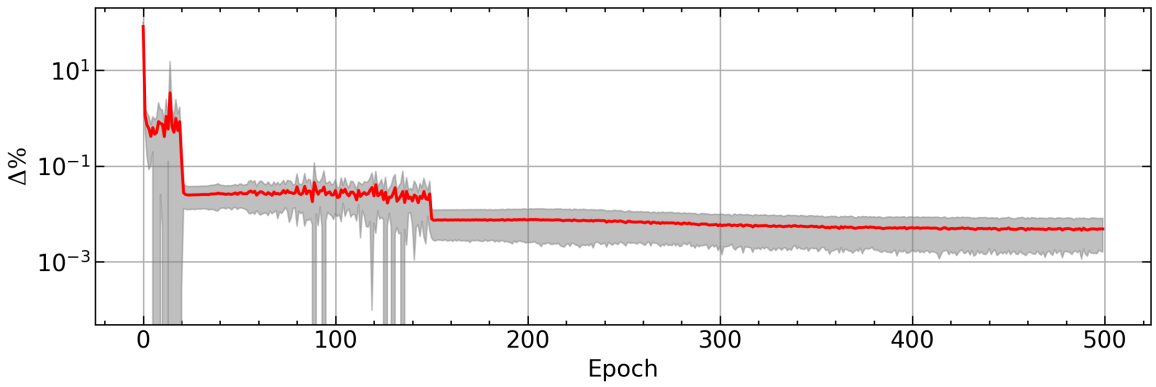


Figure 4.6: Mean value ± standard deviation of $\Delta\%$ for all 20 trials. $\Delta\%$ is defined as $100\% \cdot (T_L - V_L)/T_L$ where T_L is the training loss (blue) and V_L is the validation loss (green).

As overfitting is not present, it is justified to examine only the training loss in this analysis. The evolution of loss versus epoch for all 20 trials is shown in Figure 4.7. Despite an identical model architecture and training set, the optimization procedure differs significantly. This suggests that the initial randomization of network weights has a strong influence on the gradient descent procedure.

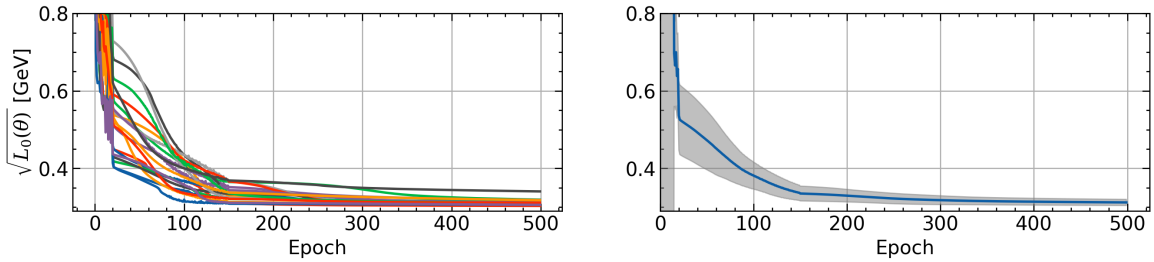


Figure 4.7: Optimization of loss function vs. epoch. Shown are all twenty trials (left) and the mean ± standard deviation of the twenty trials (right).

The distribution of $L_0(\boldsymbol{\theta})$ for all 20 trials at the end of training is another way to quantify variability in the training procedure. In Figure 4.8, this distribution is shown. The presence of the outlier point is an example that not all models converge to the lowest possible loss value.

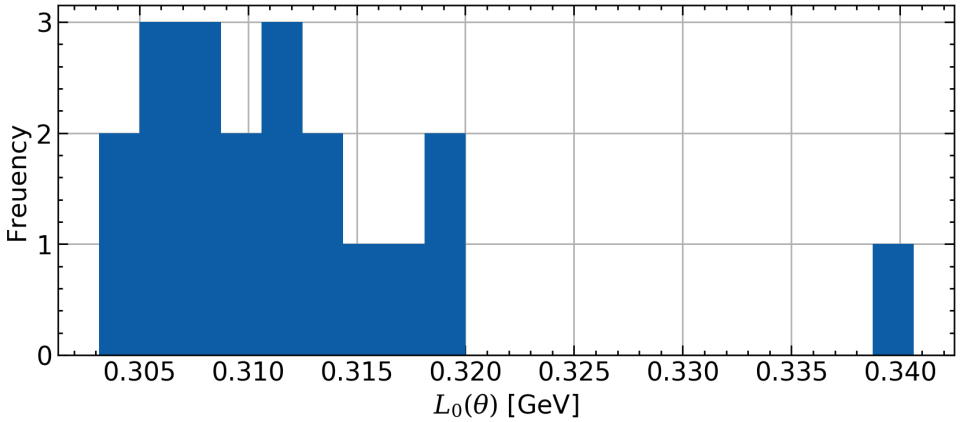


Figure 4.8: Distribution of $L_0(\boldsymbol{\theta})$ after 500 epochs of training for 20 distinct trials.

It can be informative to directly compare the training data to the model input. Figure 4.9 shows the network input $X(t)$, labels E and prediction $\hat{E}$ for one of the twenty trained models. Predictions are made at times where $E^{(s)}(t) > 0$.

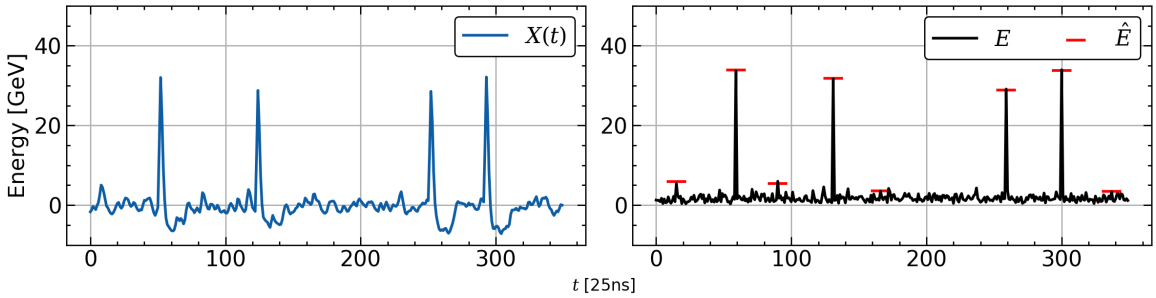


Figure 4.9: Model input (left) and model energy output vs. true energies (right) for a small portion of the data set.

The loss function $L_0(\theta)$ corresponds to one descriptive statistic of all residuals $\hat{E} - E$. As was done with the optimal filter in Figure 3.8, the complete distribution of $\hat{E} - E$ is given by a histogram of all values. In Figure 4.10, the residuals of the optimal filter are compared to those of the CNN; both were found to be Gaussian, including the tails. Since the CNN has a filter depth of $F_N = 27$, it is compared with an optimal filter with filter depth of $N = 27$. Based on Figure 3.7, it is unlikely that increasing N past 27 will significantly improve the RMSE of the optimal filter. This analysis thus tests the limitations of the optimal filter technique to the CNN.

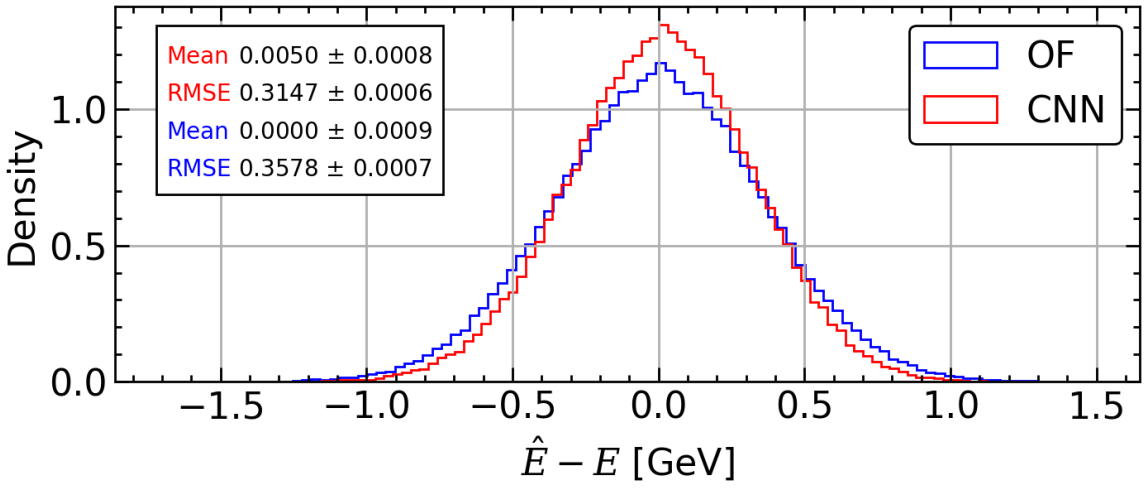


Figure 4.10: Distribution of optimal filter (OF) residuals and CNN residuals; means and RMSEs shown with units in GeV. The optimal filter uses $N = 27$ coefficients and is trained using 3000 bunch crossings. The parameters of the CNN are given by Table 4.1. This plot is also shown with logarithmic y axis scaling in Figure A.1 of the appendix.

While the CNN has a smaller RMSE than the OF, its net bias is not equal to zero. The predictions of the basic CNN suffer a more serious complication, however. As stated in Chapter 3.3, a good model must be unbiased in distinct energy regions (Figure 3.9). As such, it is useful to examine different $E^{(s)}$ regions, as is done in Figure 4.11. While the optimal filter has zero bias in all $E^{(s)}$ regions, the machine learning model does not.

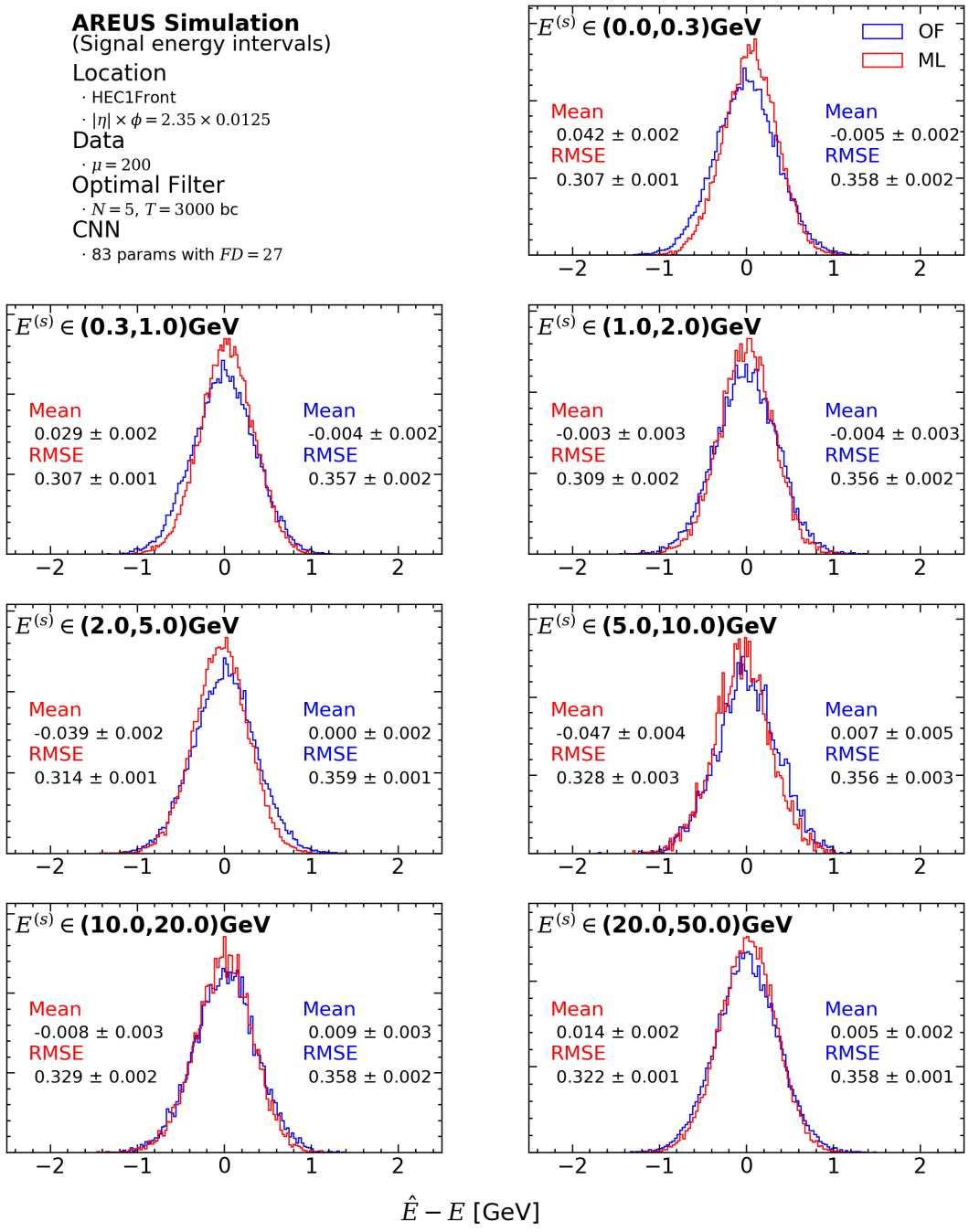


Figure 4.11: Distribution of optimal filter (OF) residuals and CNN residuals for different signal $E^{(s)}$ regions (shown in top left of each plot). Mean and RMSE units are GeV. This plot is also shown with logarithmic y axis scaling in Figure A.2 of the appendix.

In Figure 4.12, the biases of each model are plotted explicitly vs. the different signal $E^{(s)}$ injection regions. While the optimal filter makes unbiased predictions in all intervals, the CNN tends to over predict small energies and under predict large energies.

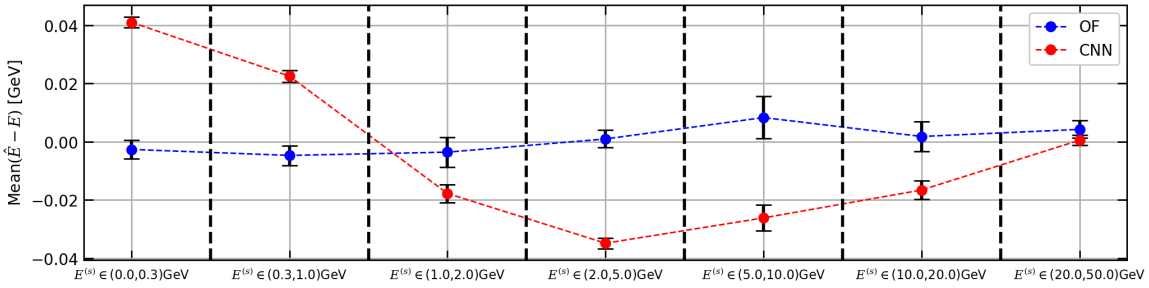


Figure 4.12: Mean value of $E - \hat{E}$ for the different $E^{(s)}$ regions shown on Figure 4.11.

Ideally, a CNN model should make unbiased predictions in all signal $E^{(s)}$ regions while maintaining a lower RMSE than the optimal filter. In Chapter 4.3, such a model is developed using a new loss function.

An intuitive way to quantify the extent to which a model makes biased predictions is to sum the absolute value of all plotted data points on Figure 4.12. This can be defined mathematically as follows: let $S' \subset S$ where S is all possible signal injection values and let $\boldsymbol{I}_{S'} = \{t | E_t^{(s)} \in S'\}$. Partition S into subsets $S_1, S_2, ...S_n$. Call this partition P. An example of various partitions are shown in Table 4.2. The sum of absolute means in P is defined by

$$\text{SAM}(P) \equiv \sum_{j=1}^{n} \left| \frac{1}{|\boldsymbol{I}_{\boldsymbol{S_j}}|} \sum_{t \in \boldsymbol{I}_{\boldsymbol{S_j}}} (\hat{E}_t - E_t) \right| \tag{4.8}$$

where $|\boldsymbol{I}_{\boldsymbol{S_j}}|$ is the number of elements in the set $\boldsymbol{I}_{\boldsymbol{S_j}}$. This quantity will be examined extensively during the rest of this thesis. A variant of it is used to construct an improved loss function in Chapter 4.3. It will be shown that models trained with the correct loss function hyperparameters have SAM scores comparable to the optimal filter.

Subset	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Values [GeV]	0-0.3	0.3-1	1-2	2-5	5-10	10-20	20-50

Table 4.2: The signal partition defined in this paper as P_0.

Hyperparameters					**Results**		
Dilations	Filters	Kernel size	FBCs	Filter Depth	Rank	RMSE [GeV]	SAM(P_0) [GeV]
(1,3)	3	7	9	27	1	0.2996±0.0016	0.1543±0.0116
(1, 2)	3	7	17	21	4	0.3057±0.0005	0.1507±0.0086
(1, 1, 2)	2	7	9	27	5	0.3057±0.0008	0.1424±0.0067
(1, 1, 3)	2	5	9	23	10	0.3101±0.0013	0.1579±0.0077
(1, 1, 2, 2)	2	5	17	27	11	0.3108±0.0010	0.1673±0.0087
(1, 1, 1, 2)	2	5	17	23	14	0.3111±0.0008	0.1579±0.0108
(1, 3)	3	5	13	19	15	0.3114±0.0003	0.1570±0.0019
(1, 2)	3	5	9	15	21	0.3140±0.0006	0.1605±0.0080
(1, 1)	3	7	9	15	23	0.3144±0.0007	0.1597±0.0108
(1, 3)	4	3	9	11	33	0.3203±0.0007	0.1747±0.0100
(1, 2)	4	3	5	9	37	0.3278±0.0013	0.1761±0.0108
Optimal Filter					38	0.5345±0.0010	0.0269±0.0119

Table 4.3: Hyperparameter search: only showing results for top future bunch crossings (FBCs) T_0 for each model. Rank is based on RMSE metric. Optimal filter shown in the bottom row

Before constructing a loss function to reduce the SAM, an experiment is first conducted to determine the optimal set of hyperparameters for minimizing the RMSE whilst using the MSE loss function. To ensure that the number of model parameters was small enough to be implemented on an FPGA (see Section 5.2), all models had 100 or less parameters. For each set of hyperparameters, 20 models were trained, and the top 5 models were used to compute the mean and standard error of both the RMSE and SAM(P_0) metrics. The results are shown in Table 4.3.

The top performing models had a large filter depth and, bounded by the 100 parameter limit, as many filters as possible. While a large filter depth allows a model to obtain more information on pile-up noise, more filters permit greater non-linear capabilities. Perhaps most notably, models performed best when the number of FBCs T_0 was 9 or greater. With a sufficient number of FBCs, the model can extract more information about the bipolar pulse caused by a particular event. The top model configuration had a filter depth of $F_N = 27$ and performed optimally with $T_0 = 9$; it likely uses eighteen prior bunch crossings to obtain information on the pileup noise and seven future bunch crossings to obtain information on the pulse shape of the simulated energy event. This is visually depicted in Figure 4.13.

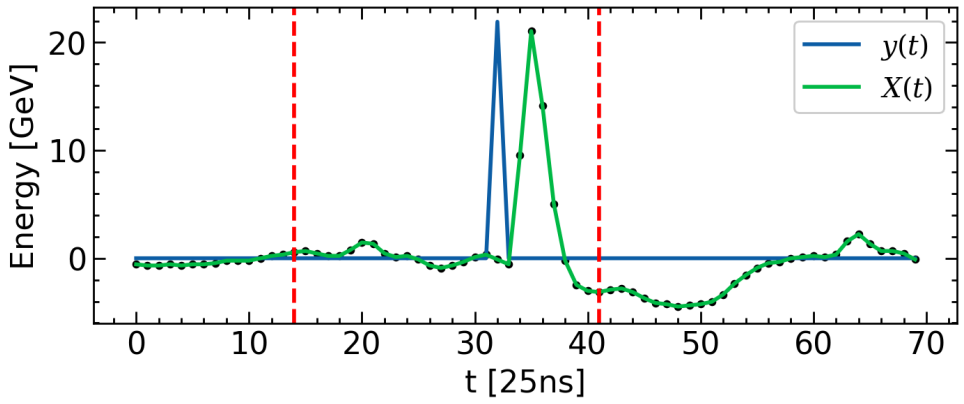


Figure 4.13: Visual depiction of the scope of the top model in Table 4.3. The model is able to see $T_0 = 9$ FBCs after the signal injection and has a filter depth of 27: this is represented by the two dotted red lines.

In Figure 4.14, the RMSE vs. FBCs is plotted for the top 3 model architectures. The significant increase in model performance as T_0 increased from 5 to 9 is likely because the model has greater access to the bipolar pulse shape caused by the simulated energy event.

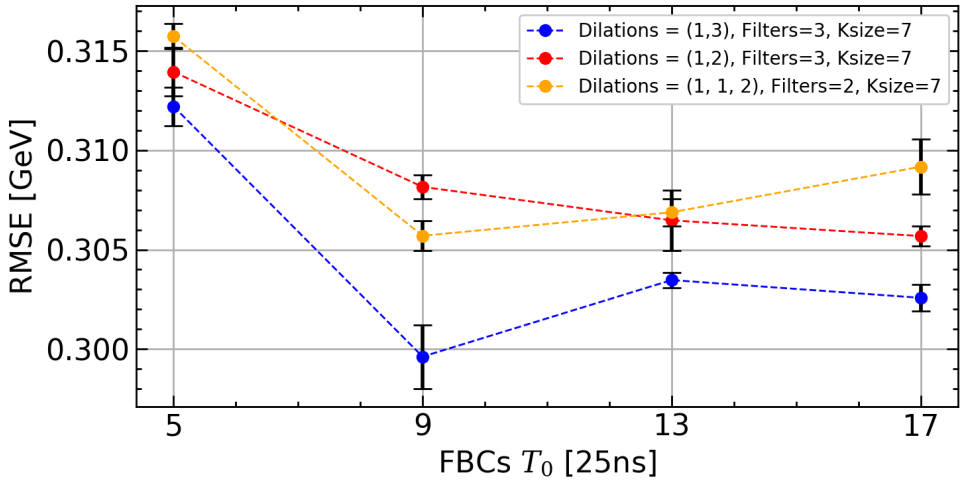


Figure 4.14: Top 3 model architectures (rows 1-3 of Table 4.3) trained with varying FBCs T_0.

For each of the top three model architectures of Table 4.3, the average loss for twenty trials is plotted on Figure 4.15. Most notably, the model architecture with

dilations (1,1,2) seems to be inconsistent during the first few epochs of training, but eventually catches up to the model with dilations (1,2) after 300 epochs. The model architecture with dilations (1,3) tends to perform the best during the entire training procedure. This is likely due to the increased filter depth and number of filters of this architecture.

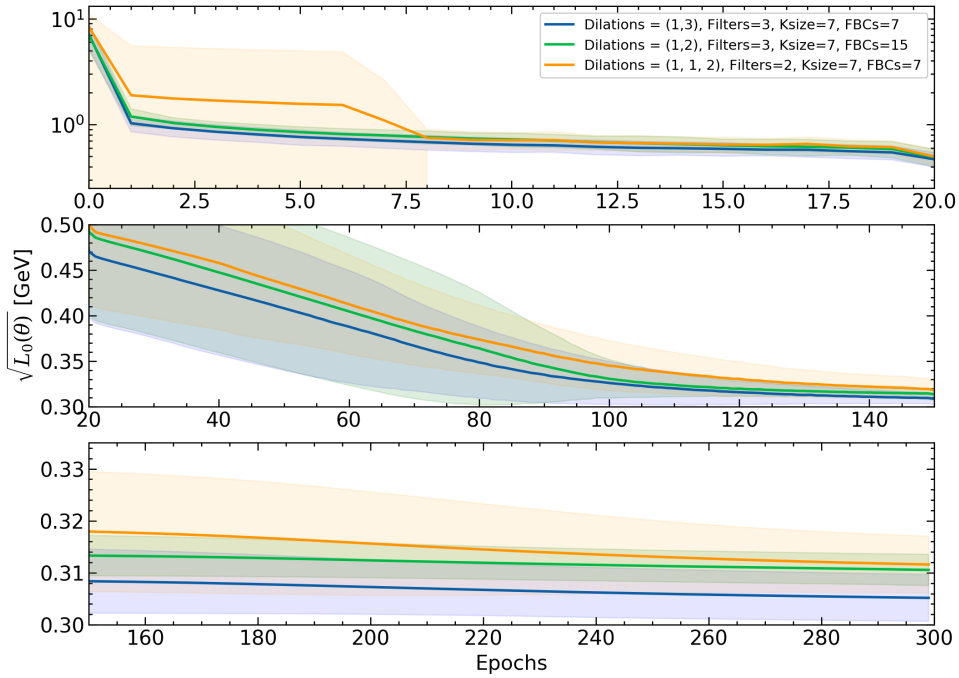


Figure 4.15: Average RMSE (with error bars $\pm 1\sigma$) during training of all twenty trials for the top 3 model architectures (rows 1-3 of Table 4.3). Shown are epochs 0-20 (top), epochs 20-150 (middle), and epochs 150-300 (bottom)

Despite significantly outperforming the optimal filter in terms of the RMSE metric, all models still had a SAM(P_0) metric far greater than the optimal filter. In the next section, a new loss function is designed such that the SAM(P_0) metric is optimized for during model training.

4.3 An Improved Approach

In the previous section, it was shown that CNNs trained using the RMSE loss function outperformed the OF in terms of the MSE metric, but failed to make unbiased

energy predictions in certain energy intervals. The extent of the bias was characterized using the SAM metric of Equation 4.8. This section begins with the derivation of a new loss function, loosely based on the SAM metric, and concludes by showing the models trained using the new loss function have significantly reduced bias.

Let $S' \subset S$ where S is all possible signal injection values and let $\boldsymbol{I}_{S'} = \{t | E_t^{(s)} \in S'\}$. Partition S into subsets $S_1, S_2, ...S_n$. Call this partition P. Consider following loss function, loosely based on the SAM metric:

$$L_1(P) = L_0 + \sum_{j=1}^{n} \alpha_j \left(\frac{1}{|\boldsymbol{I}_{\boldsymbol{S_j}}|} \sum_{t \in \boldsymbol{I}_{\boldsymbol{S_j}}} (\hat{E}_t - E_t) \right)^2 \tag{4.9}$$

where $|\boldsymbol{I}_{\boldsymbol{S_j}}|$ is the number of elements in the set $\boldsymbol{I}_{\boldsymbol{S_j}}$, L_0 is the MSE loss function given by Equation 4.5, and α_j are additional hyperparameters to adjust the relative importance of L_0 to the absolute means. For this thesis, all $\alpha_j = \alpha$ for simplicity, but it is worth noting that these parameters can be configured to give different weighting in different signal energy intervals, if needed.

The following is an analysis to determine the optimal value of α that should be used in Equation 4.9; data were simulated for the HEC1 cell located at $\eta = 2.35$ with $\mu = 200$. The CNN architecture used the configuration from row 3 of Table 4.3 with $T_0 = 13$ FBCs. For each value of α, twenty models were trained for 3000 epochs using the loss function of Equation 4.9. The total computation time was approximately 140 hours; most computation was done using parallel jobs. The learning rate scheduler used was as follows:

$$\beta(x) = \begin{cases} 10^{-2} & 0 \leq x < 60 \\ 10^{-3} & 60 \leq x < 1000 \\ 10^{-4} & x \geq 1000 \end{cases} \tag{4.10}$$

Figure 4.16 shows the mean/standard error of the RMSE and SAM(P_0) each set of twenty trials. For small values of α, the MSE component of the loss function L_0 dominates and the SAM(P_0) metric is much larger for the CNN than for the OF. As α increases from approximately 10^{-2} to 5, the SAM(P_0) metric experiences a significant improvement while the RMSE metric only experiences a slight reduction

Subset	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Values [GeV]	0-0.1	0.1-0.7	0.7-3.1	3.1-7.8	7.8-15	15-31	31-50

Table 4.4: The signal partition defined in this paper as P_1.

in performance. After about $\alpha = 5$, however, the SAM(P_0) metric and especially the RMSE metric experience a significant reduction in performance. This is likely due to an instability in training that occurs when the loss function is not sufficiently weighted by the regular MSE loss L_0. Perhaps most promising is the demonstration that models trained by Equation 4.9 with $\alpha_j = \alpha \approx 5$ significantly outperform the OF in the RMSE metric while retaining a good performance with in the SAM(P_0) metric.

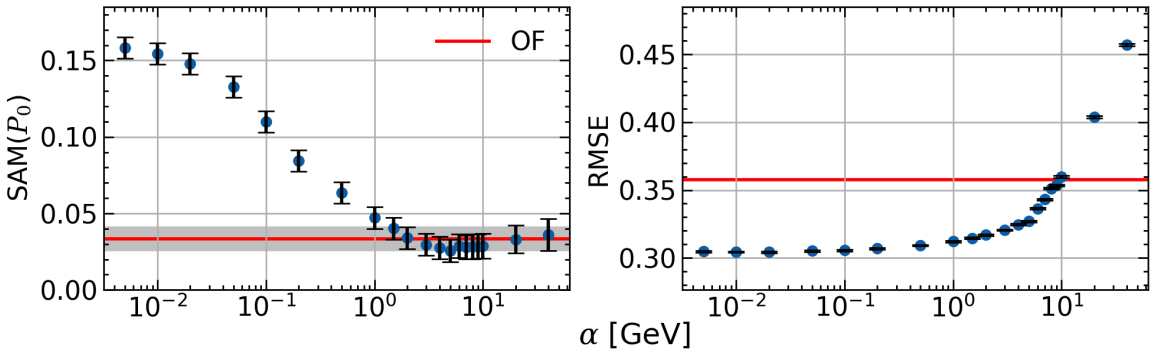


Figure 4.16: SAM(P_0) and RMSE for different α_j loss function parameters with models trained using $L_1(P_0)$. The SAM(P_0) and RMSE from the optimal filter are shown in red lines on the figure, with the grey bands showing the error.

There are potential issues with this loss function, however. During training, Equation 4.9 is only designed to penalize biased predictions for a single partition P_0. Mathematically, the model is optimized using the $L_1(P_0)$ loss and then its performance is evaluated using SAM(P_0) metric. It is theoretically possible that a model optimized by $L_1(P_0)$ may have a poor SAM(P_1) metric for another partition P_1. Fortunately, in practice, the SAM(P_1) metric if often only slightly worse than the SAM(P_0) metric. Let P_1 be defined by Table 4.4. As seen in Figure 4.17, a CNN trained using $L_1(P_0)$ has a SAM(P_1) score only marginally worse than the OF for the optimal value of α, despite the significant differences between P_0 and P_1.

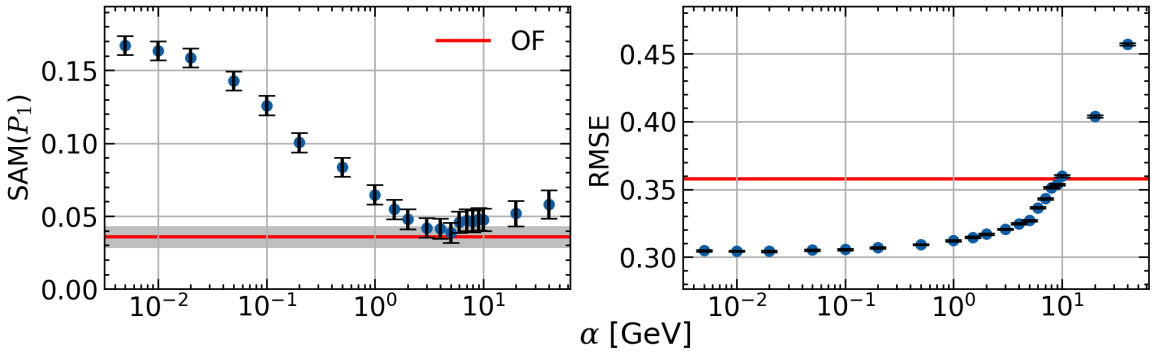


Figure 4.17: SAM(P_1) and RMSE for different α_j loss function parameters with models trained using $L_1(P_0)$. The SAM(P_1) and RMSE from the optimal filter are shown in red lines on the figure, with the grey bands showing the error.

Histograms of residuals are plotted explicitly in Figures 4.18 (where separate axes correspond to a P_0 partition) and 4.19 (P_1 partition). It is clear from the histograms that the CNN outperforms the OF in terms of width and the CNN has significantly reduced bias in all intervals.

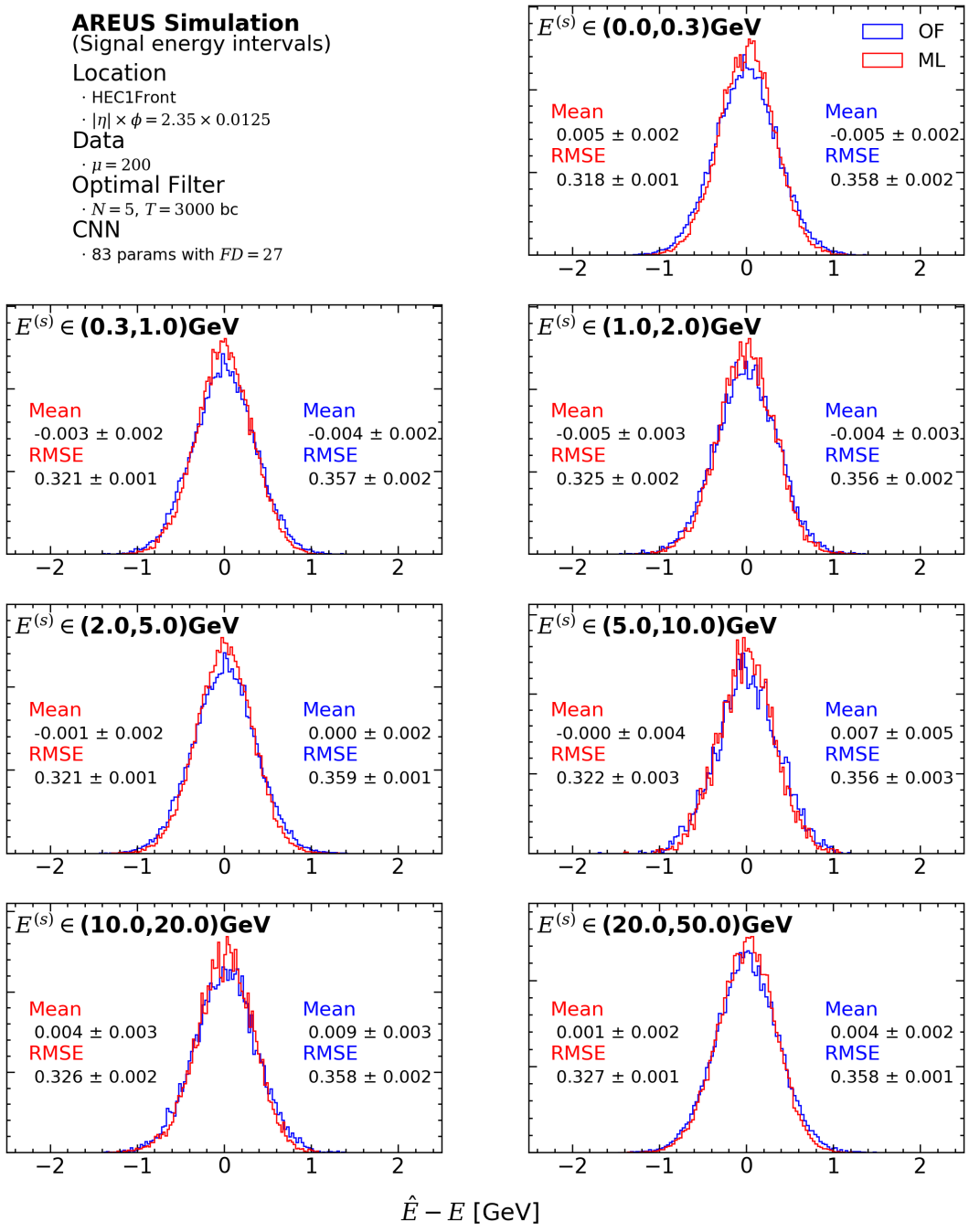


Figure 4.18: Residuals for model with the best SAM(P_0) score. Seperate axes correspond to the partition P_0. This plot is also shown with logarithmic y axis scaling in Figure A.3 of the appendix.

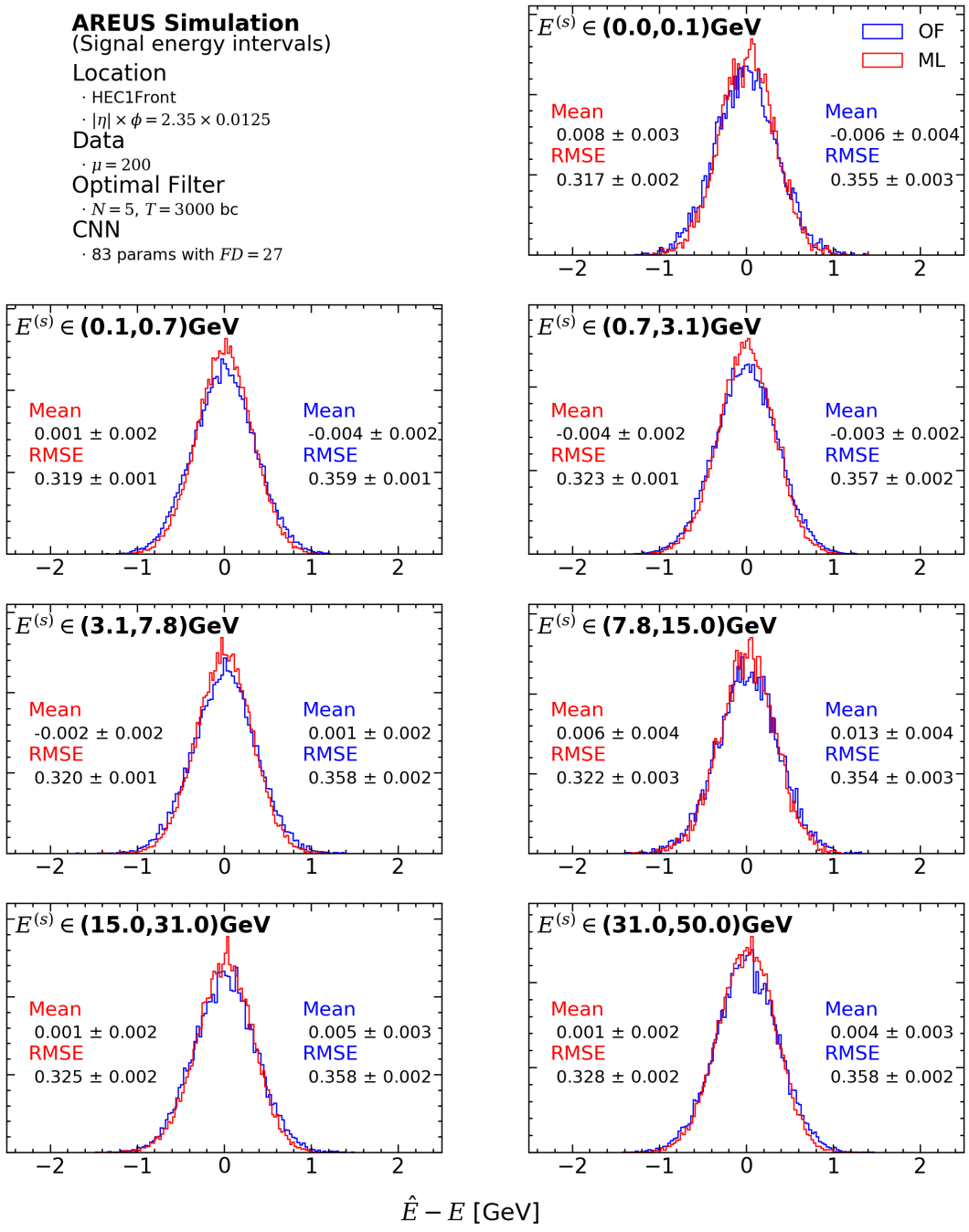


Figure 4.19: Residuals for model with the best SAM(P_0) score. Seperate axes correspond to the partition P_1. This plot is also shown with logarithmic y axis scaling in Figure A.4 of the appendix.

4.4 Global Results

With the development of a suitable model architecture and loss function, CNN models can be trained for all HEC cells. Without loss of generality, due to azimuthal symmetry in particle production, only cells with $\phi = 0.0125$ are considered in this section. All trained models have the same hyperparameters: Rank 1 of Table 4.3 with $T_0 = 11$ FBCs.

Each cell in the detector is specified by its HEC subsystem (HEC1, HEC2, HEC3, HEC4) and η. 10 models were trained for 2000 epochs for each value of η in all four subsystems of the HEC calorimeter using $L_1(P_0)$ as a loss function; the model with the best SAM(P_0) score was chosen as the best model. The total computation time was approximately 120 hours; most of the computation was done in parallel. The performance of these models is shown in Figures 4.20.

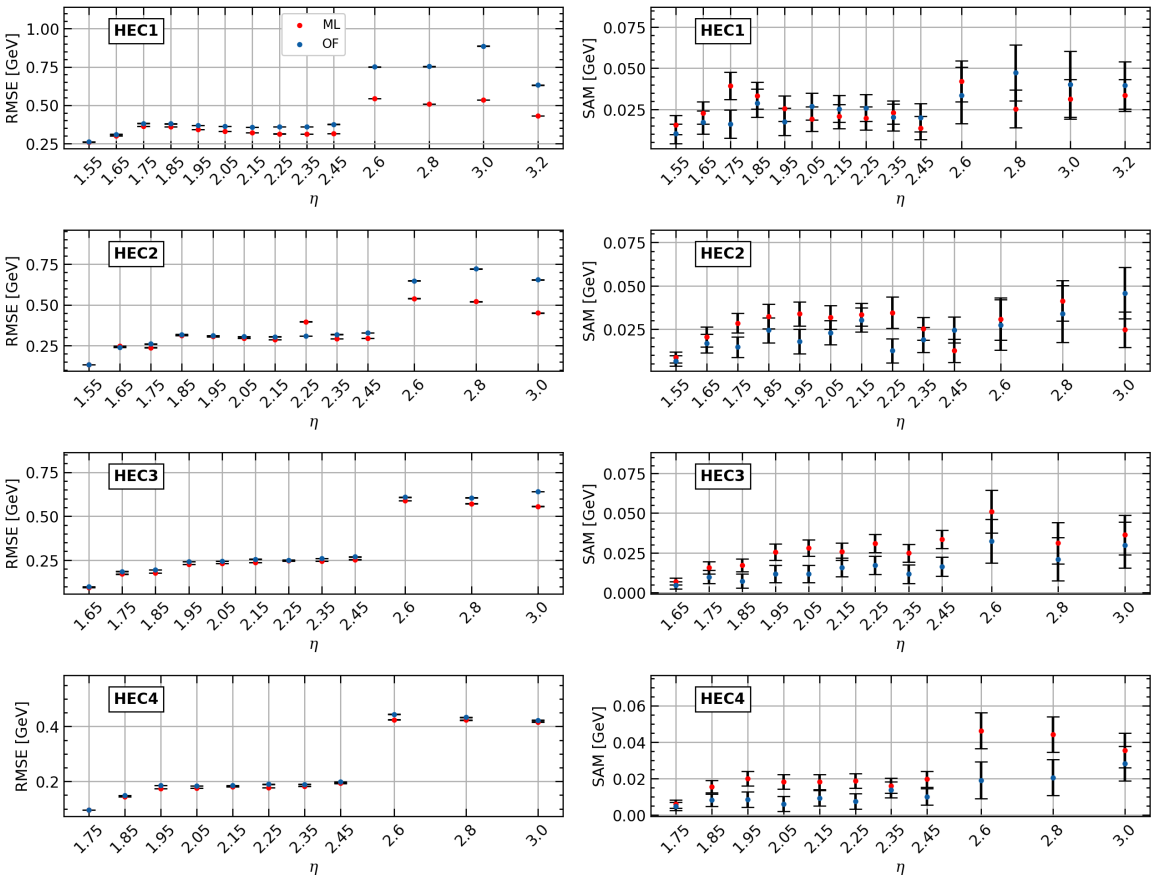


Figure 4.20: RMSE and SAM(P_0) for the model with the best SAM(P_0) score out of 10 models. Models are trained for all η values and at $\mu = 200$.

An alternative way to plot the results is to look at the relative percent difference between the CNN and OF. Defining $\Delta\% = (Q_{OF} - Q_{CNN})/Q_{OF}$ where Q is the RMSE or SAM of the two predictors, it is clear that the CNN is a better predictor for cells with a positive $\Delta\%$. Results are shown in Figure 4.21.

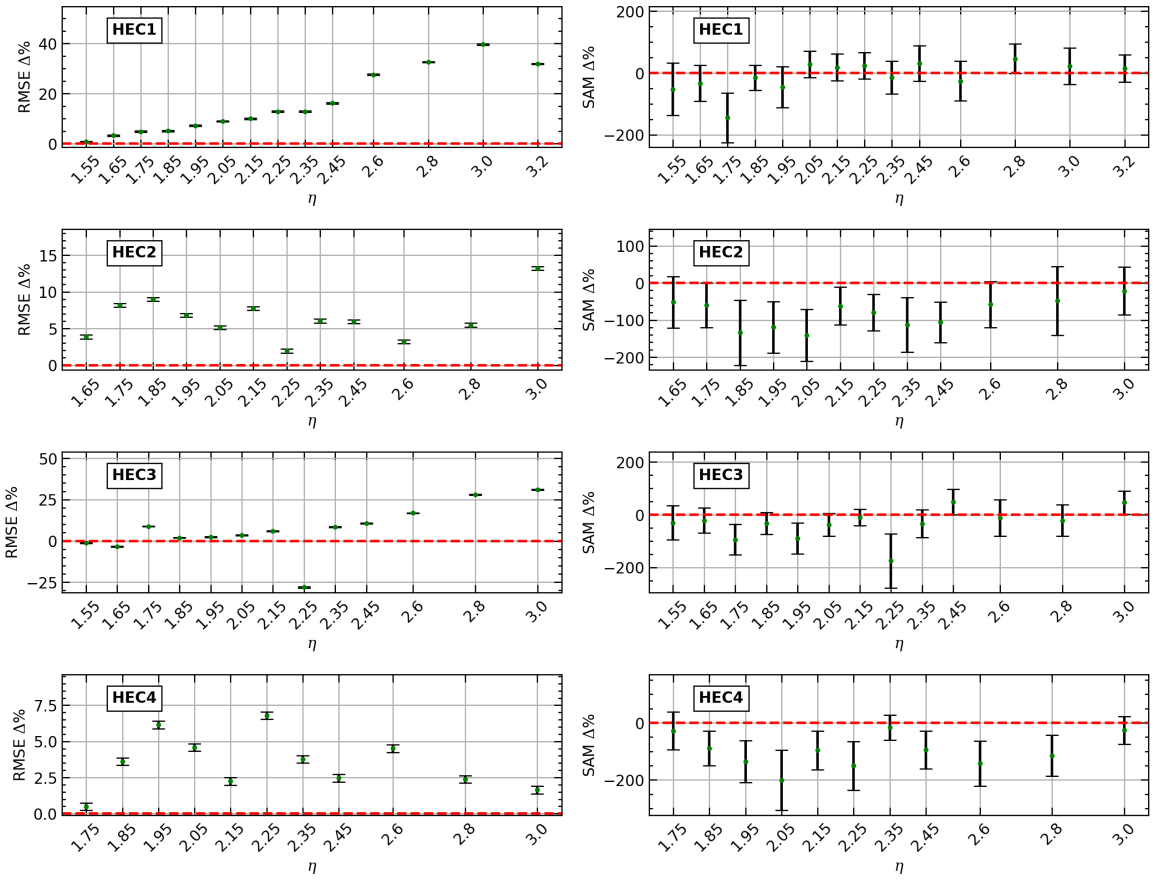


Figure 4.21: $\Delta\%$ of RMSE and SAM(P_0) for the model with the best SAM(P_0) score out of 10 models. Models are trained for all η values and at $\mu = 200$.

The differences, defined as $Q_{OF} - Q_{CNN}$ can also be compared, as shown in Figure 4.22.

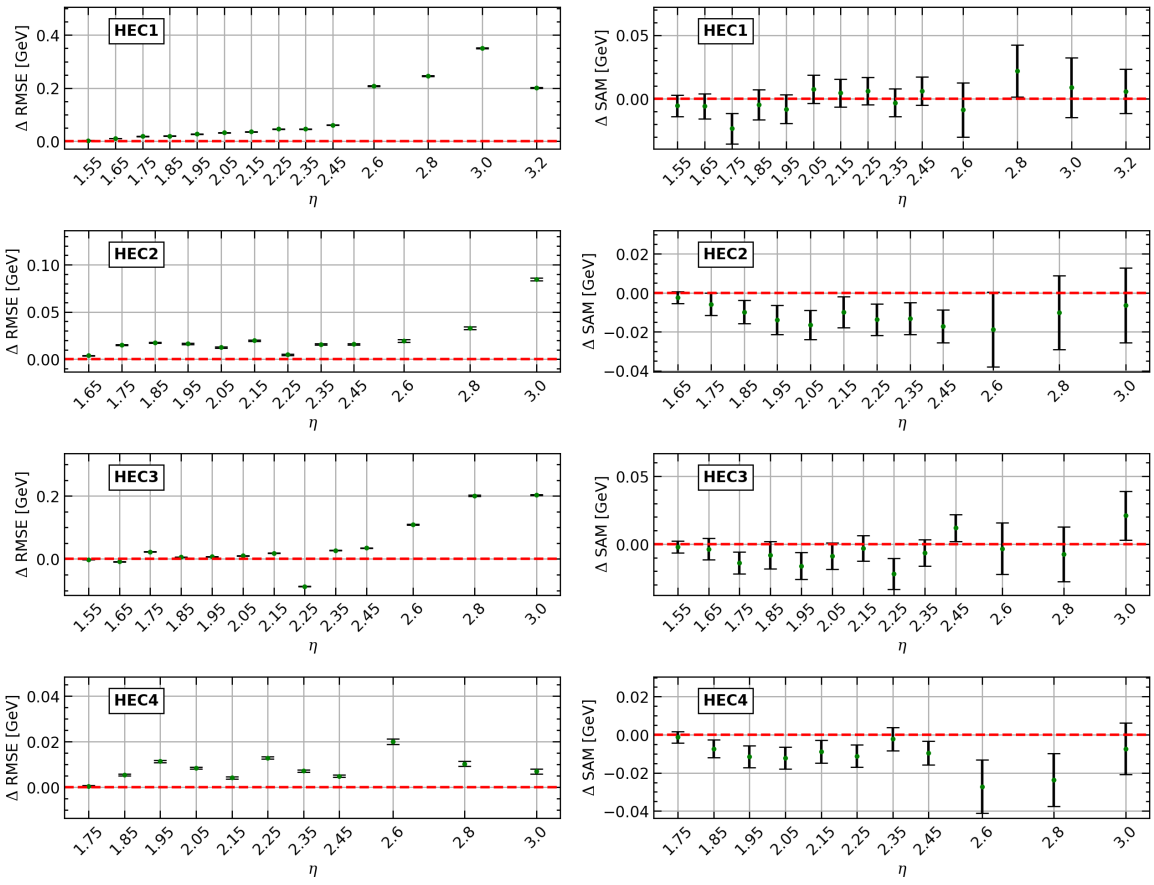


Figure 4.22: Differences of RMSE and SAM(P_0) for the model with the best SAM(P_0) score out of 10 models. Models are trained for all η values and at $\mu = 200$.

The machine learning model performs better than the optimal filter in RMSE for all values of η, especially for larger values of η, where more pileup noise is present. In the HEC1 system, the CNN and OF have consistent SAM(P_0) scores for most η values. For the HEC2, HEC3, and HEC4, however, the SAM(P_0) scores typically favour the OF. In conclusion, for the model architecture, loss function, and training scheme used, the CNN tends to outperform the OF technique at the HL-LHC: specifically for large values of η in the HEC1 system.

4.5 Additional Benefits of CNN Models

The derivation of the optimal filter estimator in Chapter 3.2 requires that signal events are spaced sufficiently far apart so that consecutive ionization currents are non-

overlapping. Throughout this thesis, data have been simulated such that consecutive signal events are spaced apart according to a uniform distribution between 30 and 50 bunch crossings. This results in non-overlapping bipolar pulse shapes. In practice, however, signal events may be very close together, as shown in Figure 4.23.

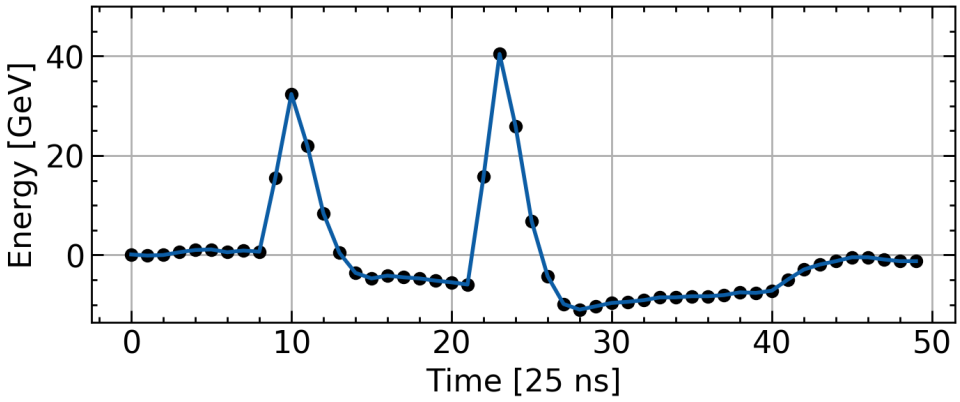


Figure 4.23: Overlapping ionization pulses from two closely spaced signal events.

In the following analysis, the generated data use signal energies injected according to uniform distribution between 3 and 50 bunch crossings, which permits the occasional overlap of signal events. The cell simulated is at $\eta = 2.35$ in the HEC1 subsystem. Ten CNN architectures with hyperparameters corresponding to Rank 1 of Table 4.3 were trained using the loss function from Equation 4.9 with $\alpha = 5$ and the learning rate scheduler from Equation 4.3; the CNN with the lowest SAM(P_0) score is used for analysis here. The CNN is compared to two configurations of the optimal filter: one with $N = 5$ coefficients and the other with $N = 25$ coefficients. While an optimal filter with larger N performs better under normal circumstances, it is reasonable that a smaller depth may prove advantageous when pulse shapes are close together. Thus both configurations are compared. The predictions of both optimal filters and the CNN are shown in Figure 4.24.

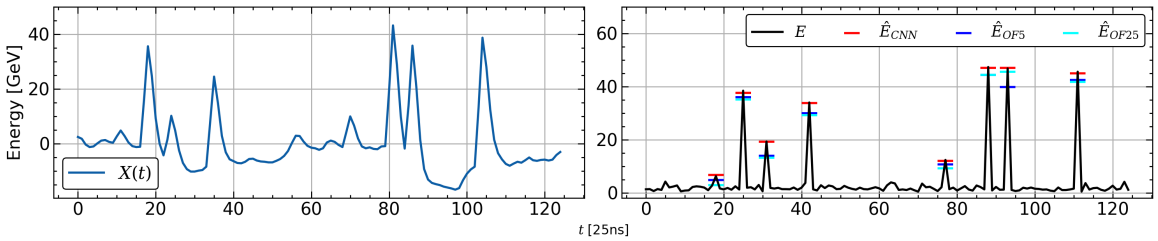


Figure 4.24: OF and CNN predictions when injected signals are close enough together such that bipolar pulse shapes overlap. OF5 corresponds to the optimal filter with $N = 5$ coefficients and OF25 corresponds to the optimal filter with $N = 25$ coefficients.

The optimal filter tends to under-predict energies when signal injections are close together for both $N = 5$ and $N = 25$ coefficients. A histogram of all residuals during signal injection is shown in Figure 4.25.

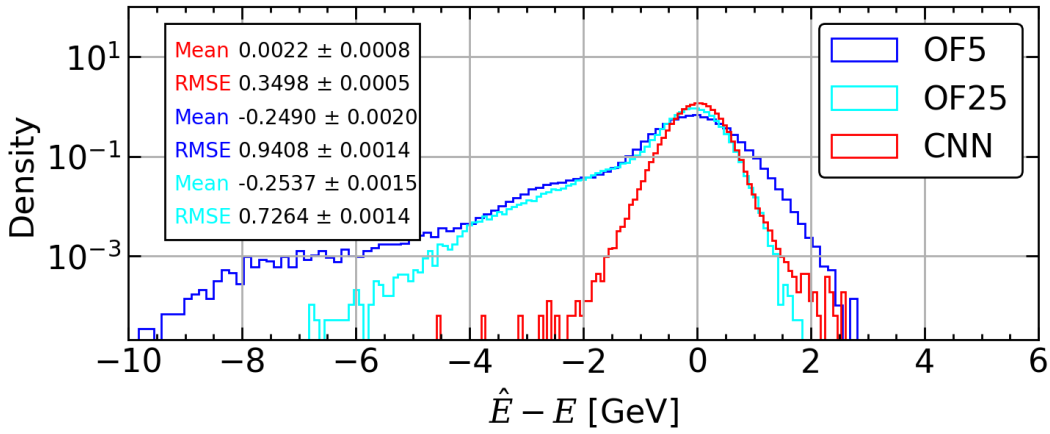


Figure 4.25: Histogram of OF ($N = 5$ and $N = 25$) and CNN residuals for data set where signal injection hits are close together. Conditions were $\mu = 200$ in cell $\eta = 2.35$ of HEC1.

The skew towards negative values for the OF for both N values is from under-predicting the energy when signal injections are close together. Interestingly, the optimal filter still performs better with a larger filter depth. While CNN models are able to make reasonable predictions in these circumstances, they still suffer a decrease in performance when used on overlapping pulse shapes. For example, the RMSE and SAM(P_0) for the CNN in this section are 0.3458±0.0005 GeV and 0.0349±0.0070 GeV.

As shown on Figure 4.20, however, the RMSE and SAM(P_0) for the CNN trained and used with the previous dataset are 0.3127 ± 0.0006 GeV and 0.0231 ± 0.0071 GeV. This is not unexpected; overlapping pulse shapes produce a more complicated data set, resulting in less adequate predictions. In conclusion, as there is no simple adjustment to the optimal filter derivation to account for such events and the CNN still retains reasonable accuracy, the CNN is undoubtedly the best model when dealing with such data.

Chapter 5

Implementation of Machine Learning in ATLAS

In Chapter 4, it was shown that CNN models can outperform the optimal filtering technique in many HEC calorimeter cells: particularly those at high η values. This chapter examines the simulation of a neural network on an FPGA; this is the means by which machine learning techniques will be implemented in ATLAS.

5.1 The FPGA

In the proposed ATLAS Phase-I upgrade, all 182468 LAr calorimeter cells will be equipped with an FPGA-based back-end system where advanced digital-filtering methods, such as the optimal filter or a CNN based filter, will be used for energy reconstruction and time alignment in each calorimeter cell [25]. While the specifications of the FPGA device are not known as of the writing of this thesis, ATLAS electronics experts recommended that all CNN models that had less than 100 parameters to ensure resource usage was not exceeded.

An FPGA is an integrated circuit designed to be configured by a customer after purchase. Through firmware configuration, it is possible to implement specialized computations that perform much faster than on a traditional CPU; this is accomplished through the use of off-load and acceleration functions. The user configuration is typically specified using a hardware description language (HDL), such as VHDL or

Verilog. The fast I/O rates of FPGAs are crucial for the required data transfer speed of 200-400Tb/s after the phase II upgrade. FPGAs also reduce the cost of ownership by supporting long lifecycles and eliminating the need of expensive software tools and specialized design teams.

FPGAs consist of many programmable logic blocks (such as AND and OR gates, for example) that can be inter-wired into a specific configuration to increase application speed. One such block, used for signal processing, is the digital signal processor (DSP). The goal of a DSP is to measure, filter, or compress real world analog signals. For the purpose of energy reconstruction in the HEC, they can process the ionization current given by Equation 3.7 and perform the filtering operations outlined in equation 4.2. By recommendation of electronics experts, since the number of configurable DSPs is limited on the FPGA used for the Phase-II upgrade and the number of filtering operations required in a CNN is large, the number of parameters for CNNs trained in this thesis were constrained to less than 100.

Perhaps the biggest challenge of implementing a CNN on an FPGA is the requirement that all neural network weights (coefficients A and b in equation 4.2) need to be converted from floating point (i.e. real numbers) to fixed point (i.e. integers). Careful tuning is required to minimize errors from rounding while also maintaining small enough outputs in the intermediate outputs of 4.1 to ensure all values can be represented by a limited number of bits. The loss of precision from the digitization of neural network weights introduces an additional uncertainty in energy reconstruction. This is examined more thoroughly in Section 5.2.

There exist many tools to simulate operation on an FPGA; one such simulation program is ModelSim. Simulation programs are used for designing and debugging an FPGA application without the use of a physical FPGA. In this thesis, the tools developed by the Liquid Argon Signal Processing (LASP) group are used. The module LASP-dacore provides simulation of basic CNNs on a generic FPGA using ModelSim. Tools in the LASP-dacore module will eventually be used to implement the new digital filtering algorithm in the ATLAS phase II upgrade; it is thus essential that all proposed CNNs are compatible with the simulation code.

5.2 Simulation Results

The following section briefly examines simulation results using the LASP-dacore module. The input data corresponds to $\mu = 200$ data at $\eta = 2.35$ in HEC1. The simulated neural network corresponds to the model with the best SAM(P_0) score in HEC1 at $\eta = 2.35$. The simulation procedure is as follows:

1. The neural network architecture, weights, and corresponding input sequence are stored in json and h5 file formats. All numerical values are floating point.

2. A configuration file from the LASP-dacore module is used to specify the desired bit-width representation of the neural network weights and input sequence. Execution of this file produces a corresponding HDL test bench file to be used with Modelsim, along with data files that store the neural network weights and input sequence in a hexadecimal format.

3. The test bench file is used in conjunction with the LASP-dacore library to simulate the output of the neural network in ModelSim. Output is given in hexadecimal format.

4. The ModelSim output is converted back to decimal and compared with the traditional neural network output of python.

To efficiently store floating point numbers in a binary representation, a certain number of bits are used to store the integer number left of a decimal point and a certain number of bits are used to store the decimal values to the right of the decimal point. This thesis typically refers to the total number of bits used to store the number, and the number of bits used to store the decimal value, referred to as fractional bits. Figure 5.1 shows an example.

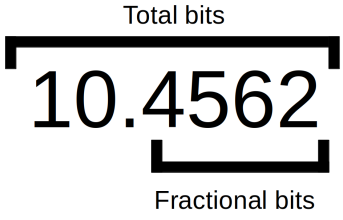


Figure 5.1: Representation of total bits and fractional bits when storing a number in binary.

The bits used to represent various parts of the weights and input sequence are mathematically defined as follows:

- $B_W \equiv$ the total number of bits used to represent the weights.
- $B_W^{(f)} \equiv$ the number of fractional bits used to store the weights.
- $B_I \equiv$ the total number of bits used to represent the input sequence.
- $B_I^{(f)} \equiv$ the number of fractional bits used to represent the input sequence.

In Figure 5.2, the output of the neural network in both python and the LASP simulation are shown. Also shown is the difference in energy predictions, defined by $\Delta E = (\hat{E}^{(p)} - \hat{E}^{(v)})$ where $\hat{E}^{(p)}$ are the CNN predictions in python and $\hat{E}^{(v)}$ are the predictions in ModelSim.

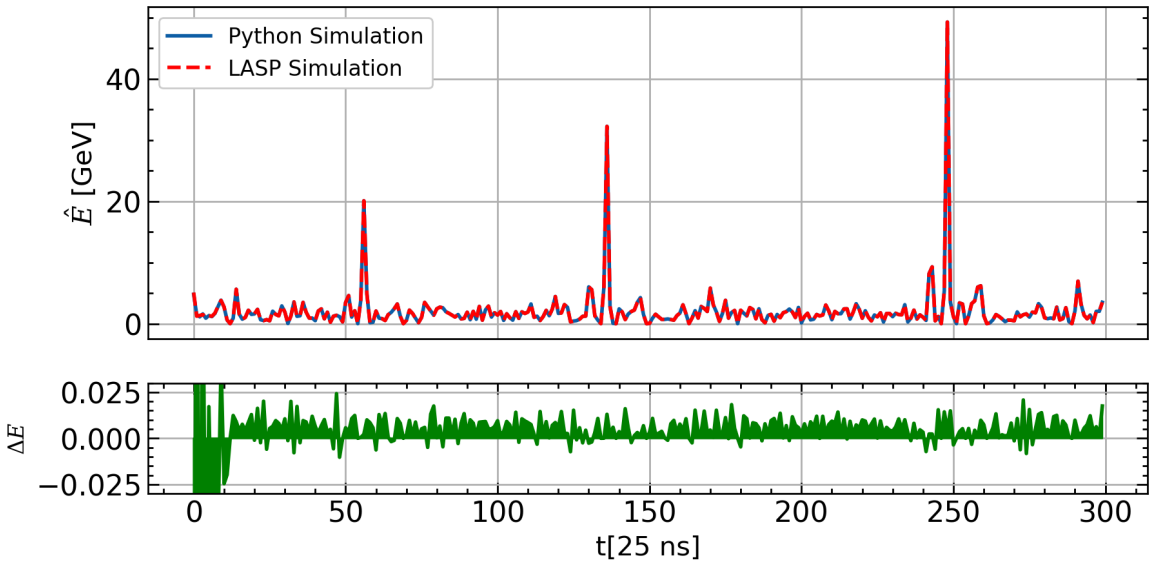


Figure 5.2: Output of CNN in python (floating point network weights and input) and ModelSim (fixed point network weights and input) and relative difference between the two. The bit configuration used was $B_W = 18$, $B_W^{(f)} = 14$, $B_I = 15$, $B_I^{(f)} = 8$.

A histogram of $\Delta E = \hat{E}^{(p)} - \hat{E}^{(v)}$ during signal events is shown in Figure 5.3. The width of the histogram can be used to estimate the error induced by rounding floating point network weights to fixed point.

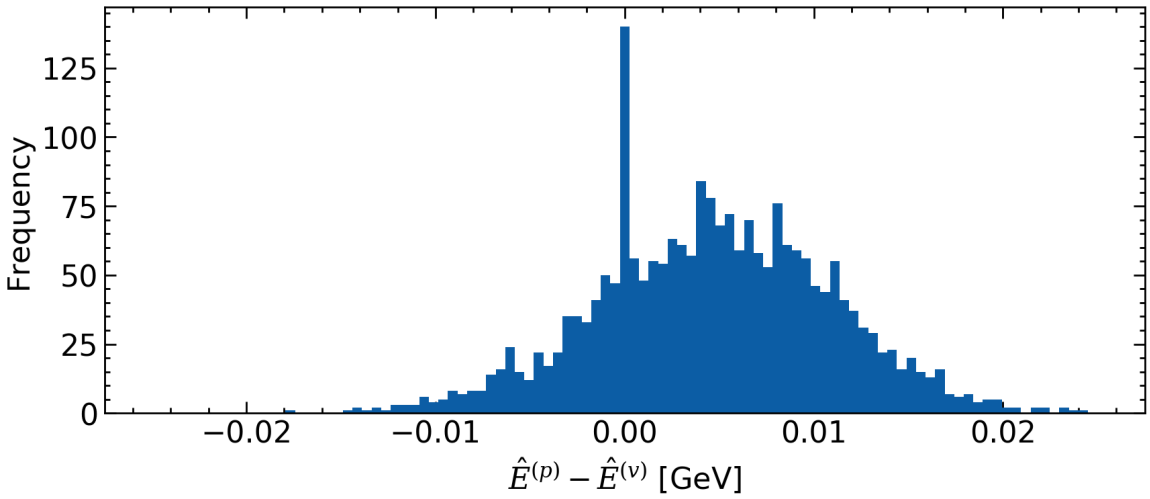


Figure 5.3: Histogram of residuals between python and ModelSim neural network energy predictions of Figure 5.2. The mean is 0.00462 ± 0.00014 GeV and the standard deviation is 0.00638 ± 0.00017 GeV.

The standard deviation of the residuals, defined here as $\delta_{\hat{E}}$, can be interpreted as an additional error in the energy reconstruction process. In Figure 4.10, for example, it implies that each prediction of $\hat{E}$ has an additional error term, and hence the computed RMSE has additional error. From Equation A.1, the error on the RMSE is given by $\delta_{\text{RMSE}} = \delta_{\hat{E}} \approx 0.006$ GeV. As seen in Figure 4.22, this additional error is negligible when the CNN and OF RMSE values are sufficiently spaced apart. However, when the OF vs. CNN RMSE metrics are close together (such as the cell $\eta = 2.25$ in HEC 2), the additional error may nullify the advantage of using a CNN due to an increased uncertainty in the network predictions. It is useful to examine ways to minimize this error.

One obvious way to minimize the error is by optimizing the usage of bits in storing the neural network weights and input sequence. For the input sequence, since all energies range between 0-50 GeV it is a safe assumption that 7 bits (which can store integer values ranging from -2^6 to 2^6) will be sufficient to store the integer portion of the number. As of writing this thesis, the maximum number of possible bits used to store the input sequence is 15, leaving 8 bits left to store the fractional bits of the input sequence. As for the neural network weights, it is sufficient to store the integer portion of the neural network weights using 4 bits (which can store integer

values from -2^3 to 2^3) since the weights and biases are small. This leaves up to 14 bits to store the decimal bits of the network weights. In Figure 5.4, the error on $\hat{E}$ is shown as a function of both the number of fractional bits used to represent the weights, and the number of fractional bits used to represent the input sequence.

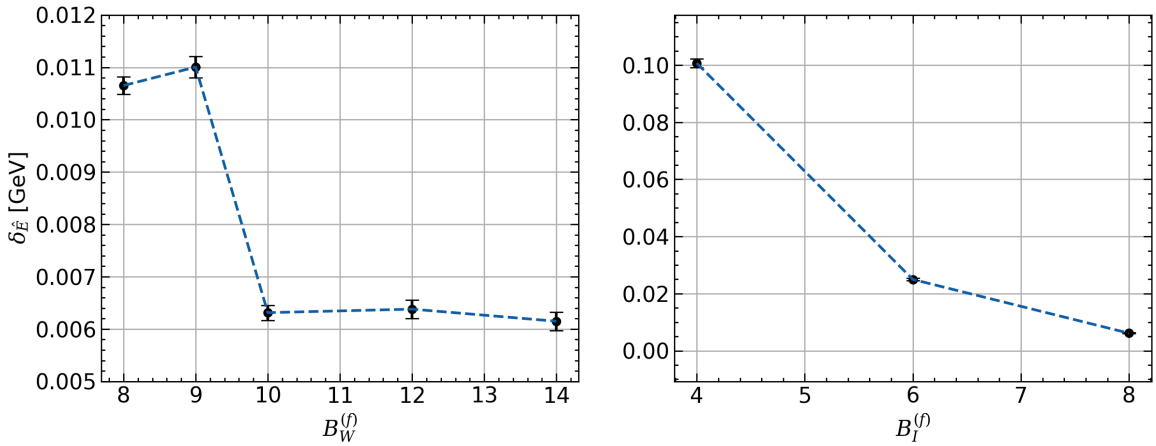


Figure 5.4: Error on $\hat{E}$, specified as $\delta_{\hat{E}}$, as a function of weight fractional bits (left) and input fractional bits (right). The left bit configuration is $B_W = 18$, $B_I = 15$, and $B_I^{(f)} = 8$. The right bit configuration is $B_W = 18$, $B_W^{(f)} = 14$, and $B_I = 15$

The plots of Figure 5.4 suggest that the limiting factor in reducing $\delta_{\hat{E}}$ is currently $B_I^{(f)}$. As 7 bits must be allocated to storing the integer portion of the input sequence, the maximum possible value of $B_I^{(f)}$ is 8, shown as the right-most point on the right plot. An increase in B_I would allow $B_I^{(f)}$ to be increased past 8. Based on the shape of the right plot, this may lead to a decrease in $\delta_{\hat{E}}$. As shown on the left plot, corresponding to $B_I^{(f)} = 8$, any increase in $B_W^{(f)}$ past 10 does not yield any improvement in $\delta_{\hat{E}}$ It is worth noting that if B_I were to be sufficiently increased, B_W may then become the limiting factor in reducing $\delta_{\hat{E}}$. When deciding whether or not to allocate more bits to the input sequence or the neural network weights, the limiting factor should be taken into account.

It is also worth noting that given a strict maximum of $B_I = 15$, there is no reason to have $B_W^{(f)}$ greater than 10, as seen on the left plot of Figure 5.4. A sufficient configuration would have $B_W^{(f)} = 10$ and $B_W = 14$, as only 4 bits are needed to store the integer portion of the neural network weights.

Chapter 6

Conclusion

The optimal filter technique and machine learning were compared as signal processing techniques to reconstruct the energy of detected particles in the HEC subsystem of the ATLAS liquid argon calorimeter. The machine learning models outperformed optimal filter in most calorimeter cells; energy resolution was improved by up to 40% and accurate predictions were made at all energies. This was largely attainable due to a unique loss function established in this thesis. Furthermore, and contrary to the optimal filter, machine learning models were found to make reasonable energy predictions when two or more particles were detected within a short period of time: these conditions may be present at the LHC during future upgrades. The simulation of the machine learning on ATLAS hardware was accomplished using the ATLAS LASP dacore software package; only a negligible decrease in model performance was observed on the hardware.

The most useful portion of this thesis for future work is likely the derivation and application of the loss function discussed in Section 4.3. Future plans to use machine learning techniques for energy reconstruction while retaining sufficient accuracy will likely benefit from the derivation. It should be additionally noted that such a loss function could potentially be improved upon, leading to a more reliable training procedure, and hence further improvements in accuracy.

Appendix A

Additional Information

A.1 Error-Propagation in the MSE

Suppose the error on each $\hat{X}_i$ is $\delta_{\hat{X}}$ and the error is needed for RMSE $= \sqrt{\sum_{i=1}^{n}(\hat{X}_i - X_i)^2}$. It is more useful to write sum as $\sum_{i=1}^{n} {R_i}^2$, where $R_i = \hat{X}_i - X_i$. It can then be identified that $\delta_{R_i} = \delta_{\hat{X}}$ and

$$\delta_{R_i^2} = 2|R_i|\delta_{\hat{X}}$$

Thus the error on the MSE is

$$\begin{aligned}
\delta_{\text{MSE}} &= \delta_{\sum_i R_i^2} \\
&= \left(\sum_i 2|R_i|\delta_{\hat{X}}\right)^{1/2} \\
&= 2\delta_{\hat{X}}\left(\sum_i R_i^2\right)^{1/2} \\
&= 2\delta_{\hat{X}} \cdot \text{RMSE}
\end{aligned}$$

Furthermore, in terms of the RMSE

$$\begin{aligned}\delta_{\mathrm{RMSE}} &= \delta_{\mathrm{MSE}^{1/2}} \\ &= \frac{1}{2}\mathrm{MSE}^{1/2} \cdot \frac{2\delta_{\hat{X}}\mathrm{RMSE}}{\mathrm{MSE}} \\ &= \delta_{\hat{X}}\end{aligned} \tag{A.1}$$

A.2 Log Scaled Plots

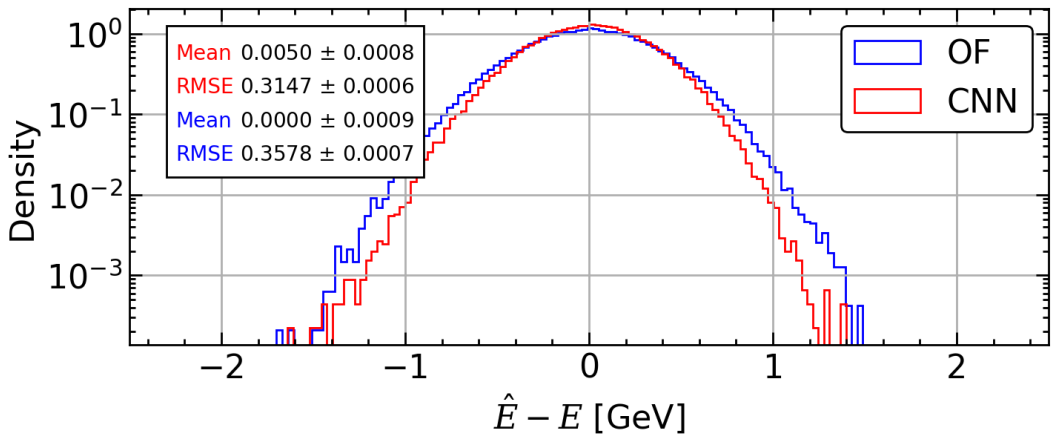


Figure A.1: Plot from Figure 4.10 with logarithmically scaled y axis.

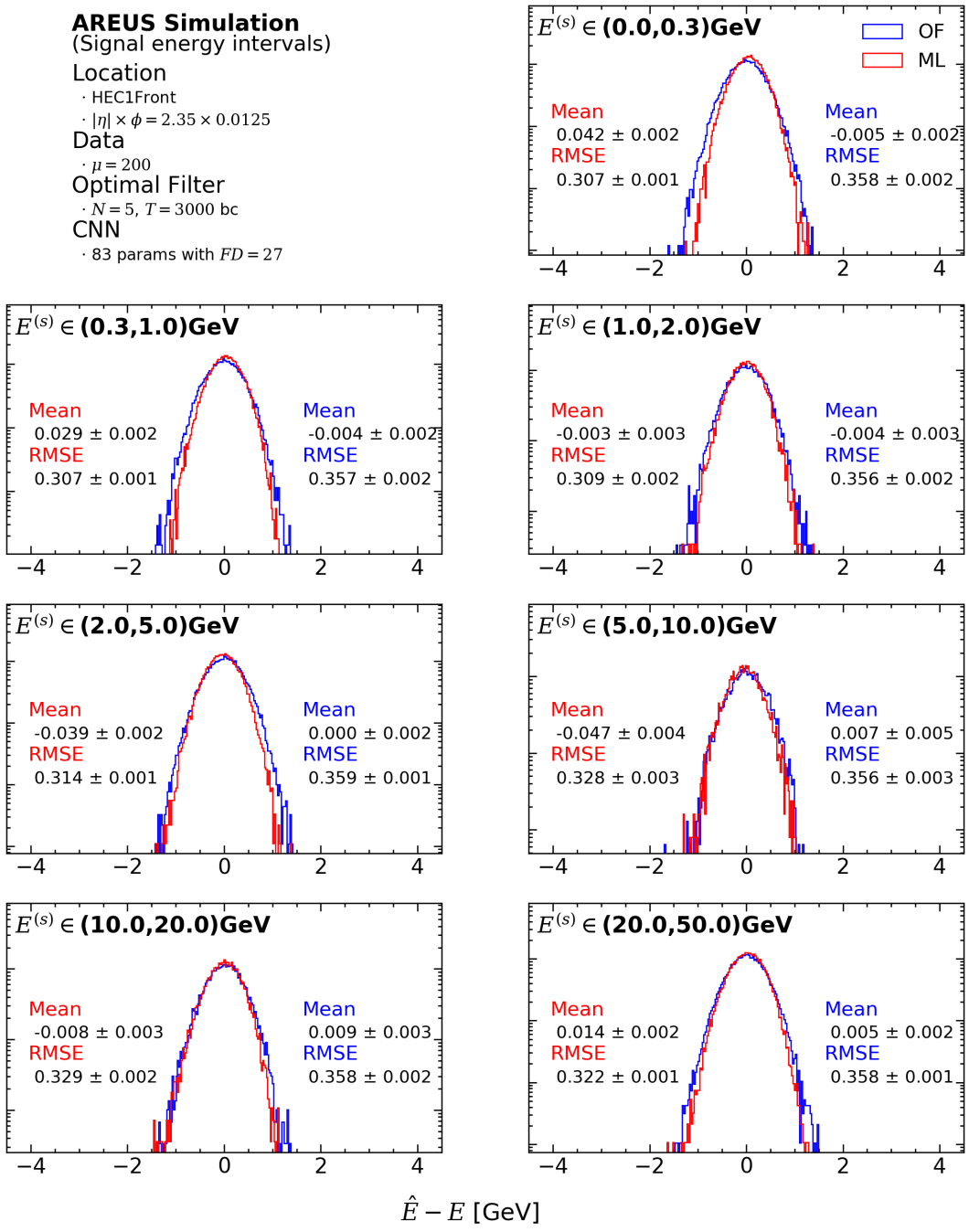


Figure A.2: Plot from Figure 4.11 with logarithmically scaled y axis.

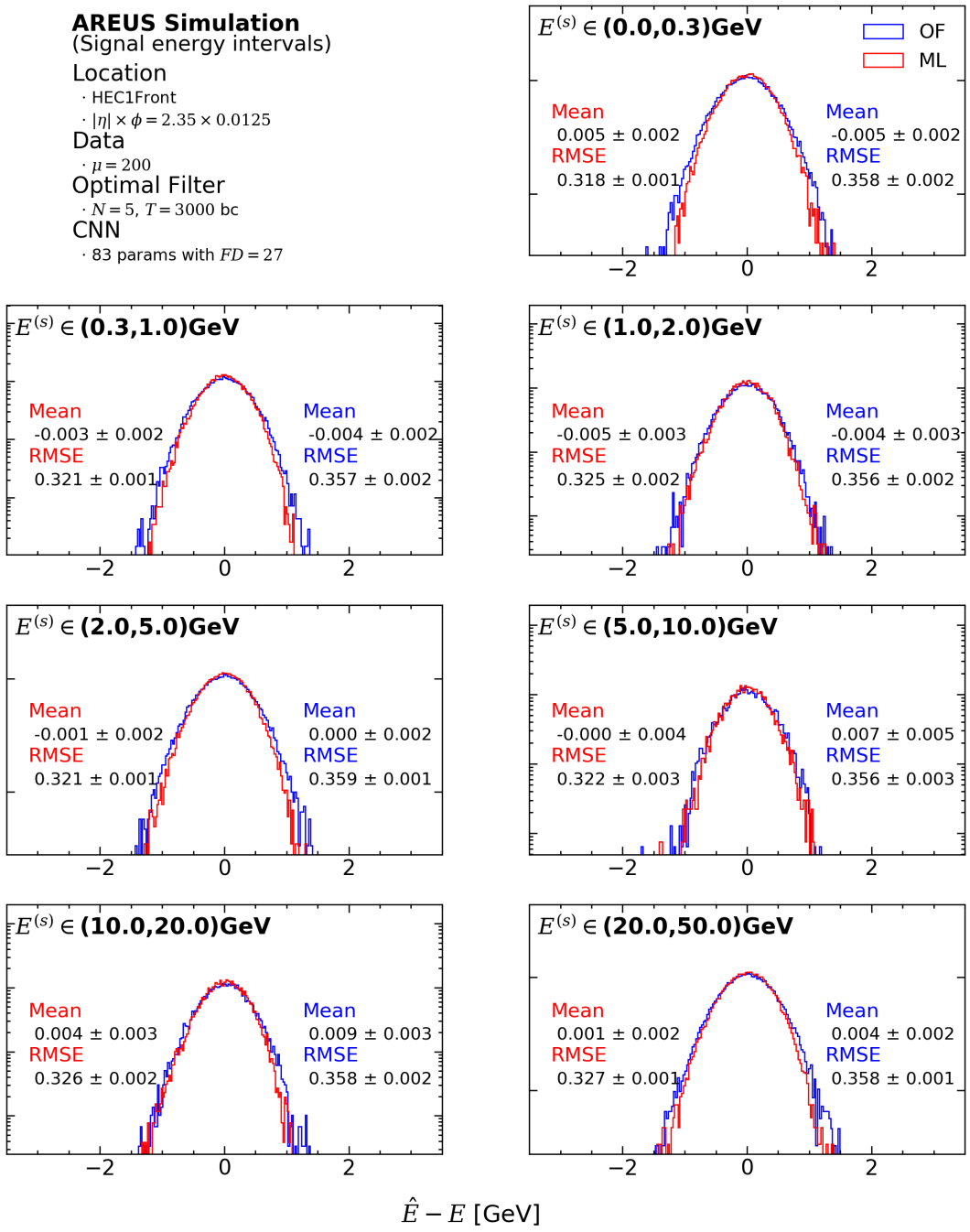


Figure A.3: Plot from Figure 4.18 with logarithmically scaled y axis.

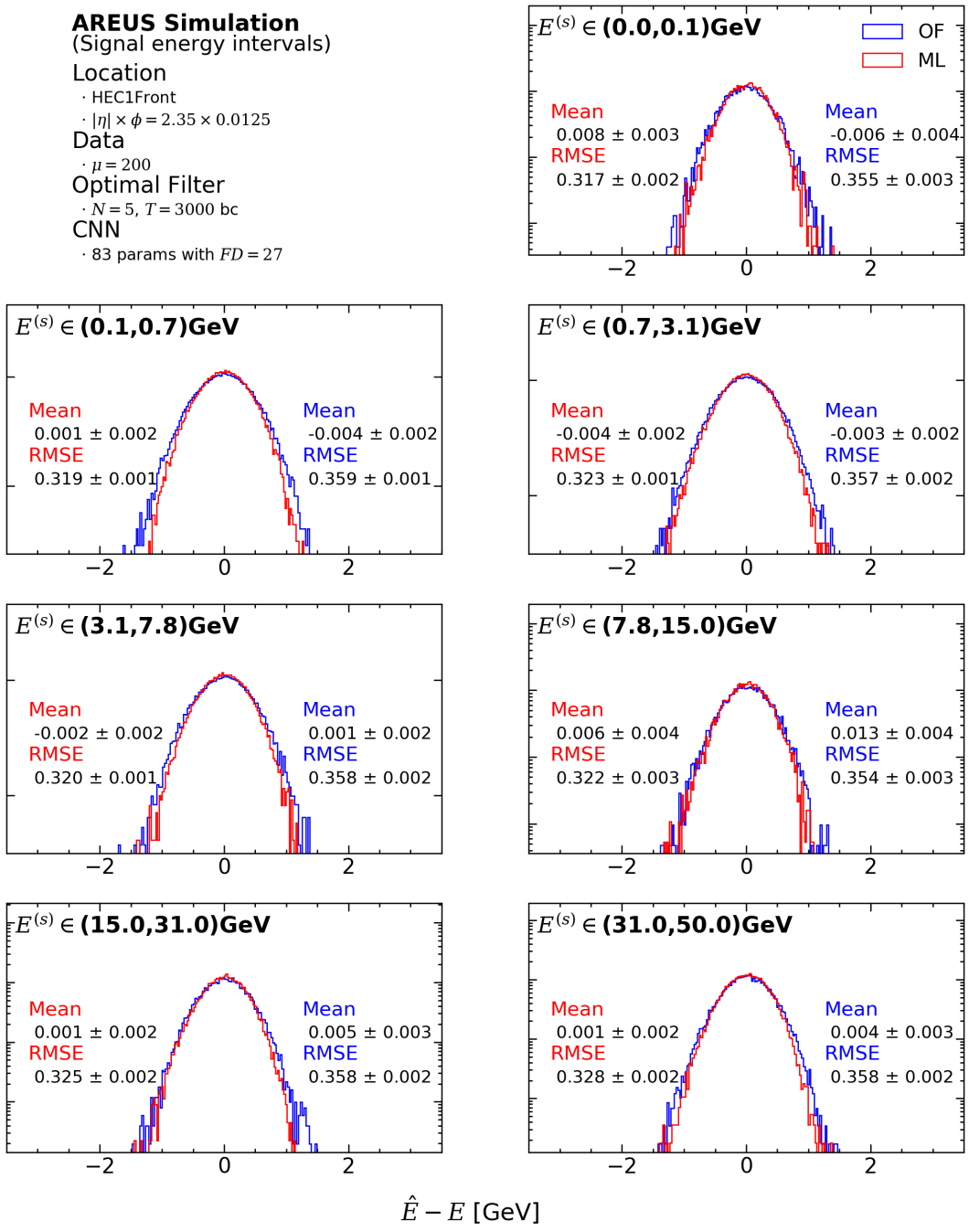


Figure A.4: Plot from Figure 4.19 with logarithmically scaled y axis.

Bibliography

[1] Lyndon Evans and Philip Bryant. LHC machine. *Journal of Instrumentation*, 3(08):S08001–S08001, Aug 2008.

[2] The ATLAS Collaboration. The ATLAS experiment at the CERN large hadron collider. *Journal of Instrumentation*, 3(08):S08003–S08003, Aug 2008.

[3] The CMS Collaboration. The CMS experiment at the CERN LHC. *Journal of Instrumentation*, 3(08):S08004–S08004, Aug 2008.

[4] The ALICE Collaboration. The ALICE experiment at the CERN LHC. *Journal of Instrumentation*, 3(08):S08002–S08002, Aug 2008.

[5] The LHCb Collaboration. The LHCb detector at the LHC. *Journal of Instrumentation*, 3(08):S08005–S08005, Aug 2008.

[6] Julie Haffner. The CERN accelerator complex. Complexe des accélérateurs du CERN. Oct 2013. General Photo.

[7] K.A. Olive. Review of particle physics. *Chinese Physics C*, 40(10):100001, Oct 2016.

[8] Eric Torrence. Luminosity Public Results Run 2. Jul 2020.

[9] G Apollinari, I Béjar Alonso, O Brüning, M Lamont, and L Rossi. *High-Luminosity Large Hadron Collider (HL-LHC): Preliminary Design Report*. CERN Yellow Reports: Monographs. CERN, Geneva, 2015.

[10] G. Aad et al. The ATLAS Experiment at the CERN Large Hadron Collider. *JINST*, 3:S08003, 2008.

[11] G. Aad, T. Abajyan, B. Abbott, J. Abdallah, S. Abdel Khalek, A.A. Abdelalim, O. Abdinov, R. Aben, B. Abi, M. Abolins, and et al. Observation of a new

particle in the search for the standard model higgs boson with the atlas detector at the lhc. *Physics Letters B*, 716(1):1–29, Sep 2012.

[12] Huaqiao Zhang and. The ATLAS liquid argon calorimeter: Overview and performance. *Journal of Physics: Conference Series*, 293:012044, Apr 2011.

[13] Morad Aaboud et al. Operation and performance of the ATLAS Tile Calorimeter in Run 1. *Eur. Phys. J. C*, 78(12):987, 2018.

[14] *ATLAS muon spectrometer: Technical Design Report.* Technical design report. ATLAS. CERN, Geneva, 1997.

[15] M (CERN) Aleksa, W (Pittsburgh) Cleland, Y (Tokyo) Enari, M (Victoria) Fincke-Keeler, L (CERN) Hervas, F (BNL) Lanni, S (Oregon) Majewski, C (Victoria) Marino, and I (LAPP) Wingerter-Seez. ATLAS Liquid Argon Calorimeter Phase-I Upgrade Technical Design Report. Technical Report CERN-LHCC-2013-017. ATLAS-TDR-022, Sep 2013. Final version presented to December 2013 LHCC.

[16] Peter Vankov and ATLAS Collaboration. ATLAS Future Upgrade. Technical Report ATL-UPGRADE-PROC-2016-003, CERN, Geneva, Jun 2016.

[17] Leonid Kurchaninov. ATLAS HEC Note-109: Modeling of the HEC Electronics Chain. Technical report, Max Planck Institute for Physics, Munich, Mar 2001.

[18] J. Thomas, D. A. Imel, and S. Biller. Statistics of charge collection in liquid argon and liquid xenon. *Phys. Rev. A*, 38:5793–5800, Dec 1988.

[19] Nico Madysa. AREUS: A Software Framework for ATLAS Readout Electronics Upgrade Simulation. *EPJ Web Conf.*, 214:02006, 2019.

[20] W.E. Cleland and E.G. Stern. Signal processing considerations for liquid ionization calorimeters in a high rate environment. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 338(2):467 – 497, 1994.

[21] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio, 2016.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

[23] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.

[24] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[25] M (CERN) Aleksa, W (Pittsburgh) Cleland, Y (Tokyo) Enari, M (Victoria) Fincke-Keeler, L (CERN) Hervas, F (BNL) Lanni, S (Oregon) Majewski, C (Victoria) Marino, and I (LAPP) Wingerter-Seez. ATLAS Liquid Argon Calorimeter Phase-I Upgrade Technical Design Report. Technical Report CERN-LHCC-2013-017. ATLAS-TDR-022, Sep 2013. Final version presented to December 2013 LHCC.