Studies of machine learning for event reconstruction in the SNO+ detector and electronic noise removal in p-type point contact high purity germanium detectors

by

Mark R. Anderson

A thesis submitted to the
Department of Physics, Engineering Physics & Astronomy
in conformity with the requirements for
the degree of Doctor of Philosophy

Queen's University
Kingston, Ontario, Canada
July 2024

Abstract

This dissertation presents two distinct topics. Both focus on the development and application of neural networks and deep learning-based methods to rare event searches in physics, specifically neutrinoless double-beta decay. In the first project, a new method for event vertex reconstruction is developed for SNO+ – a large-scale, liquid scintillator-based, multi-purpose neutrino experiment located at SNOLAB in Sudbury, Ontario, Canada. Several studies are conducted to demonstrate its performance in comparison to traditional maximum likelihood reconstruction techniques, as well as its potential to increase the sensitivity of SNO+ to neutrinoless double-beta decay. In the second project, a deep fully convolutional autoencoder is developed and applied to denoise pulses collected from a p-type point contact high purity germanium detector located at Queen's University in Kingston, Ontario, Canada and similar to the germanium detectors used in the arrays of large-scale experiments. It is shown through multiple analyses that denoising using these methods preserves the underlying pulse shape while simultaneously allowing for improvements in the energy resolution and background discrimination power in some circumstances.

Detection of the hypothetical neutrinoless double-beta decay could answer long-standing questions in physics and provide a better understanding of the Universe. As such, numerous experiments across the world are running, or under development, to search for this process. While the tools introduced here are applied to a particular liquid scintillator detector and p-type point contact germanium detector, they are broadly applicable to other experimental setups and detection technologies in addition to the specific ones utilized for each project. Furthermore, these tools can be employed to improve the sensitivity of experiments searching for other rare events, such as dark matter, using similar principles. The flexibility and straightforward transfer of these methods are discussed and some ongoing and future work is highlighted. This research is thus relevant both to and beyond the entire rare event search community and has the potential to widely improve analysis techniques, especially in light of the growing size and rates of data collection from modern particle physics experiments.

Acknowledgements

It is interesting to look back and think about how many people played a role in my life throughout the completion of this degree. A lot has happened since I began as a lowly master's student, and there are far too many people who have had a positive impact on me to name them all. I will instead limit myself to those who have been the most crucial in my success as a graduate student.

I would firstly like to thank my supervisor, Ryan Martin. I first worked with Ryan as an undergraduate summer student nearly a decade ago, and my exceptional experience there prompted me to continue my graduate studies at Queen's under his supervision. I am extremely grateful for his mentorship, insights, advice, and ability to explain complicated topics in a simple and clear way. For his willingness to help even during the busiest of times, I am particularly appreciative. I have thoroughly enjoyed our various discussions, which range from physics, to software, to the general state of the world. It would be an understatement to say that I have learned a lot from Ryan, and it is certainly not limited to particle astrophysics.

I would like to acknowledge all past and current members of the SNO+ group at Queen's for making my experience so great. Thank you to Mark Chen, Alex Wright, Peter Skensved, and Art McDonald as the senior scientists for the help and many insightful interactions over the years. As well, thank you to the numerous undergraduate summer students, fellow graduate students, and postdoctoral researchers who have come and gone (or are still around, as it may be).

I extend the above acknowledgement to the entire SNO+ collaboration. I have gotten to interact and work with many great scientists and researchers of all experience levels at many institutions around the world. I will not name them all here, but I would like to mention two people in particular. A special thanks goes to Jeff Tseng at the University of Oxford, who I have had the pleasure of working with and learning from on the software side of the SNO+ experiment. In addition to the many things I have learned from Jeff, I am extremely grateful for our recent discussions about life after the PhD. Regardless of the exact topic, our conversations are always interesting. My second special thanks goes to Mark Ward at SNOLAB. The network group at one point consisted of only us, and I appreciate the humour that Mark was able to provide in our many discussions regarding urgent problems that always managed to come up at the most inopportune of times. Our Slack thread is very long, but contains lots of useful information which I hope eventually all winds up in internally available documentation for the collaboration.

I would also like to thank everyone who was (or still is) a part of the other research group of which I was a member, GeRMLab[1]. Though local to Queen's with only several active members at any given time, my work here became a major component of my thesis. I was also able to spend some time on exciting and successful side projects resulting from the pandemic. Being a part of both a large international collaboration

[1]There are a few acronyms and abbreviations in here with different meanings that capture the nature of the work done by this group, including but not limited to germanium (detectors), Ryan Martin's lab, and machine learning!

like SNO+ and a comparatively smaller local group like GeRMLab allowed me to experience both worlds at once. I now understand many of the benefits and downsides of each, and was often able to apply skills and lessons learned from the people in one group to the other.

I am also grateful for the numerous friends I have made at Queen's during our overlapping tenures at this university. Some of these friends are also colleagues and thus covered in the acknowledgements above in addition to here. Many good times were had at our regular mandatory work meetings at the Mansion, late-night cleaning sessions in Stirling Hall, all-you-can-eat sushi (and other) outings, and hikes around the Kingston area. Particular thanks goes to Ben Tam, Brian Krar, Ian Lam, Yan Liu, Steph Walton, Chloe Lefebvre, Colin Moore, Anthony Allega, Gary Sweeney, Serena Riccetto, Debbie Morris, Jasmine Corning, Jamie Grove, Hector Hawley Herrara, Tianai Ye, and Jean-Marie Coquillat. To those listed here and many others not named directly, you have all made my time at Queen's wonderful and memorable.

Thank you also to my many friends outside of my academic life at Queen's. You are all important to me, and despite the physical distance that separates us now, I am glad that we are able to keep in such good contact. Particular thanks goes to my good friends and past housemates from the engineering program at Queen's, and to my large friend group back home in Vancouver (some of whom, like me, no longer live there). Once again, there are too many to name, but for those reading this, you know who you are.

A very special thanks goes to my immediate family. I would not be where I am today without my mother and father, Louise and Russ, who have provided me with endless support over the years and have given me a wonderful life. Though I moved across the country from Vancouver for university and have lived out here since, my home is always where they are. Thank you also to my sister Nicole, as well as her partner Jonny and their dog Marvin, both for the support and for always making my trips home that much more fun. I am furthermore extremely grateful for my grandparents – Herb, Shelagh, Ron, and Mavis – along with their role in my life, the endless support that they too have provided, and the keen interest they have always taken in my schooling. Though only my grandma, Shelagh, is still alive today, my granddad, Herb, was around for the majority of my degree. I fondly remember his curiosity in my work and him asking what I would do after I finally finished the PhD!

Family is not just those who are directly related to you by blood or law, and I consider my family to have grown since I began as a graduate student. I am very thankful to my partner's family, who I now consider as a part of my own, for the kindness and hospitality they have selflessly provided over many years. Having a family away from home made an immeasurable difference in my life, especially during the pandemic. I have always felt so welcomed by them. Lastly, a massive thank you to my partner, Sara. I would not be where I am today without you. You have supported me since I met you, nearly the entire duration of this degree, and I cannot imagine how much harder this journey would have been without you. Especially throughout the months of writing culminating in this thesis and me going slightly insane along the way, you have provided me with so much help and inspiration that I cannot even begin to describe. You have made me a better person and you motivate me to succeed. As you now work towards the completion of your own degree, I hope that I can provide the same level of support that you have given me. I look forward to everything that is to come.

Statement of Originality

This thesis is written entirely by the author. Some portions of the thesis draw from and largely mirror the author's publication in [1]. Although the author completed the vast majority of the analysis and writing for the publication, six others (Vasundhara Basu, Ryan D. Martin, Charlotte Z. Reed, Noah J. Rowe, Mehdi Shafiee, Tianai Ye) contributed to the publication in various ways, including internally reviewing it before submission to the journal. Details of specific sections that are taken and adapted from the publication in [1] are noted explicitly below and in their respective chapters.

The work described throughout this thesis is built off of the extensive achievements from the particle astrophysics community at large. Of particular note, the author is a member of the international SNO+ collaboration. Naturally, the author's work in relation to the SNO+ detector would not be possible without the contributions of all past and current collaborators, both on SNO+ and the original Sudbury Neutrino Observatory. Although not a part of a formal collaboration, the high purity germanium detector analysis similarly makes extensive use of prior work done by others. Thus, this thesis, while ultimately made possible by enormous local and global efforts, focuses on the important advances made by the author. With that in mind, a more detailed description of the author's contributions to each chapter follows.

Chapter 1 provides background on neutrinos and motivates the search for neutrinoless double-beta decay. This consists of a literature review of the history behind the discovery of neutrinos and current experimental results at the time of writing. It thus does not contain any new or original ideas.

Chapter 2 discusses the two detectors relevant for this thesis: the SNO+ detector located at SNOLAB in Sudbury, Ontario, Canada and a high purity germanium detector located at Queen's University in Kingston, Ontario, Canada. The construction of both detectors were completed before the author had begun work on either, and most of their crucial operational components were already well-established. This chapter thus does not contain any new or original ideas. Furthermore, it should be emphasized that the results of this thesis take advantage of simulations and data collected from these detectors, and thus depend on the work of many others. Being a member of the SNO+ collaboration, the author contributed in various ways to the operation of the SNO+ experiment, many of which are not pertinent to this thesis. Of relevance, the author had an extensive role in maintaining and upgrading the simulation and analysis software described in Section 2.1.6. As well, though the high purity germanium detector was not built and configured by the author, the author did contribute to the simulation and analysis software described in Section 2.2.4.

Chapter 3 provides a background on machine learning with a focus on neural networks, and as such contains no original ideas. This chapter consists of information relevant to understand the analyses of later chapters, written and summarized to a sufficient level of detail by the author. The text in Section 3.4 is adapted from the explanation in [1] and thus contains a substantial text overlap with the corresponding section in the publication.

The remainder of the thesis (except for the conclusion and endmatter) is divided into Parts I and II.

Part I

Chapter 4 describes event reconstruction in the SNO+ detector. Section 4.1 is an overview of the traditional likelihood-based reconstruction procedure, developed prior to the author's membership on the collaboration, and thus is based on the work of others on the SNO+ experiment. Section 4.2 focuses on the contributions of the author in developing a new event reconstruction method for the SNO+ experiment using deep neural networks.

Chapter 5 presents comparisons and studies of the model developed in Section 4.2, and as such is original work. In particular, Sections 5.1 to 5.3 present various results of applying and extending the author's methods. Section 5.4 summarizes these results, highlights implications for rare event search experiments, and lists ongoing and potential future work. Though no new analysis is presented in this section, many of the ideas presented (and initial feasibility studies, where applicable) are the author's, and where this is not the case, it is noted clearly.

Part II

Chapter 6 describes pulse denoising in the high purity germanium detector at Queen's University. Section 6.1 provides an overview of various well-established traditional denoising methods and thus is not original work. Section 6.2 describes the neural network denoising algorithm developed by the author, focusing on new contributions. Both sections draw heavily from the author's publication in [1], though modifications are frequently made to the text in order for the thesis to read properly. Section, figure, table, equation, and citation numbers have also been updated to match the format of the thesis.

Chapter 7 presents evaluations and studies of the model developed in Chapter 6. It largely mirrors the results presented in [1] and follows nearly the same structure. Some additional results are presented in this chapter that are not published in [1], including the analysis described in Section 7.1.4. As with Chapter 6, minor modifications to the text drawn from [1] have been made to fit with the thesis, and section, figure, table, equation, and citation numbers have also been updated for consistency. Section 7.3 summarizes this part, highlighting experimental implications for rare event search experiments using germanium detectors and demonstrating the broad applicability of the techniques developed with concrete examples. It contains no new analysis and is not published in [1], though some ideas listed there are restated here as well.

This thesis ends with Chapter 8, which summarizes both the results and various extensions of the research developed and presented throughout this thesis that are already being applied. As information is only summarized from all previous chapters, it contains no new or original work.

Table of Contents

List of Tables

List of Figures

List of Acronyms

$0\nu\beta\beta$	neutrinoless double-beta (decay)
$2\nu\beta\beta$	two-neutrino double-beta (decay)
ADC	analogue-to-digital converter
A/E	(maximum) amplitude (of the current pulse) over energy
AMoRE	Advanced Mo-based Rare Process Experiment
API	application programming interface
AUC	area under the curve (in reference to the ROC curve)
AV	acrylic vessel
β	beta (historical term for the electron)
$\beta\beta$	double-beta (decay)
BHT	butylated hydroxytoluene
BisMSB	1,4-bis(2-methylstyryl)benzene
CANDLES	Calcium Fluoride for the Study of Neutrinos and Dark Matters by Low Energy Spectrometer
CL	confidence level
CMOS	complementary metal-oxide-semiconductor
CNN	convolutional neural network
COBRA	Cadmium Zinc Telluride 0 Neutrino Double-beta Research Apparatus
CUORE	Cryogenic Underground Observatory for Rare Events
CUPID	CUORE Upgrade with Particle Identification
DAQ	data acquisition

DDA	N,N-dimethyldodecylamine
DFT	discrete Fourier transform
ELEGANT	Electron Gamma-ray Neutrino Telescope
EXO	Enriched Xenon Observatory
FEC	front end card
FFT	fast Fourier transform
FV	fiducial volume
FWHM	full width at half the maximum
GERDA	Germanium Detector Array
GNN	graph neural network
GPU	graphics processing unit
HDF	Hierarchical Data Format
HPGe	high purity germanium
ICPC	inverted coaxial point contact
KamLAND	Kamioka Liquid Scintillator Antineutrino Detector
KamLAND-Zen	KamLAND Zero-Neutrino Double-beta Decay
KATRIN	Karlsruhe Tritium Neutrino
L1	LASSO (regularization/regression)
L2	ridge (regularization/regression)
LAB	linear alkylbenzene
LASSO	least absolute shrinkage and selection operator
LEGEND	Large Enriched Germanium for Neutrinoless Double-beta Decay
LHC	Large Hadron Collider
LNGS	Laboratori Nazionali del Gran Sasso
LSTM	long short-term memory
MC	Monte Carlo
MicroBooNE	Micro Booster Neutrino Experiment
MJD	MAJORANA DEMONSTRATOR

ML	machine learning
NDF	number of degrees of freedom
NEMO	Neutrino Ettore Majorana Observatory
NEWS-G	New Experiment with Spheres – Gas
nEXO	next EXO
NEXT	Neutrino Experiment with a Xenon TPC
Nhits	number of PMT hits (in an event)
NN	neural network
NOνA	NuMI Off-axis ν_e Appearance
NuMI	Neutrinos at the Main Injector
PDF	probability density function
PL	position likelihood
PLS	piecewise linear smoothed
PMNS	Pontecorvo-Maki-Nakagawa-Sakata (mixing matrix)
PMT	photomultiplier tube
PMTIC	PMT interface card
PPC	p-type point contact
PPO	2,5-diphenyloxazole
PSUP	PMT support structure
QHL	charge high gain, long integration window
QHS	charge high gain, short integration window
QLX	charge low gain, short or long integration window
QUADIS	Queen's University Analysis of Data in Spheres
RAT	Reactor Analysis Toolkit
ReLU	rectified linear unit
RENO	Reactor Experiment for Neutrino Oscillation
RMS	root mean square
ROC	receiver operating characteristic (curve)

ROI	region of interest
SNEWS	SuperNova Early Warning System
SNO	Sudbury Neutrino Observatory
SSIM	structural similarity index measure
STFT	short-time Fourier transform
SuperNEMO	Super NEMO
TAC	time-to-amplitude converter
TPC	time projection chamber
TPU	tensor processing unit
XSNOED	X-Windows SNO(+) Event Display

Chapter 1

Introduction

This chapter is largely a review of the current status of neutrino physics and modern experimental procedures. It serves to motivate the analyses of this thesis. Sections 1.1 to 1.3 cover the basics of neutrino physics, from their discovery to the present day. Section 1.3, in particular, highlights the mysterious nature of the neutrino and how the detection of a hypothetical process called neutrinoless double-beta decay could answer long-standing questions in physics. Section 1.4 covers modern detection technologies and existing experiments searching for neutrinoless double-beta decay. This section also provides the context needed to understand the two detector technologies that are discussed in more detail in Chapter 2. Section 1.5 ties the previous sections together, providing a summary of the goal of this thesis in the context of the neutrino physics background given, and argues how machine learning can be used to help solve the most important problems in neutrino physics today.

1.1 Historical overview of neutrinos

In the early 1900s, it was thought that beta (β, historical term for the electron) decay was a two-body process. Previous studies of alpha and gamma decay showed that the energy distribution of the emitted particle was discrete. Naturally, it was expected that this would also be the case for β decay. However, multiple independent observations showed that the energy spectrum of the emitted electron was continuous [2–4]. These results implied that energy was not conserved in the β decay process. Furthermore, it was later shown that the electron always had less energy than would be expected in a two-body decay process [5], demonstrating that its energy has an upper bound. This ruled out some hypotheses to reconcile the unexpected observation.

In 1930, Wolfgang Pauli proposed a new particle – the neutrino – to explain the apparent violation of energy conservation in the two-body β decay model [6, 7]. To be compatible with conservation of charge and its lack of detection, this new particle would have to be electrically neutral and have a very small mass. In 1934, Enrico Fermi published a theory of β decay [8–10], which proposed that the neutrino (a term he coined based on its apparent properties) was created along with the electron in the decay of the neutron,

$$\mathrm{n} \longrightarrow \mathrm{p} + \mathrm{e}^{-} + \bar{\nu}_{\mathrm{e}}. \tag{1.1}$$

According to his proposition, this then-undetected neutrino carried away some fraction of the total energy released – the Q-value of the decay – and thereby explained the continuous and bounded β decay spectrum. Fermi's theory included a new coupling to explain the four-fermion interaction that is β decay, which preceded what is now known as the weak nuclear force or weak interaction. As implied by the name, the weak interaction is a short-range force with a field strength much less than the electromagnetic and strong[1] interactions. His proposal provided a theoretical framework that led to the prediction of numerous phenomena and justified many future experiments.

Since neutrinos interact only through the weak interaction, their first detection was not until 1956 by the Cowan-Reines neutrino experiment [11]. It was expected that nuclear reactors would produce a large flux of electron antineutrinos, which could be detected via the distinct signature of the inverse β decay reaction,

$$\bar{\nu}_e + p \longrightarrow n + e^+. \tag{1.2}$$

Clyde Cowan and Frederick Reines built a detector nearby a nuclear reactor and successfully demonstrated that despite having such limited interactions with matter, neutrinos were detectable. In the following years, it was shown that there was more than one type of neutrino, as had been expected from observations of muon decay and then later the discovery of the tau particle.

1.2 Neutrinos in the Standard Model

The neutrino is an elementary particle and is a fundamental building block of the Standard Model of physics. The Standard Model contains twelve fermions, or spin-½ particles. Each fermion has a corresponding antiparticle which has the opposite charge but is otherwise the same. Within the fermions, there are two families of particles: the quarks and the leptons. The former interact via the strong force, while the latter

[1]Although the strong interaction was not known at the time, with the discovery of the neutron and the knowledge that nuclei consist of protons and neutrons, it was understood that *something* had to hold nuclei together and overcome the repulsive electrostatic force of the constituent protons.

do not.

Both families contains three subgroups each consisting of two associated particles. In the lepton sector, the three subgroups are the electronic, muonic, and tauonic generations (frequently also referred to as the first, second, and third generations, which more broadly encompass the quark sector as well). As implied by their names, these subgroups consist of the electron (e^-), muon (μ^-), and tau (τ^-) particles, respectively.

There are also three “flavours” of neutrinos which correspond to the charged leptons, and thus belong to each of the generations mentioned above: the electron neutrino (ν_e), the muon neutrino (ν_μ), and the tau neutrino (ν_τ). In the Standard Model, the neutrino is electrically neutral and has a mass of zero. As with all fermions, each neutrino has a corresponding antiparticle, denoted by the complex conjugate symbol due to its lack of charge (e.g., $\bar{\nu}_e$). All leptons interact via the weak force[2], while only the charged leptons interact via the electromagnetic interaction. As the neutral neutrino then only undergoes the weak interaction, it very difficult to study, and is the reason that it took over twenty years after its initial proposal to be detected.

While the Standard Model has been extremely successful, it is also incomplete. The Standard Model fails to include, for example, gravity. It also fails to provide explanations for dark matter and the matter-antimatter asymmetry observed in the Universe. Perhaps most notably, it is now confirmed that the neutrino is massive, in contradiction with the Standard Model. While accommodations can be made to include the neutrino mass, it is not clear how this should be done. Further research is required to properly incorporate the neutrino mass term in the Standard Model and

[2]All leptons also interact via gravity, but as this is not accounted for in the Standard Model and since gravity is negligible at the subatomic scale relative to the other fundamental forces, it is ignored here.

to explain how the neutrino gets its mass.

1.3 Neutrinos beyond the Standard Model

An understanding of neutrino mass, and how neutrinos do not agree with the Standard Model, depends on neutrino oscillations – the ability of the neutrino to be observed as a different flavour than its initial state. This section introduces the observations which led to the development of the idea of oscillations, experimental confirmation of this phenomenon, and the present status of neutrino theory beyond the Standard Model.

1.3.1 Neutrino oscillations

1.3.1.1 The solar neutrino problem

As early as 1957, neutrino oscillations had been postulated [12], though this work considered oscillations between the neutrino and antineutrino. In 1962, neutrino oscillations were considered in the context of flavour mixing [13]. By 1968, the theory was beginning to gain traction and experimental setups were proposed to detect this then-hypothetical oscillation and lepton flavour nonconservation [14]. In 1969, a full theory of neutrino mixing had been developed in the two flavour case [15].

At around the same time period, the solar neutrino flux was measured by the Homestake experiment [16]. The Standard Solar Model – a broad framework to understand the energy production and properties in the Sun – was used to predict the expected neutrino flux that should be observed [17], after initially being proposed for this task in 1964 [18, 19]. The measured flux was significantly less than expected, casting doubts on both the experimental setup and the Standard Solar Model. Improvements, verification, and continued running of the Homestake detector supported the initial flux measurement and ruled out sources of detector error [20]. Other experiments [21–23] using completely different detection technologies further confirmed this deficiency, which was often referred to as the solar neutrino problem. In fact, the work in [14] and [15] predicted the solar neutrino problem before it was confirmed experimentally.

It took over forty years from the first detection of neutrinos to observe neutrino oscillations, which was done by the Super-Kamiokande experiment in 1998 [24]. The Sudbury Neutrino Observatory (SNO) experiment later confirmed that neutrinos have mass by demonstrating that neutrinos from the Sun change flavour as they propagate to the Earth [25]. These results, along with conclusively resolving the solar neutrino problem, provided the first experimental evidence that the Standard Model is incomplete.

1.3.1.2 Neutrino oscillation theory

Neutrinos oscillate between flavours because the observable eigenstates are superpositions of mass eigenstates. The flavour basis, of which the elements interact with the charged leptons, is not aligned with the mass basis. A neutrino flavour eigenstate ν_α, with $\alpha \in \{\mathrm{e}, \mu, \tau\}$, can be written in terms of the mass eigenstates ν_k, with $k \in \{1, 2, 3\}$, as

$$|\nu_\alpha\rangle = \sum_{k=1}^{3} U_{\alpha,k} |\nu_k\rangle, \tag{1.3}$$

where $U_{\alpha,k}$ is an element in U, the unitary mixing matrix. U is often called the Pontecorvo-Maki-Nakagawa-Sakata (PMNS) mixing matrix to recognize the contributions of those scientists in the formulation of neutrino oscillation theory [12–15]. U defines the relation between the flavour basis and mass basis in a vacuum.

The mass eigenstates are eigenstates of the Hamiltonian, and thus propagate through space and time. The solution to the Schrödinger equation is the plane wave; for mass states $|\nu_k\rangle$, the evolution over time t and distance $\vec{x}$ is given by

$$|\nu_k(t,\vec{x})\rangle = e^{-i(E_k t-\vec{p}_k\cdot\vec{x})}|\nu_k\rangle, \tag{1.4}$$

where $E_k = \sqrt{p_k^2+m_k^2}$ is the energy, $\vec{p}_k$ is the momentum with magnitude $p_k = |\vec{p}_k|$, and m_k is the mass. In Equation (1.4) and the remainder of this section, natural units are used such that $c = \hbar = 1$. By writing the time-evolved flavour state in terms of the mass eigenstates as in Equation (1.3), the probability of a neutrino with an initial flavour state α later being detected as a neutrino with flavour state β, such that $\alpha \neq \beta$, is given by

$$\begin{aligned}
P\left(\nu_\alpha \longrightarrow \nu_\beta\right) &= \left|\langle\nu_\beta(t,\vec{x})|\nu_\alpha\rangle\right|^2 && (1.5)\\
&= \left|\sum_{k=1}^{3}\sum_{l=1}^{3} U^*_{\beta,l}U_{\alpha,k}e^{-i(E_l t-\vec{p}_l\cdot\vec{x})}\langle\nu_l|\nu_k\rangle\right|^2 && (1.6)\\
&= \left|\sum_{k=1}^{3} U^*_{\beta,k}U_{\alpha,k}e^{-i(E_k t-\vec{p}_k\cdot\vec{x})}\right|^2 && (1.7)\\
&= \sum_{k=1}^{3} U^*_{\beta,k}U_{\alpha,k}e^{-i(E_k t-\vec{p}_k\cdot\vec{x})}\sum_{l=1}^{3} U_{\beta,l}U^*_{\alpha,l}e^{i(E_l t-\vec{p}_l\cdot\vec{x})} && (1.8)\\
&= \sum_{k=1}^{3}\sum_{l=1}^{3} U^*_{\beta,k}U_{\alpha,k}U_{\beta,l}U^*_{\alpha,l}e^{-i((E_k-E_l)t-(\vec{p}_k-\vec{p}_l)\cdot\vec{x})}. && (1.9)
\end{aligned}$$

Assuming that the neutrinos are relativistic, the approximation $p_k \gg m_k$ holds. Thus, the displacement $x = |\vec{x}|$ can be approximated as $x \simeq t$ and the momenta of the two states are roughly equal, $p_k \simeq p_l$. As well, given the tiny neutrino mass, $E_k \simeq p_k$ is used after the energy difference term is expanded. Equation (1.9) then becomes

$$P\left(\nu_\alpha \longrightarrow \nu_\beta\right) = \sum_{k=1}^{3}\sum_{l=1}^{3} U^*_{\beta,k} U_{\alpha,k} U_{\beta,l} U^*_{\alpha,l} \exp\left(-i\frac{\Delta m^2_{kl} x}{2E}\right). \tag{1.10}$$

Here, $\Delta m^2_{kl} = m^2_k - m^2_l$ is the squared mass difference, and E is the energy of the neutrino.

As can be seen in Equation (1.10), the dependence on Δm^2_{kl} means that oscillations require non-zero neutrino masses. More specifically, at least two of the three mass eigenstates must be non-zero and the masses must all be different.

1.3.2 Neutrino mass

Although it is now known that the neutrino is massive, much about the neutrino mass – both its absolute scale as well as how to incorporate the mass term into the Standard Model – remains poorly quantified or unknown.

1.3.2.1 Neutrino mass scale and known parameters

Upper bounds on the neutrino masses have been determined by direct measurements of tritium β decay, $^3\mathrm{H} \longrightarrow {}^3\mathrm{He} + \mathrm{e}^- + \bar{\nu}_\mathrm{e}$, and studying the β spectrum near the endpoint. The KATRIN (Karlsruhe Tritium Neutrino) experiment has placed the effective electron antineutrino mass at less than $0.8\,\mathrm{eV}/c^2$ (90 % CL) using this method [26]. Furthermore, cosmological observations limit the sum of the neutrino masses, $\sum_{k=1}^{3} m_k$, to be less than $0.12\,\mathrm{eV}/c^2$ (95 % CL) [27], although these bounds are heavily dependent on the model used. Oscillation experiments can place lower bounds on the neutrino masses by setting the lightest mass term to zero. These constraints demonstrate that although the neutrino mass is non-zero, it is orders of magnitude lower than the masses of even its charged lepton partners.

However, some of the mass splitting terms (or their magnitudes) have been measured, as have many of the elements of the PMNS mixing matrix. This provides some insight into the massive neutrino. To make further statements about the mass splitting terms, the neutrino mass state labels must be better defined relative to each other. The convention is that ν_1 makes up the largest portion of ν_e, $m_1 < m_2$, and $|\Delta m^2_{21}| < |\Delta m^2_{31}|$.

Δm^2_{21} is known from solar neutrino and reactor neutrino experiments. A long baseline experiment – KamLAND (Kamioka Liquid Scintillator Antineutrino Detector) – has also measured Δm^2_{21} [28]. Its value is $\Delta m^2_{21} = (7.53 \pm 0.18) \cdot 10^{-5}\,\text{eV}^2/c^4$ according to analyses of various experimental observations [29]. $|\Delta m^2_{31}|$ has been measured from atmospheric neutrino experiments and medium baseline experiments, albeit with relatively large uncertainties compared to the scale of $|\Delta m^2_{21}|$. As well, the sign of Δm^2_{31} is not known, and thus it is also not known whether $m_1 < m_3$ or $m_3 < m_1$. The former situation is referred to as the normal hierarchy, while the latter situation is referred to as the inverted hierarchy. A visual illustration of the two possible orderings is shown in Figure 1.1. In the normal hierarchy, the lightest neutrino mass eigenstate contains the largest proportion of ν_e.

Because of the large uncertainty on $|\Delta m^2_{31}|$, its magnitude is approximately equal to that of $|\Delta m^2_{32}|$. Fitted values put $|\Delta m^2_{32}| = (2.437 \pm 0.033) \cdot 10^{-3}\,\text{eV}^2/c^4$ for the

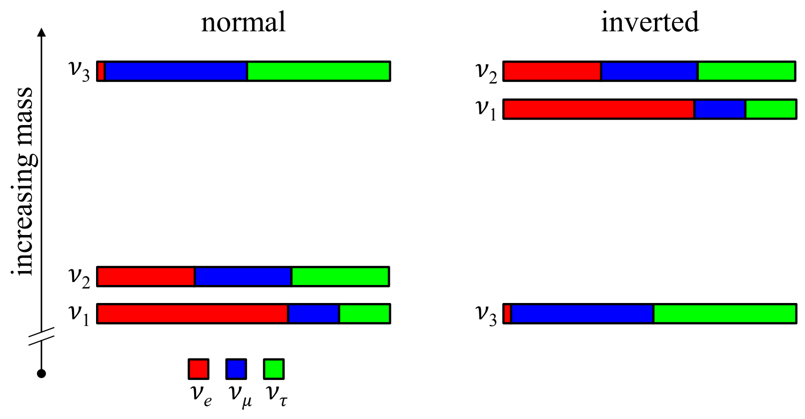


Figure 1.1: Visual illustration of the two possible neutrino mass hierarchies. The normal hierarchy is shown on the left, while the inverted hierarchy is shown on the right. Each colour represents a flavour eigenstate, with its size for each mass eigenstate corresponding to the mixing proportion. This proportion in turn corresponds to a PMNS mixing matrix element. Figure from [30].

normal hierarchy and $|\Delta m^2_{32}| = (-2.519 \pm 0.033) \cdot 10^{-3}\,\mathrm{eV}^2/c^4$ for the inverted hierarchy [29]. Combining the mass splitting measurements $|\Delta m^2_{32}|$ and $|\Delta m^2_{21}|$ constrains $\sum_{k=1}^{3} m_k$ to be greater than about $0.059\,\mathrm{eV}/c^2$.

1.3.2.2 Nature of the neutrino mass

Additionally, as previously mentioned, the detection of neutrino oscillations raises the question of how to incorporate the neutrino mass term into the Standard Model. The Standard Model predicts only left-handed neutrinos and right-handed antineutrinos, and so far, (anti)neutrinos have never been detected in the opposite helicity state [31]. This observation contributed to the prediction of massless neutrinos, which is now known to be incorrect.

Neutrino mass requires the inclusion of right-handed neutrinos and left-handed antineutrinos. The Standard Model can be most simply extended by assuming that neutrinos are Dirac particles and the Higgs mechanism is responsible for their masses, as is the case with other fermions. This requires the existence of right-handed neutrinos [29]. Such neutrinos would need to be "sterile," meaning that they only interact via gravity and none of the Standard Model forces, in order to be consistent with experimental evidence. As the neutrino is extremely small in comparison to all other fermions, this extension of the Standard Model means that the Higgs coupling must be extraordinarily small.

The Standard Model can also be extended to include a Majorana mass term [32], named after the physicist who proposed it [33,34]. In this formulation, some particles can be "Majorana" particles, meaning that the charge conjugation operator has no effect on the particle. In other words, a Majorana particle is its own antiparticle. This can only be possible for neutral particles given that the charge is reversed, and so the neutrino is the only candidate fermion satisfying this condition.

The incorporation of the Majorana mass term into the Standard Model does not necessarily exclude a Dirac mass term. If both terms are included, the Lagrangian consists of a non-diagonal mass matrix with elements of the Dirac mass, m_D, and Majorana masses, m_M^L and m_M^R (superscripts denoting left- and right-handedness, respectively). In matrix form, and considering only one neutrino flavour for simplicity,

$$\mathcal{L}_{\mathrm{D+M}} = -\frac{1}{2}\begin{bmatrix} (\bar{\nu}_\mathrm{L})^c & \bar{\nu}_\mathrm{R} \end{bmatrix} M \begin{bmatrix} \nu_\mathrm{L} \\ (\nu_\mathrm{R})^c \end{bmatrix} + \mathrm{H.c.}, \tag{1.11}$$

with

$$M = \begin{bmatrix} m_\mathrm{M}^\mathrm{L} & m_\mathrm{D} \\ m_\mathrm{D} & m_\mathrm{M}^\mathrm{R} \end{bmatrix}, \tag{1.12}$$

where H.c. is used to indicate the Hermitian conjugate of the prior term and the superscript c indicates charge conjugation. Under this theory, the chiral states are not aligned with the mass states, which are eigenstates of the Hamiltonian. In other words, the Dirac and Majorana masses are not the physical masses. By finding the eigenvalues and eigenstates of M, the mass basis can be constructed and the chiral states can be written as a superposition of the mass states.

A possible explanation for the observed tiny neutrino masses, the "seesaw mechanism" [35], imposes the conditions that $m_{\mathrm{M}}^{\mathrm{L}} = 0$ and $m_{\mathrm{M}}^{\mathrm{R}} \gg m_{\mathrm{D}}$. By applying the appropriate approximation under this assumption to the general eigenvalue solution, the resulting eigenvalues of M, corresponding to eigenstates conventionally labelled as ν and N, are

$$m_{\nu} \simeq \frac{(m_{\mathrm{D}})^2}{m_{\mathrm{M}}^{\mathrm{R}}} \tag{1.13}$$

and

$$m_{\mathrm{N}} \simeq m_{\mathrm{M}}^{\mathrm{R}}. \tag{1.14}$$

In the seesaw mechanism, the small *light neutrino* (ν) mass can be explained by the large *heavy neutrino* (N) mass, given that it is inversely proportional to $m_{\mathrm{M}}^{\mathrm{R}}$ per Equation (1.13). The neutrinos observed in the weak interaction are thus a superposition of these light and heavy neutrinos. Conversely, the light and heavy neutrinos are superpositions of the left- and right-handed neutrinos. The mixing angle would be small, with the dominant contribution to ν being from ν_{L} and the dominant contribution to N being from ν_{R}.

Due to this mixing between the chiral and mass states, and under the assumptions of the seesaw mechanism, the effective Dirac neutrino mass can be on the same order of magnitude as for the other fermions while being consistent with the neutrino mass limits from current observations. This is because the heavy, primarily right-handed, neutrino is not bound by Standard Model scales and can be very large. The seesaw mechanism is thus widely regarded as a well-motivated description of the small neutrino masses relative to all other known fermions.

Regardless of the exact mechanism, the Majorana formulation is appealing as it could also describe the dominance of matter over antimatter in our universe. It is thought that matter and antimatter were produced in equal quantities during the Big Bang. It would then be expected that the matter and antimatter should have completely annihilated. Astronomical observations have failed to detect any antimatter-dominated regions in the Universe [36], and so there must be an unexplained source of matter-antimatter asymmetry. If the neutrino is a Majorana particle, then two Majorana phases must be included in the PMNS matrix. These phases are a source of charge parity violation, which would then allow for asymmetric production of matter over antimatter [37].

Assuming the model of light and heavy neutrinos described earlier to be the case, the decay of heavy neutrinos into leptons and antileptons in the early stages of the Universe could be asymmetric. Further decays into quarks would also be necessary to explain the imbalance of matter in the baryon sector. This hypothetical process based on lepton number violation which leads to producing the observed baryon asymmetry is often called leptogenesis [38]. Even a small discrepancy in the decay rates could explain the formation of the Universe today, motivating the search for the Majorana nature of the neutrino and the first observation of lepton number violation.

1.3.3 Neutrinoless double-beta decay

In principle, one could directly observe (anti)neutrinos from a source to determine if some later behave as their antiparticle in a subsequent Standard Model reaction. For instance, if electron (anti)neutrinos are first emitted in $\beta^{\pm}$ decays and soon after absorbed via inverse $\beta^{\pm}$ decays, two leptons of the same sign would be observed, providing an experimental signature of lepton number violation. However, the neutrino must be absorbed in the correct chiral state for the second reaction to proceed, and since the typical relativistic neutrino only contains a small component of the opposite helicity, such a process is heavily suppressed.

Given the neutrino energies and mass scales involved in such an experiment, the rate of the hypothetical Standard Model-violating interaction would be many orders of magnitude lower than its Standard Model-obeying counterpart. Searching directly for lepton number violation is thus infeasible under the constraints of modern technology [39]. Furthermore, the Majorana phases cancel out in neutrino oscillation measurements, meaning that no differences should be observed in oscillation parameters whether or not the neutrino is a Majorana particle.

The most practical method of determining whether or not the neutrino is a Majorana particle is instead to observe the double-beta decay processes,

$$2\,\mathrm{n} \longrightarrow 2\,\mathrm{p} + 2\,\mathrm{e}^{-} + 2\,\bar{\nu}_{\mathrm{e}}, \tag{1.15}$$

$$2\,\mathrm{p} \longrightarrow 2\,\mathrm{n} + 2\,\mathrm{e}^{+} + 2\,\nu_{\mathrm{e}}. \tag{1.16}$$

Like the $\beta^{\pm}$ decays,

$$\mathrm{n} \longrightarrow \mathrm{p} + \mathrm{e}^{-} + \bar{\nu}_{\mathrm{e}}, \tag{1.17}$$

$$\mathrm{p} \longrightarrow \mathrm{n} + \mathrm{e}^{+} + \nu_{\mathrm{e}}, \tag{1.18}$$

which occur in nuclei with atomic mass number A and proton number Z as,

$$(A, Z) \longrightarrow (A, Z+1) + \mathrm{e}^{-} + \bar{\nu}_{\mathrm{e}}, \tag{1.19}$$

$$(A, Z) \longrightarrow (A, Z-1) + \mathrm{e}^{+} + \nu_{\mathrm{e}}, \tag{1.20}$$

double-beta decay can occur in nuclei as

$$(A, Z) \longrightarrow (A, Z+2) + 2\,\mathrm{e}^{-} + 2\,\bar{\nu}_{\mathrm{e}}, \tag{1.21}$$

$$(A, Z) \longrightarrow (A, Z-2) + 2\,\mathrm{e}^{+} + 2\,\nu_{\mathrm{e}}. \tag{1.22}$$

There exist some isotopes for which single $\beta^{\pm}$ decay to proton number $Z \mp 1$ is forbidden. This can occur if the sum of the masses of the products of the decay are greater than the mass of the parent isotope, or if the decay is disallowed due to the nonconservation of angular momentum. However, double-beta decay may be allowed.

There are only 35 isotopes for which it is possible for double-beta decay to occur [40], of which only 11 have been observed to undergo double-beta decay with typical half lives between $\sim 10^{19}$ yr to $\sim 10^{21}$ yr [41]. Furthermore, the two-electron double-beta channel in Equation (1.21), as opposed to the two-positron double-beta channel in Equation (1.22), is the only decay branch to have been experimentally detected. Thus, the two-electron double-beta channel is most typically what is referred to by the term two-neutrino double-beta (2νββ) decay.

If the neutrino is a Majorana particle, then it is possible for the decay to proceed without the emission of neutrinos. This would result in the lepton number-violating neutrinoless double-beta (0νββ) decay process (again, referring only to the two-electron channel),

$$2\,\mathrm{n} \longrightarrow 2\,\mathrm{p} + 2\,\mathrm{e}^-, \tag{1.23}$$

or, as it would occur in nuclei,

$$(A, Z) \longrightarrow (A, Z+2) + 2\,\mathrm{e}^-. \tag{1.24}$$

A Feynman diagram of this process is shown in Figure 1.2.

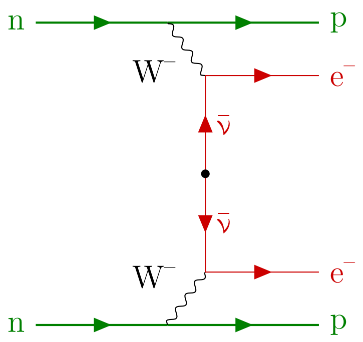


Figure 1.2: Feynman diagram of the 0νββ decay process. Corresponds to Equations (1.23) and (1.24). Figure adapted from [42].

Similarly to β decay, the energy spectrum of the two electrons emitted in the 2νββ decay process is continuous because the neutrinos carry away some fraction of the energy. With 0νββ decay, the electrons would carry all available energy, resulting in a peak in the distribution of the summed kinetic energies of the two electrons at exactly the Q-value of the decay (often abbreviated as $Q_{\beta\beta}$) and, in principle, a detectable signature. A theoretical illustration of the double-beta summed electron energy spectrum, using a highly exaggerated normalization of the 0νββ decay peak and a realistic energy resolution of 2.5 % at $Q_{\beta\beta}$, is shown in Figure 1.3. Although 0νββ decay has never been observed and remains hypothetical, given what is known about 2νββ decay half lives and limits on 0νββ half lives, the 0νββ peak would be much smaller in reality than what is represented in the figure. Non-zero energy measurement uncertainties of any real experiment, combined with the presence of backgrounds with energies near or overlapping the Q-value of 0νββ decay, make an observation of this peak extremely challenging. Experimental considerations for detection are discussed more in Section 1.4.2.

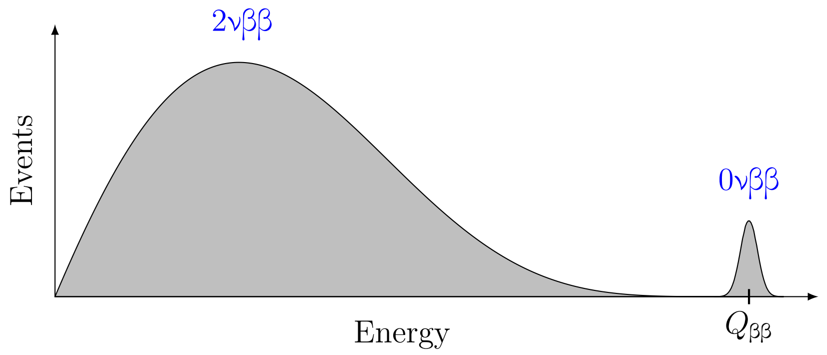


Figure 1.3: Illustrative summed electron energy spectrum from double-beta decay, assuming that 0νββ decay can occur. A peak at the Q-value of the decay would be observable given sufficient background reduction and low energy resolution, as is shown in the figure. The 0νββ decay peak here uses an energy resolution of 2.5 % and a highly exaggerated normalization relative to the contribution to the spectrum from 2νββ decay. Figure adapted from [42].

The 0νββ decay half-life is given by

$$T^{0\nu}_{1/2} = \left(G \left| \mathcal{M} \right|^2 \langle m_{\beta\beta} \rangle^2 \right)^{-1}, \tag{1.25}$$

where G is a phase space factor, $|\mathcal{M}|$ is the nuclear matrix element for the decay, and $\langle m_{\beta\beta} \rangle$ is the effective Majorana mass term given by

$$\langle m_{\beta\beta} \rangle = \left| \sum_{k=1}^{3} U_{e,k}^2 m_k \right| . \tag{1.26}$$

Here, $U_{e,k}$ are elements of the PMNS mixing matrix, as originally introduced in Equation (1.3). 0νββ decay is suppressed because the exchanged (anti)neutrino must be absorbed in the correct chiral state for the weak interaction (see Figure 1.2). As the neutrino is relativistic, it only contains a small proportion of the opposite helicity state (or conversely, a neutrino in a given helicity state only contains a small proportion of the opposite chirality state). The mass term m_k is included in Equation (1.26) due to this helicity suppression of the decay, which scales by the inverse of the squared mass. The electron neutrino flavour mixing elements arise from the fact that the electron neutrino is a superposition of mass states, and the probability of observing an electron neutrino in mass state k is given by $U_{e,k}^2$.

An observation of 0νββ decay would allow for the effective Majorana mass to be determined, as per Equation (1.25), and current limits on the decay half-life thus also constrain the mass. However, the nuclear matrix element is a large source of uncertainty in its determination. There are numerous techniques to estimate this parameter, and variations usually range by a factor of about 2 to 3 for a given isotope [42, 43]. These discrepancies mean that different models can lead to quite a range for the effective Majorana mass term. As well, such uncertainties have implications on the planning of 0νββ decay experiments due to the inverse squared dependence of the half-life on the nuclear matrix element in Equation (1.25). This affects both the amount of isotope to be used (the sensitivity, and thus amount of material to be deployed, could span about an order of magnitude for a given mass value to probe) and the choice of isotope itself (since the nuclear matrix element influences the half-life, and thus the chance of detection for a given mass value in a given amount of time).

Due to the aforementioned helicity suppression of $0\nu\beta\beta$ decay, the process – if allowed at all – is even more rare than $2\nu\beta\beta$ decay by at least several orders of magnitude. A large global effort is in place to search for $0\nu\beta\beta$ decay, with different approaches being used around the world. The next section summarizes the leading detection techniques and technologies, along with major competitive experiments and their results (if actively running or completed) or projected sensitivities (if planned or under construction).

1.4 Neutrinoless double-beta decay detection and experimental methods

The careful design of an experiment to detect $0\nu\beta\beta$ decay is crucial to its success. The choice of double-beta decay isotope and the choice of detection technology are two important considerations to make, and are described in more detail below.

1.4.1 Double-beta decay isotopes

For a number of reasons, the chosen isotope should ideally have a high $0\nu\beta\beta$ decay Q-value. Radioactive backgrounds are more prominent at lower energies and are thus more likely to overlap the energy region of interest (ROI) encompassing the $0\nu\beta\beta$ decay Q-value, making the separation of signal events challenging. Furthermore, G in Equation (1.25) scales with the Q-value of the decay [32], indicating a lower half-life and thus a better chance of detection.

In addition to the Q-value, backgrounds inherent in, or inadvertently introduced into, the material – whether it be from acquisition, transport, processing, or the isotope itself – should be considered. The irreducible 2νββ decay background is of particular concern, and isotopes should ideally be chosen to have a long 2νββ decay half-life. Given that larger values of the phase space and nuclear matrix elements imply lower half lives, and thus better detection chances, both their absolute values and the 0νββ to 2νββ ratios of these quantities must be factored into the choice of isotope.

Practical limitations must also be considered. In particular, some isotopes are easier and less costly to acquire than others. Some isotopes also have a much higher abundance in the natural form of the material, meaning that the amount of the isotope per unit mass of material is higher, thus improving the chances of detection. As well, it reduces the need for enrichment, which (if it is even possible for the isotope) can be very expensive.

As well, the choice of detection technology (discussed in more detail in Section 1.4.2) is not independent of the isotope chosen. In some instances, the isotope can be a part of the detection medium, increasing the detection efficiency. Some detection technologies require high quantities of the isotope, which limits the choice based on practical considerations.

A summary of the double-beta isotopes typically used in 0νββ decay search experiments, with their natural abundances, Q-values, and 2νββ decay half lives, is given in Table 1.1. As can be seen from the table, no isotope presents itself as an obvious choice, and experiments must make prioritizations. For instance, ^{48}Ca has by far the *highest* Q-value and yet has the *lowest* natural abundance. On the other hand,

^{130}Te has by far the *highest* natural abundance but is near the bottom of the listed candidate isotopes in terms of Q-value.

Table 1.1: Candidate double-beta decay isotopes. Table is sorted by increasing atomic mass number of the isotope. Q-values from [44], natural abundances from [45, 46], and $2\nu\beta\beta$ decay half-life values from [41].

Isotope	**$0\nu\beta\beta$ Q-value**	**Natural abundance**	**$2\nu\beta\beta$ half-life, $T^{2\nu}_{1/2}$**
	(MeV)	(%)	(yr)
^{48}Ca	4.268	0.2	$5.30^{+1.20}_{-0.80} \cdot 10^{19}$
^{76}Ge	2.039	7.8	$(1.88 \pm 0.08) \cdot 10^{21}$
^{82}Se	2.998	8.8	$0.87^{+0.02}_{-0.01} \cdot 10^{20}$
^{96}Zr	3.356	2.8	$(2.30 \pm 0.20) \cdot 10^{19}$
^{100}Mo	3.034	9.7	$7.06^{+0.15}_{-0.13} \cdot 10^{18}$
^{116}Cd	2.813	7.5	$(2.69 \pm 0.09) \cdot 10^{19}$
^{130}Te	2.528	34.1	$(7.91 \pm 0.21) \cdot 10^{20}$
^{136}Xe	2.458	8.9	$(2.18 \pm 0.05) \cdot 10^{21}$
^{150}Nd	3.371	5.6	$(9.34 \pm 0.65) \cdot 10^{18}$

1.4.2 Detection technologies

There are various technologies available for the detection of $0\nu\beta\beta$ decay. They are outlined in this section. The ones directly relevant to this thesis – liquid scintillator detectors and semiconductor detectors – are listed last and described in the most detail. As well, the experiments discussed along with the $0\nu\beta\beta$ decay limits are summarized in Table 1.2. All experiments outlined here are based on the principle of measuring the energy deposition of the two electrons from double-beta decay.

Table 1.2: 0νββ decay half-life limits for double-beta decay isotopes. Table is sorted first by increasing atomic mass number of the isotope and second by increasing half-life limit, if multiple competing experiments are using or have used the same isotope.

Isotope	**Experiment**	**0νββ half-life limit, $T^{0\nu}_{1/2}$** (10^{25} yr, 90 % **CL**)	**Citation**
^{48}Ca	NEMO-3	0.0020	[47]
	ELEGANT VI	0.0058	[48]
^{76}Ge	MJD	8.3	[49]
	GERDA	18	[50]
^{82}Se	NEMO-3	0.025	[51]
	CUPID-0	0.46	[52]
^{96}Zr	NEMO-3	0.00092	[53]
^{100}Mo	NEMO-3	0.11	[54]
	CUPID-Mo	0.18	[55]
^{116}Cd	NEMO-3	0.010	[56]
	Aurora	0.022	[57]
^{130}Te	CUORE	2.2	[58]
^{136}Xe	EXO-200	3.5	[59]
	KamLAND-Zen	23	[60]
^{150}Nd	NEMO-3	0.0020	[61]

1.4.2.1 Calorimetric detectors

A number of detectors are based on calorimetric technology. CUORE (Cryogenic Underground Observatory for Rare Events) [62,63] and AMoRE (Advanced Mo-based Rare Process Experiment) [64,65] use arrays of bolometers – calorimeters operated at extremely low temperatures on the ∼10 mK scale – containing the double-beta isotope as the detector medium and source. A thermometer is used to measure minuscule changes in temperature when energy is deposited in these bolometric absorbers.

The absorbers are grown as crystals and can be grown from many of the candidate isotopes listed in Table 1.1. In practice, TeO_2 crystals have typically been used to search for the $0\nu\beta\beta$ decay of ^{130}Te, although enriched crystals composed of the double-beta isotope ^{100}Mo are also used by the AMoRE experiment. CUORE has set the most stringent limits on the $0\nu\beta\beta$ decay of ^{130}Te with a $T^{0\nu}_{1/2}$ limit of $2.2 \cdot 10^{25}$ yr (90 % CL) and a corresponding $m_{\beta\beta}$ limit of $(90 - 305)\,\mathrm{meV}/c^2$, the wide range of which is a consequence of different nuclear matrix element values [58].

Another advantage of this technology is that more bolometers can be added to the arrays, allowing for the overall isotope mass to be increased relatively straightforwardly. However, particle identification – and thus background rejection – in such detectors is relatively poor. The successor to CUORE, CUPID (CUORE Upgrade with Particle Identification) [66], aims to address this issue by using scintillating bolometers as the absorbers. The CUPID collaboration has used the CUPID-0 and CUPID-Mo demonstrators to set limits on the $0\nu\beta\beta$ decay of ^{82}Se [52] and ^{100}Mo [55], respectively, both of which are world-leading. The CUPID tonne-scale experiment will take a phased approach to lower backgrounds and introduce more of the ^{100}Mo isotope, with the first phase, CUPID baseline, expected to have a $T^{0\nu}_{1/2}$ limit sensitivity of $>10^{27}$ yr after ten years of data collection [66]. This sensitivity corresponds to $m_{\beta\beta}$ exclusion limits of $(10 - 17)\,\mathrm{meV}/c^2$ depending on the nuclear matrix element model used [67]. The final phase, CUPID-1T, will use 1 t of ^{100}Mo and has a projected $T^{0\nu}_{1/2}$ limit sensitivity of $9.2 \cdot 10^{27}$ yr ($m_{\beta\beta}$ range $(4.1 - 6.8)\,\mathrm{meV}/c^2$; both 90 % CL) [67]. Construction and commissioning of this phase could begin in the late 2020s or early

2030s [68].

A completely different calorimetric approach is to use layers of trackers surrounded by calorimeters. The source material containing the double-beta isotope is placed in the centre of the detector as a thin foil. Surrounding the source are wire chambers, which allow for particle track reconstruction in three dimensions. Finally, calorimeters on the outside of the detector measure the deposited energy of the particles. Given the detailed event information from this approach, tracking calorimeters are able to record several observables and provide useful topological separation. The NEMO (Neutrino Ettore Majorana Observatory) experiment [69], and its successor SuperNEMO (Super NEMO) [70], use this technique. Furthermore, since the source is distinct from the detector, it is straightforward to use multiple double-beta decay isotopes with the same configuration. NEMO's detector, NEMO-3, has been used to set some of the best $0\nu\beta\beta$ decay half-life limits for multiple isotopes, including ^{96}Zr [53], ^{100}Mo [54], and ^{150}Nd [61]. However, the foil with the source isotope must be extremely thin to ensure the electrons from double-beta decay escape into the detector. Thin foils are not only difficult to produce, but also limit the target mass and thus scalability of such a detector configuration.

1.4.2.2 Time projection chambers

Time projection chambers (TPCs) consist of a detector filled with an inert substance susceptible to ionization and scintillation. A large voltage difference is applied to the ends of the detector, typically a cylinder, to create an electric field. Particles disturb the medium and ionize its atoms, producing electrons which drift to the anode. At the anode, a segmented detector – often an array of wires forming a multi-wire proportional chamber – is used to reconstruct the two-dimensional position of the interaction in the plane perpendicular to the drift direction. Three-dimensional reconstruction can be accomplished by measuring the time between the detection of scintillation photons from the interaction and the registration of a signal at the anode from the ionized electrons. The energy is reconstructed via the collected charge and scintillation light signals.

The EXO (Enriched Xenon Observatory) experiment [71], and its successor nEXO (next EXO) [72], are based on the TPC concept. The EXO-200 detector used liquid xenon enriched with the ^{136}Xe isotope as the detector medium, acting as both the source and the target. nEXO will contain a much larger mass of xenon with an even higher enrichment factor. After ten years of operation, the projected $T^{0\nu}_{1/2}$ sensitivity of nEXO is expected to be $1.35 \cdot 10^{28}$ yr (90 % CL), corresponding to $m_{\beta\beta}$ limits of $(4.7 - 20.3)\,\mathrm{meV}/c^2$ calculated using a range of nuclear matrix element models [72].

Another planned experiment, NEXT (Neutrino Experiment with a Xenon TPC) [73], uses a different approach than EXO and nEXO, instead using high-pressure gaseous xenon, to provide superior energy resolution. It also provides better topological separation, allowing for the possibility of the two electrons from the double-beta decay of ^{136}Xe to be individually identified. Such tracking capability provides excellent background discrimination from all but the $2\nu\beta\beta$ decay channel, which is limited by energy resolution. NEXT has demonstrated that searches for $0\nu\beta\beta$ decay using this technique are feasible with their smaller-scale NEXT-White detector [74]. The expected $T^{0\nu}_{1/2}$ sensitivity of NEXT's future tonne-scale detector is $>10^{27}$ yr [75].

TPCs are advantageous in that they provide excellent particle identification via the ratio of ionization to scintillation energy detected for an event. TPCs are also relatively large and can be scaled fairly straightforwardly. While their energy resolution is more modest than other technologies, the use of a gas instead of a liquid reduces this discrepancy. However, the choice of the double-beta isotope is limited as the detector medium must also contain the source. In particular, the medium should be inert due to the electric field and must scintillate for full topological reconstruction capabilities. The medium must also be extremely pure to avoid recapture and changing of its drift properties. $0\nu\beta\beta$ decay experiments use xenon enriched in the ^{136}Xe isotope for these reasons.

1.4.2.3 Inorganic scintillator detectors

As the only known double-beta isotope with a Q-value over 4 MeV, the $0\nu\beta\beta$ decay of ^{48}Ca has the smallest ROI overlap with natural radioactive backgrounds. However, its natural abundance is at least an order of magnitude lower than the other candidate isotopes (see Table 1.1), making enrichment necessary. More efficient and inexpensive enrichment processes are actively being developed for this reason.

^{48}Ca can be found in inorganic CaF_2 (calcium fluoride) crystals. These crystals undergo scintillation, and can be doped with europium to provide better light yield. The ELEGANT (Electron Gamma-ray Neutrino Telescope) experiment currently has the best limits on the $0\nu\beta\beta$ decay of ^{48}Ca using their ELEGANT VI detector [48]. Due to the relatively short attenuation lengths, such a configuration is not easily scaled to higher target masses. The CANDLES (Calcium Fluoride for the Study of Neutrinos and Dark Matters by Low Energy Spectrometer) experiment uses undoped CaF_2 crystals instead, immersed in liquid scintillator to shield the crystals from backgrounds. CANDLES uses a wavelength shifter around the crystals to provide comparable light yield to the ELEGANT VI detector with longer attenuation. The CANDLES-III detector configuration is in operation and the backgrounds have been studied to determine the feasibility of using such a detector for future 0νββ decay searches [76].

The isotope ^{116}Cd can also be found in inorganic scintillating crystals, specifically $CdWO_4$ (cadmium tungstate). The Aurora experiment used $CdWO_4$ crystals enriched with ^{116}Cd to set the best limits on the 0νββ decay of ^{116}Cd [57].

1.4.2.4 Liquid scintillator detectors

Organic liquid scintillators are a frequent choice of material used in the detection of 0νββ decay. They offer several advantages including a high light yield and a fast timing response, both of which are crucial for event reconstruction. Liquid scintillators, unlike solid materials, can be relatively easily purified, even when the detector is in operation [77–83]. As well, since the double-beta source material is dissolved into the scintillator, more options for the choice of isotope are available. Although liquid scintillator detectors have worse energy resolutions than other available technologies, they are much more easily scaled to larger target masses. Furthermore, double-beta isotope can be later removed from the scintillator. This can extend the life of the experiment for other purposes and allow for systematic checks. It also offers a practical advantage, as the isotope may be able to be resold at a later time.

Both SNO+ [84], the successor to the SNO experiment, and KamLAND-Zen (KamLAND Zero-Neutrino Double-beta Decay) [85], the successor to the KamLAND experiment, will search for 0νββ decay using this technology. SNO+ will dissolve natural tellurium directly into the liquid scintillator in the detector and search for the $0\nu\beta\beta$ decay of ^{130}Te. The expected $T^{0\nu}_{1/2}$ sensitivity after five years of data taking and operation is $2.1 \cdot 10^{26}$ yr, and sensitivities greater than 10^{27} yr are achievable due to the scalability of the detector technology [84]. More details on the SNO+ detector are given in Section 2.1. KamLAND-Zen has taken a different approach, opting to deploy a thin balloon of enriched xenon-loaded liquid scintillator in the centre of the original KamLAND detector. This balloon is itself contained within the larger main balloon of pure liquid scintillator. Currently, the best limits on the $0\nu\beta\beta$ decay of ^{136}Xe have been set by KamLAND-Zen using KamLAND-Zen 800 [60], a detector configuration with an internal balloon containing 745 kg of enriched xenon (>90 % ^{136}Xe).

Typically, a liquid scintillator contains both a bulk solvent and one or more wavelength-shifting dopants. The dopant(s) ensure that the light emitted from interactions is of a more optimal wavelength for detection. They also typically have less overlap between their emission and absorption spectra, hence increasing the overall light yield. Linear alkylbenzene (LAB) is an example of a bulk solvent and is used in the SNO+ experiment [84]. 2,5-diphenyloxazole (PPO) is a common choice for the primary dopant and is used in both the SNO+ [84] and KamLAND-Zen [60] experiments. Daya Bay [86] and RENO (Reactor Experiment for Neutrino Oscillation) [87] are two additional examples of experiments which use LAB as the base solvent and PPO as the primary dopant.

1.4.2.5 Semiconductor detectors

Semiconductor detectors are another common choice of detection technologies. High purity germanium (HPGe) detectors are of particular importance due to the fact that

^{76}Ge is a double-beta isotope, and so detectors can be fabricated with source-enriched target material. Additionally, as the name implies, HPGe detectors are frequently used for their intrinsic purity, which arises due to the crystal growing procedure [88–90]. The particularly detrimental uranium and thorium isotopes are not found at detectable levels within the HPGe detectors used in modern experiments [91]. They also offer superior energy resolution over other detector technologies and very low energy thresholds. Point-contact detectors in particular have a very low capacitance and thus even better energy resolutions compared to the traditional semi-coaxial detectors [92,93]. Although HPGe detectors offer numerous advantages, they are very costly to fabricate. As well, the Q-value of ^{76}Ge is the lowest of the candidate double-beta isotopes and its natural abundance makes enrichment necessary (see Table 1.1). HPGe detectors also need to be built individually and put together into an array of detectors. However, this modular construction allows for an experiment to be built in phases, which is useful for both testing and to spread out costs.

Currently, the MJD (Majorana Demonstrator) and GERDA (Germanium Detector Array) experiments – both of which used point-contact HPGe detectors – have set some of the best limits on the 0νββ decay half-life in the ^{76}Ge isotope space [49, 50]. The future LEGEND (Large Enriched Germanium for Neutrinoless Double-beta Decay) experiment aims to construct a tonne-scale HPGe detector array (>90 % ^{76}Ge enrichment), combining and expanding the collaborators, efforts, tools, materials, and other resources from the MJD and GERDA experiments [94]. LEGEND's ∼200 kg demonstrator, LEGEND-200, is partially operational and collecting data [95]. The LEGEND-200 experiment is located at the Laboratori Nazionali del Gran Sasso (LNGS) facility, where GERDA previous operated, and has an expected

$T^{0\nu}_{1/2}$ sensitivity of $\sim 10^{27}$ yr. The demonstrator uses a combination of semi-coaxial, p-type point contact (PPC), and inverted coaxial point contact (ICPC) [96] detectors, the latter of which is a newer development and allows for individual detectors to be a factor of 2 to 3 times larger [94]. The second phase of the project will consist of an upgrade to LEGEND-1000, characterized by its use of approximately 1 t of enriched germanium to achieve a $T^{0\nu}_{1/2}$ sensitivity of $>10^{28}$ yr and a corresponding $m_{\beta\beta}$ limit of $(9 - 21)$ meV$/c^2$ depending on the nuclear matrix element model used [94].

Other semiconductor technologies have been proposed and used, although not at large scales. The COBRA (Cadmium Zinc Telluride 0 Neutrino Double-beta Research Apparatus) experiment uses CdZnTe (cadmium zinc telluride) crystals as semiconductor detectors. One advantage of such an experiment is that CdZnTe contains nine double-beta isotopes, all of which are part of the detector medium [97]. Five of the isotopes – ^{70}Zn, ^{128}Te, ^{130}Te, ^{114}Cd, and ^{116}Cd – decay via the $2\beta^-$ mode and are thus candidates for the typical $0\nu\beta\beta$ decay channel [98]. The COBRA demonstrator, consisting of 64 CdZnTe detectors of volume 1 cm^3 each, has been used to show that searches for $0\nu\beta\beta$ decay using this technology are feasible, and noncompetitive half-life limits on the five aforementioned candidate isotopes have been set [98]. The demonstrator has been upgraded and extended with nine larger CdZnTe detectors for future searches with improved sensitivity [99].

1.5 Thesis goals and overview

The field of neutrino physics is in an exciting era. Much has been learned about the neutrino since its first detection, and yet still more remains to be understood. In addition to its unknown absolute mass scale, the nature of the neutrino is undetermined. The detection of the hypothetical 0νββ decay introduced in this chapter can help answer these questions and provide a better understanding of the Universe as a result. Many large experiments utilizing a number of different detection technologies are searching for this process today.

In order to have even a chance of detecting 0νββ decay, or other rare event processes, extraordinary physical measures must be taken to reduce backgrounds. Experiments are often located deep underground to shield from cosmic radiation which would otherwise dominate any signal and render identification completely infeasible. As well, trace amounts of radioactive impurities in any component of the detector setup, introduced at any point from the manufacturing process to the installation, can ruin the entire experiment. Cleanliness is constantly maintained at all stages of construction and operation as a result. While these physical measures can limit backgrounds to a manageable level, the vast majority of data collected are still backgrounds. As detector hardware continues to improve, allowing for increased rates of data collection, advanced analytical techniques are becoming more and more important.

Machine learning, and neural networks in particular, are frequently used in other fields to extract complex patterns from large amounts of data. Their application in certain areas of particle astrophysics, though growing, is still limited and many avenues exist where advances in data analysis can be made. This thesis focuses on the novel development of two neural network techniques with each applied to a different experimental technology: (1) event reconstruction for the SNO+ experiment, and (2) pulse denoising for a local PPC HPGe detector. Along with motivations and thorough descriptions of their development, this thesis demonstrates how these tools can be used to improve experimental sensitivities to rare events. Practical implications are also discussed. Furthermore, a strong emphasis is placed on the broad applicability of these approaches; while two specific detectors are used to demonstrate the performance and results of the methods, extensions – both ongoing and planned – of each project are described.

This chapter motivates the search for $0\nu\beta\beta$ decay and the work of this thesis. Chapter 2 introduces the SNO+ detector and the local PPC HPGe detector in the context of their experimental programs to better understand the analyses and work of the author that follows. Chapter 3 provides a background on machine learning and neural networks. While fairly comprehensive, only details directly relevant to the development of the models in later chapters are described. The author's primary contributions form the remainder of the thesis (except for the conclusion and end-matter) and are split into two parts. Part I – containing Chapters 4 and 5 – covers the development of a new neural network-based method for event reconstruction in the SNO+ detector and its various applications, results, and extensions. Part II – containing Chapters 6 and 7 and based on the author's publication in [1] – presents the development of an autoencoder for pulse denoising in a local PPC HPGe detector at Queen's University, along with how it can be used to improve the sensitivity of experiments using these types of detectors to rare events. Chapter 8 concludes the thesis and highlights some of the future prospects of the work presented in Parts I and II.

Chapter 2

Neutrino Experiments and Technologies

This chapter outlines and describes the two detector technologies in the context of their experimental programs that are utilized in this thesis. Section 2.1 describes the SNO+ detector, which is relevant for Part I. Section 2.2 describes the p-type point contact high purity germanium detector at Queen's University, for which the analysis in Part II depends on.

2.1 The SNO+ detector

2.1.1 Detector overview

The SNO+ detector is located 2 km underground at SNOLAB, an underground laboratory and clean facility in an active nickel mine near Sudbury, Ontario, Canada [84]. SNOLAB hosts several neutrino and dark matter detectors in addition to SNO+ [100, 101]. The deep underground location provides excellent shielding from cosmic radiation, with the flat overburden offering an equivalent protection of ∼6000 m of water [100]. The resulting muon flux at SNOLAB is approximately $0.3\,\mathrm{m}^{-2} \cdot \mathrm{d}^{-1}$ [100, 102], implying a reduction in flux by a factor of nearly $5 \cdot 10^7$ in comparison to at sea level with no shielding[1]. Furthermore, the entire laboratory space is a class-2000 clean room, with even more stringent classes maintained closer to experimental infrastructure [100, 101].

The SNO+ detector consists of a 12 m diameter, 5.5 cm thick spherical acrylic vessel (AV) filled with approximately 780 t of liquid scintillator. Approximately 10 000 photomultiplier tubes (PMTs) surround the AV to register light emitted from particle interactions in the detector medium. These PMTs are supported by a geodesic steel structure called the PSUP (PMT support structure) at a radius of approximately 8.4 m. A cavity dug out from the rock houses the entire detector configuration. It is roughly cylindrical in shape with a maximum diameter of 24 m and a height of 30 m. The AV is immersed and suspended (via hold-up and hold-down ropes for the weight of the AV and buoyancy of the liquid scintillator, respectively) in 7 kt of ultra-pure water filling the cavity to provide shielding from radioactivity present in the rock and PMTs.

Much of the hardware used for SNO+, as well as the detector's location in what is now SNOLAB, is from its predecessor, the SNO experiment. This includes the PMTs and AV. However, SNO+ has several major upgrades from the original SNO experiment to account for its new physics goals, such as faster electronics to handle the increased rate of data collection and an improved cover gas system to further reduce radon backgrounds. Additionally, completely new systems were built and deployed for the purposes of the new detector media. This includes the aforementioned hold-down ropes to account for the buoyancy of the AV when filled with liquid scintillator, which has a lower density than the water which it is immersed in, and purification

[1]This reduction factor is calculated using the standard muon flux of $1\,\text{cm}^{-2} \cdot \text{min}^{-1}$ on surface at sea level [29].

and processing plants for the scintillator and tellurium. An illustration of the SNO+ detector containing the main components is shown in Figure 2.1.

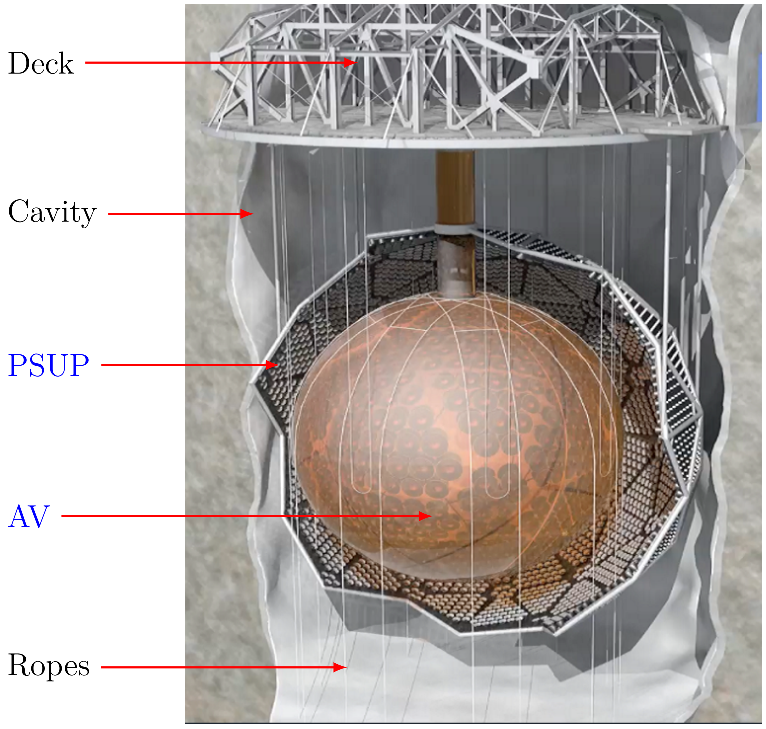


Figure 2.1: Cross-sectional illustration of the SNO+ detector. Included are the AV, PSUP, rope systems, cavity, and upper deck. Figure adapted from [84].

The AV contains the detection medium, discussed more thoroughly in Section 2.1.3 in relation to the physics goals highlighted in Section 2.1.2. The SNO+ electronics and data acquisition (DAQ) system are discussed in Section 2.1.5, and the data processing and analysis software are introduced in Section 2.1.6. A more detailed description of the SNO+ detector, its parts, and its systems is given in [84].

2.1.2 Physics goals

The SNO+ experiment is multi-purpose and has a number of physics goals throughout its operation. More details regarding these goals can be found in [84] and [103].

2.1.2.1 Neutrinoless double-beta decay

The primary goal of SNO+ is to search for the $0\nu\beta\beta$ decay of ^{130}Te. As described in more detail in Section 2.1.3, the SNO+ detector will be loaded with natural tellurium, which contains 34.1 % of the double-beta isotope (see Table 1.1) and thus requires no enrichment. SNO+ thus has the potential to fill the detector with a large amount of ^{130}Te which will allow the experiment to set competitive limits on, or improve the chances of detecting, $0\nu\beta\beta$ decay.

2.1.2.2 Secondary physics goals

Although the key objective for SNO+ is to search for $0\nu\beta\beta$ decay, the SNO+ experiment has a broad physics program that is allowed by its low backgrounds, low energy threshold, and large detector volume. The experiment is set to undergo three phases of operation (described in Section 2.1.3), each of which is optimal for a subset of objectives (some of which overlap between phases). Some of the prominent secondary physics goals are described below.

Antineutrinos Measuring the flux of antineutrinos can help us to constrain neutrino oscillation parameters and understand more about the inner workings of the Earth. Two sources of antineutrinos at SNO+ and their importance include:

- **Reactor Neutrinos** A major source of antineutrinos at SNO+ is from nuclear reactors due to the decay of isotopes in the core and subsequent β decays of the products. The flux of antineutrinos is directly related to the thermal power of the nuclear reactor and can be calculated precisely. Since the distance from SNO+ to the nuclear reactors is also known, reactor neutrinos are useful for the study of neutrino oscillations. As the antineutrino flux scales with the inverse squared distance from the source, the majority (60 %) of the antineutrino flux at SNO+ is from the Bruce, Pickering, and Darlington nuclear reactors in Canada.

- **Geoneutrinos** Another source of antineutrinos is from radioactivity in the Earth. Beta decay of isotopes in the mantle and the crust (^{238}U, ^{232}Th, ^{40}K) produce a "geoneutrino" flux that can be measured by SNO+. Geoneutrinos are of interest because a large fraction of the heat produced in the Earth is thought to be from such decays. Furthermore, a study of geoneutrinos will provide a better understanding of the backgrounds for SNO+ and other experiments.

Solar neutrinos SNO+ is also sensitive to solar neutrinos, the study of which can provide a better understanding of the Sun and lead to improved measurements of neutrino mass splitting and mixing angle parameters. With low backgrounds in the detector, SNO+ is able to measure ^{8}B solar neutrinos across a wide range of the energy spectrum. This is important as oscillation probabilities depend on the neutrino energy and the effects of matter as the neutrinos pass through the Sun and the Earth. In particular, the lower energy "transition regime" – where matter effects become less prominent and the electron neutrino survival probability increases notably with decreasing energy – may offer insight into subdominant oscillation effects and new physics beyond the Standard Model.

Supernova neutrinos Neutrinos produced from supernovae provide an excellent opportunity to learn more about neutrino physics as well as to further study core collapse explosion mechanisms. Furthermore, the neutrinos from supernovae are emitted almost immediately after a core collapse, whereas the photons may be emitted hours or days after. As a result, the detection of such neutrinos may provide advanced warning of a supernova explosion. “Pre-supernova neutrinos” from the carbon, neon, oxygen, and silicon burning phases, as well as pair production and electron capture in the accretion and cooling phases, may also be detectable. The SNO+ collaboration, at the time of writing, is in the process of joining a global consortium of experiments sensitive to supernova neutrinos – the SuperNova Early Warning System (SNEWS) project [104, 105] – which aims to provide advanced warnings of supernovae to the astronomical community and coordinate efforts to optimize the detection of the emitted neutrinos (e.g., reduction of thresholds based on coincident signals or external triggers to other detectors).

Exotic physics Due to the large volume of low background material deployed in the detector, SNO+ can search for, and place limits on, many other beyond the Standard Model physics processes. This includes invisible modes of (di)nucleon decay and the detection of axion-like particles.

2.1.3 Operating phases

The SNO+ detector is planned to undergo three phases of operation and is currently in its second phase. Each phase is distinctly marked by the target material in the AV. The three phases consist of:

1. **Water phase** Initially, the AV was filled with approximately 900 t of ultra-pure water. The SNO+ experiment has completed the water phase, with data taking occurring from September 2017 to July 2019.

   In this phase, measurements of the backgrounds were performed and several calibration systems were deployed. As well, several physics studies (e.g., solar neutrinos [106], nucleon decay [107], measurement of the neutron-proton capture [108], observation of reactor antineutrinos [109]) were conducted.

2. **Scintillator phase** At the time of writing, the AV is completely filled with approximately 780 t of liquid scintillator. SNO+ is actively taking data and thus currently in the scintillator phase. As briefly mentioned in Section 1.4.2.4, the SNO+ liquid scintillator is composed of LAB as the base solvent and PPO as the dopant. LAB was chosen due to its compatibility with acrylic, non-toxic nature, comparable light yield to other liquid scintillators, and general availability and low cost of the product [81]. PPO was also chosen for its low cost and availability. Additionally, it is well-studied and has been used and successfully deployed in liquid scintillators for many years (e.g., [110]).

   The transition from the water phase to the scintillator phase required slowly replacing the ultra-pure water inside the AV with the SNO+ liquid scintillator. This was done through the top of the AV neck while draining water from the bottom. As liquid scintillator is both less dense than and immiscible with water, the scintillator floated on top and produced a clear boundary at the interface. Unfortunately, this transition was interrupted by the COVID-19 global pandemic, resulting in a halt to the fill and an unplanned "partial fill" period from March 2020 to October 2020. During this time, the boundary sat at approximately 75 cm above the equator of the AV and the concentration of PPO was 0.6 g/L[2].

[2]g/L in the context of scintillator means grams of solute per litre of solution.

Eventually, operations were resumed and the initial scintillator fill was concluded in April 2021, with the PPO concentration remaining the same. Additional PPO was added until April 2022, bringing the concentration to about 2.2 g/L. 1,4-bis(2-methylstyryl)benzene (BisMSB) – a secondary wavelength shifter – is now being added to the scintillator to improve the light yield for when tellurium loading begins in the next phase [81]. The first batch of BisMSB was added in July 2023, with subsequent batches following in September – November 2023 and bringing the current concentration to 2.2 mg/L. As well, another chemical – butylated hydroxytoluene (BHT) – was introduced into the detector in June 2023. While BHT should not impact the light yield of the medium, it will help to prevent oxidation. The net concentration of BHT is currently 6.5 mg/L.

With a much higher light yield than water, liquid scintillator allows lower energy events to be detected and improves the sensitivity to certain physics processes. The primary goals of this phase are to continue to understand the backgrounds in the detector, measure the flux of low energy solar neutrinos, measure the geoneutrino flux, and potentially observe neutrinos from supernovae. Completed objectives include a measurement of the antineutrino flux from nuclear reactors [111]. As well, a study on directionality was conducted using data from the partial fill periods due to the lower concentration of the dopant, demonstrating event-by-event direction reconstruction for the first time in a large liquid scintillator detector [112].

3. **Tellurium phase** The final phase of the SNO+ experiment is the tellurium-loaded scintillator phase, or simply the "tellurium phase." Initially, approximately 3.9 t of natural tellurium will be dissolved into the scintillator and introduce about 1.3 t of the ^{130}Te isotope [84]. This corresponds to 0.5 % natural tellurium by mass, and higher concentrations of up to 3 % are planned. The main focus of this phase is to search for the $0\nu\beta\beta$ decay of ^{130}Te and to improve measurements of the ^{130}Te $2\nu\beta\beta$ decay lifetime. Many of the goals from the scintillator phase will also continue in the tellurium phase.

   ^{130}Te was chosen as the double-beta isotope due to its high isotopic abundance in natural tellurium and compatibility with liquid scintillator. As well, only two candidate isotopes in Table 1.1 have a longer $2\nu\beta\beta$ decay half-life than ^{130}Te, minimizing the impact of this irreducible background.

   At the time of writing, the tellurium phase has not begun. Preparation is in progress, with additions to the scintillator such as BisMSB and BHT having been completed and further additions such as the N,N-dimethyldodecylamine (DDA) stabilizer [113] to precede the tellurium loading. Tellurium is expected to be added to the detector in early 2025.

The SNO+ experiment has the advantage of being able to dissolve additional tellurium straightforwardly and inexpensively, which would allow for better sensitivity to the $0\nu\beta\beta$ decay signal. Future phases or subphases are possible for the SNO+ experiment and could entail various improvements such as increased tellurium loading and upgrades to higher quantum efficiency PMTs [84].

2.1.4 Interaction and detection principles

Detection of particle interactions in SNO+ relies on the scintillation light emitted from ionizing radiation. Materials that can emit light via scintillation are called "scintillators." In a scintillator, charged particles passing through the medium excite its molecules, which then quickly de-excite and emit energy in the form of photons [114, 115]. This process occurs over a period of ∼10 ns or less, and any scintillator is thus fluorescent. Scintillators can be inorganic or organic, each with different scintillation mechanisms. The scintillation process is isotropic and independent of the direction of travel of the particle[3].

As previously discussed in Section 2.1.3, the SNO+ liquid scintillator consists of LAB as the bulk solvent and PPO as the primary dopant, or fluor. The addition of a fluor allows for an excited scintillator molecule to non-radiatively transfer its energy to a molecule of the fluor, which then undergoes the de-excitation instead of the bulk scintillator to emit photons. The dopant is usually selected to have a higher quantum yield than the bulk scintillator. As well, the dopant will typically have a different emission spectrum from the solvent and reduce the occurrence of self-absorption, improving the light yield of the mixture.

Prior to the liquid scintillator phase, SNO+ operated as a water Cherenkov detector. The primary source of light in this phase was from Cherenkov radiation, which occurs in a material when a particle travels faster than the speed of light in the medium [118]. Cherenkov radiation is a result of the constructive interference of the electromagnetic waves that occurs in this scenario. The light is also directional

[3]This is not strictly true for all types of scintillators. In particular, the response of both organic and inorganic crystal scintillators can depend on the relative orientation of the particle direction and crystal axes [116, 117].

and emitted in the path of a cone. Figure 2.2 illustrates the concept of Cherenkov radiation in two dimensions. The angle at which the light is emitted is referred to as the Cherenkov angle, or θ_{Ch} in the figure, and it depends on the speed of the particle and the refractive index of the medium.

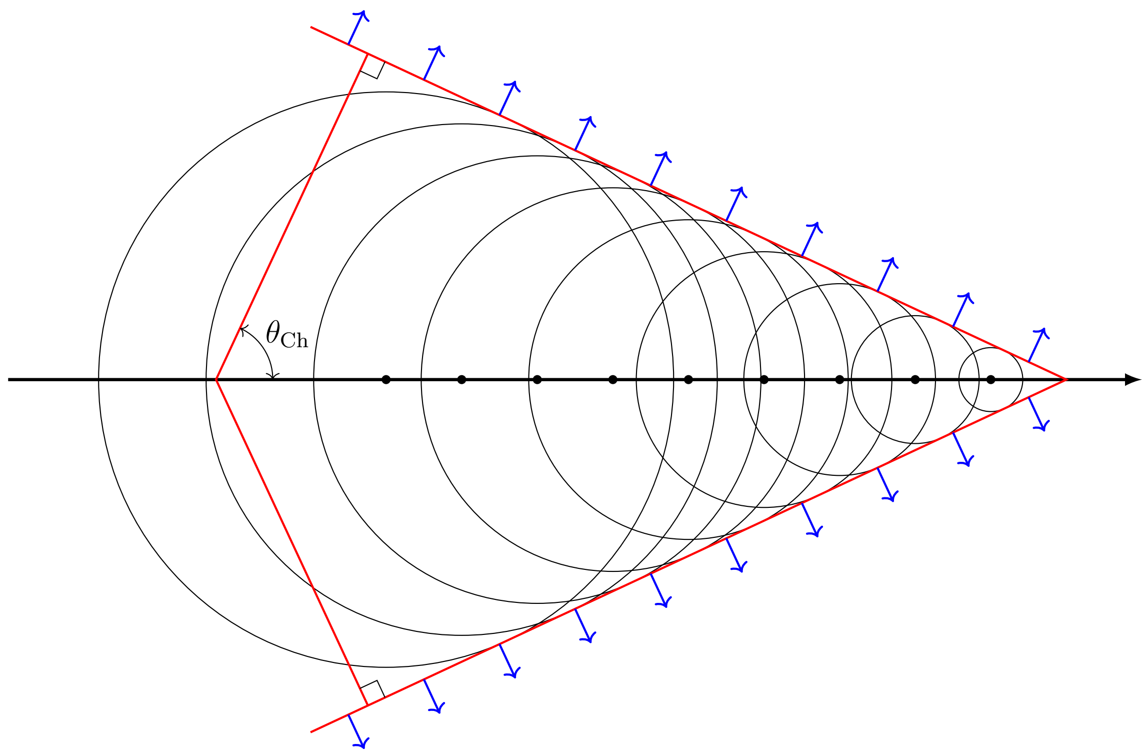


Figure 2.2: Illustration of Cherenkov radiation in two dimensions. Particle trajectory is shown by the thick black arrow, along which are equidistant points of emission. Black circles indicate electromagnetic waves originating from points on the trajectory, while blue arrows indicate the direction of light propagation (perpendicular to the red lines, which are in turn perpendicular to the wavefronts).

While the vast majority of light from an interaction in the SNO+ detector (in the current operating phase) is from scintillation, Cherenkov radiation is a component of the overall light produced when the particle exceeds the threshold velocity. However, scintillation produces far more photons for a given energy deposition, and Cherenkov radiation usually accounts for $<5\,\%$ of the total light output. The Cherenkov light, in addition to being anisotropic, is prompter than scintillation light due to its direct production as the particle moves.

Only charged particles can produce scintillation light. Electrons and positrons will commonly interact in the scintillator medium, travelling short distances of order <1 cm for energies of order 1 MeV and emitting scintillation light in the process. Heavier charged particles, such as alphas, also create scintillation light and deposit more energy per distance travelled. Although neutral particles such as gammas do not excite the scintillator directly, pair production and Compton scattering produce secondary charged particles which do. Gammas travel much further than electrons and positrons, with average path lengths $\sim$50 times greater. Higher energy gammas may also Compton scatter multiple times across a large distance, depositing energy and resulting in light emission at each interaction site.

2.1.5 Data acquisition and processing

In the SNO+ detector, the PMTs detect minuscule amounts of light from particle interactions. Photons will strike some of the PMTs, resulting in a small electrical signal that is sent to the SNO+ trigger system. PMT measurements are temporarily cached by the front end electronics in wait of a global trigger, which is determined by coordinated logic applied to all PMT signals. If a global trigger is not received within a given time window, the cache is cleared. The SNO+ DAQ system is responsible for this logic, and more broadly, for collecting and grouping data from the PMTs into analyzable sets. This section provides an overview of the SNO+ DAQ and processing pipeline with a sufficient level of detail to understand the work in this thesis. More information on these systems can be found in [84].

If the signal from a hit PMT passes the adjustable discriminator threshold, a time-to-amplitude converter (TAC) is started to record the PMT hit time information. Three values of the charge registered at the hit PMT using different integration windows and gain factors – QHS (charge high gain, short integration window), QHL (charge high gain, long integration window), and QLX (charge low gain, short or long integration window) – are also written to the cache. Simultaneously, two fixed-length square pulses of 20 ns and 100 ns are immediately created. These pulses and both a high- and low-gain version of the original hit signal are passed through various stages of the electronics to determine whether a global trigger should be issued. Ultimately, each class of pulse is summed over all hit PMTs, and if any such resulting summed pulse exceeds an adjustable threshold for that class, a global trigger is issued. For example, the sum of all 100 ns square pulses is equivalent to the number of hit PMTs within a 100 ns window and the threshold is set accordingly.

A chain of electronics is used to provide better control over these sums, as well as to divide the detector into more manageable components for maintenance and upgrades. Each PMT is connected to one of nineteen crates, which are in turn grouped into either singles or pairs as part of a rack. PMT interface cards (PMTICs) link a subset of the individual PMTs to the multiple associated front end cards (FECs) of a crate. The PMTICs filter the PMT signals before passing them to the corresponding FECs. These FECs are responsible for buffering raw hit data for the PMTs connected to it, as well as digitizing the data via an analogue-to-digital converter (ADC). Each FEC contains four daughter boards responsible for setting the discriminator threshold that determines whether a PMT has been hit. The daughter boards also issue the trigger signals described in the previous paragraph and contain both an integrator (responsible for the various integral charge values) and complementary metal-oxide-semiconductor (CMOS) chip (responsible for the TAC).

Pulses are then summed at the crate level and passed to the analogue master trigger cards. Each analogue master trigger card is responsible for summing a class of pulse over all crates and passing the result to the digital master trigger card. The digital master trigger card then compares the amplitude of the received pulses against the adjustable thresholds and issues a global trigger if any thresholds are crossed. It can also filter out a given pulse type if specified in the controllable trigger mask. There is a 450 ns delay until it can send a new global trigger once one has already been issued.

Upon receipt of the global trigger by the front end electronics, the TAC is stopped, the charge integration is finished, and all the information in the caches is read out and recorded. This information, which includes the relative time at which each PMT was hit and the charge registered at each PMT, is defined as an “event.” The “event window” over which PMT information is included to create a single event is partially defined by the length of the cache timeout. It takes a total of approximately 220 ns for the digital master trigger card to receive the pulses and the front end electronics to receive the corresponding global trigger. Given that the cache timeout is approximately 400 ns, the event window begins approximately 180 ns before the hit which resulted in a global trigger and ends approximately 220 ns after. Only hits occurring within this window make up an event.

As events are collected, they are bundled together and saved to disk. Events are grouped into short time periods of consistent operation called a “run.” Typically, runs correspond to one hour of detector data, although runs can be manually incremented if detector conditions change or maintenance needs to be performed. A single run consists of one or more files, the number of which is determined by size on disk. The files which make up a run are referred to as subruns if more than one file is needed. The resulting data files are transferred off-site, processed, and stored. Processing consists of various checks and converting the raw data into formats that are more easily analyzable. Included in this procedure is the conversion from raw ADC counts of the PMT timing (TAC) and charge (QHS, QHL, QLX) information to calibrated values in interpretable units (e.g., ns for time).

2.1.6 Simulation and analysis software

SNO+ uses an internal software package for both simulation and analysis called RAT (Reactor Analysis Toolkit). The framework is primarily written in C++ and has several major dependencies. Geant4 [119] is used for the Monte Carlo (MC) simulations and provides detailed modelling of the detector geometry. Interactions within the detector and the corresponding optical photons are produced via GLG4Sim [120], which is directly integrated into RAT for full compatibility. The data are written, stored, and read with ROOT [121]. As well, much of the code within RAT, as well as outside, make use of ROOT's data structures and analysis functionality.

2.2 The PPC HPGe detector at Queen's University

2.2.1 Detector overview

The HPGe detector is located at Queen's University. It is a cylindrical 1 kg PPC detector with a height of approximately 5 cm and a radius of 3 cm. The detector was manufactured by ORTEC/AMTEK [122] and is operated in a PopTop configuration connected to a cryostat/dewar at a temperature of approximately 110 K. The detector is also operated at the manufacturer-recommended bias of 3700 V. The depletion voltage of the detector is 2750 V.

The PPC detector is similar to the ones used in the array of the MJD (now inherited by the LEGEND experiment and used in the LEGEND-200 demonstrator). The bulk of the material is germanium, with the n^+ contact held at a positive potential and the small p^+ point-contact held at ground. The sharper weighting potential over the semi-coaxial detector results in better identification of event site multiplicity, and thus excellent pulse shape discrimination and background rejection [123]. A schematic of a typical PPC detector is shown in Figure 2.3 (correct aspect ratio, but exaggerated thickness of the n^+ contact). The diagram is a vertical cross-section through the central axis of the cylinder (plane perpendicular to the bases) and is azimuthally-symmetric.

Section 2.2.3 describes the DAQ and electronics setup for the detector, and Section 2.2.4 contains details of the processing, analysis, and simulation software. The HPGe detector is, of course, at a much smaller scale than SNO+ and will not be used directly as part of a $0\nu\beta\beta$ decay search experiment. As such, this section does not have an experimental program like in Section 2.1 for SNO+. However, the detector serves as an excellent test of the methods developed in this thesis, which can be applied to large-scale $0\nu\beta\beta$ decay experiments such as LEGEND.

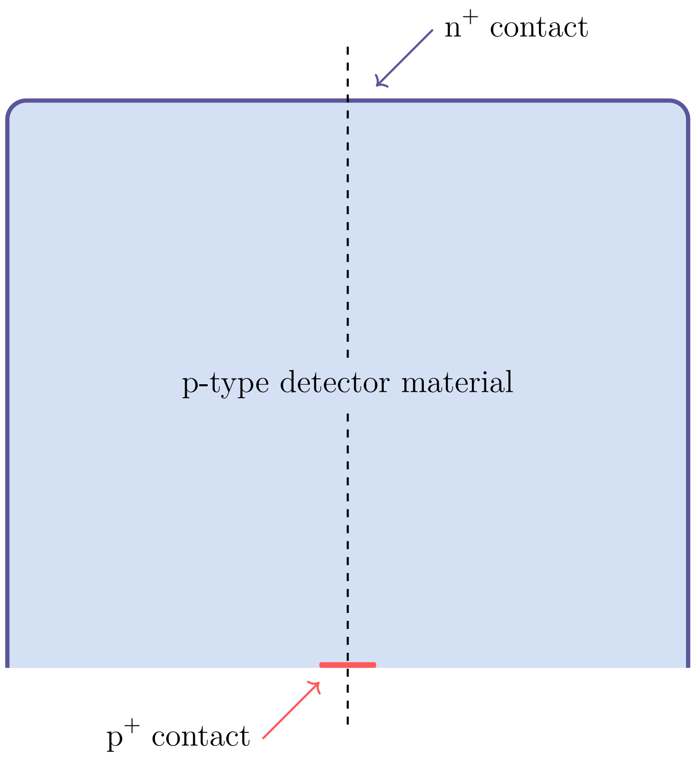


Figure 2.3: A PPC HPGe detector cross-sectional illustration. The cross-section is formed from the intersection of a plane perpendicular to the bases of the cylindrical detector passing through its central axis, which is indicated by the dashed line in the diagram to show the symmetry. The n^+ and p^+ contacts are labelled and shown in blue and red, respectively. The illustration has the correct aspect ratio of the HPGe detector at Queen's University, but the thickness of the n^+ contact is not to scale.

2.2.2 Interaction and detection principles

In HPGe detectors, particles deposit energy by ionizing the germanium atoms in the medium. This releases charge carriers, which are transported by an electric field created by external electrodes. The incoming charge is collected and converted to a voltage, as described in more detail in Section 2.2.3, to create a pulse which constitutes an *event*.

Electrons and positrons do not travel far in the germanium, with typical path lengths under 1 mm. In contrast, gammas may travel a larger distance and deposit energy at multiple sites via Compton scattering. Interactions for which the energy deposition is localized to one site in the detector are referred to as *single-site*, whereas more than one interaction at distinct sites within a given window of time – the event window – is referred to as *multi-site*. The definitions of "event" and "event window" are more formally defined next in Section 2.2.3, but are analogous to the corresponding definitions in Section 2.1.5 for SNO+. Importantly, multi-site events are characterized by the offset collection of charge at different times due to the different distances from the interaction site to the contact, and thus different drift times for the charge carriers. Example simulated single- and multi-site events in the form of pulses are shown in Figure 2.4.

The distinguishing feature between the two classes of events is most noticeable in the current pulse due to the arrival of charge carriers at different times for the multi-site event. For single- and multi-site interactions of identical energy, as shown in Figure 2.4, the multi-site event current pulse will have multiple separable peaks, each with amplitudes smaller than the individual peak in the current pulse of the single-site event.

2.2.3 Data acquisition and processing

As with SNO+, the HPGe detector has a DAQ system responsible for the read out and collection of data into a usable form. The charge carriers released from an interaction in the detector are collected and converted to a voltage by an integral charge sensitive preamplifier. The preamplifier signal is then passed to a 16-bit 125 MHz SIS3316 digitizer from Struck Innovative Systems [124], where the data are sampled

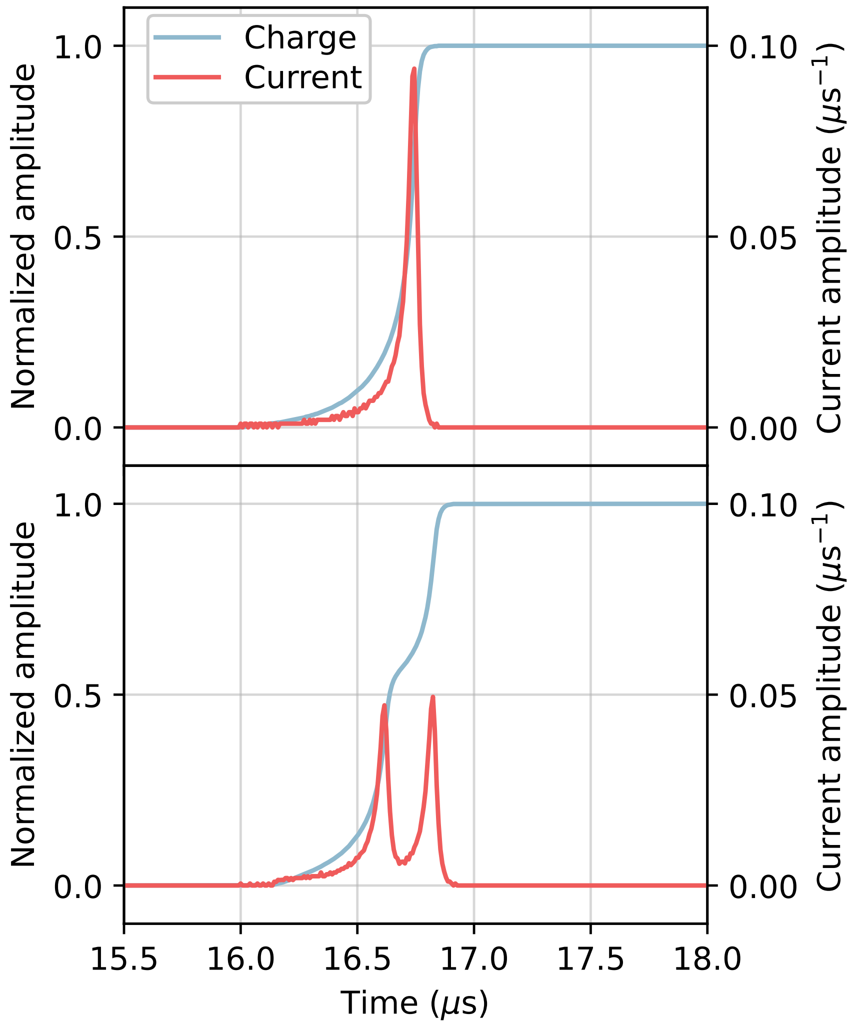


Figure 2.4: Examples of simulated PPC HPGe single- and multi-site events. The single-site event is contained in the top plot, while the multi-site event is contained in the bottom plot. The charge (blue, left y-axis) and current (red, right y-axis) pulses are shown for each.

and converted from an analogue to a digital signal via an ADC. The digitizer has 16 channels, each with its own ADC, allowing for any of these channels to be disabled or disconnected individually.

Data are continuously written to dual circular buffers on the digitizer to facilitate parallel reading and writing. Once one buffer is full, the writer switches to the other buffer to allow for the full buffer to be read, minimizing dead time. A per-channel trapezoidal filter built into the digitizer is continuously applied to the digitized data and is used to create a trigger signal. This signal is compared to a set per-channel threshold, and if it exceeds the threshold, a global trigger is issued. In addition to the internal triggering mechanism built into the digitizer, external triggering logic can be used to supply a global trigger if desired. The global trigger is then sent to all connected and enabled channels, instructing the buffer to be read out and the data to be recorded and saved.

The total number of samples and the number of pre-trigger samples are tunable parameters that determine the exact pulse to be recorded from the buffer upon receipt of a global trigger, and thus fully define the "event window." An "event" is then defined as the information contained within a single event window, typically consisting of one or more charge pulses like those in Figure 2.4 (although real pulses will also contain electronic noise). Although the event window can be any length, recorded pulses are typically a power of 2, specifically either 4096 or 8192 samples long. This allows for easy downsampling without clipping. Given the rate of the digitizer, each pulse consists of a series of voltages at 8 ns intervals. As the voltages are discretized by the ADC, the values of each sample are in units of counts (arbitrary units).

The digitizer also calculates and saves the amplitude of the event using the maximum of a trapezoidal filter, among other metadata, although offline analysis is used in determining a more accurate pulse amplitude with optimized shaping parameters. This trapezoidal filter, unlike the one used for triggering, can have a pole-zero parameter specified.

2.2.4 Simulation and analysis software

The siggen simulation software [125] is used to create highly detailed sets of position-dependent basis pulses, often referred to as "library" pulses. siggen models the propagation of charges in azimuthally-symmetric point-contact (either p-type or n-type) HPGe detectors. A number of parameters, ranging from the detector geometry to operating characteristics such as the temperature and bias, can be tuned to model a specific detector and configuration. Interactions are modelled on a 1 mm × 1 mm grid in radius and height.

Typical data processing and analysis uses custom internal software called QUADIS (Queen's University Analysis of Data in Spheres), which as the name implies, was developed largely at Queen's University. As with RAT for SNO+, QUADIS is largely written in C++ and is responsible for converting the raw data from the digitizer into a format compatible with ROOT. This higher level processed data are typically used for analyses, rather than the raw data. Much of QUADIS uses other ROOT functionality in addition to its data structures, for example for computing the fast Fourier transform (FFT) on pulses and creating histograms.

Chapter 3

Machine Learning

This chapter is largely a review of machine learning with a focus on the aspects relevant to the analyses in this thesis. Section 3.1 contains a basic overview of machine learning to motivate the general formulation of a learning problem. Section 3.2 introduces the concept of neural networks, including how they are trained and some general guidelines on effective optimization. Section 3.3 describes convolutional neural networks, a particular class of neural networks, along with motivations and benefits for this type of architecture. Section 3.4 focuses on autoencoders in the context of neural networks. It largely mirrors the corresponding portion of the author's publication in [1]. Section 3.3 is relevant for Part I, while both Sections 3.3 and 3.4 are needed to follow Part II. This chapter ends with Section 3.5, which discusses considerations on how the data should be divided and preprocessed for model training, validation, and testing.

3.1 The goal of machine learning

Machine learning is a broad field that applies statistical algorithms to learn and extract useful patterns from data. A learning problem can typically be formulated as follows: some response function, f, takes an input, denoted by the matrix X, and produces a meaningful output, denoted by the matrix Y,

$$Y = f(X) + \varepsilon, \tag{3.1}$$

where ε is the irreducible noise term not modelled by f. If ε has a mean of zero, the best prediction is then given by applying f to X,

$$\hat{Y}^{*} = f(X). \tag{3.2}$$

Typically, the true response function f is unknown, complicated, and/or does not have an analytical or otherwise easily expressible form. Instead, some estimation of f is used to model the response, Y, given X. Denoting this approximation of the response function as $\hat{f}$ and calling it the *prediction function*, the predicted output, which can only be as good as $\hat{Y}^{*}$, is instead given by

$$\hat{Y} = \hat{f}(X). \tag{3.3}$$

The goal of machine learning is to *learn* the best prediction function, and thus obtain the predictions, $\hat{Y}$, closest to the true response (or *targets*), Y. $\hat{f}$ is a general class of model parameterized by some number of parameters which can be tuned to best fit the data. The process of optimizing the parameters in $\hat{f}$ by fitting it to the data is often called *training*. The data used to train, or fit, the model (or conversely, for the model to "learn from") are called the *training data*, or collectively the *training set*.

Machine learning is typically divided into two broad categories: supervised learning and unsupervised learning. Supervised learning uses a set of training data with corresponding input and target pairs to find the best mapping between them, as shown in Equations (3.1) and (3.3). In particular, supervised learning algorithms use these known pairs to produce a prediction function with the goal of predicting the targets accurately and generally. With unsupervised learning, the data provided are unlabelled and no known target is given. Instead, unsupervised learning algorithms aim to extract patterns from the data itself, rather than from the explicit mapping of the inputs to their outcomes. Still, the problem can often be formulated as learning a prediction function as per Equation (3.3) where the desired responses are unlabelled. A prominent example of this is clustering, where the data are divided into classes based on similarities in their features. Without knowledge of the target, clustering allows a prediction function to be learned that organizes the data into distinct groups.

Supervised learning is typically further divided based on the type of target. Regression problems are those where the target is continuous, while classification problems are those where the target is a discrete value in a finite set (typically binary or with a few class labels). For classification problems, the output of a model is often a continuous value, but a threshold is applied to divide the prediction into classes based on this value.

To quantify how well a supervised machine learning approach is performing, a loss function (commonly also referred to as an objective function or cost function) between the prediction and known target, $L(\hat{Y}, Y)$, is chosen. The loss function is optimized with respect to the parameters of the model in order to fit the data, and so the choice of this function can have a large effect on the parameters that are determined. In regression problems, the mean squared error is common, while the cross-entropy loss is frequently used for classification tasks. For more complicated problems in modern deep learning, different and specialized loss functions may be required to achieve optimal performance.

There exist many frameworks to learn and build these predictive models. The focus of this thesis is on deep neural networks, and so the following sections introduce neural networks along with explanations of more advanced architectures needed to provide a solid background for later chapters. Reference material is linked throughout for more thorough descriptions of concepts. As well, much of the information provided, such as discussions on neural network optimization and hyperparameters in Sections 3.2.4 and 3.2.5, are more broadly applicable to other machine learning algorithms.

3.2 Neural networks

3.2.1 Concept

Neural networks are a machine learning approach loosely inspired by the workings of the brain. The brain consists of many interconnected neurons that will send signals when excitatory inputs are received from other neurons. A neural network consists of many *artificial neurons*, or *nodes*, whose connections to other nodes are weighted. These nodes are grouped into at least two *layers*, where nodes in a given layer are linked to nodes in the adjacent layer(s). A single node accepts an arbitrary number of input values and computes their weighted sum to produce one output value. Effectively, any given node is making a simple decision by weighing the inputs, or evidence, that it receives. The weights determine how important each connecting node is and how it influences the result. By combining layers of many nodes, more complicated decisions can be made, and with each additional layer, these decisions become more abstract. An advanced predictive model can thus be constructed in this way. Setting the weights of each node manually to fit a certain model quickly becomes untenable as the number of nodes and layers grows, and so a learning algorithm is used to update the weights based on patterns in the inputs such that the neural network can produce a useful output. More details on the optimization process are given in Section 3.2.4.

Layers are processed in order and sequentially[1] starting from the initial inputs. Layers between the input and the output are called *hidden* layers. When many hidden layers are stacked between the inputs and outputs, it is called a *deep* neural network. When each node in a given layer is connected to every node in the previous layer, the layer is referred to as *fully-connected* or *dense*. The node outputs in a layer are often called *features*, such that each feature is an individual value or property corresponding to some characteristic of the data. For the input and output, features are typically real-world quantities, as opposed to the more abstract representations in hidden layers.

A basic illustration of a neural network is shown in Figure 3.1. The input layer x is connected to hidden layer h_1, which is then connected to the subsequent hidden layer. This continues until h_l, which is connected to the final output layer, $\hat{y}$. The output of this layer is the prediction, which is evaluated against the target y. Note that all variables, including the output, are generally vectors to represent a single input and target training pair, of which each contains an arbitrary number of features. The

[1]In general, connections need not be sequential, and more complicated architectures may have multiple connections across layers. However, for the purposes of introducing the topic, this rule will be assumed.

typical vector arrow is omitted in the figure to avoid clutter, though will be used throughout the remainder of Section 3.2.

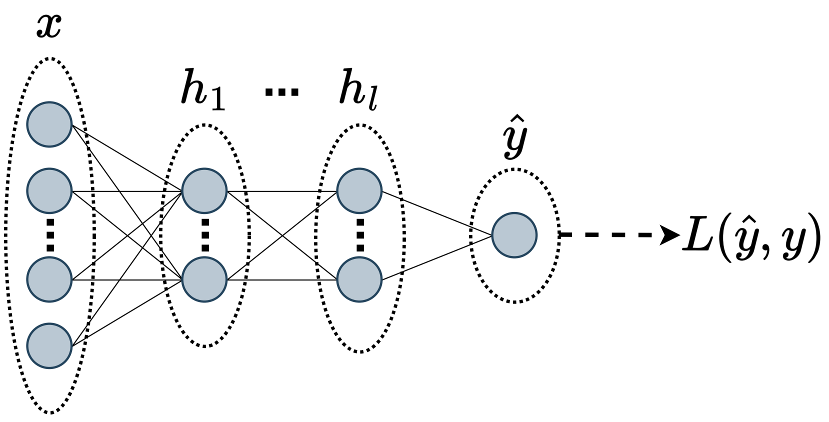


Figure 3.1: Illustrative concept of a basic dense neural network. Every node in a given layer is connected to every node in the previous layer. The network consists of an arbitrary number of hidden layers, h_1 to h_l, each with arbitrary numbers of nodes. The output of the network, $\hat{y}$, is evaluated by computing the loss function, L, with $\hat{y}$ and the target, y. The output layer shown here is of size one, but this is not required.

3.2.2 Node weighting and the forward pass

As previously mentioned, each connection between nodes is weighted. More formally, the input to the j^{th} node in the l^{th} layer (not the input layer) is given by

$$z_j^{(l)} = \sum_{i=1}^{N^{(l-1)}} w_{i,j}^{(l)} x_i^{(l-1)} + b_j^{(l)}, \tag{3.4}$$

where $z_j^{(l)}$ is the input to the node in question, $x_i^{(l-1)}$ is the output from the i^{th} node in the previous layer, $w_{i,j}^{(l)}$ is the weight of the connection from $x_i^{(l-1)}$ to $z_j^{(l)}$, and $b_j^{(l)}$ is a biasing term. The summation is over all nodes in the previous layer,

$N^{(l-1)}$. In general, N with a superscript is used to indicate the number of nodes in the corresponding layer. As per Figure 3.1, $l = 0$ is reserved for the input layer and so Equation (3.4) is only valid for $l \geq 1$.

In vector notation, if $W^{(l)} \in \mathbb{R}^{N^{(l-1)} \times N^{(l)}}$ is the matrix of weights with elements $w_{i,j}^{(l)}$, and $\vec{x}^{\,(l-1)}$ is the column vector of outputs from layer $l-1$, Equation (3.4) can be written more compactly as

$$z_j^{(l)} = \vec{w}_{\star,j}^{\,(l)} \cdot \vec{x}^{\,(l-1)} + b_j^{(l)}, \tag{3.5}$$

where $\vec{w}_{\star,j}^{\,(l)}$ is a column from the weight matrix $W^{(l)}$ corresponding to all weights connected to node j in layer l from the previous layer.

Equation (3.5) can be written yet even more compactly for all nodes in the layer using matrix multiplication,

$$\vec{z}^{\,(l)} = \left(W^{(l)}\right)^T \vec{x}^{\,(l-1)} + \vec{b}^{\,(l)}, \tag{3.6}$$

where $\vec{z}^{\,(l)}$ is the vector of inputs to the layer and $\vec{b}^{\,(l)}$ is the vector of biases for the layer.

Mathematically, a neural network is simply consecutive applications of Equation (3.6) to the inputs of the preceding layer. The entire procedure of calculating the weighted sum at each layer until the predicted output is obtained, along with evaluating the output against the corresponding target by calculating the loss function, is called the *forward pass*. The parameters of the network are updated in what is called the *backward pass*, described in Section 3.2.4.

Given that each layer is simply a series of linear combinations of the previous layer, such a network built from these connections is limited to linear (regression) or linearly separable (classification) problems. Of course, this is quite a stringent restriction, which is remedied by *activation functions*, described next.

3.2.3 Activation functions

Typically, a non-linearity is applied to the weighted sum of each hidden layer (and perhaps the output layer as well) – this allows the neural network to theoretically approximate any function [126,127]. This result applies even to networks with no hidden layers (although this does not mean that accurately modelling any such function is practical).

Denoting the non-linearity as σ and applying it to Equation (3.5) produces the output of the node, which is often called the *activation*. Correspondingly, σ is often called the *activation function*. Using $a_j^{(l)}$ to represent the activation of the j^{th} node in layer l,

$$a_j^{(l)} = \sigma\left(z_j^{(l)}\right) = \sigma\left(\vec{w}_{\star,j}^{(l)} \cdot \vec{x}^{\,(l-1)} + b_j^{(l)}\right), \tag{3.7}$$

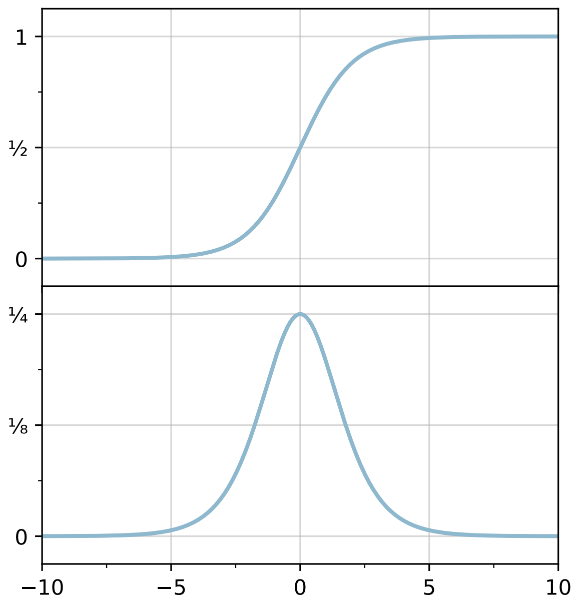
or, applying the function elementwise to Equation (3.6) and using matrix notation, the vector of activations, $\vec{a}^{\,(l)}$, is given by

$$\vec{a}^{\,(l)} = \sigma\left(\vec{z}^{\,(l)}\right) = \sigma\left(\left(W^{(l)}\right)^T \vec{x}^{\,(l-1)} + \vec{b}^{\,(l)}\right). \tag{3.8}$$

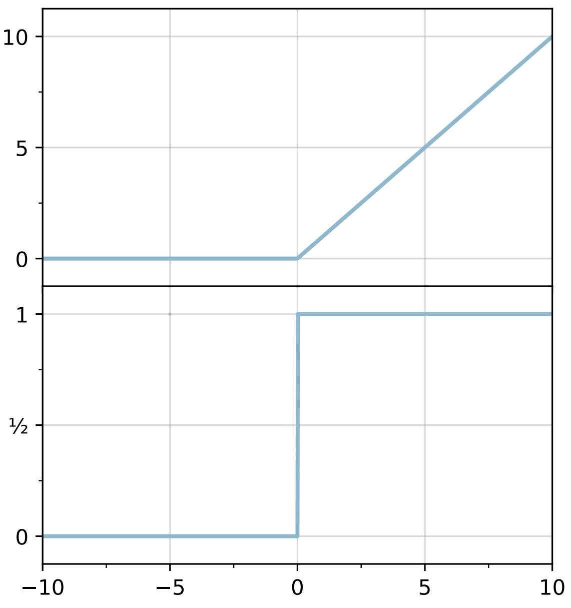
A common activation function is the logistic (specifically the sigmoid) function. Noting that the variable x here is arbitrary and not related to the notation used above, it is given by

$$\sigma_{\text{sigmoid}}\left(x\right) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x}. \tag{3.9}$$

The output of the sigmoid function is bounded between zero and one. However, it suffers from the *vanishing gradient problem* in deep neural networks due to the steep increase to its bounds, and therefore tiny derivatives at larger magnitudes of the input. This has consequences on the optimization procedure and will be discussed in more detail in Section 3.2.4. Figure 3.2a shows the sigmoid function and its first derivative to illustrate these properties.



(a) Sigmoid activation function



(b) ReLU activation function

Figure 3.2: Example activation functions and their first derivatives. 3.2a shows the sigmoid function, while 3.2b shows the ReLU function. Each subfigure contains a plot of the function itself (top) and the function's first derivative (bottom). Both activation functions, and their derivatives, are evaluated over the domain $[-10, 10]$.

An extension of the sigmoid function is the hyperbolic tangent, or tanh, activation function,

$$\sigma_{\text{tanh}}\left(x\right) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}. \tag{3.10}$$

It can shown that the tanh function is simply a scaled and shifted version of the sigmoid function[2] with a range between negative one and one. As such, it still suffers from the problem of vanishing gradients. Empirically, however, the tanh function often performs better than the sigmoid function as an activation, which may be due to fact that its output can be both positive and negative, allowing the gradients for the weights connecting to a node in the following layer to differ in sign [128].

Another common non-linearity is the rectified linear unit (ReLU) function, first employed in [129] and popularized as an activation function for deep neural networks in [130]. The ReLU function is zero for negative inputs and the identity function for positive inputs. The equation is given by

$$\sigma_{\mathrm{ReLU}}\left(x\right) = \max\left(0, x\right). \tag{3.11}$$

The ReLU function is popular as it has the advantage of computationally fast evaluation and its gradient does not saturate with large inputs, thereby avoiding the vanishing gradient problem [130–132]. However, if its input is negative, both its output and gradient will then be zero. In turn, the weights connected to the node will not be updated. When most of the inputs to the node are negative – as can happen with a large negative bias – the node will remain stuck with little chance of an update forcing it out of the zero regime. In this situation, since the node always outputs zero while simultaneously not allowing its connecting weights to be updated, the node is said to be *dead*. These properties are clear from graphical representations of the

[2]Specifically, $\sigma_{\mathrm{tanh}}\left(x\right) = 2\sigma_{\mathrm{sigmoid}}\left(2x\right) - 1$. To prove this, start by factoring out e^{-x} from the numerator and denominator of Equation (3.10) and separating the resulting fraction into two terms. With some more algebra and the definition in Equation (3.9), this relation between the sigmoid and tanh function can be verified.

ReLU function and its derivative, as shown in Figure 3.2b. Despite the aforementioned issues, from empirical observations, the ReLU activation tends to perform well on a wide variety of problems.

There exist many more activation functions that work well in certain situations. Additionally, there exist numerous extensions of the ReLU activation specifically to counter the issues it introduces. Activation functions have limited restrictions, but should have a number of mathematical properties that will positively impact the performance of the learning (described in Section 3.2.4). Additionally, for gradient-based optimization algorithms, the activation function should be continuously differentiable[3].

3.2.4 Optimization and the backward pass

The weights between the connections of a neural network must be optimized for the data in order to obtain a useful prediction function. There are a variety of optimization methods available. However, the most common by far in the field is gradient descent and its extensions.

Without knowledge of gradient descent or any other iterative optimization algorithms, the natural approach to finding the best parameters of a network would be to calculate the partial derivatives of the loss function with respect to each parameter, set them to zero, and solve the series of equations. However, neural networks typically have far too many parameters for this to be feasible. An analytic approach is further complicated by potentially complex activation functions and loss functions.

[3]Practically speaking, the activation function may have a finite number of non-differentiable points, as is the case for the ReLU function. The derivatives at these points can be approximated in various ways, such as by using the left- or right-derivatives around the discontinuity, or the point's subderivatives.

Gradient descent, on the other hand, is an iterative method of finding optima for arbitrary differentiable functions. An intuitive explanation using the notation of Section 3.2.2 is as follows. A small change in the loss function can be written in terms of a small change in one of the weights of the network,

$$\Delta L = \frac{\partial L}{\partial w_{i,j}^{(l)}} \Delta w_{i,j}^{(l)}. \tag{3.12}$$

To minimize the cost function, ΔL should decrease until L reaches its lowest value. To guarantee that ΔL is negative, the change in the weight can be set to

$$\Delta w_{i,j}^{(l)} = -\eta \frac{\partial L}{\partial w_{i,j}^{(l)}}, \tag{3.13}$$

where η is a scaling parameter called the *learning rate*. This learning rate is important as it will determine how fast a solution is converged upon, or whether convergence will occur at all. A learning rate too low may result in the optimization procedure taking an impractically long time to converge on a minimum. The optimizer may also get stuck in a shallow local minimum with no chance of escaping. A learning rate too high may cause the optimizer to never converge as parameter updates will oscillate around, or far overshoot, a minimum in the loss (Equation (3.12) does not hold if $\Delta w_{i,j}^{(l)}$ is large). Some additions to gradient descent adapt the learning rate throughout the optimization procedure, such as by adding a decay factor to the learning rate as the loss function approaches a minimum to increase stability.

With gradient descent, every weight (and bias) is updated simultaneously via the change given in Equation (3.13). This is referred to as the *backward pass*. After the weights are updated, the forward pass is computed again with these new parameters to produce the new predictions, the gradients are recalculated, and the weights are once again updated simultaneously. For a large network, the gradient in Equation (3.13) can be quite complicated. An algorithm called backpropagation is used to efficiently compute the gradients. This algorithm is based off of the chain rule in calculus, and was popularized in [133], although first introduced in [134] and later published in [135].

The implementation of gradient descent used in almost all scenarios is *stochastic* gradient descent. The preceding "stochastic" term refers to evaluating the average gradient over randomly selected batches of data, rather than over the entire training set. As the overall loss is typically an average over the loss function evaluated at each training example, the overall gradient will be an average over the gradient evaluated at each training example. When computed over the entire training set, this can be computationally expensive and slow down learning.

Instead, the overall gradient can be estimated by computing the gradient over a small, random *mini-batch* of data. Over many iterations, since the mini-batch gradient should be statistically close to the real gradient, stochastic gradient descent will learn quicker. Usually, the mini-batches are drawn from the training set without replacement until the training set is exhausted. A full iteration over the training set is referred to as an *epoch*.

A mini-batch size orders of magnitude lower than the number of examples in the training set is typically all that is needed to compute a reasonably stable approximation of the gradient. However, a balance needs to be obtained in computing a statistically accurate gradient while also keeping the mini-batch size low to ensure learning is quick. In the extreme case of using mini-batches of one training example each, the approximation to the overall gradient will be poor and unstable. The approximation will also be particularly prone to outliers. This could result in difficulty converging on a solution. Fortunately, somewhat large mini-batch sizes are computationally feasible due to efficient matrix multiplication implementations in modern machine learning libraries that take advantage of certain hardware processors such as graphics processing units (GPUs) and tensor processing units (TPUs).

In addition to stability and speed, an important property of any optimization algorithm is to find the global optimum of a function, or at least one of the better potential optima. The choice of algorithm, as well as the parameters which define it such as the learning rate, are made in part to avoid getting stuck in poor local optima. Numerous extensions of gradient descent optimization, such as first-order methods like momentum [136] and algorithms utilizing second-order information like those derived from Newton's method to be practical for deep neural networks [137], have been proposed and shown to improve training in many instances.

3.2.4.1 Gradient stability

The gradient of Equation (3.13) can become quite complicated in a large network, as due to the chain rule, it will involve the multiplication of partial derivative terms from all layers that come after. This is particularly prominent in the early layers of deep networks. A number of approaches can be taken to counter this numerically unstable situation. One that has already been mentioned is the choice of activation function. The sigmoid and tanh functions saturate quickly with inputs of large magnitudes, and as a result, their derivatives become too small to appreciably change the optimizable parameters. For a neural network with many layers, each of which contains sigmoid or tanh activations, gradients can rapidly approach zero and prevent learning. This is the reason that the ReLU activation function tends to work so well in practice: it does not saturate, and thus does not as easily lead to vanishing gradients.

Another technique to help improve the stability of training neural networks is to choose an appropriate parameter initialization procedure. When a neural network is constructed, the weights and biases must be initialized to some value for the first forward pass. The initial parameters should all be different, and thus typically random, to avoid nodes in a layer taking on the exact same values. Carefully considering the distribution from which the weights are drawn turns out to be very important. For example, if each weight is initialized from a zero mean unit variance Gaussian distribution, then the resulting summation (as in, e.g., Equation (3.4)) will have a distribution with a much wider variance. Depending on the inputs, the exact weight and bias initialization scheme, and the choice of activation function, large output values can lead to the already discussed vanishing gradient problem due to saturation. Another frequent instability observed is the *exploding gradient problem*, where the gradient grows exponentially due to its dependence on the values of the parameters in layers that follow. Exploding gradients produce weight updates that are too large and never converge.

Whether the gradients “vanish” or “explode” is a symptom of the fundamental numerical instability inherent to the product of many terms. Appropriately initializing the weights to be scaled by the number of nodes in the surrounding layers has a large effect on improving the stability of the gradients, particularly for the sigmoid and tanh activations [138]. Other works are based off of this idea and have taken to developing robust weight initialization schemes for various scenarios, such as when the ReLU activation is used [139] and for other optimization methods [140].

Even scaling the inputs to an appropriate range has a significant impact on the stability of training neural networks. This is discussed in more detail in Section 3.5, which also highlights the different choices that can be made in scaling. As the gradient of the weights connecting the inputs to the first hidden layer will depend on the inputs themselves, it makes sense that the choice of scale will impact the numerical stability of the gradient. More generally, this principle can also be applied to the hidden layers of a network. A procedure called *batch normalization* is used to rescale the inputs to hidden layers [141], and is often applied to deep neural networks in practice.

3.2.5 Hyperparameters

The optimization process described in Section 3.2.4 is applied to all weights and biases of the network. However, the number of nodes in a layer is itself a parameter of the network, as is the number of layers. The learning rate η in Equation (3.13) is also a parameter of the network (or more accurately, the optimization process if using gradient descent). Even the choice of using neural networks over some other technique is a sort of parameter choice. These parameters are not updated during the training procedure, but rather they instead control the training procedure. Such parameters are called hyperparameters to distinguish them from the directly optimizable parameters of a model.

Hyperparameter optimization is a challenging task and currently, there is no straightforward, general approach to selecting hyperparameters of a neural network that generalize to a given arbitrary problem. Empirical evidence and intuition is typically used to guide the selection of hyperparameters. A robust approach is to perform a grid search over all hyperparameters in the model. A *validation set* of data, completely separate from the training set, can be used to select the best performing model on the basis of generalization to unseen data (more on this in Section 3.5.1). However, a full grid search quickly becomes computationally infeasible given more than a few hyperparameters. For example, testing a combination of ten learning rate values with ten different network configurations will require 100 training iterations. Neural networks are known to have both many parameters and many hyperparameters relative to other model choices. The network topology itself can be varied in numerous ways, and more complicated extensions of gradient descent require yet more choices to be made prior to training.

Determining good hyperparameters is a part of the model selection process. Understanding the data can be useful in eliminating a set of model classes or selecting a set which might perform well. A typical first step is to carefully determine the problem to be solved and how to best quantify the performance of a given model. Care must be taken to select a metric which makes sense for the task. Empirical evidence from prior publications can also be used to help guide the selection process. For example, it is now widely understood that convolutional neural networks (described in Section 3.3) perform much better on image classification tasks than dense neural networks, and so there is little point in spending any significant amount of time testing variations of dense neural networks for a similar problem. As well, equivariance and known symmetries in the data may help in determining a specific architecture.

More details on practical considerations for training neural networks, including the selection of hyperparameters and discussions on methodology, can be found in [131, 132, 142].

3.2.6 Overfitting

Overfitting occurs when the model fails to generalize well to new and unseen data. For example, data that are modelled well by a linear function of its independent variables should not use a complicated neural network with many parameters. In this scenario, the increased complexity does not offer much advantage over the simpler model, while also increasing the chances of fitting the data too well. For an overfit model, even the local noise becomes important and the model is then sensitive to small fluctuations. In the extreme case, if the model has far too many parameters for the amount of data, it may "memorize" the exact mapping from the inputs to the outputs without actually learning anything useful.

Overfitting is a problem common to all machine learning algorithms. Even in linear regression, if the data contains the same number of features as data points, a fit can be found that passes through all points. This is most obvious in polynomial regression – a generalized linear model – where the degree of polynomial can be chosen to match the size of the data, and therefore fit the data perfectly. However, neural networks are particularly prone to overfitting due to the large number of parameters in typical network architectures. As such, neural networks tend to be used for complex problems involving large amounts of data.

The most obvious way (not specific to neural networks) to reduce overfitting is to simply provide more data. However, acquiring or generating more data is not always practical or feasible. Data augmentation – transforming existing data slightly to mimic the generation of new data without changing the target – is another method of reducing overfitting. Again, this may not be possible in all situations, particularly if the data are not invariant to certain transformations. There will also be a limit to how well this scales, and eventually, marginal improvements in the performance will become small.

Yet another simple way to reduce overfitting is to reduce the number of parameters in a given model (e.g., layers and/or nodes in a neural network). However, this may result in *underfitting* (insufficient complexity to model the data), slow learning, or other problems. It may instead be easier to have a model with a large number of parameters, and then place constraints or penalties – either directly or indirectly – on their values. Numerous methods of countering overfitting in this way have been proposed both in general and for neural networks to help them generalize better. These processes, known broadly as *regularization*, restrict the model in some way to effectively reduce its complexity. Some of the most common regularization techniques which are used throughout the analyses of this thesis are described below.

3.2.6.1 Kernel regularization

L1 and L2 regularization (corresponding to LASSO (least absolute shrinkage and selection operator) and ridge regression, respectively) place penalties on the size of the weights, or *kernel*, of the network. This takes the form of adding a penalty term directly to the original loss function, which for L1 regularization is

$$L_{\mathrm{L1}} = L + \frac{\lambda_{\mathrm{L1}}}{N} \sum_{i,j,l} \left| w_{i,j}^{(l)} \right| , \tag{3.14}$$

and for L2 regularization is

$$L_{\mathrm{L2}} = L + \frac{\lambda_{\mathrm{L2}}}{N} \sum_{i,j,l} \left(w_{i,j}^{(l)} \right)^2 , \tag{3.15}$$

where λ_{L1} and λ_{L2} are hyperparameters to determine the scale of the respective penalty terms, N is the size of the training set, and the summation is computed over all indices of each weight matrix, i and j, and all indices l corresponding to layers in the network. Inspecting Equations (3.14) and (3.15), it is clear that the terms penalize large weights. In general, large weights may result in small fluctuations of the model producing large changes to the output. Conversely, small weights should result in smaller changes to the output, given small fluctuations. The network is thus less complex if it has smaller weights.

The way in which the weights are minimized is different between Equations (3.14) and (3.15), however. L1 regularization results in the least important weights being shrunk to zero, effectively deactivating the connections between some nodes and therefore acting as a feature selector. L2 regularization, on the other hand, shrinks the weights proportionally to the magnitude of the weight.

Empirical evidence supports the use of kernel regularization in neural networks, and many models in the literature use it to obtain better performance. For further discussions and motivation on L1 and L2 regularization, refer to [131].

3.2.6.2 Dropout

Dropout is another regularization technique that can help to prevent overfitting in deep neural networks [143, 144]. With dropout, a random subset of the nodes in a hidden layer are temporarily removed while the model is trained. Both a forward and backward pass are computed without these nodes, and so all weights connecting to them are not updated. The withheld subset is changed on every iteration (whether it be per mini-batch or per epoch), and the process is repeated. The *drop rate* is the fraction of randomly selected nodes that are removed in a given iteration.

Because the omitted subset is always changed, all weights and biases of the network will eventually be updated. When the trained network is used for inference, all nodes are restored. However, since more nodes, and thus weights, are used in the inference process rather than in training, it can be expected that the input to a given node where the previous layer used dropout will have a larger magnitude. To counter this effect, the parameters are scaled by an appropriate factor to compensate for the change in magnitude of the resulting summation (e.g., in Equation (3.4)).

Dropout is conceptually similar to training multiple separate neural networks at once and taking the average output as the final prediction. Mistakes made by one trained model are less likely to be made by another trained model, even if both models are of the same class and use the same training data. This is because there will be some variance in the results due to statistical fluctuations of training (assuming that something is stochastic, such as the weight initialization or the shuffling of the training data), and so averaging can help to remove these effects. Dropout is not exactly the same, of course, as each "effective" network is not completely independent of the others. Increasing the drop rate will reduce the correlation between the effective networks, although it will also reduce the number of effective nodes in training and may result in slower learning.

Dropout also helps to ensure that no single node is significantly more important than any other node in the network. Like L1 and L2 regularization, dropout reduces how much nodes rely on other nodes, leading to a more robust model. Therefore, small changes in the inputs should result in small changes in the outputs, making the network more stable to unimportant fluctuations. In some sense, dropout is a form of noise addition that helps the network learn to be insensitive to small changes. It is a specific case of the stochastic delta rule [145], as shown explicitly in [146], which samples each weight from a random distribution with a mean of the weight on the forward pass. For further discussions and motivation on dropout regularization in deep neural networks, refer to [131].

3.3 Convolutional neural networks

Convolutional neural networks (CNNs) are an extension to the basic dense neural networks described in Section 3.2.2. While this class of networks work very well for many tasks, they are also quite limited in what they can learn with a given set of data. Dense neural networks cannot easily take advantage of the relation between nodes, which can be important depending on the problem. It is often the case that nodes that are closer together have a closer relationship than nodes further away. For example, with two-dimensional images, neighbouring pixels are usually part of some subfeature. With one-dimensional time-series signals, points far away in time may very well be less important to points in the present. It may make sense to have this information present in the network structure, rather than having the network learn that many relations between nodes are unimportant. Dense networks also have the drawback of having a large number of parameters to optimize, and thus more potential for overfitting. CNNs are, in one sense, a method of implicit regularization.

3.3.1 The convolution operation

To understand CNNs and their benefits, it is helpful to first write the definition of the convolution. Formally, the convolution is between two functions, p and q, and is denoted by the $*$ symbol,

$$(p * q)(t) = \int_{-\infty}^{\infty} p(\tau)q(t-\tau)\mathrm{d}\tau. \tag{3.16}$$

In practical applications, typically an analytic continuous function does not exist and the data are discretized. In this common scenario, the convolution is written as a discrete operation,

$$(p * q)(t) = \sum_{\tau=-\infty}^{\infty} p(\tau)q(t-\tau). \tag{3.17}$$

The function p is called the input, and the function q is called the kernel (often also referred to as the filter or window). Although the summation is from negative to positive infinity, both the input and kernel are typically limited to a much smaller domain. By assuming that terms outside of the domains are zero, the summation can be done over a finite range. The area over which the kernel fully overlaps the input (or vice versa if the kernel size is greater than the input size) is called the *valid* region.

A similar operation to the convolution is the cross-correlation, which for a real-valued function p is given by (in its discrete form)

$$(p * q)(t) = \sum_{\tau=-\infty}^{\infty} p(\tau)q(t+\tau), \tag{3.18}$$

The only difference between Equations (3.17) and (3.18) is that the kernel is not flipped for the cross-correlation. Most machine learning libraries implement the cross-correlation rather than the convolution, but still refer to it as a convolution. This convention will be used throughout the remainder of the thesis in regards to machine learning and neural networks.

Inspecting Equation (3.18), the convolution operation can be thought of as a sliding weighted average, applied over signal p using the weights given by q. An example discrete convolution in one-dimension is shown in Figure 3.3, which uses a kernel of size three applied to an input of arbitrary length. The highlighted region is the convolution evaluated at a single example point. Note that although the stride – the number of discrete units to shift the kernel – is typically one, as is the case in the figure, this value can be changed. Increasing the stride is a form of downsampling, described in more detail in Section 3.3.4.

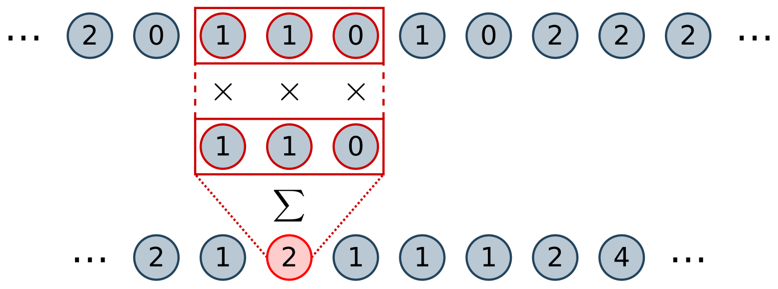


Figure 3.3: Example one-dimensional discrete convolution. The convolution is applied to an input of arbitrary length using a kernel of size three. Highlighted in the figure is an example of the convolution operation evaluated at a single point, including the appropriate operations on the nodes.

For a one-dimensional convolution, the size of the output is directly related to the size of the input, but affected by the window size, stride length, and any potential padding. Formally, the size of the output, O, is given by

$$O = \frac{I - K + 2P}{S} + 1, \tag{3.19}$$

where I is the size of the input, K is the size of the kernel, S is the stride length, and

P is the symmetric padding applied to both ends of the signal. Here, it is assumed that $I \geq K$. When the convolution is computed over the valid region (i.e., $P = 0$) and either the window size or stride length are greater than one, the size of the input will be smaller than the size of the output.

The input signal can also have multiple features, sometimes referred to as *channels*. In this scenario, every entry in the signal has multiple values that correspond to a single point in space or time. A well-known example of data with multiple features are typical two-dimensional images, which can have three channels for the red, green, and blue colour values. Each entry, or pixel, in the image is then represented by a triple, rather than a single value. As well, the digitizer used with the PPC detector for data acquisition supports up to 16 channels, meaning that one event could have multiple corresponding pulses, with each point in time thus having multiple features.

When the input signal has more than one channel, the kernel is extended with independent parameters such that its channel dimension is identical. Mathematically, p and q can then be represented as functions of both the primary dimension and the channel, which is conventionally last. Indexing the D channels by d (colloquially, the "depth" of the input), Equation (3.18) can be generalized to

$$(p * q)\,(t) = \sum_{d=1}^{D} \left(\sum_{\tau=-\infty}^{\infty} p(\tau, d) q(t + \tau, d) \right), \tag{3.20}$$

where brackets are included to support the intuition of computing the convolution at the same point in space or time for each channel and summing the results.

Regardless of the number of channels, the signal (and convolution) is still considered one-dimensional, which refers to the spatial or temporal dimension. In general, an N-dimensional convolution only refers to the N spatial or temporal dimension(s) and never to the channel dimension because the number of channels does not affect the size of the output, per Equation (3.20).

3.3.2 Definition of a convolutional neural network

A CNN is defined as a neural network with at least one layer using the convolution operation, rather than the standard fixed-size weight matrix with unconstrained entries. A single convolutional layer may contain more than one kernel, each with a different set of weights. The resulting output of a convolutional layer will then contain the same number of signals as the number of kernels. The number of weights per kernel is also set in advance of training the network, just as with the number of nodes in a dense layer. As with dense neural networks, the weights of each kernel are optimized in the training procedure, and the number of weights and number of kernels determine the complexity of the network.

A useful implementation detail is that the convolution operation can still be implemented as a matrix multiplication, but with some weights in the matrix fixed between certain entries. Thus, Equation (3.8) still applies when referring to convolutional layers in addition to dense layers.

3.3.3 Benefits of convolutional neural networks

With both the convolution operation and the CNN formally defined, it is easier to understand why CNNs are oftentimes superior to dense neural networks at certain tasks. Because a convolutional layer is only parameterized by the weights of the kernel, rather than a weight representing every connection of every node to the nodes in the previous layer, the number of parameters for a convolutional layer is typically much smaller. This makes a convolutional layer more computationally and memory efficient. Furthermore, these weights are shared in that they are applied to the entire input. With a dense neural network, each weight is applied only once to a single node. Weight sharing can be advantageous as it constrains the model and may force the network to learn more informative weights. This is particularly important for extracting features from the input, especially when such features may appear more than once throughout a given signal. As well, convolutions are translation equivariant, meaning that shifting the input and applying the convolution will produce the same output as applying the convolution and shifting the result. This is useful when the absolute location of a certain feature or property in a signal is unimportant. Finally, convolutions can be used to handle inputs of variable length. As can be intuitively understood in Figure 3.3 and mathematically understood in Equation (3.18), a change in the input dimension simply changes the output dimension. A fixed-size weight matrix, on the other hand, cannot handle data of differing input dimensions without cropping or padding the input. This property of CNNs is utilized for both projects of this thesis.

3.3.4 Downsampling in convolutional neural networks

Typical CNNs use pooling layers after each convolution layer to downsample the output. Pooling takes a neighbouring subset of features and reduces the dimensionality of them to one via an invariant operation. For example, max pooling takes the maximum value of the inputs in the subset, while average pooling takes the mean value of the inputs in the subset. Often, pooling is implemented as an operation applied over a sliding window much like the discrete convolution described in Section 3.3.1.

In order to downsample, the stride must be greater than one[4]. An example of average pooling with a window size and stride length of two is shown in Figure 3.4.

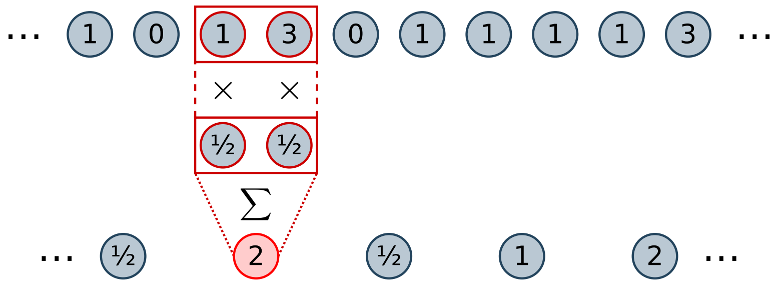


Figure 3.4: Example one-dimensional average pooling. The pooling operation is applied to an input of arbitrary length using a kernel size and stride length of two. Highlighted in the figure is an example of the average pooling evaluated at a single point. The implementation shown here is a specific convolution with fixed kernel parameters and is thus similar to Figure 3.3. The size of the output after average pooling using these parameters is half of the input size.

There are two purposes to pooling. The first is compression, which is applicable to all forms of downsampling. Taking the feature output from a convolutional layer and applying a pooling layer will reduce the dimensionality of the output, removing some information and thereby compressing it. Pooling does not reduce the number of parameters in a CNN itself, but it does reduce the number of overall computations needed. Compression thus decreases the computational expense of a network.

The second purpose of pooling is to make the output approximately invariant to small changes in a local region. This makes the network more robust to shifts, noise,

[4]If using only the valid region and a window size greater than one with a stride of unity, the size of the output will be reduced. However, if the size of the input is much larger than the window, as is typically the case, the reduction in the output size relative to the input size will be small (Equation (3.19) reduces to $O = I - K + 1$, and so with $K + 1 \ll I$, $O \lesssim I$). This situation is thus not considered downsampling in the discussion.

and other small fluctuations in a neighbourhood. Local invariance may or may not be desirable depending on the problem. The exact type of pooling layer will also affect the properties of the invariance. For example, max pooling is insensitive to the exact location of a given feature and thus small shifts, while average pooling is more dependent on a feature's location, but less sensitive to noise within the local region.

Average pooling can be implemented as a general convolution with fixed weights, as highlighted in Figure 3.4. Conversely, the convolution operation is itself a form of average pooling, and with a stride size greater than one, it also acts to downsample its input. Instead of taking a simple average, the convolution takes a weighted average of the inputs in its window. With nonuniform weights, however, the operation is no longer truly invariant, though it can be a good approximation in instances where the input is known to follow a certain pattern.

3.4 Autoencoders

3.4.1 Overview

An autoencoder is an unsupervised machine learning algorithm used to encode data by learning from the data. The idea of the autoencoder was introduced in the 1980s to learn an efficient coding for orthogonal inputs [133, 147, 148]. The concept was further explored and developed in the mid to late 1980s and early 1990s; for example, coupled hierarchical autoencoders were demonstrated to converge much faster on encoding a representation than a typical multi-hidden layer feedforward network at the time [149]. As well, the encoder-decoder structure was made more explicit and deeper networks with multiple hidden layers were used to obtain better nonlinear encodings [150]. Today, autoencoders are widely used for dimensionality reduction, anomaly detection [151], and generative modelling [152]. Autoencoders are also still frequently used for obtaining useful features or representations for other tasks. In [153], for instance, an autoencoder was trained to learn a compressed representation of pulses from germanium detectors, and the encoded input was then used to train another network for pulse shape discrimination.

An autoencoder has two main components: the encoder is a function, f, that produces a latent representation of the input, while the decoder is another function, g, which takes the latent representation and generates a reconstruction of the original input. Using x as the input, y as the internal representation, and z as the reconstructed output[5], the encoder transforms x to y by $f_\theta(x) = y$ and the decoder transforms y to z by $g_{\theta'}(y) = z$. The optimal parameters θ and θ' are selected by minimizing the reconstruction error, L, between x and z. Typically, the encoder and decoder are connected and trained as one model. In almost all references to autoencoders in the literature, both f_θ and $g_{\theta'}$ are neural networks, and further discussion herein will assume this.

3.4.2 Denoising autoencoders

Standard autoencoders can remove some of the input noise if the latent representation is highly compressed, as the network will be forced to extract only the most useful features. For example, in [153], it was observed that the autoencoder removed some of the noise from the inputs, despite not being trained to do so. Denoising autoencoders take this a step further and make the objective to remove noise explicit.

[5]The notation in this discussion differs from that of Section 3.2, and the reader should carefully follow the text as certain variables are reused but with slightly different purposes. As well, lowercase variables representing inputs and outputs are considered as vectors, and the typical arrow is omitted to avoid clutter with other mathematical accents used later.

With denoising autoencoders, the input is instead an artificially corrupted version of the clean signal, denoted by $\tilde{x}$. The encoder is thus a mapping $f_\theta(\tilde{x})$ of the noisy signal to the latent representation, and the decoder a mapping $g_{\theta'}(f_\theta(\tilde{x}))$ of the latent representation to a reconstruction of the clean signal. As the output is the same, the reconstruction error between x and z, $L(x, z)$, is still minimized to train the network. The basic concept of the denoising autoencoder is illustrated in Figure 3.5.

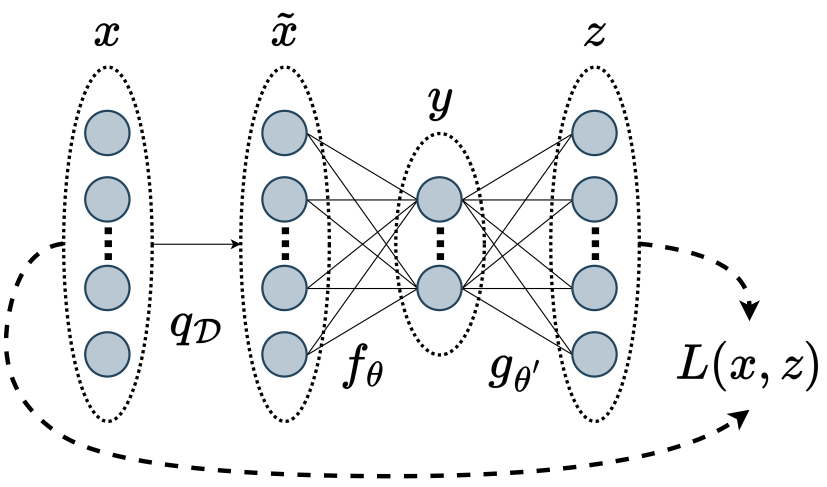


Figure 3.5: Basic concept of denoising autoencoders. An input x is artificially corrupted by some noise process $q_\mathcal{D}$ to become $\tilde{x}$. The encoding portion of the autoencoder, f_θ, produces a new representation y. The decoder portion of the network, $g_{\theta'}$, attempts to reconstruct the clean input. Its estimate is given by z, and a loss function, $L(x, z)$, quantifies the reconstruction. Notation is largely based off of discussion in [154]. Figure adapted from [1].

Denoising autoencoders were originally proposed to extract robust features from the inputs [154, 155]. The primary goal was not to remove noise. Rather, denoising was used as a criterion to produce encoded representations which performed better on a variety of classification tasks.

3.5 Data preparation

The discussion on neural networks so far has largely ignored the data itself and practical considerations with regards to the optimization procedure. This section discusses the general procedures for splitting data and preprocessing it. While the focus in on neural networks, it is applicable to any machine learning technique.

3.5.1 Dividing the data

With machine learning, the data are generally split into three sets: a training set, a validation set, and a test set. The former two terms have already been mentioned in this chapter. As their names imply, the training set is what is seen during training, while the validation set is used to validate the model. However, the same can be said about the test set. These two sets have different purposes, even if both are used to see how well the model generalizes when presented with unseen examples. The validation set is used to select the best hyperparameters of the model. It is typically also used during the training procedure and evaluated at each epoch. This allows for the performance to be evaluated relative to the training set, and makes it easier to determine if overfitting is occurring (if the loss on the training set is substantially lower than the loss on the validation set, the model is failing to generalize and is thus overfitting). It also allows for training to be stopped when the validation loss no longer improves, even if the training loss continues to improve. The test set is used when the final model, including the best hyperparameters according to the validation set, has been selected. It should *never* be used in the training procedure or to select hyperparameters to avoid biasing the results. All final results should be reported on the test set. One must also be careful to avoid *leakage* – using information from the test set in the training procedure inadvertently.

A fairly standard train-validation-test fractional split is 80-10-10 (%). However, the exact fractions used will depend on the size of the datasets available. For example, it may be possible to generate more data at any point, as is the case with simulations. This would allow for the fractional split to be updated dynamically. There may also be a set of data which is unsuitable for training, but useful for validation and/or testing, such as calibration data (due to the fact that events from calibration sources typically do not vary in energy and are only deployed at certain positions, thus leading to biases in the training).

Cross-validation is a slightly alternative approach to the standard train-validation-test split. It instead divides the data into only a training and test set. The test set functions the same as described above. However, a fixed-fraction subset of data is sampled from the training set and withheld from the procedure. Once the model is trained on the complementary subset, it is validated on the withheld data. This process is repeated with a different selection until the entire training set is exhausted. The validation performance is then taken as the average over the results from each iteration. As cross-validation requires training the same model with the same hyper-parameters multiple times, the choice of the fraction, or whether to split the data into folds or to test all combinations of leaving out data, will impact how quickly it can be performed. The advantage of cross-validation is that it reduces the variance of the estimate of the prediction error and takes advantage of training with more data (or allows for more data to be withheld for the test set).

Cross-validation is particularly useful when the amount of available data is low and when the machine learning algorithm is quick to optimize. Cross-validation is not used in the analyses of this thesis as they are not limited by the quantity of data available. It is also cost-prohibitive with even modestly sized neural networks.

3.5.2 Preprocessing

Most machine learning algorithms, including neural networks, require that the inputs are on a "reasonable scale." What a "reasonable scale" constitutes is not well defined and can change depending on the problem and network architecture. Usually, at minimum, the inputs should roughly range between 0 and 1 or -1 and 1. Theoretically, rescaling may not be strictly necessary because any scaling in the inputs can be countered by a scaling of the weights and biases, leading to exactly the same outputs. This becomes more challenging when techniques such as L1 and L2 regularization are used, although in principle, an appropriate scaling of the penalty term can be done. Practically, however, such scaling helps with convergence in the optimization process, as was explained in the discussion on gradient stability in Section 3.2.4. Furthermore, if a certain feature is numerically larger, the loss function will naturally be dominated by the term involving that feature and will thus place less importance on the other features, which is likely undesirable. Typically, the outputs should be rescaled as well for the same reasons as for the inputs: to ensure gradient stability and (if multiple independent outputs) to avoid one output dominating the loss function and being erroneously prioritized.

There are a variety of methods to scale the inputs. Min-max normalization simply rescales the data to the range $[0, 1]$ based on the minimum and maximum of the data. This method can be very sensitive to outliers. However, it is useful if there is some known bound on the data that ensures outliers will not be present. These bounds can be used in place of the computed minimum and maximum.

Standardization is another rescaling method in which the data are normalized to have a mean of zero and a standard deviation of unity. Again, this method can be sensitive to outliers as the sample mean and standard deviation must be computed on the data to estimate the true corresponding terms. It does have the advantage of centring the data, and thus allowing the inputs to the first hidden layer to take on different signs, which may have a similar effect to the tanh function described in Section 3.2.3.

Any sort of preprocessing must be done carefully to avoid leakage. For example, if normalizing or standardizing the data, only the training data should be used to compute the sample estimates.

Part I

Applications of Neural Networks and Deep Learning to the SNO+ Detector

Chapter 4

Event Reconstruction in the SNO+ Detector

The raw data collected from experiments such as SNO+ do not contain information that is immediately usable for physics analyses. Useful properties of each event must first be calculated in order to be easily interpretable by analyzers. Some of these properties are fundamental to the interaction, and they are said to be *reconstructed* from the raw data. These properties, which include the position, direction, time, energy, and particle type, collectively form the *event vertex* for one or more interactions making up an event in the detector.

This chapter explores the use of neural networks for event reconstruction in the SNO+ detector with a focus on position reconstruction. Section 4.1 provides a background on traditional, likelihood-based vertex reconstruction in SNO+. This section highlights the importance of event reconstruction and provides a background to justify the development and adoption of complementary approaches. Section 4.2 presents a novel deep learning-based approach for event reconstruction, describing the motivation, model development (including design choices based on the data and problem), and training procedure. Results from applying the model developed here are presented in the next chapter.

4.1 Traditional event reconstruction

The traditional vertex reconstruction approach that is used for events in the SNO+ detector is based on a maximum likelihood estimation. The most probable event vertex is determined by the pattern of PMT hits and their relative hit times (along with, perhaps, the charge collected at each hit PMT). The likelihood approach used by SNO+ makes the assumption that an event is caused by a single electron interacting in the detector. While this is obviously not always the case, other methods are used to predict the particle identity.

More formally, to determine the position of a given event, a likelihood function based off of the *time residuals* for each hit PMT is maximized. The time residual is defined between an event vertex and PMT indexed by i as

$$T_{\mathrm{res},i} = T_{\mathrm{PMT},i} - T_{\mathrm{fit}} - T_{\mathrm{transit}}\left(\vec{x}_{\mathrm{PMT},i}, \vec{x}_{\mathrm{fit}}\right) \tag{4.1}$$

where $T_{\mathrm{PMT},i}$ is the hit PMT time, $\vec{x}_{\mathrm{PMT},i}$ is the hit PMT position, T_{fit} is the fitted time of the event, and $\vec{x}_{\mathrm{fit}}$ is the fitted event position. The last term of Equation (4.1), T_{transit}, is known as the *transit time* or *time of flight* and is a function of $\vec{x}_{\mathrm{PMT},i}$ and $\vec{x}_{\mathrm{fit}}$. In the simplest case, it is given by

$$T_{\mathrm{transit}}\left(\vec{x}_{\mathrm{PMT},i}, \vec{x}_{\mathrm{fit}}\right) = \frac{\left\|\vec{x}_{\mathrm{PMT},i} - \vec{x}_{\mathrm{fit}}\right\|}{c_{\mathrm{avg}}}, \tag{4.2}$$

where c_{avg} is the mean group velocity of light in the medium. However, with the SNO+ detector, the time of flight is more complicated as the photons travel through the detector medium, the AV, and the external water outside of the AV to reach the PMTs. Furthermore, refraction at the AV boundary will occur unless the photon hits directly perpendicular to the surface, although this effect is typically ignored and $T_{\text{transit}}\left(\vec{x}_{\text{PMT},i}, \vec{x}_{\text{fit}}\right)$ is usually represented as the sum of the individual time of flight values between the three media under the assumption of a straight line path.

The distribution of $T_{\text{res},i}$ is expected to be centred about zero with a prominent Gaussian peak, since the difference between the PMT hit time and event time should be the time of flight in the simplest case. The full probability density function (PDF), $P\left(T_{\text{res},i}\right)$, is more complicated and accounts for reflections, properties of the PMTs such as noise, and other effects. The result is a distribution with a long tail and several secondary peaks. It is derived from simulations and calibration sources.

T_{fit} and $\vec{x}_{\text{fit}}$ are chosen to maximize the likelihood function and thus represent the most probable time and position for a given event. The likelihood function is defined as the product of the individual probabilities over the number of PMT hits (Nhits),

$$\mathcal{L}_{\text{vertex}} = \prod_{i=1}^{\text{Nhits}} P\left(T_{\text{res},i}\right). \tag{4.3}$$

Typically, the problem is instead formulated as a minimization, and is done of over the logarithm of the likelihood for numerical stability,

$$\begin{aligned} -\log\left(\mathcal{L}_{\text{vertex}}\right) &= -\log\left(\prod_{i=1}^{\text{Nhits}} P\left(T_{\text{res},i}\right)\right) && (4.4) \\ &= -\sum_{i=1}^{\text{Nhits}} \log\left(P\left(T_{\text{res},i}\right)\right). && (4.5) \end{aligned}$$

Equation (4.5) is minimized with some optimizer which iterates over the parameters and performs updates that increase (decrease) the likelihood (negative log likelihood). The SNO+ technique uses Powell's method [156], rather than something like gradient descent, due to the fact that the PDFs are not differentiable. Still, some of the same ideas and general issues discussed in Section 3.2.4 are relevant, even if the details differ (e.g., numerical stability).

The current approach to reconstruction used by the collaboration involves a cascade of various algorithms. The main algorithm in the chain minimizes the negative log likelihood function as in Equation (4.5) to determine the event position. However, although the likelihood optimization is among the first in the procedure, several algorithms are run beforehand. The reason for this is that the parameters to be fit must be initialized to some value, which is often referred to as the *seed*. As was the case with backpropagation for neural networks, the initial values can have a large effect on the success of the optimizer. Rather than initializing the positions and times randomly, a simpler and quicker method is used to compute a "reasonable" position that is better than an uninformed guess. This algorithm is based on the principle that only four PMT hits are needed to analytically compute the position and time of an event[1]. Additionally, uncalibrated or poorly calibrated PMTs are filtered out prior to the likelihood optimization, and some basic (unseeded) classifiers are run first.

Other quantities of the vertex, such as the energy, are also optimized. The most probable energy is computed after the best event position and time have been calculated. In SNO+, the energy is estimated based off of Nhits. For moderate energies in the range of ~1 MeV to ~3 MeV, it is expected that the relation is mostly linear. A correction is applied to account for multiple hits on the same PMT in a given event, which is not resolvable using the PMT charge information due to its low accuracy. The deviation from a linear relationship between energy and Nhits increases for

[1]Only four hits are needed as there are four parameters to be fit when setting $T_{\mathrm{res},i}$ to zero in Equation (4.1). Of course, in practice, the *accuracy* of the computed position and time depends on which four PMTs are chosen, the path of the light from the event to the hit PMTs, and the noise associated with the hit times.

higher energies, where multiple photons hitting the same PMT becomes more likely. As well, the correction factor is dependent on the position of the event. For decreasing energies below the ∼1 MeV scale, the relationship again becomes more nonlinear due to scintillator quenching.

In the SNO+ water phase, a likelihood approach based off of information from the Cherenkov light of an interaction was used to reconstruct the event direction. Cherenkov light is directional and emitted at an angle dependent on the refractive index of the medium and speed of the particle, allowing for the most probable direction to be calculated based on the observed PMT hit data. The likelihood is thus a function of the inner product of the fitted event direction and the vector pointing from the fitted event position to a hit PMT,

$$\cos\left(\theta_{\gamma,i}\right) = \vec{u}_{\text{fit}} \cdot \frac{\vec{x}_{\text{PMT},i} - \vec{x}_{\text{fit}}}{\left\|\vec{x}_{\text{PMT},i} - \vec{x}_{\text{fit}}\right\|}, \tag{4.6}$$

where $\vec{u}_{\text{fit}}$ is the (normalized) direction vector of the event and $\theta_{\gamma,i}$ is the angle between this direction vector and the straight line path from the event position to the i^{th} PMT. Though it ignores refraction at the AV and other effects, the latter term is an approximation of the emitted photon direction. As such, $\theta_{\gamma,i}$ is often called the *photon angle*.

The position can either be optimized simultaneously with the direction, or remain fixed to its value as determined from minimizing Equation (4.5) first. In the latter scenario, the likelihood function for the direction would look similar to Equations (4.3) to (4.5), with the PDF replaced with $P\left(\cos\left(\theta_{\gamma,i}\right)\right)$ and $\vec{x}_{\text{fit}}$ held fixed at its previously estimated value. The distribution of Equation (4.6) is centred around the Cherenkov emission angle in the medium since the light in the water phase for a typical event was from Cherenkov radiation. As with the time residual PDF used for position reconstruction, the $\cos(\theta_{\gamma,i})$ PDF is created from simulations of the detector and models numerous other effects.

The directional Cherenkov light in the scintillator and tellurium phases, on the other hand, is only a small fraction of the total light emitted for a typical interaction. Isotropic scintillation produces the vast majority of the light in these media, making the distribution of $\cos(\theta_{\gamma,i})$ nearly flat. A likelihood function based solely on Equation (4.6) thus offers almost no directional information in these phases. However, Cherenkov emission is prompt while the molecular excitation and de-excitation process that produces scintillation light is comparatively slow. The difference in emission times will depend on the exact scintillator and its timing profile, but the Cherenkov light should be separable given sufficient photon detection time resolution. Using this principle, event-by-event directional reconstruction has been demonstrated to be possible for the SNO+ scintillator data using a more complicated two-dimensional PDF consisting of $T_{\text{res},i}$ in one dimension and $\cos(\theta_{\gamma,i})$ in the other [112]. As shown in Figure 4.1, a clear Cherenkov peak can be seen at the lower time residuals in the PDF for liquid scintillator simulations.

After the event position and time are optimized, their values are used to compute numerous other quantities which are then applied to analyses. Some of these quantities include other components of the event vertex (e.g., direction and energy), as already discussed. Other parameters which depend on the event position include the goodness of fit and the uncertainties associated with the respective vertex reconstruction. The event vertex is also used to compute classifier values, such as the fraction of hits within a given time residual window or likelihood ratios to identify

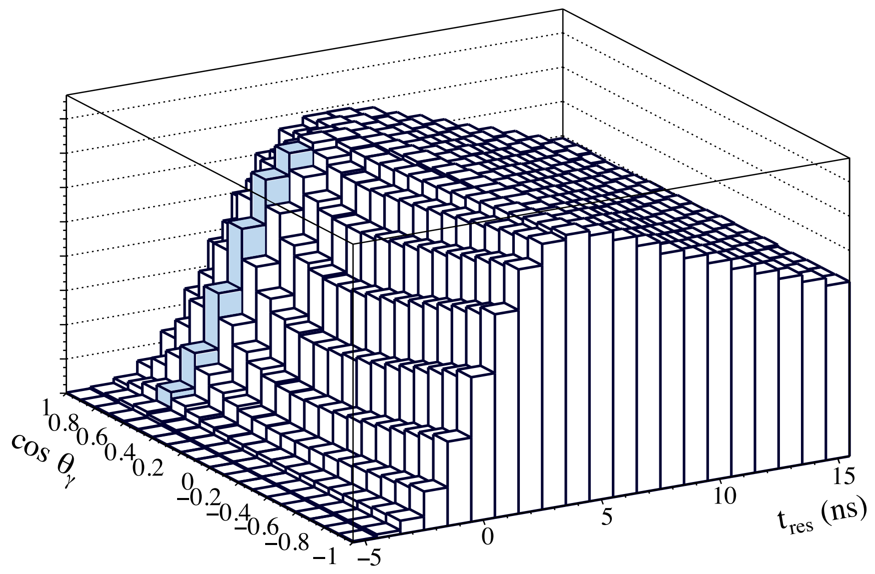


Figure 4.1: Joint PDF of the photon angles and time residuals of hit PMTs. The PDF is generated for 6 MeV electrons simulated uniformly in LAB with 0.6 g/L PPO in the partial fill phase. The Cherenkov peak is visible in the photon angle for early time residuals in the −3 ns to 2 ns range and highlighted in blue. Figure from [112].

multi-site events, which are in turn used to remove backgrounds. Additionally, the event position is directly used to reject events outside of a specified fiducial volume (FV) in order to remove radioactive backgrounds from the AV and PMTs. The event direction could be used to reject solar neutrinos in the 0νββ decay ROI via a correlation with the position of the Sun at the time of the event, or to identify high energy solar neutrinos as is done for ^{8}B in [112]. Overall, the performance of the optimization algorithms in determining accurate event vertices are very important in order to analyze SNO+ data.

4.2 Deep-learning based event reconstruction

4.2.1 Motivation

Given the importance of vertex reconstruction, especially the position, researching methods to achieve superior performance (in terms of accuracy, convergence success, and/or speed) is worthwhile. One relatively unexplored area for SNO+ is different optimization techniques, which may offer improvements in convergence and finding global (or better local) optima. Another approach is to consider completely different alternatives to the traditional maximum likelihood estimation method. Given the success of machine learning – particularly deep learning – in other domains, its use for reconstruction at the Large Hadron Collider (LHC) and more broadly in the high-energy physics community [157–159], and its ability to extract complex patterns from very large datasets, this section focuses on developing deep neural networks for event reconstruction in the SNO+ detector as a secondary and complementary method.

The largest problem for the SNO+ chain of reconstruction algorithms is the speed of computation. The likelihood optimization to calculate the position and time takes an average of about 0.05 s/ev for events at 1 MeV, and nearly every other algorithm down the chain depends on this result. Furthermore, many of the classifiers that run afterwards are slow and dominate the overall event reconstruction time. The traditional approach is also difficult to parallelize for a single event, given the sequential structure and dependence of algorithms on prior results in the chain. At the typical $\sim$2 kHz rate of virtually continuous data collection, improvements in computation time are valuable. Ignoring issues of speed, it can also be useful to have a secondary, independent algorithm for vertex reconstruction in order to compare it to the traditional method and determine any discrepancies or issues with the performance of either approach.

With machine learning, the bulk of the time is often spent training the model. Once a trained model exists, inference is typically much faster. Neural networks are particularly well known for requiring a fairly large computational investment in exchange for fast inference later. Furthermore, neural networks consist of sequences of matrix multiplications, for which existing efficient implementations exist. These matrix multiplications can take advantage of GPUs, making them even faster, and modern libraries provide interfaces to ensure that utilizing GPUs is relatively straightforward. Overall, a trained neural network can be hundreds to thousands of times faster than the traditional maximum likelihood estimation approach.

4.2.2 Development of the model

Developing a machine learning approach to vertex reconstruction is not straightforward as it requires the use of low-level detector data. Many machine learning frameworks require that the inputs to the algorithm are of a fixed length. However, for data from SNO+, the value of Nhits is dependent on the energy of the event and will vary even for the same energy due to the stochastic nature of the interaction and emitted light. One way to handle this problem is to provide a fixed-length vector corresponding to each PMT to the algorithm, padding the non-hit PMTs with a default value such as zero. However, this naive solution performs poorly due to the sparsity of the inputs. In the scintillator phase, the ratio of Nhits to energy is approximately 250 Nhits/MeV. For a 2.5 MeV event (about the Q-value of the $0\nu\beta\beta$ decay of ^{130}Te), the typical fraction of hit PMTs will only be $\sim$6 %. Furthermore, in the tellurium phase where the light yield is lower, the sparsity of the data will become even worse.

Another approach that was considered is to create a separate neural network for each possible value of Nhits. This was only considered in the water phase of the experiment, where Nhits for a typical event was an order of magnitude lower. Despite the fact that it performed reasonably well in the water phase due to completely eliminating the issue of sparsity, this approach has numerous disadvantages. In particular, it reduces the size of the training set for each individual network, is computationally expensive for training and hyperparameter optimization, and does not scale well with increasing Nhits, making it completely infeasible in the scintillator and tellurium phases.

Other experiments, such as MicroBooNE (Micro Booster Neutrino Experiment) [160–162] and NOνA (NuMI Off-axis ν_e Appearance) [163–165], have had success with CNNs for various tasks, especially particle identification. CNNs, due to their shared weights and equivariant nature, are particularly good at image classification and segmentation by learning abstract features about the data (e.g., identifying edges). However, the detector structures of experiments like MicroBooNE and NOνA naturally lead to two-dimensional rectangular images that are suitable for the standard CNN architecture. Projecting the three-dimensional spherical SNO+ detector onto a two-dimensional image is not straightforward, as any projection will always result in distortion[2] (i.e., no projection can map great circles to straight lines *and* preserve angles).

Using instead a three-dimensional grid and applying three-dimensional convolutions is computationally inefficient as the light is registered on the PSUP, which is

[2]This can be shown or proven in a number of ways; for example, by comparing the sum of angles of a planar triangle (always π radians) and a spherical triangle (always greater than π radians). More generally, it is a consequence of Gauss's *Theorema Egregium*.

essentially a two-dimensional surface, rendering most of the volume unused. Furthermore, with nearly 10 000 PMTs, resolving each PMT individually would require a grid consisting of $\sim 10^{15}$ individual points. A network of even moderate size applied to such an input would be computationally prohibitive without utilizing an intelligent sparse matrix representation. A coarser grid would be more tractable, but information would be lost due to binning.

In [166], the authors applied conventional CNNs to identify a particular background against the $0\nu\beta\beta$ decay signal in a spherical liquid scintillator detector with a similar configuration to KamLAND-Zen. Structurally, events from the KamLAND-Zen detector are comparable to those from the SNO+ detector, and so this work is perhaps the most directly applicable use of deep learning to the experiment. In their publication, each event is represented by a series of two-dimensional image maps to be fed into the CNN, where each map corresponds to the hit PMTs in a fixed interval in time within the event. The map itself is structured as a grid with the polar angle on one axis and the azimuthal angle on the other, much like a pixelated image. This has the same issues of distortion mentioned above, though the approach still worked reasonably well for the task.

More recently, the KamLAND-Zen collaboration has developed a more complicated spherical CNN with a long short-term memory (LSTM) mechanism to handle the rotational symmetry of the detector and to better preserve the temporal ordering of the feature maps, respectively [167]. It was used in the first results from KamLAND-Zen 800 for identifying time periods of high-backgrounds, leading to improvements in the ^{136}Xe $0\nu\beta\beta$ half-life limits [60]. While this technique performs well on classification problems, and would presumably extend to SNO+, position reconstruction does not necessitate spherical symmetry as it depends on the absolute positions of the hit PMTs. In fact, spatial invariance is *not* a desirable property for a model when its goal is to determine the position; while shifting the location of an event should produce the same particle identification under ideal circumstances, such a shift should produce a different event position by definition. It will become evident in the next section that an LSTM mechanism, or similar, is also not needed to preserve temporal ordering for the architecture developed in this thesis.

CNNs can be used in a different way, as was alluded to in Section 3.3.3. Because of their shared weight feature, they can accept inputs of variable size. While this property is not usually needed for a detector configuration which can easily be projected onto a two-dimensional surface, it can be used to construct a network to handle data of a variable input length. This completely eliminates the issue of sparsity in the detector. The next section provides details on the model and network architecture that is used to reconstruct events. Although it is applied specifically to position reconstruction, the architecture is flexible, and with some minor changes can be used to reconstruct other quantities in an event vertex.

4.2.3 Description of the model

4.2.3.1 Inputs and outputs

Low-level detector data from the PMT hits are used in primary vertex reconstruction, as described in Section 4.1. Each hit PMT contains a corresponding hit time, along with three values of charge which use different integration windows and gain factors. An example of two events in the SNO+ detector are given in Figure 4.2. Figures 4.2a and 4.2b show an event near the centre of the detector, while Figures 4.2c and 4.2d show an event near the AV. Each pair of images consist of a mapping of the event onto a two-dimensional surface and a histogram of the PMT hit times for the event. The sinusoidal projection is used for illustration, and the colour indicates the value of the hit time (converted from the uncalibrated ADC TAC value to ns), which also corresponds directly to the units of the x-axis in the histograms. XSNOED (X-Windows SNO(+) Event Display) – an event visualization software tool originally developed for the SNO experiment and now adapted for SNO+ [168, 169] – was used to generate these plots. No charge information is included in the figure.

Each PMT is also located at a fixed position, which can be incorporated into the individual hit information. Since a given PMT position does not change on an event-by-event basis, it can either be explicitly provided to the model or implicitly provided via the structure of the inputs and network. As discussed in Section 4.2.2, there are several reasons to prefer the former, and so the model developed here directly includes PMT positional information. Further justification will also become apparent when the architecture is presented in Section 4.2.3.2 which follows.

The charge information in SNO+ is not considered reliable. Furthermore, it is not used in the likelihood-based optimization since it is expected to have little effect on position reconstruction. Therefore, charge information is not included in the data used to train the model. The inputs to the network thus consist of the Cartesian coordinates of each PMT along with the PMT hit time. For a given event, the input is represented by a two-dimensional array of size $(\text{Nhits}, 4)$. Essentially, the input is treated as an unstructured point cloud. The ordering of the rows is arbitrary and it will be mathematically shown to have no effect on the output.

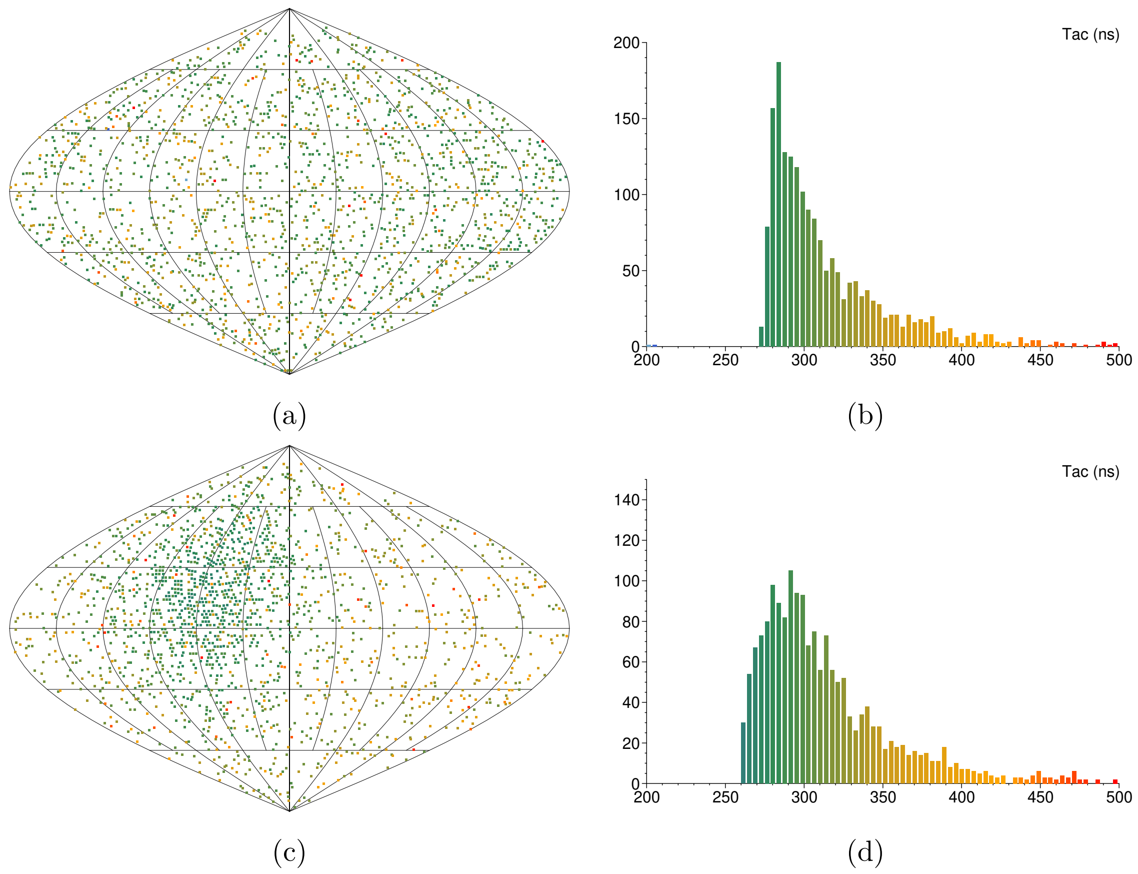


Figure 4.2: Example simulated events in the SNO+ detector visualized using XSNOED. 4.2a and 4.2b show an event near the centre of the detector. 4.2c and 4.2d show an event near the AV. The projections in 4.2a and 4.2c are sinusoidal. The histograms in 4.2b and 4.2d are of the PMT hit times in ns. Each bin is given a unique colour to identify the PMT hit time on the event projections.

Using the terminology defined in Section 3.3.1, each position coordinate of a hit PMT and the hit time can be thought of as separate channels. Providing additional channels to the data is straightforward and may be useful in certain scenarios. The most obvious extension would be to provide the charge information, which although not included for the purposes of the analyses in this thesis, could be useful for energy reconstruction and in background rejection tasks.

The output of the model is the position component of the event vertex, again in Cartesian coordinates. The reconstructed position corresponds to a single point within the detector volume in three-dimensional space[3]. Section 4.2.3.2 describes the network architecture along with more details on how the inputs and outputs are structured for training and inference.

4.2.3.2 Network architecture

The network can be divided into two distinct components: a feature extractor and a predictor.

Feature extractor The feature extractor takes in the (Nhits, 4) vector of inputs for a given event. The feature extractor consists of several convolutional layers applied sequentially, starting with the input. Each convolution is done with a stride and window size of unity. This transforms each hit PMT into another representation. Since the window size is one, information between PMTs is never pooled together – a window size greater than one would remove the network's permutation invariance property, and so this is handled later.

At each convolutional layer, the number of filters can be arbitrarily chosen as this parameter is not constrained by the shape of the previous layer. However, it will impact the size of the kernel in the next layer and thus the overall number of parameters in the network. Due to the one-by-one convolution implemented, the shape of the output of a given convolution layer will be (Nhits, $F^{(l)}$), where $F^{(l)}$ is the number of filters in the l^{th} layer of the feature extractor (the first dimension does not

[3]Though not shown explicitly in Figure 4.2, the event position could be projected onto the PSUP and represented in these maps as a single point, though radial information would be lost in such a representation.

change per Equation (3.19) with $K = 1$, $S = 1$, and $P = 0$). A diagram of the feature extractor architecture is shown in Figure 4.3. This illustration contains an arbitrary number of layers (left-to-right), an arbitrary number of filters per layer (in-to-out), and an arbitrary number of PMT hits (top-to-bottom). The input corresponds to the PMT hit information from one event. F (without a superscript) denotes the number of filters in the final convolutional layer.

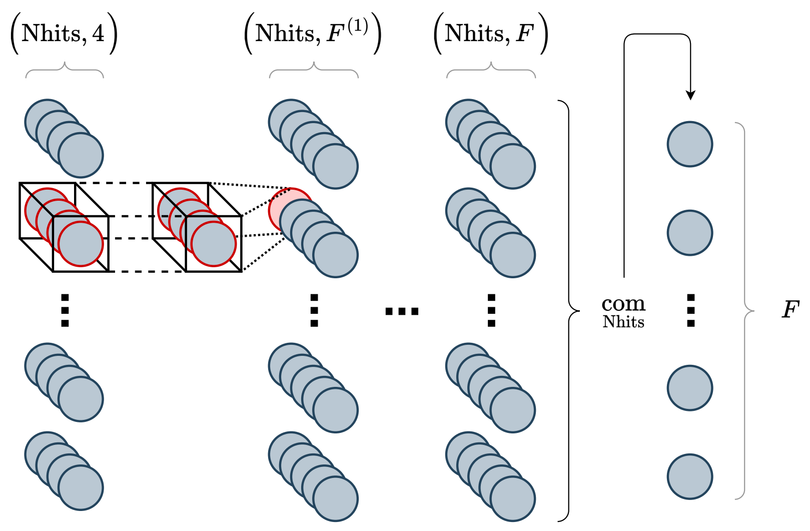


Figure 4.3: Illustrative diagram of the CNN feature extractor concept. The architecture contains a sequence of convolutional layers and a final commutative operation to produce a fixed-length vector of size F. Each group of nodes constitutes the PMT information of one hit (or its transformation). An example convolution is illustrated in the first layer (product and summation symbols omitted), applied to the second hit, to produce the first transformed output node in the second layer. This node, highlighted in red, is a component of the new, intermediate representation of the second PMT hit. The convolution is applied across the Nhits dimension (top-to-bottom) for each layer.

Generally, the selection of hyperparameters $F^{(l)}$ and F should balance the number of optimized parameters that result from this choice. In particular, very large numbers of filters per layer will lead to very large kernel sizes, which may be prohibitive in training and in extreme cases could lead to overfitting.

A commutative operation is next applied to the output of the final convolution layer. Using $\phi\left(\text{PMT}_k\right) \in \mathbb{R}^F$ to denote the transformation of PMT_k (a group of nodes in the first layer of Figure 4.3; the hit information) from the successive convolutions, the function to produce the fixed-length vector in Figure 4.3 can be written as

$$\underset{\text{Nhits}}{\text{com}} := \underset{k \in \{1,\ldots,\text{Nhits}\}}{\text{com}} \left(\phi\left(\text{PMT}_k\right)\right) \tag{4.7}$$

where the commutative operation, com, is applied elementwise in the filter dimension and reduced across the Nhits dimension, resulting in a vector of shape $(1, F)$. Typically, the first dimension is removed, and the output becomes a vector of length F, as per Figure 4.3. Again, F can be arbitrarily selected with no constraints imposed by prior layers, but should be chosen to be reasonably large such as to represent the data in a useful way.

This commutative operation is key to ensure that the network is both able to accept an input of variable length and to ensure that the output is invariant to the ordering of the PMTs. Given any length of input and any ordering of the PMTs, the final output of the feature extractor is always of length F. The idea is inspired by the results from a paper called Deep Sets [170], which studies the application of neural networks to sets and provides a theoretical framework. By definition, a commutative operation is insensitive to the order of operands, which for the problem of position reconstruction are the PMT representations are transformation.

The commutative operation is chosen to be the mean, which is simple and easily explainable given that it weights every PMT equally. It also ensures that the scale of the output is independent of the number of hit PMTs in an event. Thus, for example, even if the network is only trained on low-energy events, it should generalize reasonably well to high-energy events which typically have a much higher value of Nhits. In principle, any commutative operation can be used to preserve the invariance and arbitrary input length properties of the network. Some options – including the sum or multiplication – will be dependent on Nhits, while others – such as the average or maximum – will not. A dependence on Nhits could be advantageous, for example, in energy reconstruction.

The network architecture also contains a masking component to each layer which is not shown in Figure 4.3. If the data were fed in event-by-event, such a mask would not be needed in principle. However, there are two problems with this. Firstly, sometimes random hits in an input vector are bad, even after preprocessing and selecting only well-calibrated PMTs. Secondly, in practice, the data must be grouped into batches for efficient training and inference. In fact, stochastic mini-batch gradient descent requires that data be grouped together randomly. Batches must have a fixed shape, yet the chance of all events in a batch having the same Nhits value is practically zero. Therefore, given a batch b of size $N^{(b)}$, the shape of the input tensor is $\left(N^{(b)}, \text{Nhits}^{(b)}_{\text{max}}, 4\right)$, where

$$\text{Nhits}^{(b)}_{\text{max}} = \max_{i \in \left\{1,\ldots,N^{(b)}\right\}} \left(\text{Nhits}_i\right) \tag{4.8}$$

is the maximum value of Nhits in the batch. Each row corresponds to a single event. For events with a value of Nhits that is less than the maximum, the remaining hit quantities are padded with zeros. This is chosen because a hit PMT at coordinate $(0, 0, 0)$ is physically impossible due to the PSUP, and the hit times are never zero.

The chosen padding also has the advantage of producing a weighted sum of zero for the unhit PMTs regardless of the weights. However, the bias term can make the input to the node, and thus the activation, non-zero. Thus, a binary mask is applied after every convolution layer. The mask is created for each batch prior to passing the inputs to the network, and simply masks out unhit PMTs (defined by the length four vector of zeros) by setting those entries in the mask to zero and the remaining entries to one. The mask is then multiplied with the output of the convolution layer.

As well, the mask is used in computing the average to obtain the fixed-length feature vector. Again, due to both bad hits that are not removed and the batching of the data, the denominator of the average may be larger than it should be, while the numerator will be as expected due to the final masking layer forcing the output to zero for unhit nodes. Using the number of hit PMTs according to the mask ensures that the average is computed correctly.

The mask also provides flexibility in changing the preprocessing procedure. For example, it allows for random PMT hits to be removed from the data as a method of regularization, should it be desired. It also allows early or late PMTs to be masked out of the calculation with a simple threshold.

As previously mentioned, extending the feature extractor to support additional channels is straightforward. However, since the inclusion of more channels necessitates filters of a different size in the first layer, retraining of at least the first set of parameters would be required. It would be possible to transfer the remaining weights from an existing model, although fully retraining the network from scratch may be the most direct way to obtain optimal performance.

Predictor The second portion of the network is the predictor, which accepts as input the output of the feature extractor – a vector of length F. The predictor is a fully-connected neural network with several hidden layers. The output of this network is of size three, as it predicts the Cartesian coordinates of the event position. For numerical stability and bounding, the outputs are rescaled to fall within the interval $[-1, 1]$ and the tanh activation function is applied to the final layer. This procedure is described and justified in Section 4.2.5.

The feature extractor and predictor, while abstractly are two distinct networks, are connected and trained as one network. This ensures that the feature extractor learns to produce a representation that is useful for predicting the position. More details on the training procedure are given in Section 4.2.6.

In principle, due to the flexible nature of the model, *any* quantity can be predicted from the PMT hit data. All that is required is to change the size of the output and the data used for training. While this thesis focuses on primarily position reconstruction, the architecture is easily extended to numerous other problems, such as signal and background classification. Furthermore, because the feature extractor is, as the name implies, supposed to extract a useful representation of the data, those weights from a trained model could be transferred to a different task. Even though, in such a scenario, the feature extractor will be optimized for producing a representation useful for position reconstruction, it may very well provide better results than randomly initializing the parameters. The weights could also be held fixed and not updated in the training procedure, if the representation is general enough.

4.2.3.3 Network architecture specifics

The discussion so far has avoided any mention of numbers except where fixed as per the requirements of the network. This abstraction is intentional, for the architecture – while designed for SNO+ data – could easily be applied to numerous other experiments. An overview of the full network architecture used in the analyses of Part I of this thesis is shown in Table 4.1. The table is separated into two sections, dividing the feature extractor and the predictor.

Each convolution layer in Table 4.1 is defined to contain the 1×1 convolution itself, a ReLU activation applied to the output of the convolution, a batch normalization operation applied to the output of the activation, and a multiplication with the hit mask (in that order). The final multiplication with the hit mask in each layer is fundamental to the architecture (and thus listed explicitly in the table), whereas the activation function and batch normalization that precede it are choices made for the specific application in this thesis. The first element of the output size of the feature extractor portion is the spatial length after the convolution is applied, while the second element is the number of filters, $F^{(l)}$, at the l^{th} layer. As each convolution is one-by-one, the spatial dimension – $\text{Nhits}_{\text{max}}^{(b)}$ – never changes. The last layer of the feature extractor is the masked per-channel mean across the Nhits dimension, reducing the size of the output to a single dimension of length $F = 512$. The batch size is not included in the output shape as it is arbitrary and does not affect the network structure. If the batch size is one, the first element of the output shape entries for the feature extractor in the table can be simplified by the fact that $\text{Nhits}_{\text{max}}^{(b)} = \text{Nhits}$, and the final mask of each convolution layer and the mean operation becomes redundant.

For the specific model in Table 4.1, $F^{(l)}$ is chosen to increase by a factor of two

Table 4.1: Summary of the full position reconstruction CNN architecture. Table contains both the feature extractor and predictor components of the network. The type and output shape O of each layer are also included. The output shape does not include the batch dimension, and the architecture is valid for any batch b of arbitrary size.

Layer	**Output**
Input	$\text{Nhits}_{\text{max}}^{(b)}$, 4
Convolution – Masked Output	$\text{Nhits}_{\text{max}}^{(b)}$, 8
Convolution – Masked Output	$\text{Nhits}_{\text{max}}^{(b)}$, 16
Convolution – Masked Output	$\text{Nhits}_{\text{max}}^{(b)}$, 32
Convolution – Masked Output	$\text{Nhits}_{\text{max}}^{(b)}$, 64
Convolution – Masked Output	$\text{Nhits}_{\text{max}}^{(b)}$, 128
Convolution – Masked Output	$\text{Nhits}_{\text{max}}^{(b)}$, 256
Convolution – Masked Output	$\text{Nhits}_{\text{max}}^{(b)}$, 512
Masked Mean	512
Fully-connected	800
Fully-connected	600
Fully-connected	400
Fully-connected	200
Fully-connected	100
Fully-connected	80
Fully-connected	60
Fully-connected	20
Fully-connected (Output)	3
Total number of parameters: 1 434 475	

in each layer up until $F = 512$, resulting in seven convolutional layers. $F = 512$ was deemed to be a reasonable size for the output of the feature extractor, considering all factors including obtaining a sufficiently useful representation of the data, the speed of training, and the general size of the predictor network. A representation of that size is approximately 5 % of the number of PMTs, which is highly compressed for high energy events and of comparable size to the number of hit PMTs for low energy (recalling that the ratio of Nhits to energy is of order 500 Nhits/MeV in the scintillator phase).

For the predictor network, the input size is given by the output of the feature extractor. The first layer is chosen to be of the same order as the output of the feature extractor, and subsequent layers are chosen to gradually decrease to the size of the output, which in the case of position reconstruction is three. Again, each fully-connected layer consists of a ReLU activation applied to its output, except for the final layer which uses the tanh function. All but the last three layers also use batch normalization applied after the activation.

4.2.4 Datasets

In order to train the model to work with data from the SNO+ detector, data are simulated using RAT, described in Section 2.1.6. In this supervised machine learning approach, the underlying truth information is known. Simulations from RAT are extremely detailed and are expected to model the detector well under most circumstances. Single-electron events are simulated to create the dataset used by the model to learn position reconstruction. Other particle types could be simulated, though electrons are produced in β decay processes – including 2νββ/0νββ decay – and other important interactions. As well, the typical initial assumption for likelihood-based vertex reconstruction in SNO+ hypothesizes a single-electron event, making this dataset useful for a direct and fair comparison between methods.

To ensure that the network sees a variety of data and does not overfit, events are generated uniformly in volume throughout the detector. As well, particles are generated with isotropic momenta from their initial position and with an energy drawn from a bounded uniform distribution. A summary of the important parameters of the simulations used for training and evaluation are given in Table 4.2. The table is split into three sections, consisting of software properties, detector properties, and simulation/event properties. The run number is considered a detector property as it sets numerous values in the simulation to match the detector operating conditions at the time the run occurred. Detector settings affected by the run number include trigger masks and thresholds, PMT noise levels, and PMT channel statuses. It could also include the level of the scintillator in the detector (applicable to only the partial fill period), the AV offset value (applicable to the partial fill period and afterwards), and more. Various components of the reconstruction procedure also use run-dependent values.

In Table 4.2, the number of events simulated differs from the number of events that are actually part of the final dataset. This is because some simulated events fail to trigger the detector, particularly those at low energies. As well, certain events, such as those occurring in the neck, are removed.

4.2.5 Data preprocessing

All data are preprocessed prior to training the model. However, instead of normalizing or standardizing the data using computed sample quantities from the training set, known physical information regarding the detector is used to bound the inputs. In

Table 4.2: SNO+ detector simulation parameters for single-electron events. Rows are split into three sections, consisting of software properties, detector properties, and event properties.

Parameter	**Value/Description**
RAT version	7.0.9
ROOT version	5.34.38
GEANT4 version	10.00.p02
Run number	300 000
Detector medium	LAB PPO (2.2 g/L)
Event type	Electron
Position distribution	Uniform
Position range	Inner AV volume
Energy distribution	Uniform
Energy range	(0.5 – 20) MeV
Direction distribution	Isotropic
Number of simulated events	1 260 000
Number of triggered events	1 257 989
Number of triggered events passing cuts	1 214 749

particular, the Cartesian coordinates of the hit PMTs, which by default are in units of mm, are all scaled by a factor of $8412 \cdot 2/\sqrt{12} \approx 4856.67$. This factor is chosen because the average radius of the PSUP is approximately 8412 mm, and the standard deviation of a uniform distribution with lower bound a and upper bound b is $(a-b)/\sqrt{12}$. The distribution of hits is roughly uniform over each of the coordinates, and so this scale factor makes the standard deviation of the inputs approximately equal to one. As the mean of the distribution is also approximately zero due to the symmetry of the PSUP, these components of the input are effectively standardized.

The PMT hit times, which are by default in units of ns, are simply scaled by a factor of 500. While the trigger window is approximately 400 ns, the minimum is rarely below 100 ns, making the effective upper bound approximately 500 ns. The distribution is much more complicated than those of the PMT coordinates, with dependencies on both properties of the events and the detector conditions, and so this feature is not standardized. Since the hit time is effectively bounded by the trigger window, this scaling is a form of min-max normalization.

Event positions prior to preprocessing are in units of mm and thus are typically on the order of 10^3 to 10^4. As such, the outputs are also normalized to fall within the range $[-1, 1]$. While not strictly necessary given that all three coordinates are on the same scale, normalized outputs ensure that the gradients computed in backpropagation are numerically stable. Normalization also allows for the outputs to be bounded with an activation function, removing the occurrence of extreme values and enforcing physical bounds on the predicted positions. For these reasons, a scale factor of 6000 is applied to each coordinate individually, limiting components of the prediction to be less than the AV radius. However, no further restrictions are applied to the overall event radius, which can technically exceed the AV radius even with the application of an activation function. Despite this, predictions with radii greater than 6000 mm were found to be infrequent. As well, the sharp gradient decline near values of ± 1 for the tanh activation is not a significant issue as each coordinate individually is statistically unlikely to fall so close to ± 6000 mm (± 1 after scaling) given that the distribution of simulated events is uniform in volume within a sphere.

4.2.6 Training procedure

Data files are converted from the default ROOT format that is generated from RAT simulations to a tabular structure in the HDF5 (Hierarchical Data Format) file standard [171]. This is done for better compatibility with Python and its numerical, scientific, data analysis, and machine learning libraries. Data are also split into the three typical sets: the training, validation, and test sets. Each is used according to the definitions in Section 3.5.1. The test set consists of 10 % of the overall simulated dataset, while the training and validation sets consists of 90 % and 10 % of the remaining data, respectively. The split is completely random, and as a result shuffles all events.

The loss function used is the mean squared error. The Adam algorithm – a stochastic gradient-based adaptive optimization procedure [172] – with a learning rate of $5 \cdot 10^{-4}$ is used to minimize the loss function, as it was found (through the validation set) to outperform standard stochastic gradient descent. A batch size of 128 is used in training, as it provides an adequate approximation of the true gradient while not exceeding memory limits of the GPUs used for training.

For regularization and numerical stability, batch normalization is used in all but the last two layers of the predictor network. During training, the sample mean and standard deviation of a given mini-batch are used to standardize the outputs of the layers. Simultaneously, a moving mean and standard deviation are maintained and updated with the training data. During inference, these values are used instead of their respective mini-batch sample estimates to eliminate batch dependence. Although L1 and L2 regularization applied to the predictor network layers was tested, neither was found to have an observable effect on the final results. For the same reasons, dropout was tested but ultimately excluded from training.

100 epochs of training are done for a given model. Metrics, including the loss function, calculated on the training data are updated after each batch is processed until the end of the epoch. The model in its state at that point is also run on the validation set. Both values are recorded into a separate file. The loss metric on the validation set is used to select the best epoch for a given model, and that value is then used to select the best overall model. Each model has different hyperparameters, such as the network architecture or learning rate, and so this process selects the optimal hyperparameters of the ones varied. As well, the validation loss history is used to determine the number of epochs of training to perform. For training, 100 epochs was found to be more than sufficient for convergence; the optimal validation loss usually occurred well before 100 epochs had elapsed. The validation loss also began diverging from the training loss which continued to improve, indicating some level of overfitting and justifying the selection.

The networks are implemented using Keras [173], a high-level and flexible application programming interface (API) for TensorFlow [174]. Training and inference are conducted on NVIDIA *GeForce GTX Titan X*[4] and *GeForce RTX 3090*[5] GPUs.

The following chapter focuses on the application of this model and its performance. All results presented next and throughout this thesis are conducted on the withheld test set or otherwise data that was not present during training. Several additional steps were taken to prevent accidental leakage, such as removing read and write permissions from test data until all training was completed.

[4]https://www.nvidia.com/en-us/geforce/graphics-cards/geforce-gtx-titan-x/

[5]https://www.nvidia.com/en-us/geforce/graphics-cards/30-series/rtx-3090-3090ti/

Chapter 5

Deep Learning-Based Event Reconstruction: Results and Analysis

This chapter presents the results of evaluating the performance of the SNO+ neural network reconstruction model developed in Chapter 4. Section 5.1 focuses on the performance of the position reconstruction neural network algorithm in comparison to the traditional maximum likelihood estimation. This section uses a subset of the data described in Table 4.2, meaning that the evaluation concerns only data from the same distribution as the training set. Section 5.2, in contrast, evaluates the performance of the neural network on reconstructing the position of gammas originating from PMTs that travel into the AV. These gammas are from a completely different class of events than what is seen during training, and the results in this section show that the neural network can offer improvements in reconstructing a prominent background. Section 5.3 shows how the neural network architecture and training methodology can be extended to direction reconstruction in liquid scintillator with promising results. Chapter 5 ends with a summary and discussion in Section 5.4, focusing on the implications and future work of the methods developed in Part I.

5.1 Position reconstruction evaluation on simulated electron data

The test data used for evaluation are taken from the simulations described in Table 4.2. For comparison, only events where the position likelihood algorithm converged on a solution are considered. Evaluations are made over the entire test set, as well as further subsets to quantify the dependence of both algorithms on the energy and radial position of an event. The distribution of events as a function of energy is shown in Figure 5.1, while the distribution of events as a function of radius is shown in Figure 5.2.

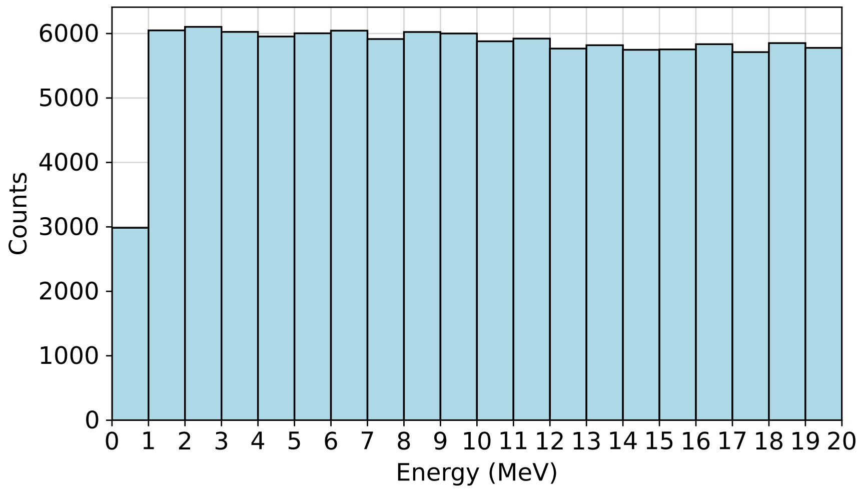


Figure 5.1: Distribution of simulated event energies in the test set.

The distribution across energies is roughly uniform, with proportionally fewer events at energies less than 1 MeV due to the lower triggering efficiency at these energies. The distribution across radii is cubic as the events are simulated uniformly in volume. Due to the small number of events at low radii, the first bin is larger than the others, spanning from 0 mm to 2000 mm.

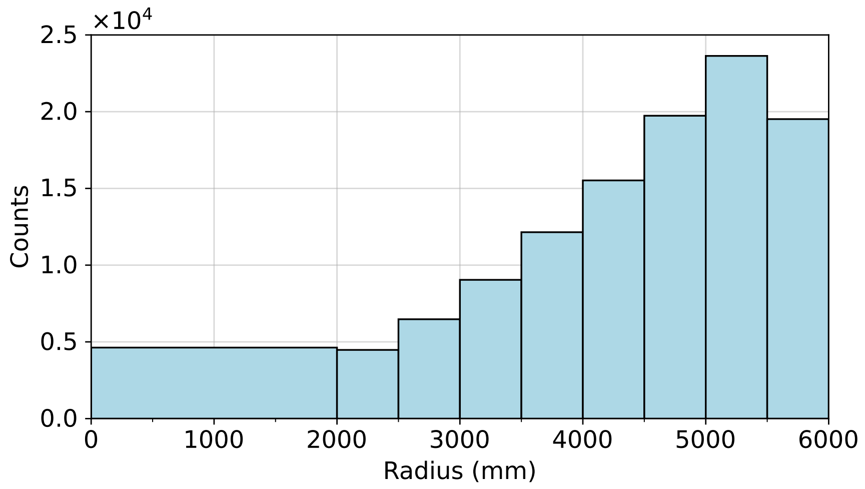


Figure 5.2: Distribution of simulated event radii in the test set.

5.1.1 Position bias and resolution

The performance of each algorithm is quantified by the distribution of the difference between the true and reconstructed position – the residuals. In particular, the position bias and resolution of the residual distribution are compared between the two methods. The position bias can be defined in several ways, including:

1. μ_{data}– the mean of the residuals, calculated on the data
2. μ_{fit}– the mean of the residuals, as determined by a fit to the residual distribution
3. The *mode* of the residual distribution

The bias definition according to Item 1 does not depend on the how the residual distribution is binned, in contrast to Items 2 and 3. However, it is the most prone to outliers. Items 2 and 3 require the residuals to be binned appropriately in a histogram. Problems with both the fit and the mode can arise if the binning is too fine given the amount of data, with a larger dependence on random fluctuations in the residuals. Binning too coarse will mask details in the structure of the distribution and make the bias measurement too uncertain to be useful. Item 2 has the additional requirement that the fit to the distribution is good. For example, if the residuals are fit to a normal distribution when the data are not normally distributed, the mean may poorly reflect the true bias.

The position resolution can similarly be defined in numerous ways, with methods corresponding to the items in the list above:

1. σ_{data}– the standard deviation of the residuals, calculated on the data
2. σ_{fit}– the standard deviation of the residuals, as determined by a fit to the residual distribution
3. The *full width at half the maximum (FWHM)* of the residual distribution

These resolution definitions have the same advantages and drawbacks of the corresponding bias definitions. The results thus contain evaluations using all three definitions, though a particular emphasis is placed on the FWHM due to its robustness to outliers and lack of dependence on the exact distribution of the residuals.

5.1.1.1 Overall performance

Figure 5.3 shows the overall residual distributions for the x, y, and z event position coordinates for both reconstruction methods. Figure 5.4 shows the overall residual distributions using the true and reconstructed event radius – calculated directly using the Cartesian coordinates – for both reconstruction methods. These evaluations are over all events in the test set, meaning that energies and radii over the full range specified in the simulation are included. While this obstructs the dependence of the performance on the event properties, the overall evaluation provides a good initial indication of the benefits of the neural network reconstruction algorithm.

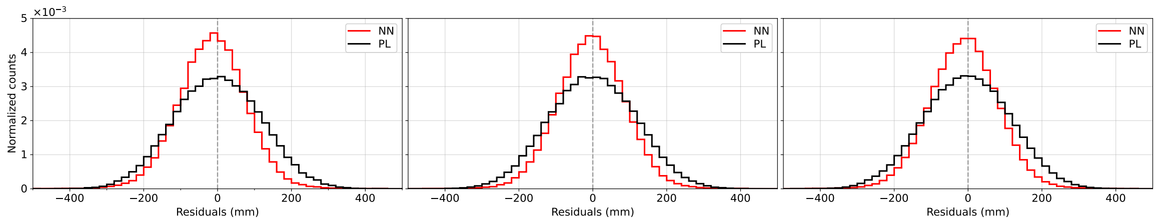


Figure 5.3: Overall residual distributions (x, y, z coordinates). The x, y, and z position coordinate residual distributions are shown in the left, middle, and right subplots respectively. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

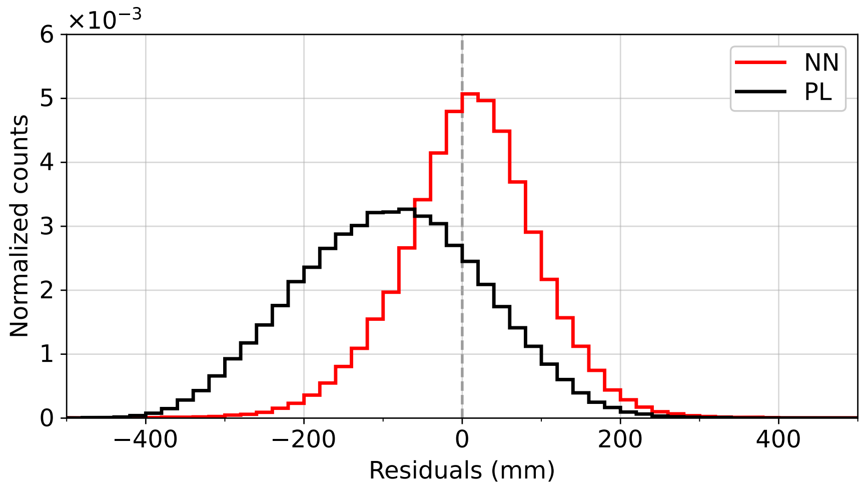


Figure 5.4: Overall residual distributions (radius). Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

In terms of resolution, the neural network performs better than the position likelihood method overall. As well, the neural network has a much lower radial bias. To quantify the performance, Tables 5.1 and 5.2 show the position bias and resolution (respectively) using all three definitions described earlier. Metrics are shown for each position coordinate individually and for the radius. Uncertainties in the tables are statistical and arise from the standard error for data-based metrics, fit error for fit-based metrics, and bin widths for histogram-based metrics.

Table 5.1: Overall position bias comparison for reconstruction methods. Includes the bias as defined by the mean of the data, the mean of the fit to the residual distribution, and the mode of the residual distribution for the neural network (NN) and position likelihood (PL) position reconstruction methods. Uncertainties are statistical only.

		PL bias (mm)	**NN bias** (mm)
x	$\boldsymbol{\mu}_{\text{data}}$	0.31 ± 0.34	-14.50 ± 0.28
	$\boldsymbol{\mu}_{\text{fit}}$	-0.09 ± 0.47	-13.31 ± 0.27
	Mode	$10 \quad \pm 10$	$-10 \quad \pm 10$
y	$\boldsymbol{\mu}_{\text{data}}$	-0.10 ± 0.34	-4.02 ± 0.28
	$\boldsymbol{\mu}_{\text{fit}}$	-0.17 ± 0.43	-3.62 ± 0.19
	Mode	$-30 \quad \pm 10$	$-10 \quad \pm 10$
z	$\boldsymbol{\mu}_{\text{data}}$	-2.84 ± 0.34	-11.50 ± 0.28
	$\boldsymbol{\mu}_{\text{fit}}$	-2.28 ± 0.34	-10.67 ± 0.20
	Mode	$-10 \quad \pm 10$	$-10 \quad \pm 10$
r	$\boldsymbol{\mu}_{\text{data}}$	-90.15 ± 0.34	9.39 ± 0.26
	$\boldsymbol{\mu}_{\text{fit}}$	-89.63 ± 0.61	12.66 ± 0.62
	Mode	$-70 \quad \pm 10$	$10 \quad \pm 10$

The results in the figures and tables demonstrate that improvements with the neural network are particularly prominent in the residual distributions and metrics of the radius, rather than any of the individual Cartesian coordinates. Note that for bias and resolution calculation methods which depend on the binning of the residuals, the histograms use bin edges ranging from -1000 mm to 1000 mm in increments of 20 mm. This is the same as is done in Figures 5.3 and 5.4, though they are restricted

Table 5.2: Overall position resolution comparison for reconstruction methods. Includes the resolution as defined by the standard deviation of the data, the standard deviation of the fit to the residual distribution, and the FWHM of the residual distribution for the neural network (NN) and position likelihood (PL) position reconstruction methods. Uncertainties are statistical only.

		PL resolution (mm)	**NN resolution** (mm)
x	$\boldsymbol{\sigma}_{\text{data}}$	117.03 ± 0.25	93.39 ± 0.28
	$\boldsymbol{\sigma}_{\text{fit}}$	119.73 ± 0.38	88.30 ± 0.22
	FWHM	293 ± 14	203 ± 14
y	$\boldsymbol{\sigma}_{\text{data}}$	116.84 ± 0.25	93.53 ± 0.28
	$\boldsymbol{\sigma}_{\text{fit}}$	119.25 ± 0.35	89.11 ± 0.15
	FWHM	292 ± 14	208 ± 14
z	$\boldsymbol{\sigma}_{\text{data}}$	116.65 ± 0.25	94.56 ± 0.29
	$\boldsymbol{\sigma}_{\text{fit}}$	118.72 ± 0.28	90.05 ± 0.16
	FWHM	288 ± 14	212 ± 14
r	$\boldsymbol{\sigma}_{\text{data}}$	117.06 ± 0.24	89.56 ± 0.28
	$\boldsymbol{\sigma}_{\text{fit}}$	121.00 ± 0.50	80.51 ± 0.50
	FWHM	295 ± 14	174 ± 14

to a viewing range of -500 mm to 500 mm.

To further understand why the neural network performs so well, Sections 5.1.1.2 and 5.1.1.3 investigate these metrics as a function of event energy and radius, respectively.

5.1.1.2 Energy-dependent performance

Figures 5.5 and 5.6 show the position bias and resolution (respectively) as a function of the simulated event energy for each of the neural network and position likelihood methods. The x-axis value of every point represents the energy bin with the same upper edge as in Figure 5.1, and metrics are evaluated over the complete set of events in the corresponding bin. As well, all radii are included for each energy bin. From left to right, the figures contain the results for the x, y, and z coordinates of the event position. The resolution is the FWHM for reasons discussed earlier. However, the bias definition used here is actually the mean of the residuals calculated on the data, μ_{data}. The reason for this choice is because the uncertainty of the mode and FWHM are dependent on the bin width of the residual distribution, which is relatively large due to the number of events available in each energy or radius bin. While the uncertainty of the FWHM is proportionally small to its actual value, the uncertainty of the mode is typically of order, or greater than, the value itself. To reduce the dependence on outliers, a residual limit of 1000 mm is set when calculating the bias and resolution according to the mean and standard deviation. Any residual exceeding this threshold is excluded from the calculation.

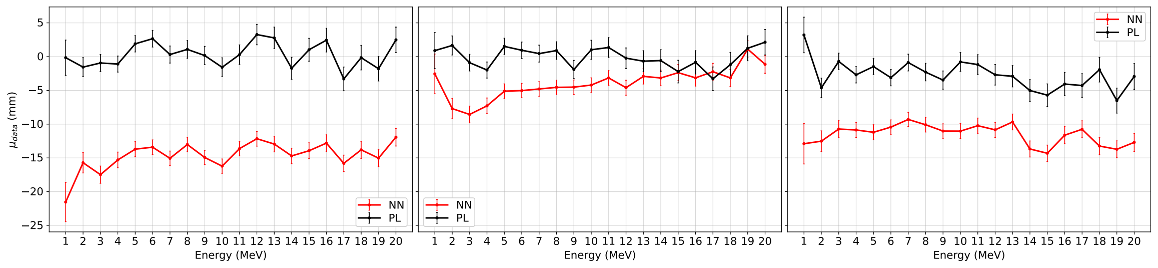


Figure 5.5: Position bias (x, y, z coordinates) as a function of true energy. The x, y, and z position coordinate residual distributions are shown in the left, middle, and right subplots respectively. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

Figures 5.7 and 5.8 show the bias and resolution (respectively) for each reconstruction method, but for the overall radius rather than the individual position coordinates.

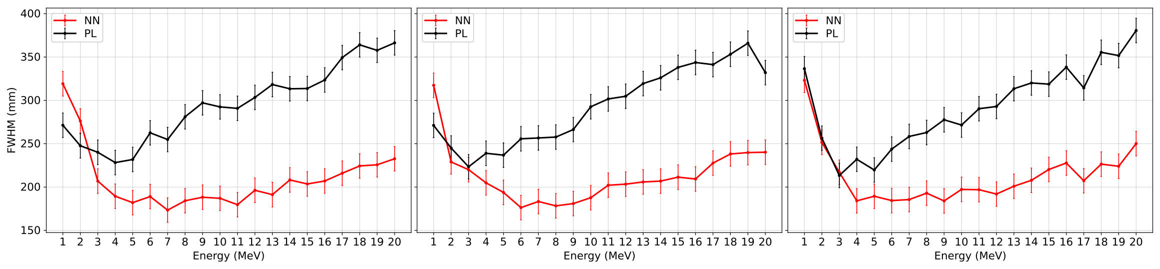


Figure 5.6: Position resolution (x, y, z coordinates) as a function of true energy. The x, y, and z position coordinate residual distributions are shown in the left, middle, and right subplots respectively. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

The bias and resolution definitions used are the same as those from Figures 5.5 and 5.6. Again, all radii are included for each energy bin.

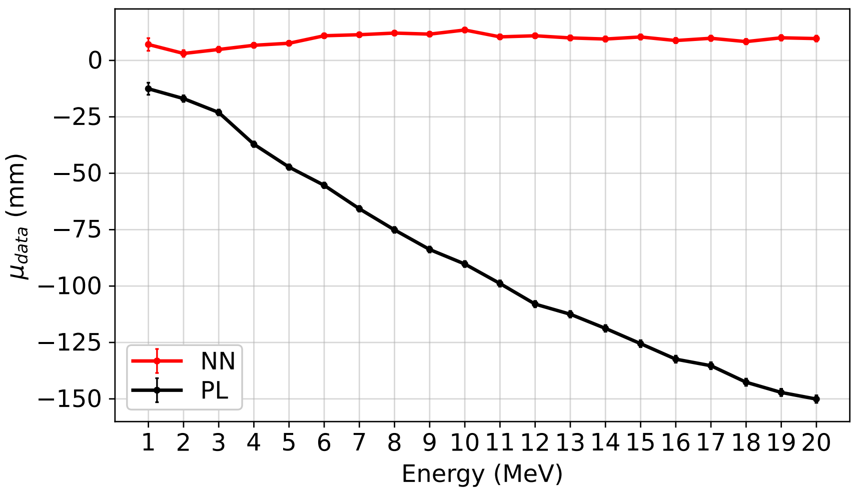


Figure 5.7: Position bias (radius) as a function of true energy. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

From these results, the neural network position resolution is comparable or better than the position likelihood method at all energies evaluated. As the event energy increases, a proportionally greater improvement is observed with the neural network.

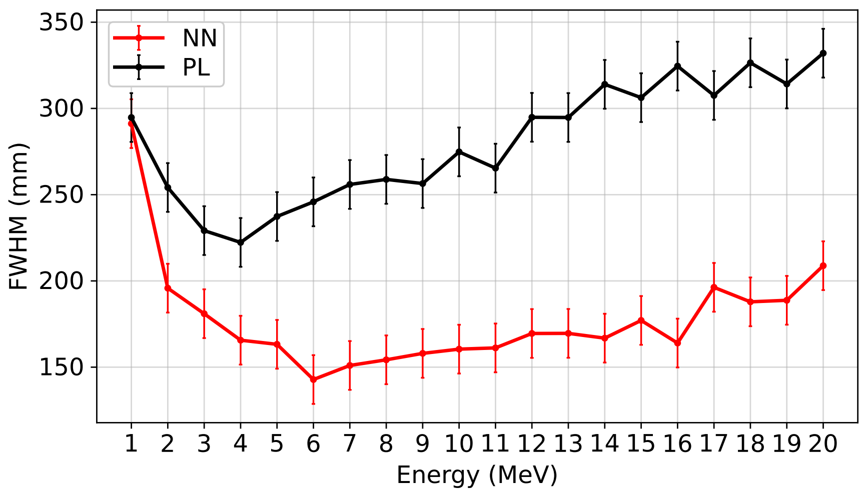


Figure 5.8: Position resolution (radius) as a function of true energy. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

Improvements beyond the uncertainties in the FWHM begin at energies greater than 3 MeV. Above approximately 10 MeV, the relative performance of the neural network is higher due to the substantial increase in position resolution as a function of energy for the position likelihood method which is much less pronounced for the neural network. This trend is likely in part due to the fact that the position likelihood algorithm operates under the assumption that events are point-like. This is a good approximation at lower energies, but at higher energies, the electrons travel further and create a track. The neural network is able to implicitly learn this effect and accurately reconstruct the initial event position.

For each individual position coordinate, the neural network is more biased than the position likelihood method at all energies, though this bias is proportionally small compared to the resolution. However, the neural network is far *less* biased in radius, as observed in Figures 5.4 and 5.7. The position likelihood radial bias also grows with energy, whereas a relatively stable bias is observed across the entire energy range evaluated for the neural network.

5.1.1.3 Radius-dependent performance

Figures 5.9 and 5.10 show the bias and resolution (respectively) as a function of radius for each of the neural network and position likelihood methods. The x-axis value of every point represents the radial bin with the same upper edge as in Figure 5.2, and metrics are evaluated over the complete set of events in the corresponding bin. As well, all energies are included for each radial bin. The bias is defined as μ_{data}, while the resolution is defined as the FWHM – the same as is done in Sections 5.1.1.1 and 5.1.1.2. The quality of the residual distributions, and thus the bias and resolution metrics using any definition, are more adversely prone to statistical fluctuations when the number of events in the radius bin is low.

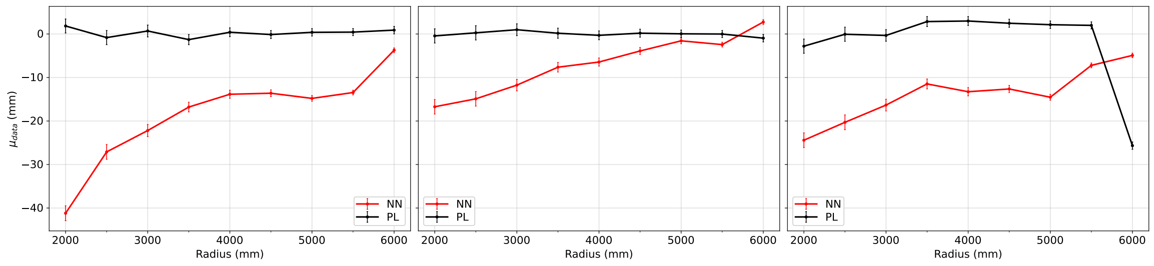


Figure 5.9: Position bias (x, y, z coordinates) as a function of true radius. The x, y, and z position coordinate residual distributions are shown in the left, middle, and right subplots respectively. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

Again, Figures 5.11 and 5.12 show the bias and resolution (respectively) for each reconstruction method, but for the overall radius rather than the individual position coordinates. As with Figures 5.9 and 5.10, all energies are included for each radial bin.

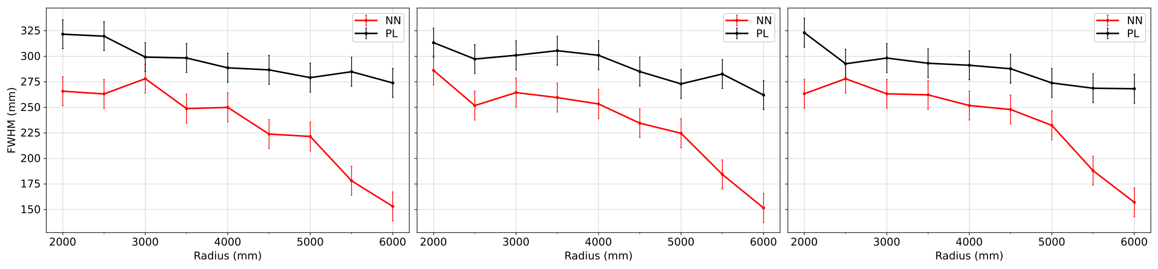


Figure 5.10: Position resolution (x, y, z coordinates) as a function of true radius. The x, y, and z position coordinate residual distributions are shown in the left, middle, and right subplots respectively. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

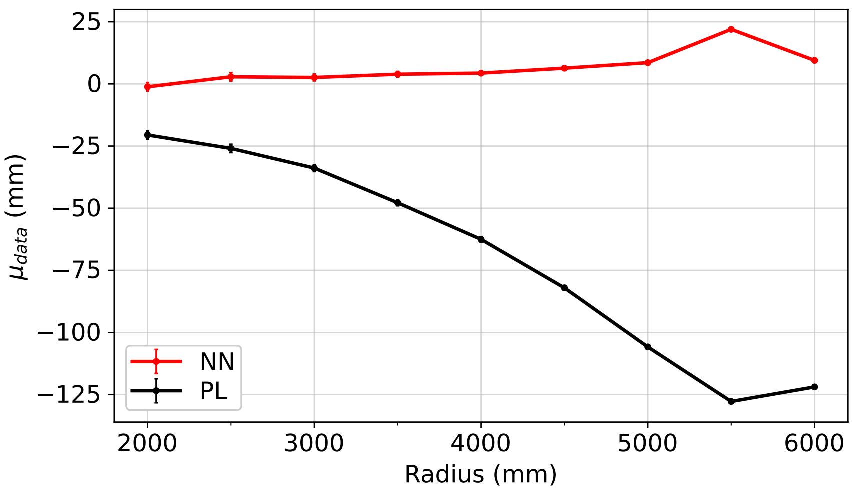


Figure 5.11: Position bias (radius) as a function of true radius. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

For both the neural network and position likelihood methods, the general trend is that the resolution decreases with increasing radius. Unlike with energy, the proportional improvements from the neural network remain fairly stable until about 5000 mm, after which the relative improvement grows larger. However, the neural network is more biased than the position likelihood method, especially at lower radii, for each individual position coordinate. As with the energy-dependent results, this

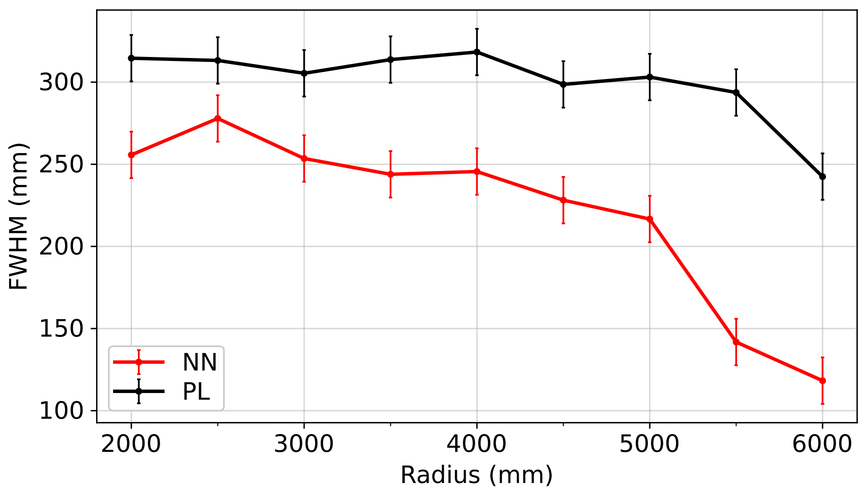


Figure 5.12: Position resolution (radius) as a function of true radius. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

bias is proportionally small in comparison to the resolution. Furthermore, when evaluating radial metrics, the bias is actually much lower with the neural network, and this improvement grows with increasing radius. There is likely a global bias due in part to the deviation from the point-like event assumption at higher energies (as observed in Figure 5.7), but does not explain the radial dependence.

The biases evaluated in Sections 5.1.1.1 to 5.1.1.3 are not considered in the context of direction. As with uniform energy and radius, the isotropic distribution of events can mask subtle effects in the bias and resolution. This is explored thoroughly in the next section.

5.1.2 Position drive

The performance of each algorithm is next quantified by the *drive* – the projection of the distance vector from the true to the reconstructed position onto the true event direction vector. Mathematically, the drive is given by

$$d = \vec{u}_{\text{true}} \cdot (\vec{x}_{\text{fit}} - \vec{x}_{\text{true}}) \, , \tag{5.1}$$

where $\vec{u}_{\text{true}}$ is the true event direction with $\|\vec{u}_{\text{true}}\| = 1$, $\vec{x}_{\text{true}}$ is the true event position, and $\vec{x}_{\text{fit}}$ is the reconstructed event position. The drive is a scalar quantity, with a positive value indicating a forward bias in the direction of the event and a negative value indicating a backward bias. Visually, the concept of drive is depicted in Figure 5.13.

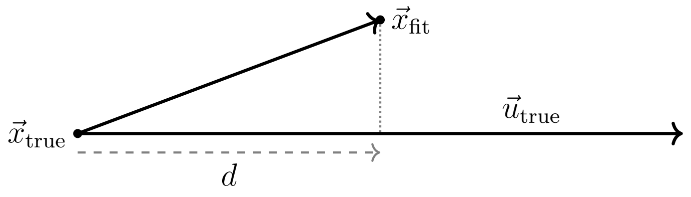


Figure 5.13: Illustration of position drive. The two distance vectors, $\vec{x}_{\text{fit}} - \vec{x}_{\text{true}}$ and a scaled $\vec{u}_{\text{true}}$, are shown as thick solid lines. The drive, d, is represented by the dashed grey line in the figure. As it is always along $\vec{u}_{\text{true}}$ by definition, the drive is a scalar quantity.

Though it might be assumed that the drive should have an expected value of zero, a positive bias is always observed. The main cause is early directional Cherenkov light which is emitted faster than isotropic scintillation light. For a PMT hit by a photon from Cherenkov radiation, a lower time of flight will maximize the PDF for the time residual corresponding to the hit PMT, $P\left(T_{\text{res},i}\right)$ (see Equation (4.1) and the discussion in Section 4.1). This is because Cherenkov light is subdominant relative to scintillation light, which is in turn reflected in $P\left(T_{\text{res},i}\right)$. In the overall maximum likelihood estimation, the Cherenkov hits produce a "pull" towards those PMTs and a reconstructed position positively biased along the direction of the event.

A non-zero track length, deviating from the point-like reconstruction assumption, also contributes to this bias, though the main cause is from the Cherenkov light [175]. Due to its inherent positive bias in liquid scintillator detectors, the term "drive" is often used to refer to both the numerical value of the projection as well as the general concept of position reconstruction bias in the direction of the event. Also note that drive is not seen in the bias plots of Section 5.1.1 due to the fact that the electrons are simulated uniformly in direction throughout the detector volume.

Drive is an important metric to study. A notable impact is on the PDFs used to reconstruct the event direction, as is briefly mentioned in [112] and discussed more thoroughly in [175]. Using the reconstructed position from the likelihood-based method results in undesirable peaks in the angular distributions at earlier time residuals. As well, substantial differences in the performance of direction reconstruction by substituting the reconstructed position for the true position in the likelihood optimization are observed. In particular, distributions of the angle between an event's true and reconstructed direction, θ, are more sharply peaked and contain less poorly reconstructed directions (defined as backward-pointing, $\frac{\pi}{2} < \theta \leq \pi$ or $\cos(\theta) < 0$) when the optimization is done using the true event position. Using the neural network architecture to improve upon direction reconstruction is described in more detail in Section 5.3. This section only presents drive bias studies in the context of improving position reconstruction.

5.1.2.1 Overall performance

The distributions of the drive from the neural network and position likelihood methods are shown in Figure 5.14. As with Section 5.1.1.1, these distributions are created from the full test set and are thus over all energies and radii. Sections 5.1.2.2 and 5.1.2.3 highlight the dependence of the drive on the event properties.

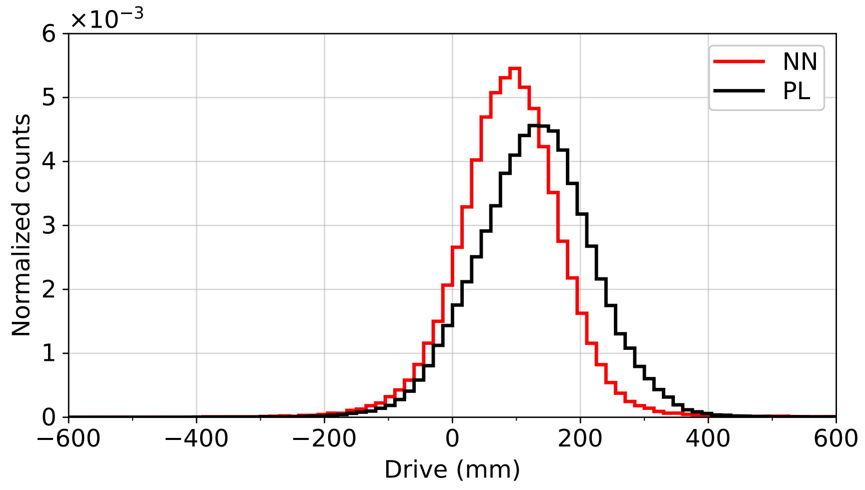


Figure 5.14: Overall drive distributions. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

Notably, the trend is that the drive is less prominent from the neural network reconstruction, indicating that it is better able to separate the Cherenkov information from events and identify that the relatively small proportion of those hits should be treated differently. Developing a rule-based system for the position likelihood method to reduce drive would be difficult, though possible with an iterative approach using the initial position to compute and cut on the time residuals. The neural network instead learns this information implicitly and reduces the drive as a consequence of minimizing the loss function. An explicit penalty term could be incorporated into the loss function to directly penalize drive, though this is not done here.

To quantify the overall reduction in drive, the drive bias – as defined by both the sample mean and the mode of the drive distribution – is provided in Table 5.3. These results show that the drive bias is reduced by about 30 % with the neural network, depending on how the bias is calculated. As well, the neural network position reconstruction method has a lower *drive spread*. This is shown in Table 5.4, using both the standard deviation of the data and the FWHM of the drive distribution as spread definitions. Note that neither the drive bias nor spread tables include Gaussian fit metrics as the drive distributions are asymmetric.

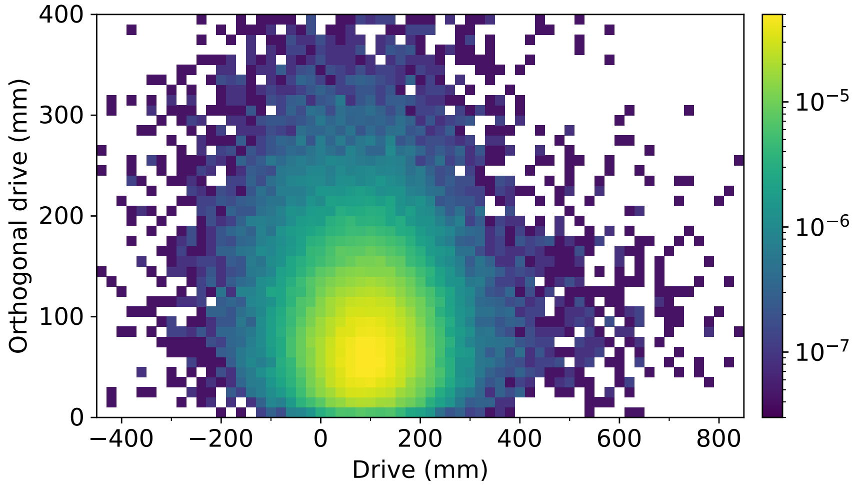
Table 5.3: Overall drive bias comparison for reconstruction methods. Includes the drive bias as defined by the mean of the data and the mode of the drive distribution for the neural network (NN) and position likelihood (PL) position reconstruction methods. Uncertainties are statistical only.

	PL drive bias (mm)	**NN drive bias** (mm)
$\boldsymbol{\mu}_{\text{data}}$	128.50 ± 0.27	91.02 ± 0.26
Mode	127.5 ± 7.5	97.5 ± 7.5

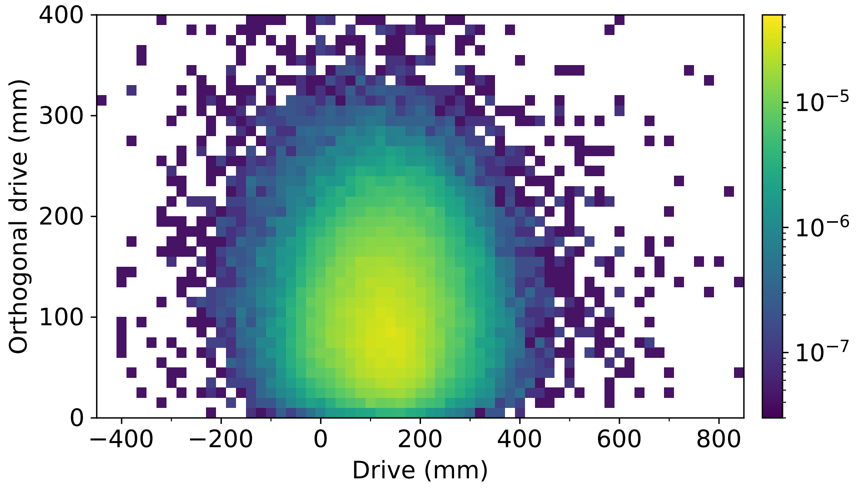
Table 5.4: Overall drive spread comparison for reconstruction methods. Includes the drive spread as defined by the standard deviation of the data and the FWHM of the drive distribution for the neural network (NN) and position likelihood (PL) position reconstruction methods. Uncertainties are statistical only.

	PL drive spread (mm)	**NN drive spread** (mm)
$\boldsymbol{\sigma}_{\text{data}}$	93.18 ± 0.26	87.77 ± 0.48
FWHM	200 ± 11	164 ± 11

A two-dimensional histogram of the reconstructed position relative to the true event position is shown in Figure 5.15. The coordinate system is such that the x-axis is always in the direction of the event, while the y-axis is perpendicular to the direction vector (and the y-axis plane passes through both the true and reconstructed position). By construction, the perpendicular component can only be positive. See Figure 5.13 for a visualization of an event in this coordinate system.



(a) Neural network



(b) Position likelihood

Figure 5.15: Reconstructed position relative to the true event position. The coordinate system is such that the origin is the true event position, the x-axis is along the direction of the event, and the y-axis plane passes through the reconstructed position. Figure 5.15a shows the neural network reconstruction distribution in this arrangement, while Figure 5.15b shows the position likelihood reconstruction distribution. Note that the colour bar, representing normalized counts, is logarithmic.

As previously observed in Figure 5.14, the overall drive is lower for the neural network. The perpendicular component to the drive is also substantially lower, which reflects the overall reduction in position resolution discussed in Section 5.1.1. The neural network reconstructed position is thus more concentrated towards the origin than that from the position likelihood method.

5.1.2.2 Energy-dependent performance

Figure 5.16 shows the drive bias as a function of energy for the neural network and position likelihood methods. Here, the mean of the data is used to define the bias. Error bars are shown on the figure, but are small and not visible on all points. Both the neural network and position likelihood methods exhibit the same trend of increasing drive as a function of energy. However, the drive is substantially smaller with the neural network. Although not shown here, using other bias definitions such as the mode in place of the mean does not produce any substantial changes, and the pattern observed is the same.

The drive spread, defined here as the FWHM of the drive distribution, as a function of energy for each reconstruction method is shown in Figure 5.17. Both the neural network and position likelihood methods produce similar values for the FWHM of the drive within uncertainty and follow a similar pattern of decreasing with increasing energy. The distribution width thus follows the opposite trend of the bias.

The relative reduction in the drive bias as a function of energy is shown in Figure 5.18, demonstrating that the largest improvement from the neural network occurs at lower energies. Still, reductions greater than 25 % are observed even at the highest energies tested.

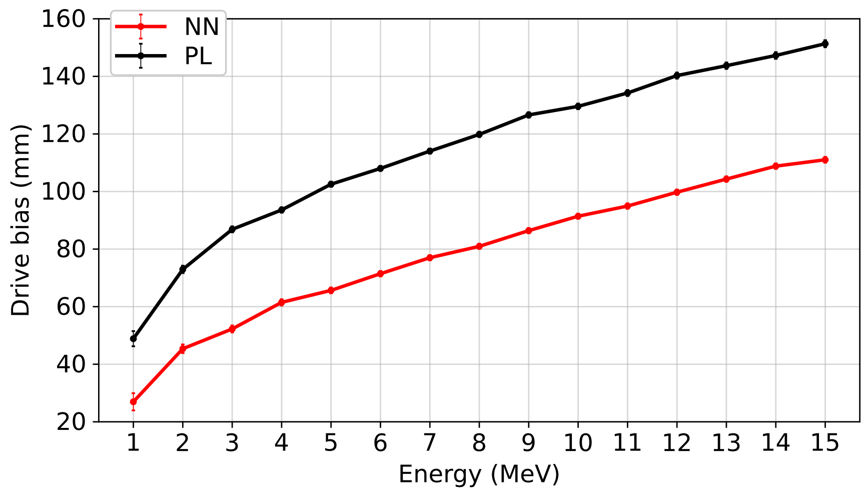


Figure 5.16: Drive bias as a function of true energy. Here, the bias is defined as the mean of the drive distribution, calculated directly on the data. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods. Error bars are present, but are difficult to see for all but the lowest two energy bins which contain fewer events.

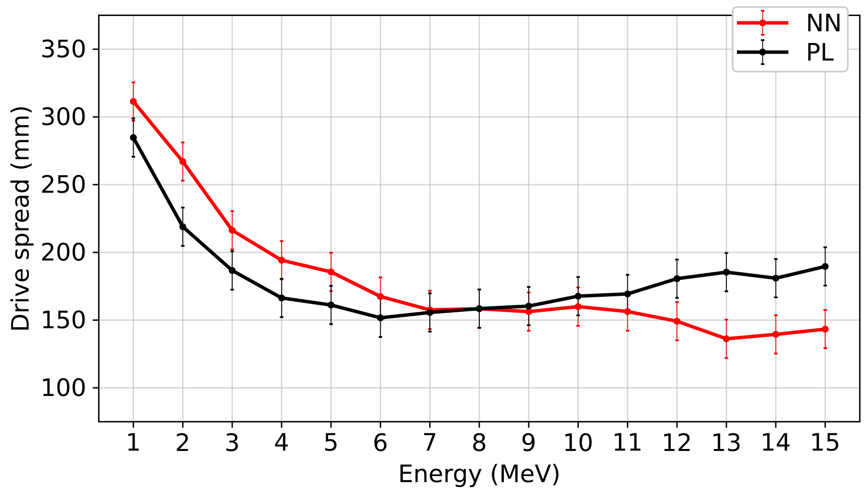


Figure 5.17: Drive distribution width as a function of true energy. Here, the distribution width is defined as the FWHM of the drive distribution. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

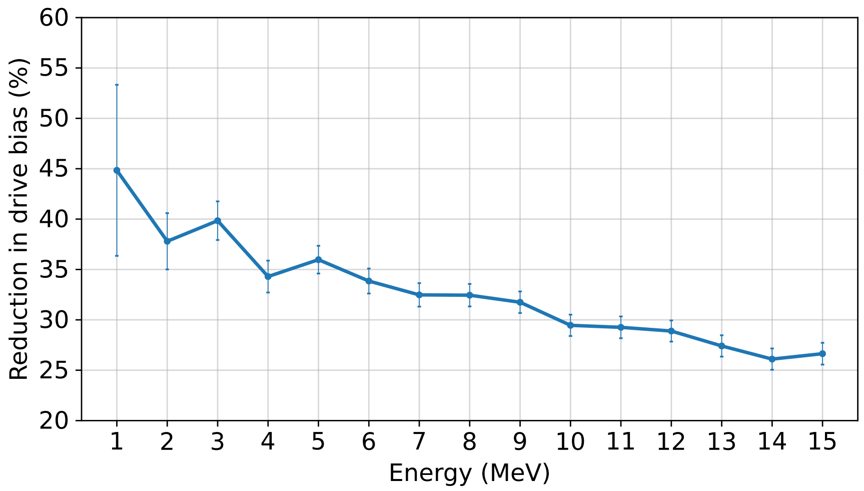


Figure 5.18: Relative reduction in drive bias as a function of true energy. Corresponds to the percentage reduction in the mean drive of the neural network (NN) over the position likelihood (PL) reconstruction method.

5.1.2.3 Radius-dependent performance

Figure 5.19 shows the mean drive as a function of radius for the neural network and position likelihood methods. Error bars are shown, though are only visible at lower radii where the number of events in the subset is small. The dependence on radius is relatively stable until about 5000 mm, after which the drive substantially decreases near the AV. Similar to the energy-dependent results, the neural network reconstructed position has a lower drive at every bin in radius.

The drive spread for each reconstruction method as a function of radius is shown in Figure 5.20. The neural network exhibits a fairly stable distribution width across all radii, with a slight decrease in the width near the AV. In contrast, the position likelihood algorithm has a clear dependence on radius, and the width of the distribution increases substantially as the event is closer to the AV. It is unclear what causes the difference in trends between the two methods.

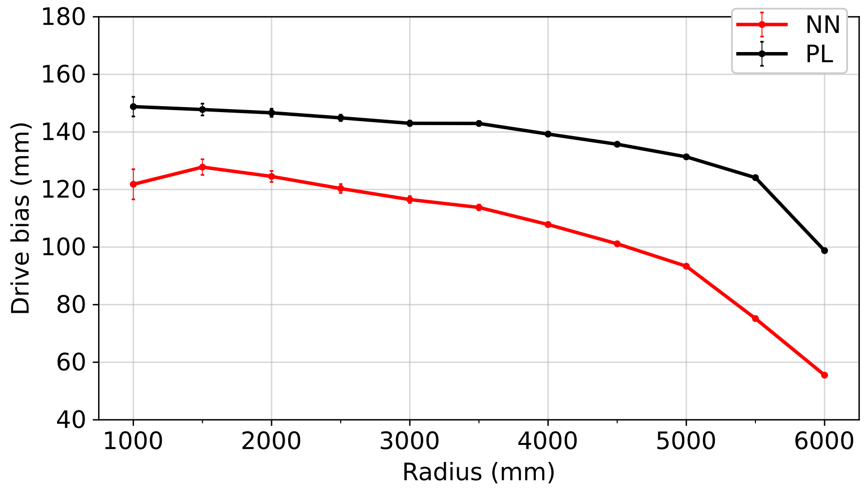


Figure 5.19: Drive bias as a function of true radius. Here, the bias is defined as the mean of the drive distribution, calculated directly on the data. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods. Error bars are present, but are difficult to see at higher radii due to the increasing number of events in those subsets.

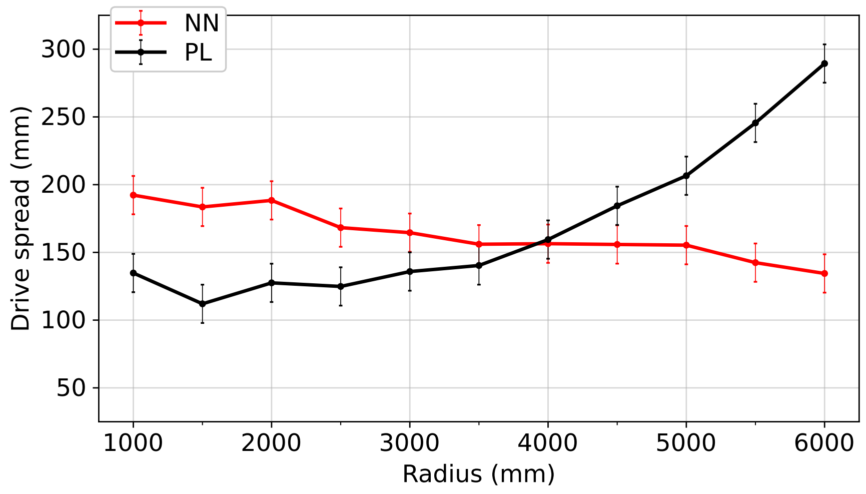


Figure 5.20: Drive distribution width as a function of true radius. Here, the distribution width is defined as the FWHM of the drive distribution. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

The relative reduction in drive bias as a function of radius is shown in Figure 5.21. Reductions in the mean drive range from about 10 % to 30 %, with even larger improvements near the AV.

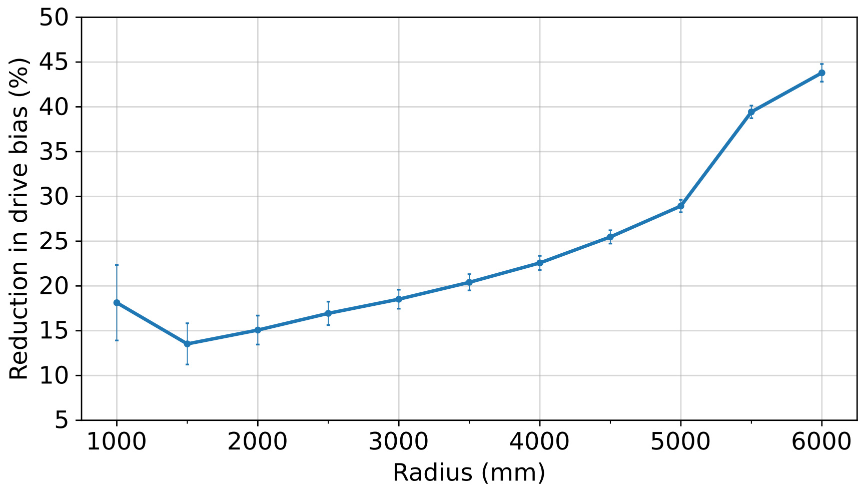


Figure 5.21: Relative reduction in drive bias as a function of true energy. Corresponds to the percentage reduction in the mean drive of the neural network (NN) over the position likelihood (PL) reconstruction method.

At lower radii, since the relative improvement in the bias is smallest while the width of the distribution is actually larger than the position likelihood method, the neural network does not offer a substantial reduction in drive. However, near the AV where both the bias and width of the drive distribution are smaller, the neural network performs notably better.

5.1.3 Speed of reconstruction

Table 5.5 shows the average fit speed of each position reconstruction method, tested on four sets of 75 000 simulated electrons at different energies. The term “fit speed” is the time per event to reach convergence for the position likelihood algorithm, and the time per event for evaluation of the trained neural network algorithm to produce a prediction. The methods in the table were run on the exact same set of data, on the same computer, and at the same time (as close as possible, given that all methods are run on a single event before proceeding to the next). The data were processed using RAT, with the neural network method using a TensorFlow framework integrated into the software by the author. The average fit speeds are calculated via a tool within RAT that measures the computational performance of all steps in the processing chain.

Table 5.5: Average fit speed comparison for reconstruction methods. Calculated as the mean speed over distinct sets of 75 000 simulated electrons, each with a fixed energy as indicated in the table.

Energy (MeV)	**PL fit speed** (s/ev)	**NN fit speed** (s/ev)
1	0.04568	0.00499
2	0.08130	0.00620
6	0.17103	0.00934
12	0.23701	0.01222

The neural network is much quicker than the position likelihood algorithm, with speed improvements on the order of 10 to 20 times. The largest improvements are seen at the highest energies, presumably due to a constant overhead in processing an event with the neural network that dominates the fit time at low energies. Furthermore, event processing in RAT is sequential, and the TensorFlow framework within RAT cannot utilize GPUs. As such, the results in Table 5.5 are representative of the worst case scenario of sequential event-by-event processing with no parallel data prefetching and model execution. Preliminary measurements suggest speeds of $\sim$0.5 ms/ev (corresponding to an improvement of order 400 times at high energies) are possible using a moderate batch size and a GPU. However, this test relies on the data first being converted to a format suitable for processing with TensorFlow in Python and ignores the overhead of batching the data and GPU initialization. Nonetheless, these results indicate that a more efficient way of processing could offer further speed improvements.

5.2 Position reconstruction evaluation on simulated gamma data

SNO+ is sensitive to the trace natural radioactivity present in the Earth and in all materials both in and surrounding the detector. Of particular concern are the long-lived ^{238}U and ^{232}Th isotopes, both of which have half lives on the order of 10^{10} yr [44] and long decay chains[1] [176]. While a great effort is made to reduce the quantity of these contaminants in detector materials from their construction to their installation, trace amounts will always be present.

The water encompassing the detector shields from much of the *external backgrounds*, which originate from all components outside of the detector medium including the PMTs, rope systems, as well as the surrounding rock. As a result, alphas and betas from most of the outer detector configuration are attenuated quickly and never reach the detector volume. Even for contamination in the acrylic of the AV or the external water itself, alphas and betas will not penetrate far into the medium. However, gammas have a much longer attenuation length, and some may make it deep into the detector volume even if originating from outside. For this reason, ^{208}Tl of the ^{232}Th chain is a particularly concerning background due to its β decay to an excited state of ^{208}Pb and the subsequent emission of a 2.6 MeV gamma from the ^{208}Pb in

[1]See `https://periodictable.com/` for full interactive decay chain diagrams of each isotope. (`https://periodictable.com/Isotopes/092.238/index.full.html` for ^{238}U) (`https://periodictable.com/Isotopes/090.232/index.full.html` for ^{232}Th)

its transition to the ground state. Though the beta will deposit its energy near the PSUP, never reaching the detector volume, the high energy gamma may travel a long distance before interacting. As well, at 2.6 MeV, the gamma can trigger the detector and reconstruct in the $0\nu\beta\beta$ decay ROI, depending on where and how it deposits its energy. This type of event is referred to as a PMT β-γ, though only the gamma is of concern in this analysis.

With ^{232}Th present in the nearly 10 000 PMTs that surround the detector, the 2.6 MeV gamma from ^{208}Tl is a non-negligible external background. Despite the large distance between the PSUP and detector volume, the shear number of PMT β-γ decays means that a non-negligible number of gammas will penetrate the detector volume. However, given the exponential dependence on the attenuation, most of these problematic gammas will deposit their energy closer to the AV than the centre of the detector. Thus, implementing an FV cut based on the reconstructed event radius will substantially reduce the rate of these backgrounds. An FV cut of 3.5 m is typically used to reduce the external background level to a rate comparable to other backgrounds in $0\nu\beta\beta$ decay analyses of SNO+ data.

An FV cut has the substantial downside of excluding a large volume of the sensitive region from analysis. Due to the spherical detector configuration and cubic dependence of volume on radius, a 3.5 m cut removes over 80 % of the overall detector volume. With the deployment of tellurium, this will exclude a substantial amount of the ^{130}Te isotope and lead to a corresponding statistical reduction in the $0\nu\beta\beta$ decay search.

Any improvements in optimizing this FV cut could result in substantial gains in volume. This section applies the neural network position reconstruction algorithm to the reconstruction of gammas and investigates the possibility of an increase in the FV cut. It also serves to test the generalization of the neural network on a completely different type of interaction in the detector from that which the model was trained on.

5.2.1 Dataset

For the same reason that PMT β-γ events are a problematic background, they are difficult to study in simulations. In particular, accurately modelling the interactions of the gammas in the detector requires a large number of decays to be generated to obtain a sufficient amount of gammas which both reach the detector medium and result in a trigger. Simulating a meaningful quantity of events that meet these conditions, especially events far into the detector volume, is computationally infeasible. Instead of a direct simulation of ^{208}Tl decays in the PMT glass, an approximation is used to generate gammas originating from a radius within the PSUP and near, but outside of, the AV which are equivalent to those produced in the PMTs. This is done by calculating the angular distribution of gammas originating from a larger radius (in this case, the PSUP) that make it to a specified radius closer to the AV [177, 178]. The functionality for this type of generator exists in RAT.

Using this approximation allows for events equivalent to PMT β-γ interactions to be simulated with a much higher fraction that trigger the detector and penetrate the detector volume. However, it still requires a large number of simulations to generate a reasonable number of interactions at low radii. Table 5.6 contains the main parameters of the simulations generated and used for the analysis in this section.

Software properties and detector properties of the simulations are the same as

Table 5.6: SNO+ detector simulation parameters for PMT β-γ events. Rows are split into three sections, consisting of software properties, detector properties, and event properties.

Parameter	**Value/Description**
RAT version	7.0.9
ROOT version	5.34.38
GEANT4 version	10.00.p02
Run number	300 000
Detector medium	LAB PPO (2.2 g/L)
Event type	PMT β-γ (approximation)
Gamma radius	6200 mm
Gamma energy	2.6 MeV
Number of simulated events	12 240 000
Number of triggered events	8 528 140
Number of triggered events passing cuts	2 334 840

those used in Table 4.2 to train the neural network position reconstruction model. While software properties should generally not have an effect on the results, updates to RAT could potentially change the optics and thus the simulation details. Furthermore, this eliminates the chances of new bugs incorrectly changing the results. The detector run number and detector medium are also kept the same, the former to keep consistent detector conditions and the latter as only models trained on the same detector medium will ever be used on that type of data.

A radius of 6200 mm is used in the generation of the gammas as it is sufficiently close to the AV. Only events which reconstruct with a valid position and energy fit, as determined by the fitter built into in RAT (which itself includes the position likelihood-based optimization), are retained. Furthermore, events with a reconstructed energy less than 2.4 MeV are discarded as that is the lower bound of the 0νββ decay ROI for SNO+, and the most problematic gamma interactions will have energies *within* the ROI.

5.2.2 Reconstructed radius distributions

The reconstructed radius distributions of both the position likelihood and neural network methods applied to the gamma interactions from PMT β-γ events are shown in Figure 5.22 with the cuts described in Section 5.2.1. The exponential attenuation of the gamma rays is clear from both fitters with the logarithmic y-axis.

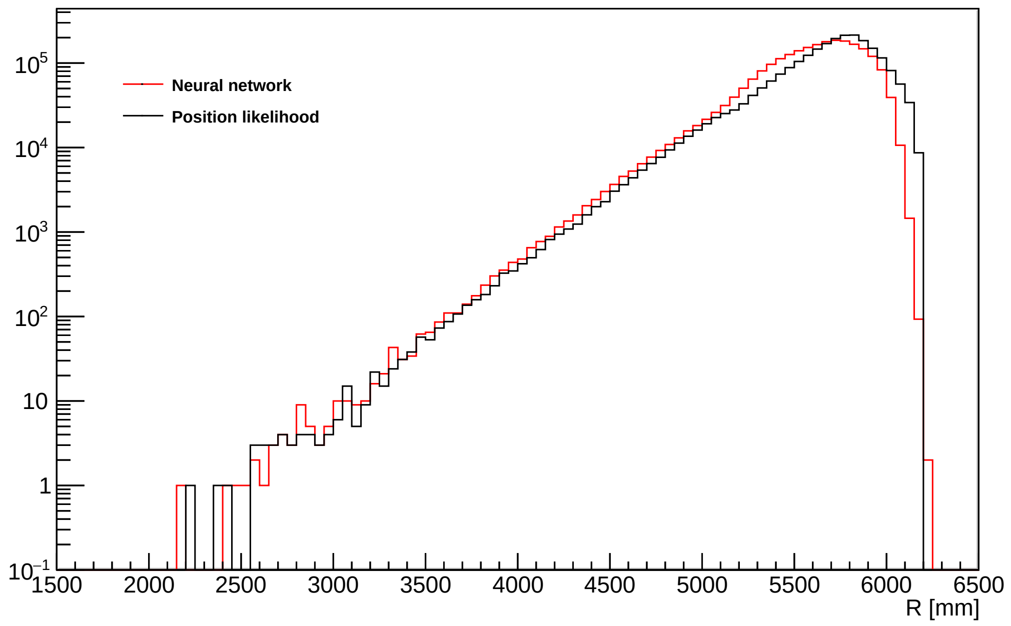


Figure 5.22: Distributions of reconstructed radii for PMT β-γ events. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

It was hypothesized that the neural network may be less likely to reconstruct in the FV and could therefore be used to extend the FV. The equivalent FV would then be the radius cut which allows the same number of events into the FV as with the position likelihood method. Even small improvements in the radius could lead to substantial gains in the FV. However, the opposite trend is observed, with the neural network being *more* likely to reconstruct in the FV. Even taking the maximum of the two fitters for each event offers no improvements.

As shown in Section 5.1.1, the magnitude of the radial bias of the neural network is less than that of the position likelihood method. The position likelihood method also has a significant negative bias, as shown explicitly in Figures 5.7 and 5.11. Since the residuals are defined as the true minus the reconstructed position, a negative bias means that the reconstruction tends to be closer to the AV than it should be on average. The results of Section 5.1.1 are evaluated on electron simulations, so to see if this is also the case for gammas, Figure 5.23 shows the distribution of the final position of the gamma particle overlayed on the reconstructed radius distributions of Figure 5.22. The "final position" is defined as the last interaction site of the gamma, which can Compton scatter multiple times and deposit its energy at distinct locations in the detector medium. This class of event is different from the electron interactions studied in Section 5.1.1 since electrons travel a much shorter distance and deposit their energy in a single localized area. The normalized radius – defined as the radius cubed divided by the radius of the AV, or R^3/R^3_{AV} – is used to better distinguish between the histograms. Since the volume of a sphere is proportional to its radius cubed, bin widths in Figure 5.23 correspond to equal volumes and can be directly interpreted as fractions of the total detector volume.

The distribution of the final gamma position is more spread out over volume than

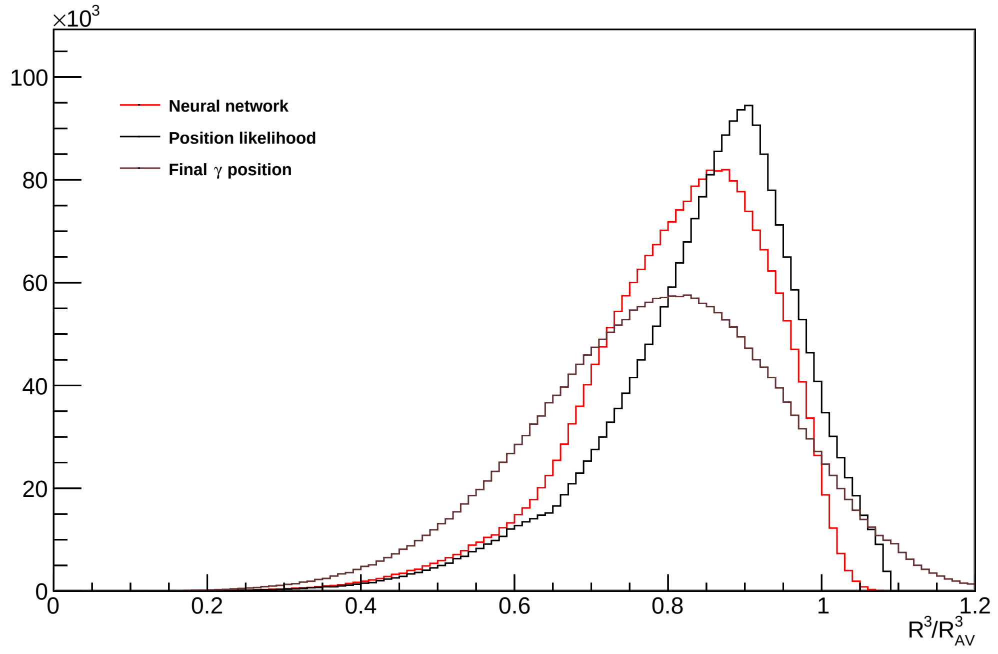


Figure 5.23: Comparison of the final gamma radius distribution to corresponding reconstructed radius distributions for PMT β-γ events. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

either reconstruction method. Still, the neural network normalized radius distribution appears to be closer to the final gamma position than the position likelihood method. To confirm this, the difference between the reconstructed and final gamma position is shown in Figure 5.24 for both the neural network and position likelihood methods.

These results indicate that the neural network is reconstructing closer to the final interaction site of the gamma than the position likelihood method. Despite being “more correct” in that sense, the reduction in bias actually negatively impacts the FV given that more gammas from radioactive backgrounds reconstruct in a given FV.

It could be argued that the final position of the gamma is not a good representation of the “true” position, given that it can Compton scatter multiple times and distribute

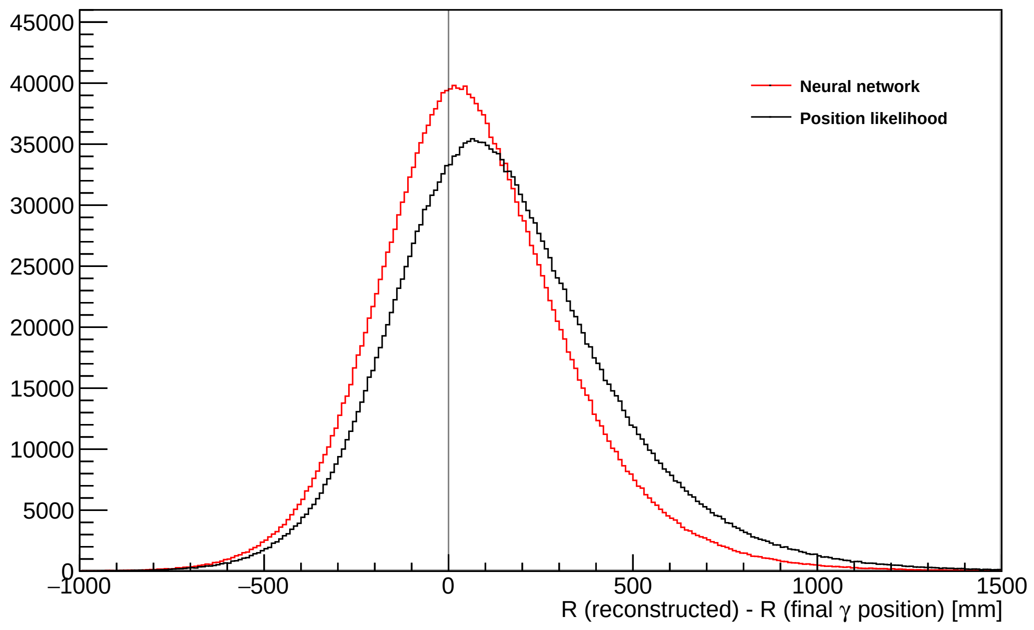


Figure 5.24: Reconstructed and final gamma radius residual distributions for PMT β-γ events. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

its energy deposition. This will in turn affect the time residual distribution and the performance of the reconstruction, both of which assume a single localized energy deposit which is not necessarily the case for gamma interactions. In particular, for gammas originating outside of the detector volume, early energy deposits will smear the time residual distribution and affect the position likelihood method by pulling the reconstructed vertex towards the AV. For these reasons, the energy-weighted radius is also used for comparison based on particle tracking in the simulations. It is defined as

$$R_\gamma = \frac{\sum_i E_{\gamma,i} R_{\gamma,i}}{\sum_i E_{\gamma,i}}, \tag{5.2}$$

where the sum is over all depositions of energy $E_{\gamma,i}$ at radius $R_{\gamma,i}$ from the gamma.

Figure 5.25 shows the distribution of the energy-weighted position of the gamma particle overlayed on the reconstructed radii distributions, as well as the final gamma radius distribution, of Figure 5.23.

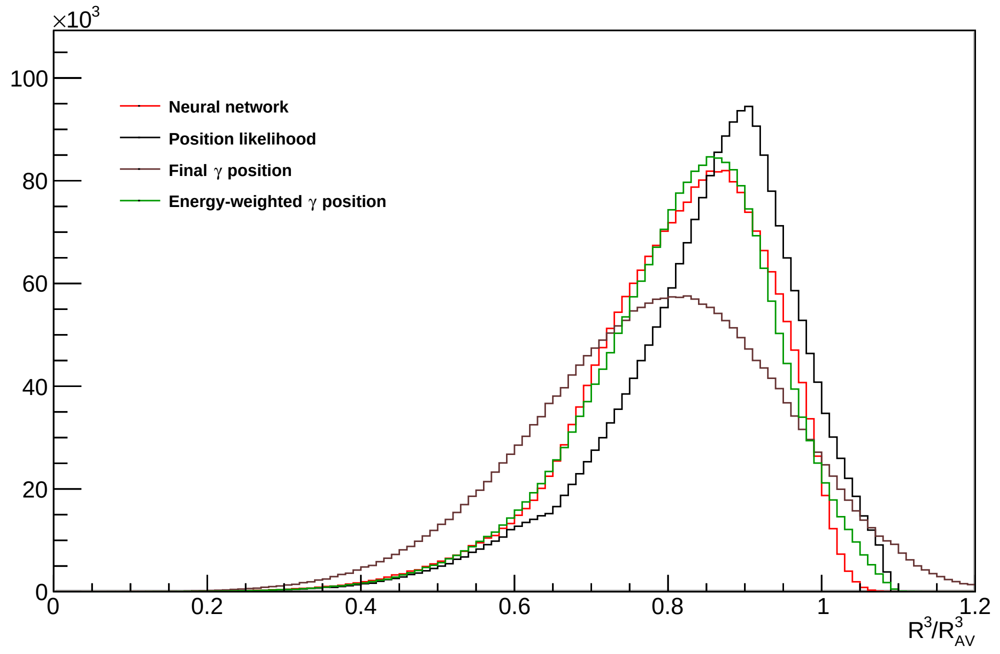


Figure 5.25: Comparison of the energy-weighted and final gamma radius distributions to corresponding reconstructed radius distributions for PMT β-γ events. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

The reconstructed radius distribution from the neural network closely matches the energy-weighted radius distribution of the gammas. This provides some insight into the physical quantity that the neural network is measuring. It is not clear what is determining the shape of the position likelihood distribution, though most likely it relates to early light near the AV from the first energy deposits of the gamma and the substantial radial bias observed in Section 5.1. Figure 5.26 shows the residuals using the energy-weighted gamma position in place of the final gamma position.

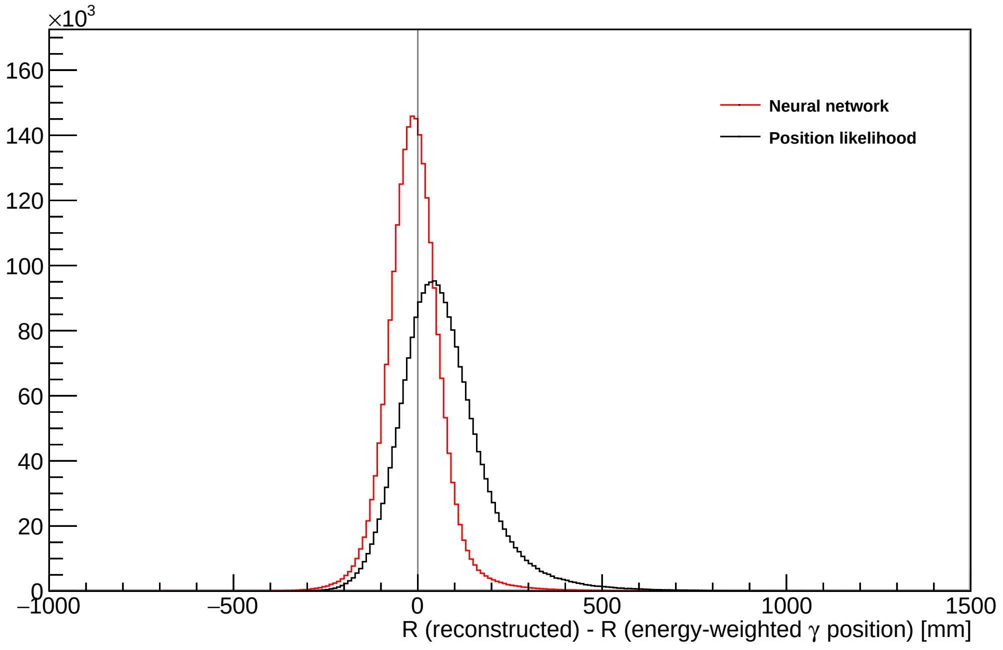
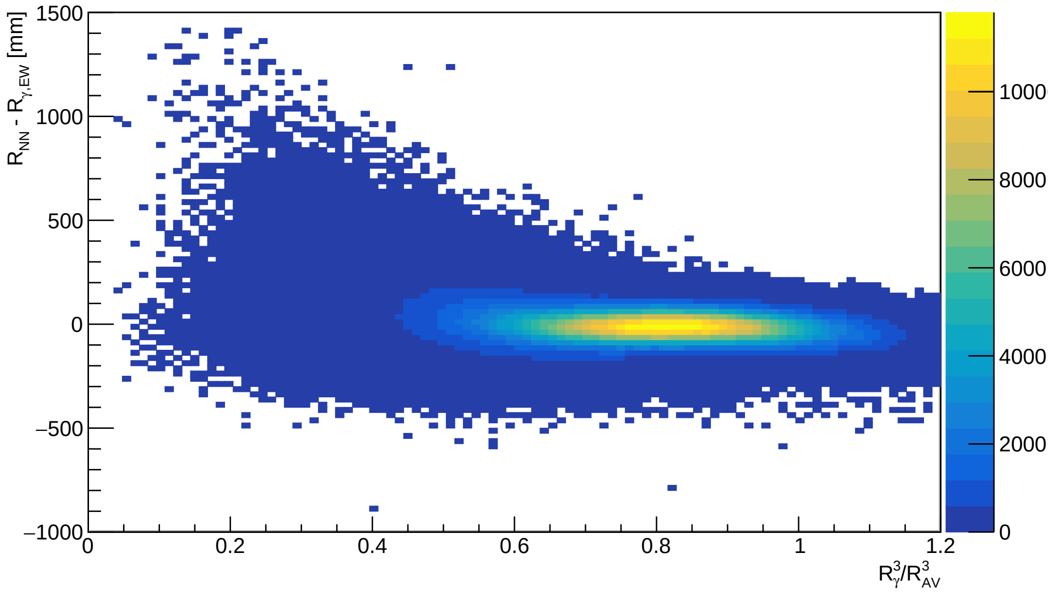


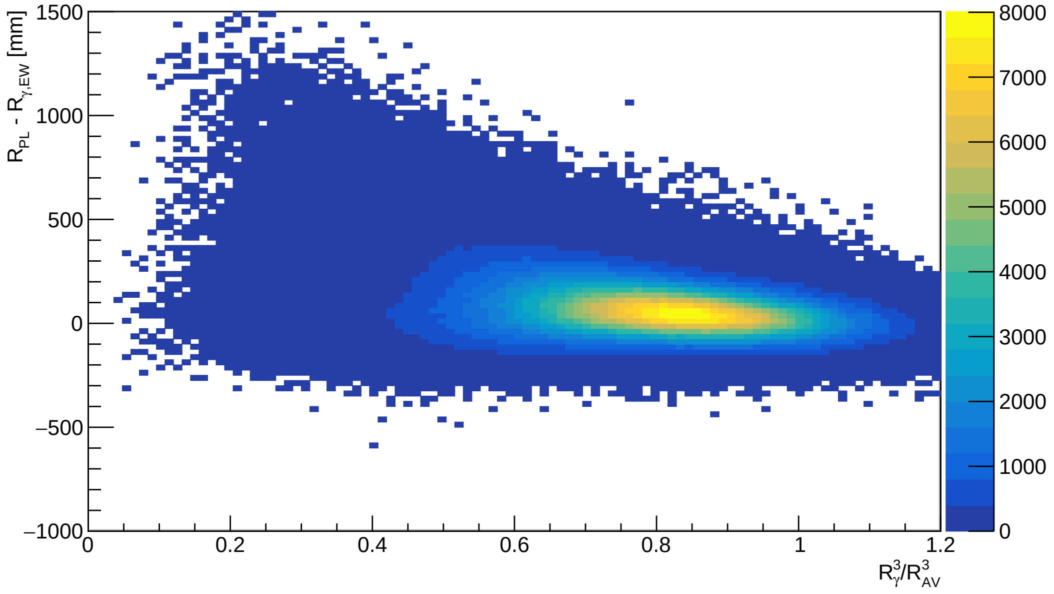
Figure 5.26: Reconstructed and energy-weighted gamma radius residual distributions for PMT β-γ events. Comparison is made between the neural network (NN) (red) and position likelihood (PL) (black) position reconstruction methods.

The residual distribution using neural network is much narrower and less biased than the same distribution using the position likelihood method, confirming that the neural network produces a reconstructed radius closer to the energy-weighted radius of the gamma. The residual distributions as a function of normalized radius (using the final gamma position) for each method are also shown in Figure 5.27. This shows the dependence of the residuals on the how far the gamma penetrates the detector. The trend is the same at all radii, with the neural network having a smaller bias and shorter tails than the position likelihood method.

Both reconstruction algorithms also exhibit the same trend of a positive tail, which intuitively makes sense: with a final gamma position far into the detector, there



(a) Neural network



(b) Position likelihood

Figure 5.27: Residuals as a function of normalized final gamma radius. 5.27a shows the distribution for the neural network, while 5.27b shows the distribution for the position likelihood method.

is a higher chance that at least one Compton scatter occurred previously, pulling the reconstruction towards the AV. Likely, the residual distribution using the neural network is tighter as the network does not depend directly on a PDF of the time residuals and is less prone to the effect of smearing. The position likelihood fitter assumes that a single electron caused the event, and the time residual distribution of gamma interactions that Compton scatter multiple times will have more hits that appear early if calculated using a single vertex.

While results presented here do not allow for a direct improvement in the FV using the neural network as originally hypothesized, this investigation has led to the discovery that the neural network can accurately reconstruct gamma vertices, assuming the "true vertex" is the energy-weighted version of all its depositions. This also indicates that the neural network is reconstructing a physical quantity without explicitly being trained to do so. The consequences of these results are unclear. Likely, better PDFs can be constructed using improved position reconstruction and the FV for the $0\nu\beta\beta$ decay analysis can be extended as a result. For example, the use of two-dimensional PDFs in energy and radius cubed may allow for much more detector volume to be included in the fit. As external backgrounds, including the gammas from β-γ events, have a clear radial dependence, improvements in position reconstruction may make this additional dimension in the fit useful.

5.3 Extensions to direction reconstruction

Likelihood-based direction reconstruction is heavily dependent on position reconstruction, both for constructing the PDFs and in setting a fixed position during the optimization. This can most easily be seen by using the true position in place of the reconstructed position in the maximum likelihood estimation for direction [112, 175]. Figure 5.28 from [112] demonstrates this effect clearly in the distribution of the angle between the true and reconstructed direction[2] under different conditions. Notably, the distribution using the reconstructed position has a broader tail and undesirable peaks at large angles. This figure also includes the effect of multiple scattering, though it is negligible in comparison to the impact of the reconstructed position and is not considered in this thesis.

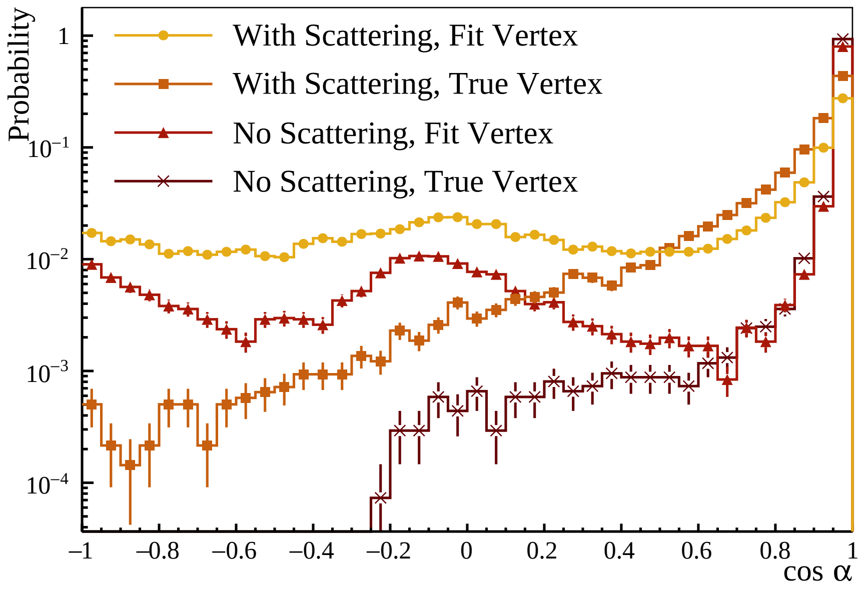


Figure 5.28: Effect of position reconstruction on direction reconstruction. These distributions are generated for 6 MeV electrons simulated uniformly in LAB with 0.6 g/L PPO in the partial fill phase. The distribution using the true position in the direction fit (orange square) is notably sharper than the distribution using the reconstructed position in the direction fit (yellow circle). Equivalent plots with multiple scattering disabled show a similar trend, demonstrating that multiple scattering is a negligible effect in comparison. Figure from [112].

[2]The notation in Figure 5.28 differs from the notation introduced and used in Section 5.3.2 for the angular residual distributions. Specifically, the variable α in the figure is used equivalently to θ in this thesis.

Due to the large drive bias in the position likelihood method, which is on the same order as the position resolution, studies to simultaneously fit for the position and direction, rather than individually with the direction depending on the result of the position, are ongoing by collaborators at the time of writing of this thesis. Given the strong interdependence of the two vertex components, it is expected that a simultaneous fit can both reduce the drive of the fitted position while also reducing the occurrence of the erroneous backward-pointing fitted directions. However, as shown in [175], the conclusions from this reasoning are not supported by preliminary results. As well, with SNO+ having transitioned to higher concentrations of PPO in preparation for tellurium loading, direction reconstruction becomes more challenging due to the faster scintillation timing profile.

Given the comparable or better performance of the neural network over the position likelihood reconstruction method, especially at higher energies, this section investigates an approach to direction reconstruction using the neural network architecture described in Section 4.2. The methodology of extending position reconstruction to direction reconstruction is described first. In the results, direct comparisons are not made between the likelihood-based direction reconstruction algorithm in [112] and [175] as it is not fully integrated into RAT and therefore difficult to run. Instead, this section aims to primarily demonstrate the feasibility of using machine learning and neural networks for direction. Some qualitative statements regarding the distributions of the two approaches are also made.

5.3.1 Methodology

5.3.1.1 Direct neural network prediction

The methodology developed in Section 4.2 can be applied to direction reconstruction with some modifications. In particular, the architecture in Table 4.1 is used with a similar training procedure described in Section 4.2.6 on the simulated electron data described in Section 4.2.4 by simply changing the targets in order to reconstruct the Cartesian coordinates of the initial event direction, rather than position. As such, the shape of the last layer does not need to be modified. However, an additional layer to normalize the output is included as the direction vector prediction, $\vec{u}_{\text{fit}}$, should always satisfy $\|\vec{u}_{\text{fit}}\| = 1$. The loss function is also modified to include a term for the negative inner product between the true initial direction of the electron and its reconstructed direction, rather than the mean squared error. This is closer to what is important in a physics analysis, as further described in Section 5.3.2.

It was quickly found that the neural network was unable to predict the direction accurately enough to be useful. This is likely due to the fact that most of the information provided in the inputs is from scintillation hits, whereas only the Cherenkov hits can be used to extract directionality. As can be seen in Figure 4.1, the Cherenkov peak is present only in the earliest time residuals. To assist the neural network, a simple time residual cut is used to remove most of the scintillation hits while keeping the Cherenkov hits. The first 8 % of PMT hits, according to the sorted time residuals in ascending order, are kept and fed to the network while the rest are discarded. This fraction is chosen by training networks with differing proportions of hits to keep and selecting the network that has the smallest loss function according to the validation set. Though 8 % was found to optimize this metric, any fraction on the order of 5 %

to 15 % was found to be sufficient to obtain good direction reconstruction.

As per Equation (4.1), the time residuals depend on the fitted event position, meaning the performance of the direction neural network also depends on the position reconstruction accuracy. The position neural network developed in Section 4.2.3.3 and presented in Section 5.1 is used in the time residual calculation to select the subset of primarily Cherenkov hits. It also has the benefit of always converging and requiring no cuts based on the fit validity. Furthermore, a custom implementation of the time residuals in Python, rather than the more advanced calculator available in RAT, is used for easy integration into the training procedure. The custom implementation assumes a straight line path from the event position to a given hit PMT, ignores transit through the AV, does not account for the AV offset, and uses an average speed of light in the two bulk media. A systematic bias on the order of 2 ns and a standard deviation on the order of 0.5 ns is observed between the custom implementation and the RAT implementation. Given this small deviation and the fact that only the relative values of the time residuals are important for the fractional cut, the simplifying assumptions made by the custom implementation are justified.

Though not explored here in detail, more advanced methods of selecting a sample composed of primarily Cherenkov hits could be implemented. Another time residual cut based on some window about the mode of the distribution was tested, but not found to offer any advantages over the simpler fractional cut. An energy- or Nhits-dependent fraction may help in reducing the number of Cherenkov hits removed by the cut. As well, with more Cherenkov photons at higher energies, a larger subset of hits (even if less pure) could still result in better performance. Introducing a sequential system that first uses a fixed fraction of PMT hits to determine the direction, followed by applying a cut on $\cos(\theta_{\gamma,i})$ and retraining the network with an even smaller subset, could afford some benefits by providing a purer selection of Cherenkov hits to the neural network. A more advanced method would be to train a new neural network to identify Cherenkov and scintillation hits individually, or force the direction network itself to choose a fraction of hits to use in the fit and learn which ones are most useful.

5.3.1.2 Drive-based prediction

As discussed in Section 5.1.2, position drive is a notable bias that the position likelihood fitter is subject to. This is also the case for the neural network, but as shown in Sections 5.1.2.1 to 5.1.2.3, the drive is reduced substantially at all energies and radii. However, the drives of each fitter are still correlated, as shown in Figure 5.29.

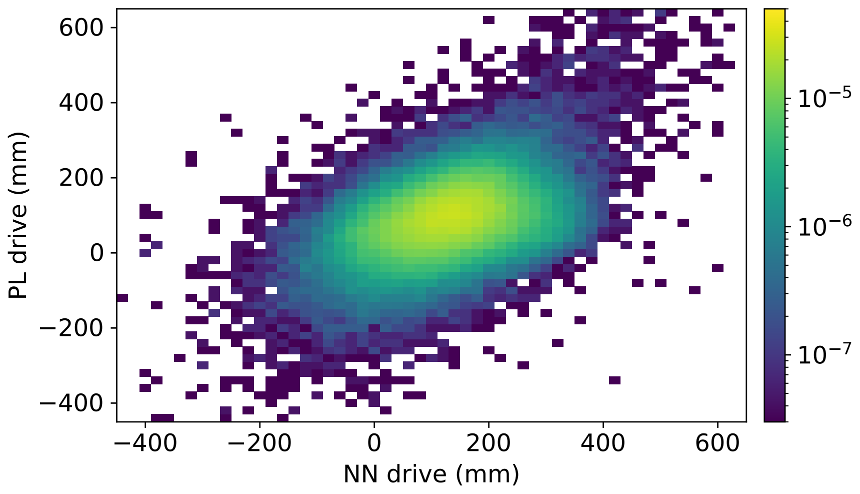


Figure 5.29: Drive correlation between the NN and PL reconstruction methods.

With correlated drives of different magnitudes, the difference in the position reconstruction values of the two methods can be used to extract some directional information. The ideal case is where one method is completely free of drive bias. Using this approach, the fitted direction can be defined as

$$\vec{u}_{\text{fit}} = \frac{\vec{x}_{\text{fit,PL}} - \vec{x}_{\text{fit,NN}}}{\|\vec{x}_{\text{fit,PL}} - \vec{x}_{\text{fit,NN}}\|}, \tag{5.3}$$

where $\vec{x}_{\text{fit,NN}}$ and $\vec{x}_{\text{fit,PL}}$ are the reconstructed positions using the neural network and position likelihood methods respectively. In Equation (5.3), the more drive biased fitter should be the first term in the subtraction, as is the case for the position likelihood method. This drive-based direction reconstruction technique was previously demonstrated on simulations using another reconstruction algorithm which analytically computes the position with multiple subsets of four PMT hits and takes the median value of each coordinate as the best estimate of the event position [179]. This technique has a higher resolution than the position likelihood method, but also has a lower drive due to the fact that selecting four scintillation hits is more probable than selecting at least one Cherenkov hit in the subset[3].

While this approach is fairly rudimentary, it is more likely to be insensitive to the exact parameters used in simulation. Both the neural network direction reconstruction approach and the likelihood-based approach are sensitive to small changes in the optics, and so any mismodelling of the simulations can drastically impact the performance on real detector data. For the former method, the training procedure depends directly on simulations and providing a reasonable time residual cut on its inputs that removes most scintillation hits. For the latter method, the shape of the PDF – and in particular the Cherenkov peak like that shown in Figure 4.1 – is dependent on

[3]This can be shown using the binomial distribution, which is a good approximation of the sampling scenario given that Nhits $\gg 4$ for a typical event. Assuming the probability of selecting a scintillation hit is $p = 0.95$ (Cherenkov radiation is <5 % of the total light output), the probability of $k = 4$ "successes" in $n = 4$ draws is $P(k; n, p) = \binom{n}{k} p^k p^{n-k} = \binom{4}{4} \cdot 0.95^4 \cdot 0.05^0 = 0.95^4 \simeq 0.8145$, or 81.45 %.

the optics. It has already been shown that changes in the PDF can have a noticeable effect on direction reconstruction [175]. Position reconstruction, on the other hand, is less prone to small changes in the simulation, and any changes in the optics would likely affect both methods in a similar way. A drive-based reconstruction approach can thus be useful on data, despite poorer performance on simulations.

5.3.2 Direction angular distribution

While the individual biases and resolutions of each Cartesian coordinate of the direction could be compared, it is simpler to look at the distribution of the angle between the true initial direction of the electron, $\vec{u}_{\text{true}}$, and its reconstructed direction, $\vec{u}_{\text{fit}}$. Denoting this angle by θ (see also Figure 5.30), a good direction reconstruction method will produce a distribution over θ that is closely centred about zero.

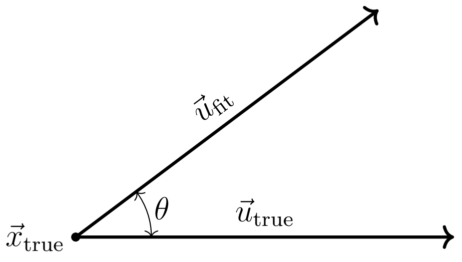


Figure 5.30: Diagram of the angle between the true and reconstructed initial direction of an event. This angle between $\vec{u}_{\text{true}}$ and $\vec{u}_{\text{fit}}$ (both corresponding to the initial electron direction) is defined as θ.

The results in this section instead use $\cos(\theta)$ to account for the symmetry of positive and negative angles. This is easily calculated using the direction vectors as

$$\cos(\theta) = \vec{u}_{\text{fit}} \cdot \vec{u}_{\text{true}}, \tag{5.4}$$

where the variables are defined as per Figure 5.30 and $\cos(\theta)$ is defined as the angular residual of an event.

The two methods described in Section 5.3.1 are next evaluated and compared by their angular residual distributions. For optimal reconstruction, the distribution of $\cos(\theta)$ should be sharply peaked at one with minimal counts in the tails.

5.3.2.1 Direct neural network prediction

Distributions of the inner product between the true and reconstructed event direction for different energy bins using the neural network are shown in Figures 5.31 and 5.32. Two figures are used to avoid clutter, with the first figure corresponding to "low energy" events and the second corresponding to "high energy" events. As the direction is normalized, the histograms are constrained to $[-1, 1]$, and a bin width of 0.2 is used. The dataset used is the same as that in Section 5.1, consisting of uniformly and isotropically distributed electron events. Similarly, labels follow the same convention as described in Section 5.1, meaning that a given energy label indicates a bin width of 1 MeV with an upper edge corresponding to the listed value. As mentioned previously, a comparison is not made against the likelihood-based method developed in [112] and [175], and these figures only contain the results using the neural network. The distribution using a randomly generated direction is also shown, and is flat across $\cos(\theta)$ as expected.

For high energies, there is a clear peak at $\cos(\theta) \simeq 1$, indicating that the neural network is able to reconstruct the direction well. Only a selection of energy bins between 10 MeV to 20 MeV are used as the difference in performance with an increase in only 1 MeV is marginal at these energies. Even at low energies, the peak at $\cos(\theta) \simeq 1$ is prominent. The angular distribution using the lowest energy range, $\leq$1 MeV, still

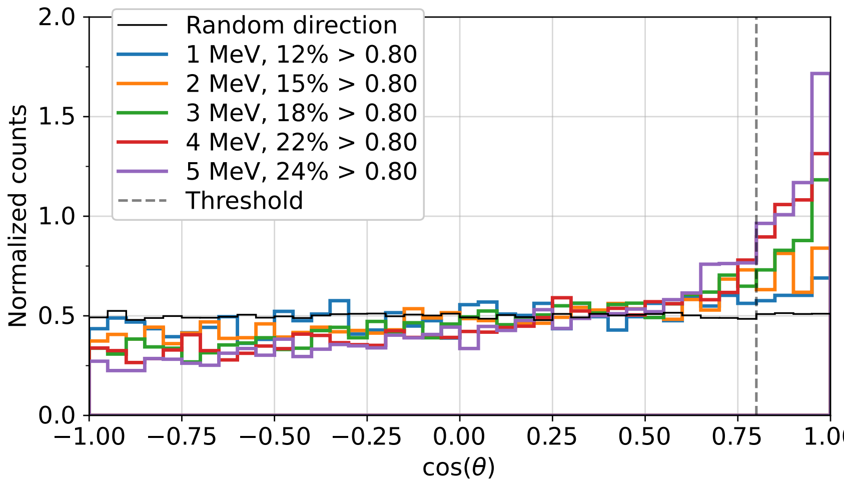


Figure 5.31: Angular residual direction distribution (low energies). Each distribution corresponds to the angle between the true and reconstructed direction using the neural network for a given energy bin. For each bin, the fraction of events with a value greater than a threshold (dashed grey line) of 0.8 is included in the legend. The distribution using a randomly generated direction for each event is also shown.

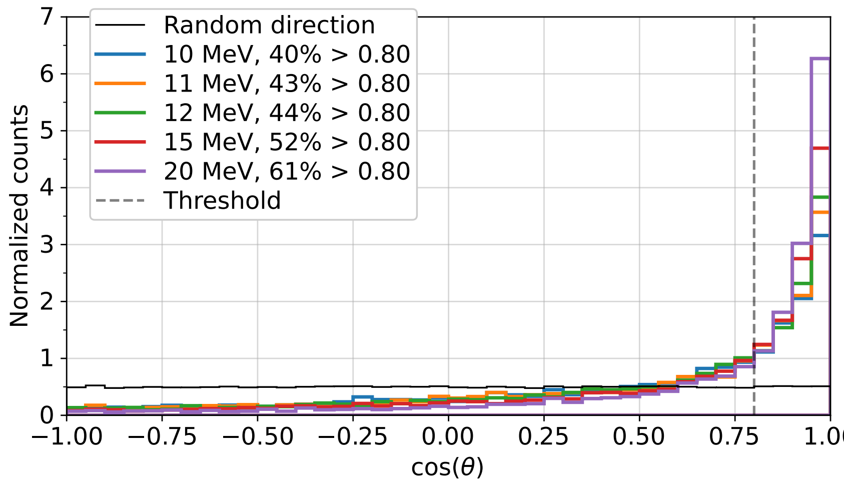


Figure 5.32: Angular residual direction distribution (high energies). Each distribution corresponds to the angle between the true and reconstructed direction using the neural network for a given energy bin. For each bin, the fraction of events with a value greater than a threshold (dashed grey line) of 0.8 is included in the legend. The distribution using a randomly generated direction for each event is also shown.

shows some preference toward the true direction. In the 2 MeV to 3 MeV range, which contains the ROI of a typical $0\nu\beta\beta$ decay analysis, the peak is distinct. All distributions in Figures 5.31 and 5.32 also contain the fraction of events with $\cos(\theta) > 0.8$, representing the "peak" of the angular residual distribution. Under the hypothesis of complete misreconstruction and a flat distribution in $\cos(\theta)$, the expected proportion of events would be 10 % in this bin. At every energy range evaluated, the proportion of events in the peak exceeds this percentage. Overall, these results demonstrate that the neural network is able to extract directional information from the raw PMT hit data.

As expected, the performance improves with increasing energy. As well, the distributions all have a maximum at the highest bin and decrease as the angle grows. The decrease is not strictly monotonic due to fluctuations in the bins, especially at lower energies and negative values of $\cos(\theta)$, though with coarser binning the distributions become closer to following a strict monotonic decrease from the peak. Notably, there are no peaks in the tails of the distributions, which is an improvement over the peaks observed at $\cos(\theta) \simeq 0$ and $\cos(\theta) \simeq -1$ in [175]. Qualitatively, the shape of the distribution using the neural network is more similar to the distribution of the likelihood method using the true event position in the direction reconstruction likelihood optimization. Given the reduction in position drive bias demonstrated in Section 5.1.2, the standard direction likelihood optimization with the neural network predicted position may perform similarly to the neural network predicted direction.

5.3.2.2 Drive-based prediction

The angular residual distribution using the drive-based method is shown in Figure 5.33. It is evaluated over the entire energy range.

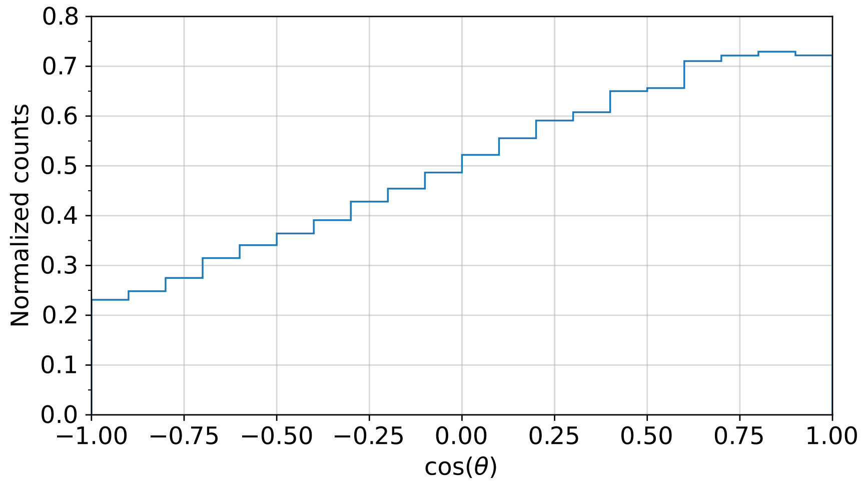


Figure 5.33: Angular residual direction distribution using the drive-based method.

Comparing to Figures 5.31 and 5.32, the peak is much less defined, and at energies >2 MeV, the bin with upper edge $\cos(\theta) = 1$ has a larger value than the distribution in Figure 5.33. While this drive-based method is substantially worse than the direct neural network predictions, there is a clear preference toward the true direction of the event.

Figures 5.34 and 5.35 show the angular residual direction distributions using the drive-based method for different energy ranges.

The performance at lower energies between the direct and drive-based methods are relatively close, with absolute differences in the fractions above the threshold ranging from 2 % to 3 %. Above 3 MeV, the discrepancy between the two methods grows. The drive-based method stops improving at >4 MeV (not shown in the figure)

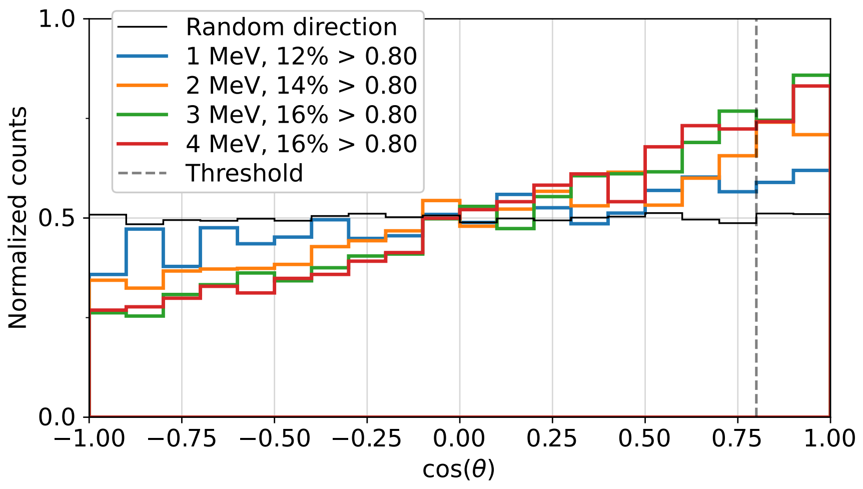


Figure 5.34: Angular residual direction distribution (low energies) using the drive-based method. Corresponds directly to Figure 5.31, though only a subset of the energy bins are included.

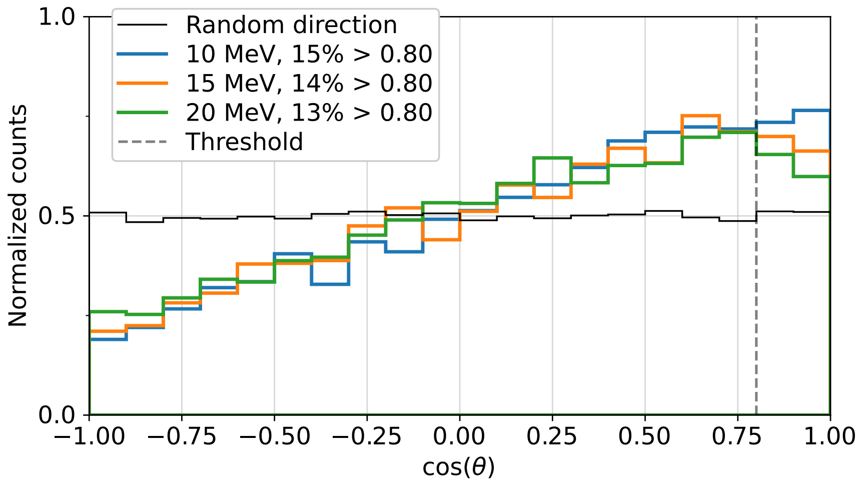


Figure 5.35: Angular residual direction distribution (high energies) using the drive-based method. Corresponds directly to Figure 5.32, though only a subset of the energy bins are included.

and worsens at very high energies, following the complete opposite pattern observed with the neural network direction method. Notably, the mode of the angular residual distribution no longer occurs at the highest $\cos(\theta)$ bin in Figure 5.35. These effects are due to the fact that the relative reduction in drive from the neural network position reconstruction method decreases as a function of energy (see Figure 5.18), making the drive-based direction method less effective. The drive-based method would thus be most suitable for use at lower energies.

5.4 Summary and discussion

This chapter presents results of a novel neural network-based event reconstruction architecture. For position prediction, it performs comparably to or better than the standard maximum likelihood approach used by the SNO+ experiment on simulated electron data. This is true for both the resolution and the drive bias. It also evaluates much faster, even in the most inefficient event-by-event inference procedure. Furthermore, it is shown that the neural network generalizes well to interactions for which it was not trained. This is demonstrated by reconstructing the gammas from PMT β-γ events which make it into the detector volume. Though the original hypothesis of increasing the FV by accepting less gamma interactions from radioactive decays in the PMTs was invalidated, the analysis led to the discovery that the neural network is far better at reconstructing the energy-weighted radius of all deposits made by gammas. This suggests that improvements will lie in developing different and better PDFs for the fit values – for example, fitting and cutting in energy-weighted radius. Finally, direction reconstruction is shown to be possible using an extension of the architecture and training procedure originally developed for position reconstruction. Results indicate the neural network has some prominent advantages over the likelihood-based approach and can be used in complementary ways given that it provides additional information.

5.4.1 Experimental implications

There are numerous implications of this work to improving the sensitivity of SNO+ and similar experiments to rare events. With better position reconstruction, improvements may be realized in the PDFs used for creating multi-dimensional fits which depend on radius or for other likelihood-based event vertex algorithms and classifiers. For example, ongoing work is being done by the collaboration in fitting $0\nu\beta\beta$ decay ROI backgrounds in both energy and radius cubed. In such fits, the FV cut is extended to encompass a much larger volume of the detector. With a much smaller radial bias and better modelling of the energy-weighted radius of gamma interactions, the neural network could allow for the FV to be extended further. As well, multi-site event identification and discrimination depends on the time residual PDFs of the two classes of events. In particular, multi-site events have more hits which appear early relative to the event time, smearing their time residual distributions. Yet another area where improved PDFs from more accurate position reconstruction could be of benefit is in the maximum likelihood estimation of direction in liquid scintillator. The performance of direction reconstruction is shown to have a strong dependence on the position used in both generating the two-dimensional PDF and in the optimization procedure itself [112,175]. Given the reduction in drive from the neural network position reconstruction algorithm, a bias which was found to be particularly problematic for maximum likelihood direction estimation, improvements are expected by simply swapping the position values in the optimization routine.

Direction reconstruction can be used for either signal identification or background removal, depending on the analysis. Solar neutrinos are an irreducible background in the $0\nu\beta\beta$ decay search as their energies, particularly those from ^{8}B, overlap with the ROI. Where solar neutrinos are the signal, direction reconstruction can help to identify those interactions and remove events with directions anti-correlated with the distance vector from the Sun to the detector. This could, in turn, allow for constrained solar neutrino fluxes to be measured. Direction reconstruction is challenging in liquid scintillator, but has been demonstrated using both a maximum likelihood estimation and with the deep learning framework developed here. The neural network was shown to overcome some issues observed in the likelihood-based approach, including a notable reduction in the frequency of backward-pointing predictions.

In terms of practical implications, a trained neural network is substantially faster to evaluate than the likelihood-based optimization routines. While neither position nor direction reconstruction using deep learning are currently applied to incoming data as part of the processing pipeline, the neural network framework has already been integrated into RAT by the author which should make either model's inclusion into the processing chain straightforward. As well, the methods developed here are complementary to existing algorithms and can be used to identify shortcomings and possible improvements. The discussion which follows touches on both current and potential future work where the complementary nature of this architecture and technique could stand out.

5.4.2 Future work and prospects

In addition to the above experimental implications, there are several avenues of extension – both on the SNO+ experiment and on other rare event search experiments – that could be explored. Some preliminary work has been proposed and done by the author in evaluating the feasibility of these extensions.

The first extension is to seed the likelihood-based fitter with the neural network predictions. This could be applied to either position or direction, since the optimization routine requires an initial "guess" of the quantity or quantities to be optimized. Starting at a guess closer to an optimal value should reduce the number of iterations needed to converge on a solution. This was tested for position reconstruction, although results did not agree with this hypothesis, and, in fact, seeding with the neural network substantially *increased* the average fit time per event. Likely, the step size of the optimizer needs to be adjusted to account for the fact that the guess is closer to the optimal value on average. More work should be done in this area. Additionally, as previously discussed for likelihood-based direction reconstruction, there is a strong dependence on the reconstructed position used in the fit. Given the lower drive bias of the position neural network, it is expected that the likelihood-based direction reconstruction procedure could be improved further, even without using the neural network-based direction reconstruction.

Another extension investigated by the author is providing an uncertainty quantification of the predicted results of the neural network. This was done using the principle of dropout (discussed in Section 3.2.6) and theoretical results that show neural networks with dropout approximate a Gaussian process [180]. To obtain uncertainty estimations, a distribution of position predictions for a given event can be created by applying the neural network to that input multiple times with dropout enabled on inference. Summary statistics of the output can then be used to estimate the mean value of the prediction and its uncertainty. Preliminary results produced sensible uncertainties that decreased as a function of energy and Nhits (following the same trend of resolution and the intuitive explanation that more hits provides more information) and increased substantially with random inputs (both uncharacteristic to what the network was trained on and unphysical). However, dropout also made the performance of the neural network noticeably worse. Uncertainty estimation was not investigated further, but would be useful for future studies. In addition to dropout as an uncertainty estimator, Bayesian neural networks [181] are another method which can provide uncertainties on their prediction and are used in various applications [182].

One straightforward extension that has not yet been tested thoroughly is a modification to the loss function of the neural network. Some effort was made into incorporating a penalty term based on the time residuals, as per Equation (4.1). This was not found to be effective, likely due to the complexities of the time residual PDF that are not accounted for in a simple minimization of the absolute value of the term. Another addition that could be incorporated into the loss function is a term that explicitly penalizes the drive of the neural network, as was first proposed in [179] for a related but different purpose. As shown in this chapter, two position reconstruction methods of differing but correlated drives can be used to extract directional information, particularly at low energies. Such a term may also aid the network in reducing its overall bias and resolution by nature of decreasing the drive bias and providing more assistance to the network.

An extension actively being pursued by other SNO+ collaborators is the use of different architectures for event reconstruction. Much of this work is being done in coordination with the author. As an example, graph neural networks (GNNs) [183–185] have been investigated in terms of their performance against the architecture here in both accuracy and evaluation time [186]. While results are preliminary, they are very promising, specifically at lower energies. The author proposed this technique and helped to begin the project.

Finally, this work is broadly applicable to other problems both in and beyond the SNO+ experiment. The network architecture developed in Chapter 4 is designed for flexibility of the input data and the outputs. The prediction of another variable, such as energy, could be done with some modifications to non-fundamental components of the architecture (e.g., the commutative operation and the number of output nodes for energy reconstruction) and the inclusion of more channels (e.g., at least one of the integrated charge variables – QHS, QHL, QLX – for energy). With some additions, the flexibility of the architecture is shown directly to be successful with direction reconstruction. Furthermore, this flexibility allows for other experiments to straightforwardly adapt the network structure to their data and analysis tasks. The architecture presented here does not depend on any symmetry, does not require a fixed length input vector, is permutation invariant to the inputs, and is easily scalable. Thus, other experiments can benefit from this transferable tool.

Part II

Applications of Neural Networks and Deep Learning to PPC HPGe Detectors

Chapter 6

Pulse Denoising in PPC HPGe Detectors

Events from HPGe detectors are collected as one-dimensional traces of charge as a function of time. All pulses contain electronic noise of a fixed amplitude, which can mask the detailed properties of an event, and at the lowest energies (where the electronic noise is proportionally large compared to the signal) mask the event completely. Removing this noise can help to advance numerous physics outcomes by revealing the otherwise obscured pulse shape, which depends on the position(s) of the energy deposit(s) in the detector.

This chapter develops a new method for removing electronic noise from one dimensional signals. While the developments and results are focused on PPC HPGe detectors, the approach is highly flexible. Section 6.1 introduces some traditional, non-machine learning signal denoising techniques that are later used for benchmarking. Section 6.2 presents the deep learning-based denoising approach developed by the author. It includes how the model was developed, the datasets used for training and evaluation, and the training procedures implemented. This chapter is heavily adapted from the author's publication in [1]. Chapter 7 then presents results of applying this methodology to simulations and data.

6.1 Traditional pulse denoising

There exist numerous techniques to remove electronic noise from time-series data. The first – and simplest – one is a moving average filter over w samples. The selection of its only hyperparameter, w, requires a trade-off between the level of noise reduction and the preservation of details such as edge sharpness in the pulse shape. As a low-pass filter, it removes high frequency noise. It is also a specific case of a convolution, where the weights defining the kernel are all equal and sum to unity. Moving average filters can be simple, as described here, or weighted, where the parameters of the kernel can differ. One common extension to the moving average for time-series signals is the exponential filter [187], where the smoothed output of a given sample is the weighted sum of the current sample and the previous prediction. The algorithm is recursive and is defined by the smoothing constant $\alpha \in (0, 1)$. For a signal $\vec{x}$ and a smoothed representation $\vec{s}$, beginning at index $i = 0$ with $s_0 = x_0$, the recursive function

$$s_i = \alpha x_i + (1 - \alpha)s_{i-1}, \tag{6.1}$$

fully defines the exponential filter for $i > 0$. Because of the repeated multiplication of prior terms by $1 - \alpha$, the weighting of prior points decreases exponentially with distance.

The Savitzky-Golay filter [188] is another noise removal technique that is more complicated than either the moving average or exponential filter. It removes noise by fitting a degree-p polynomial to w adjacent samples centred about a given point of the signal and evaluating the fitted polynomial at this point. The Savitzky-Golay filter can be implemented as a weighted moving average with coefficients that depend on w and p [188].

Another denoising method investigated in this thesis applies thresholding rules to the wavelet decomposition of the noisy signal [189]. The wavelet decomposition, or transform, is a type of mapping to a domain which allows for analysis of localized frequencies. This is in contrast to the Fourier transform, which is applied globally to the signal and loses all temporal information as a result. For this reason, sharp or discontinuous data are represented poorly in Fourier space as short frequency components are comparatively small relative to frequency components which persist across the entire signal. The short-time Fourier transform (STFT) was invented to better represent the frequency components of localized areas by applying the Fourier transform separately to equal-sized time windows of a signal. However, the STFT has limitations due to the trade-off required between the size of the window and the time and frequency resolutions of the resulting spectrum. The wavelet transform further extends the STFT idea by using different window functions depending on the frequency band [190]. The basis functions of the Fourier transform are sine waves, while the basis functions of the wavelet transform are short, oscillating and decaying waves called wavelets. Similarly to how a signal in Fourier space is represented by the sum of sine waves of differing frequencies, a signal in wavelet space consists of the sum of wavelets of differing shifts and scales.

Wavelet-based denoising requires a choice of mother wavelet and order as well as thresholds for the wavelet coefficients. Table 6.1 lists a set of common mother wavelets. With wavelet-based denoising, the method of thresholding can also be varied. VisuShrink [191] applies a global threshold to the wavelet coefficients, while BayesShrink [192] determines thresholds at each subband of the wavelet by minimizing

Bayesian squared error risk. Once a threshold is computed, either hard thresholding (keeping the coefficient if greater than the threshold) or soft thresholding (shrinking the coefficient toward zero by the threshold) can be applied to the wavelet coefficients. After thresholding, the inverse wavelet transform is applied to reconstruct the signal. Due to the numerous choices that can be made in wavelet denoising, the technique has a large number of hyperparameters.

Table 6.1: Common wavelet functions for denoising. Table includes the mother wavelet and order for each method (where applicable).

Mother Wavelet	**Order**
Haar	N/A
Daubechies	2 – 38
Coiflet	1 – 17
Symlet	2 – 19
Biorthogonal	1.1, 1.3, 1.5, 2.2, 2.6, 2.8, 3.1, 3.3, 3.5, 3.7, 3.9, 4.4, 5.9, 6.8
Reverse Biorthogonal	1.1, 1.3, 1.5, 2.2, 2.6, 2.8, 3.1, 3.3, 3.5, 3.7, 3.9, 4.4, 5.9, 6.8
Meyer	N/A (finite impulse response approximation)

The last technique examined in this work is the Kalman filter [193], which relies on knowledge of an underlying model of the data. Kalman filtering provides an estimate of the true value based on a model of the event itself and the measured samples. A weighted interpolation can be made between the two, with the weights depending on the uncertainty of the event modelling technique and measurement error. A hyperparameter can also be set to control the weighting.

These traditional denoising techniques are later compared in Section 7.1.2 to the autoencoder developed next.

6.2 Deep learning-based pulse denoising

6.2.1 Motivation

An efficient denoising algorithm can help advance searches for rare event interactions. Noise reduction techniques can allow one to identify low-energy signal events that would otherwise be dominated by electronic noise. This is of direct relevance to experiments searching for rare events at low energies, such as solar axions, dark matter, violation of the Pauli Exclusion Principle, and electron decay [194–197]. A reduction in noise would also allow for better background rejection techniques that are based on pulse shapes, such as the rejection of slow energy-degraded pulses in germanium detectors searching for signals at low energies [198, 199].

Denoising could also provide more accurate measurements of pulse amplitudes, leading to a better energy resolution. Many pulse height estimation algorithms [200, 201] use effective averaging windows with a given shaping time. While an overall reduction in the energy resolution is difficult to achieve compared to these highly efficient algorithms, denoising the pulses beforehand can reduce the shaping time required to obtain a comparable energy resolution. This can allow for shorter traces to be collected (more efficient data storage), a smaller sampling period to be used (more detailed pulses), and/or a higher data collection rate (lower energy thresholds).

Typically, noise will not be Gaussian and may contain different components from the environment of the detector. With machine learning and neural networks, a noise removal model can be learned that adapts to the data and the noise. Some of the simpler traditional denoising algorithms outlined in Section 6.1 are very fast, though a trained neural network may offer some performance gains over the more complicated wavelet threshold denoising method.

6.2.2 Development of the model

Denoising autoencoders are a promising way to extract the most useful information from data while discarding irrelevant features such as noise. As described in Section 3.4, this is due to the general premise of autoencoders preserving only the most important components of a signal by nature of its compressed structure.

The architecture that defines the encoder and decoder must also be carefully designed. Due to the importance of fine features in the pulse, preservation of the underlying signal is crucial. For instance, single- and multi-site event discrimination (as illustrated in Figure 2.4) and position reconstruction (which depends on how fast the pulse rises) both rely on small details in the signal that typically fall within a short time window of $<1\,\mu$s. As well, efficient learning is important. A fully-connected neural network is not suitable for the task of denoising because it considers all nodes in the context of each other. For a given input sample, only nearby samples are relevant, and a fully-connected architecture would be forced to learn this property of the data inefficiently (and perhaps incompletely).

Reflecting locality in the structure of the network is a more reasonable approach which should allow for faster and more effective learning. The architecture used in this thesis is thus fully convolutional and benefits from superior computational and memory efficiency over fully-connected networks [132]. The parameter sharing aspect of convolutional networks is particularly important as it forces the network to learn to remove noise in a consistent manner across the entire pulse, emphasizing feature locality. It also ensures that the network is equivariant to shifts on the time axis and thus independent of the relative position of the pulse in the trigger window. Finally, weight sharing greatly reduces the number of parameters to train, effectively acting as a regularizer to prevent overfitting.

The fully convolutional nature of the model furthermore allows for a variable length input pulse, regardless of the size of the pulses it was trained on. This is extremely useful for applications involving complicated sampling procedures or the processing of continuous streams of data. It furthermore eliminates the need for retraining solely based on changes to the size of the event window, and can allow for better weight initialization schemes in other detector setups where this window size is different.

The next section provides details on this fully convolutional architecture. Though developed for HPGe detector signals, it is versatile and applicable to general one-dimensional signals.

6.2.3 Description of the model

6.2.3.1 Inputs and outputs

Signals are collected from the PPC HPGe detector described in Section 2.2. Each signal is a pulse that consists of M voltages sampled at 8 ns intervals and encoded with 16 bits. M can technically be any number, but is typically either 4096 or 8192 samples. Thus, the inputs and outputs of the models are M-long vectors of voltage samples.

All data pulses are preprocessed to have unit amplitude using a trapezoidal filter [200], as described in more detail in Section 6.2.5. An example of three preprocessed detector pulses with different rise times – indicating different event positions in the detector – and different signal-to-noise ratios is shown in Figure 6.1. The full length pulses are plotted and the “rise region” between approximately 16 µs and 17 µs is highlighted in the inset.

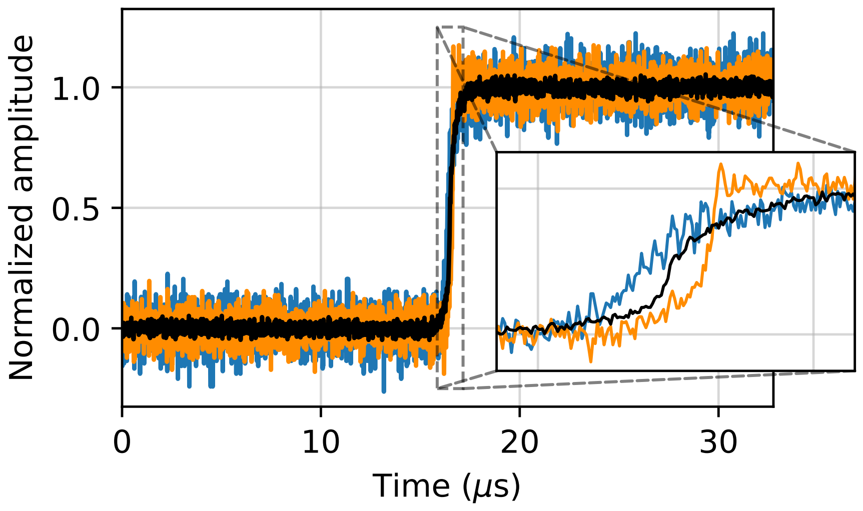


Figure 6.1: Three example pulses from the PPC HPGe detector. Each pulse has a substantially different rise time. All pulses are preprocessed such that they have a baseline of zero and an amplitude of one. The black pulse has apparently less noise because it was originally a higher amplitude pulse, so that the electronic noise is smaller relative to its amplitude.

6.2.3.2 Network architecture

For a general one-dimensional convolution, the size of the output O is given by Equation (3.19). For convenience, it is restated here as

$$O = \frac{I - K + 2P}{S} + 1, \tag{6.2}$$

where I is the size of the input, K is the size of the kernel subject to $I \geq K$, S is the stride length, and P is the symmetric padding applied to both ends of the input. For a stride length or window size greater than one, the size of the output of the convolution may be different from the input, which has consequences on the selection of layer parameters. While the output size can be forced by padding the input, it is not apparent how to pad the pulses. Prepending or appending a constant value does not account for misalignment due to the imperfect normalization. Padding also ignores the noise.

An alternative approach is to use transposed convolution layers in the decoder to "undo" the size changes caused by non-padded convolutions. The transposed convolution operation uses the transpose of the convolution matrix – a sparse matrix containing elements of the filter for computing the convolution using matrix multiplication – to switch the forward pass with the backward pass [202]. This effectively acts as a form of upsampling.

Both the encoder and decoder have "blocks" of layers, where a "block" consists of a convolution followed by a two-fold average pooling operation or a two-fold upsampling operation followed by a transpose convolution for the encoder and decoder respectively. The number of encoder blocks must be the same as the number of decoder blocks. Every convolution and transpose convolution layer *except* for the last is followed by a ReLU activation function [130]. No activation is applied to the final convolution layer as pulses can have values outside of the range $[0, 1]$ or $[-1, 1]$, and bounding the output was found not to be necessary.

The sizes of the windows in each layer can be chosen arbitrarily. However, the window size in the transposed convolution of the decoder must match the window size of the corresponding convolution in the encoder to ensure that the output sizes match. The window sizes are also subject to some restrictions to ensure that the output of the decoder matches the size of the input to the encoder. This can be seen from Equation (6.2), which simplifies to

$$O = I - K + 1, \tag{6.3}$$

for a stride length of one and no padding. If I is even (odd), K must be odd (even) in order for O to always be even. O must be even to ensure that the size of the input to the subsequent downsampling layer is divisible by two. By writing Equation (6.3) more generally for hidden layer l in the encoder and including the two-factor downsampling operation in the size,

$$O_l = \frac{O_{l-1}}{2} - K_l + 1, \tag{6.4}$$

and then applying this equation recursively until the input, indexed by $l = 0$ such that $I = O_0$, the form of the output size of layer l in the encoder can be written as

$$O_l = \frac{I}{2^{l-1}} - \sum_{i=1}^{l} \frac{1 - K_i}{2^{l-i}}. \tag{6.5}$$

Here, i indexes all hidden convolutional layers prior to and inclusive of layer l. Equation (6.5) demonstrates that I must be divisible by 2^{l-1} and each kernel (minus one) must be divisible by a power of two dependent on its index and how many layers deep the encoder spans. The window size in each convolution layer is thus selected such that these conditions are true for each layer in the network. Conversely, pulses of variable length are subject to this requirement for the chosen window sizes.

6.2.3.3 Network architecture specifics

Section 6.2.3.2 presented the general formulation and properties of the network architecture with no specific numbers. As was done in Sections 4.2.3.2 and 4.2.3.3, this abstraction is deliberate because the architecture can easily be extended to other problems, and the specific hyperparameters defining the kernel size, the number of filters, and the number of layers may need to be modified.

The specific architecture of our autoencoder is described in Table 6.2, with each layer consisting of its stride, window, and output size. The first element of the output size is the temporal length after the operation is applied, while the second element is the number of filters. The batch size is not included in the output shape as it is arbitrary and does not affect the network structure. For illustration, the table uses a fixed input length of 4096 samples. A different input shape will only change the temporal dimension of the output shape per Equation (6.5) as the number of filters does not depend on the input. As well, lines after the header of the table indicate separation of the input, encoder, decoder, and output.

The network begins with an eight-filter convolution operation with a stride length and window size of one. This does not change the temporal dimension of the input, but rather acts as a sort of "preprocessing" layer that increases the overall dimensionality of the signal. The decoder also ends with a convolution layer where the stride length, window size, and number of filters are set to unity to recover the original shape of the pulse.

Both the encoder and decoder have three blocks of layers, where the definition of a block is given in Section 6.2.3.2. As each block compresses its input across the

Table 6.2: Summary of the convolutional denoising autoencoder architecture. Included in this table is the type, stride length S, window size K, and output shape O of each layer. All shapes are calculated using a fixed input pulse size of 4096 samples. The output shape does not include the batch dimension, and the architecture is valid for any batch size.

Layer	**Stride**	**Window**	**Output**
Input	N/A	N/A	4096, 1
Convolution	1	1	4096, 8
Convolution	1	9	4088, 16
Average Pooling	2	2	2044, 16
Convolution	1	17	2028, 32
Average Pooling	2	2	1014, 32
Convolution	1	33	982, 64
Average Pooling	2	2	491, 64
Convolution	1	33	459, 32
Transpose Convolution	1	33	491, 32
Upsampling	2	2	982, 64
Transpose Convolution	1	33	1014, 64
Upsampling	2	2	2028, 64
Transpose Convolution	1	17	2044, 32
Upsampling	2	2	4088, 32
Transpose Convolution	1	9	4096, 16
Convolution (Output)	1	1	4096, 1
Total number of parameters: 286 145			

temporal dimension by a factor of approximately two due to the downsampling operation, the encoder compresses the original input by a factor of approximately eight. The compression factors are both slightly greater than stated due to Equation (6.3) for the convolution layers with stride one, which still reduce the output size slightly.

The window sizes are chosen to *increase* for each layer in the encoder and *decrease* for each layer in the decoder. As well, they are all smaller or approximately the same size as a typical pulse rise time. Odd numbered sizes are used in each convolution layer to satisfy the condition described at the end of Section 6.2.3.2 for each layer in the network starting with an input of 4096 samples.

6.2.4 Datasets

Both simulations and real data from the PPC detector are described here. Each set is used in at least one of the model training or evaluation phases.

6.2.4.1 Detector data

^{241}Am source Data were collected from the detector using a 10 µCi ^{241}Am encapsulated source, which produces 60 keV gamma rays. At this energy, electronic noise is a significant component of the pulse. A collimator was used to direct the emitted gammas to a specific location along the longitudinal axis in the cylindrical detector. The data thus consist of runs which correspond to discrete locations of the collimator spanning the entire height of the detector, ensuring that the set contains a representative distribution of events. Furthermore, the interactions of these gamma rays in the detector are almost entirely single-site since Compton scattering is unlikely at these energies. Each trace is 8192 samples in length. Data were collected in December 2021.

Figure 6.2 shows the energy spectrum of the entire ^{241}Am dataset over an energy range of 45 keV to 75 keV. Events within ±2 keV of the 60 keV peak are highlighted in the figure and taken to be associated with the source gamma rays. The event energies are estimated using a trapezoidal filter.

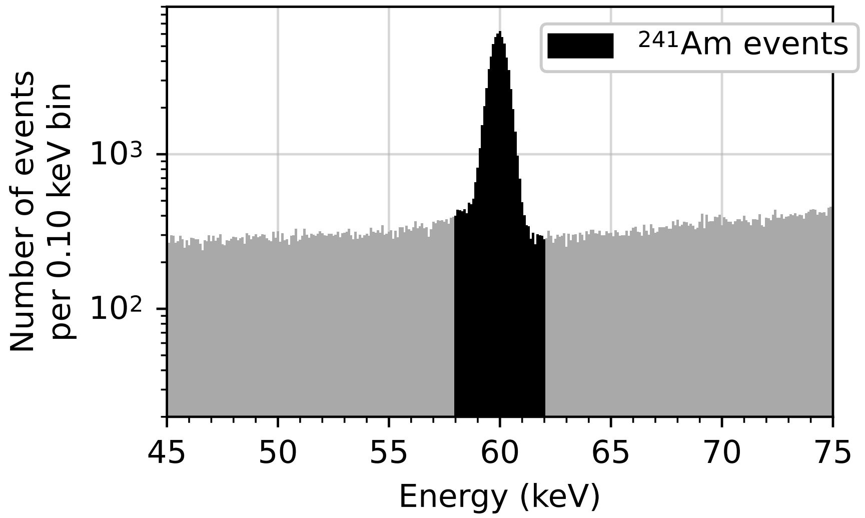


Figure 6.2: Energy distribution of the ^{241}Am dataset. Amplitudes are converted to energies via a trapezoidal filter. Events in the dark region are within ± 2 keV of the peak and taken to be signal events from the ^{241}Am source, while events in the lighter region are taken to be backgrounds. Note that the y-axis is logarithmic. Figure adapted from [1].

^{60}Co source Data were also collected from the detector using a ^{60}Co source that produces gamma rays with energies of 1173 keV and 1332 keV. In practice, data were collected over energies ranging from approximately 0.5 MeV to 3 MeV, including events from room backgrounds and the ^{60}Co source. These data include many multi-site events from gamma rays Compton scattering in the detector. Each trace is 4096 samples in length. Data were collected in June 2020 and October 2020.

Figure 6.3 shows the energy spectrum of the ^{60}Co dataset over an energy range of 1 MeV to 3 MeV. As with the ^{241}Am dataset, event energies are estimated using a trapezoidal filter. Events within ± 20 keV of the two gamma peaks are shown by a darker fill in the histogram of the figure. Other peaks in the data can also be observed, including at 2.6 MeV from the ^{208}Tl isotope of the thorium decay chain.

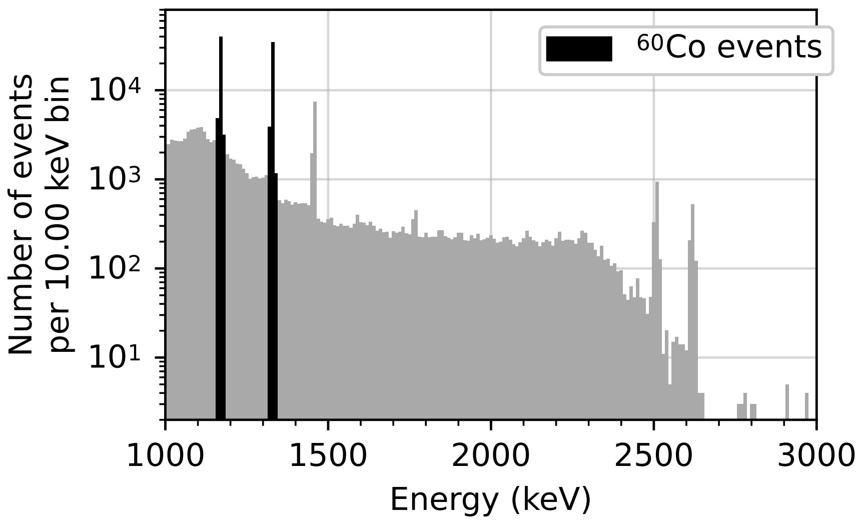


Figure 6.3: Energy distribution of the ^{60}Co dataset. Amplitudes are converted to energies via a trapezoidal filter. Events in the dark region are within ±20 keV of the two peaks at 1173 keV and 1332 keV from the ^{60}Co source. Note that the y-axis is logarithmic.

Noise A large number of noise traces were collected from the detector in order to train the models with realistic electronic noise. These were obtained by randomly triggering the digitizer to read out signals from the PPC detector. Noise data were collected at three different times over a period of two and a half years: July 2019, January 2021, and December 2021. At each time point, the detector was under different operating conditions, resulting in a diverse noise set. Traces collected in December 2021 are 8192 samples in length, while the remaining sets are 4096 samples in length.

Signals in the noise data were filtered to remove any events that occasionally occur in the same trigger window. This was done by rejecting outliers based on the baseline and root mean square (RMS) of the baseline, calculated over the first 1000 samples of each trace. Similarly, pulses with outliers in the minimal or maximal values from a trapezoidal filter with a gap window of 1.8 µs and a rise window of 6 µs were removed from this dataset.

Detector datasets summary The real detector datasets that are outlined in Section 6.2.4.1 are summarized in Table 6.3. The table contains the type of data, the primary energy peak(s) that are a direct result of gamma interactions if from a calibration source, and the total number of events collected.

Table 6.3: PPC HPGe detector calibration and noise datasets. Includes the type of data (either from a calibration source or noise), the energy peak(s) that result from the source gammas (if applicable), and the total number of events collected.

Data	**Energy peak(s)** (keV)	**Number of collected events**
^{241}Am	60	292 861
^{60}Co	1137, 1332	216 963
Detector noise	N/A	21 394 829

6.2.4.2 Simulated data

Library pulses A detailed physics-based simulation was used to create a set of 1724 “library” pulses on a 1 mm × 1 mm grid in radius and height to represent pulses uniformly in the azimuthally-symmetric PPC detector [203]. This position-dependent basis set was created using the siggen simulation software described in Section 2.2.4. The library pulses can be used to infer realistic clean signals underlying actual single-site events in the detector using a χ^2 minimization between a normalized data pulse and every library pulse in the basis set [203],

$$\chi^2 = \sum_{j}^{M} \frac{(z_{k,j} - x_{i,j})^2}{\sigma_i^2}, \tag{6.6}$$

where the $x_{i,j}$ is the j^{th} sample of the i^{th} data pulse, $z_{k,j}$ is the j^{th} sample of the k^{th} library pulse in the basis set, and the quantity is evaluated over M samples. σ_i represents the noise level in the data pulse, which is generally taken as the RMS of the noise in the pre-trigger baseline.

Multiple sets of library pulses were generated with different preamplifier time constants to further expand this dataset. For a given position, this affects the rise time and the curvature of the pulse. Using preamplifier time constants of 0 ns to 80 ns in increments of 10 ns, the library pulse dataset was expanded to 15 516 unique traces.

Piecewise linear smoothed pulses A set of pulses that look similar to the library pulses was generated by using piecewise mathematical functions. These pulses mimic the shape of the library pulses without requiring any complex physics simulations, and do not depend on the details of a specific detector. Each pulse was composed of two linear pieces connected at a varying fixed point to mimic the slow and fast portions of the rise. All pulses have an amplitude of unity and a rise time between 25 samples and 125 samples. The pulses were smoothed using a moving average with a window size of 5 % of the rise time. A total of 20 070 unique traces were created using this procedure. These signals are referred to as piecewise linear smoothed (PLS) pulses.

Figure 6.4 shows an example library pulse and PLS pulse, each with a short rise time of 280 ns. The PLS pulses tend to be noticeably sharper than the library pulses and only roughly approximate their general shape.

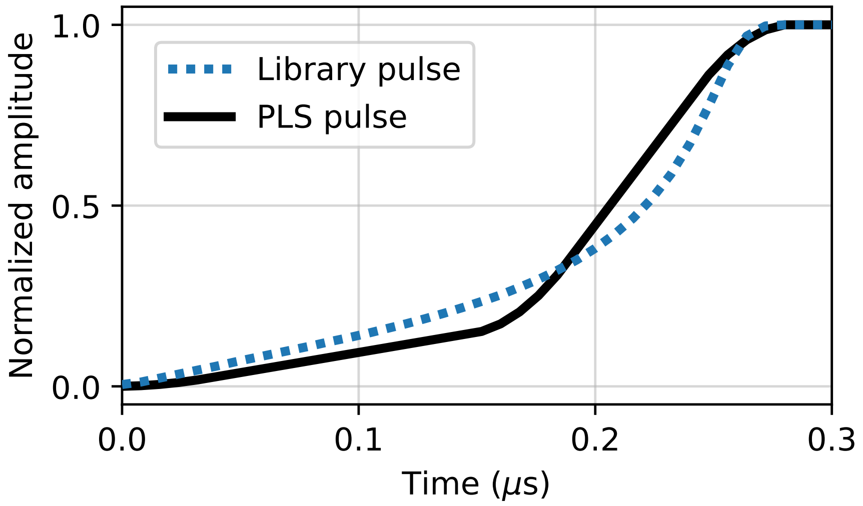


Figure 6.4: Example simulated PLS pulse. A library pulse (dotted line) is included for comparison to the PLS pulse (solid line). Both pulses have a rise time of 280 ns.

Simulated datasets summary The simulated detector datasets that are outlined in Section 6.2.4.2 are summarized in Table 6.4. The table contains the type of data and the total number of unique single-site events generated.

Table 6.4: PPC HPGe detector base simulated datasets. Includes the base class of simulated data (library or PLS) and the total number of unique simulated single-site events.

Data	**Number of simulated events**
Library	15 516
PLS	20 070

6.2.5 Data preprocessing

The baseline, defined as the mean over the first 1000 samples of a pulse, is calculated and subtracted from each signal. Pulse amplitudes are calculated by applying a trapezoidal filter to each trace [200], and the baseline-removed pulses are then scaled to an amplitude of one. Additionally, a pole-zero correction is applied to each pulse to remove the main component of the exponential decay from the resistive feedback preamplifier that was used to read out the detector. The entire preprocessing procedure is referred to as "amplitude normalization," noting that it includes the pole-zero correction for data pulses (the simulated pulses do not have an exponential decay). Because of the noise, the amplitude normalized pulses roughly, but not exactly, range in height from zero to one. Two example data pulses before and after preprocessing are shown in Figure 6.5.

Another preprocessing method explored in this thesis is standardizing each pulse to have a mean of zero and standard deviation of 0.5 after the pole-zero correction. A value of 0.5 is chosen to ensure that horizontally centred pulses have an amplitude of roughly one. This method of preprocessing is referred to as "standardization," noting again that it includes the pole-zero correction for data pulses. However, models trained with standardized pulses were found to perform similarly or even slightly worse in most circumstances. Additionally, these models were found to depend heavily on the absolute position in the pulse where the rise region begins. Due to the lack of robustness to horizontal shifts, all models described in the results chapter can be assumed to have been trained with amplitude normalized pulses unless mentioned otherwise.

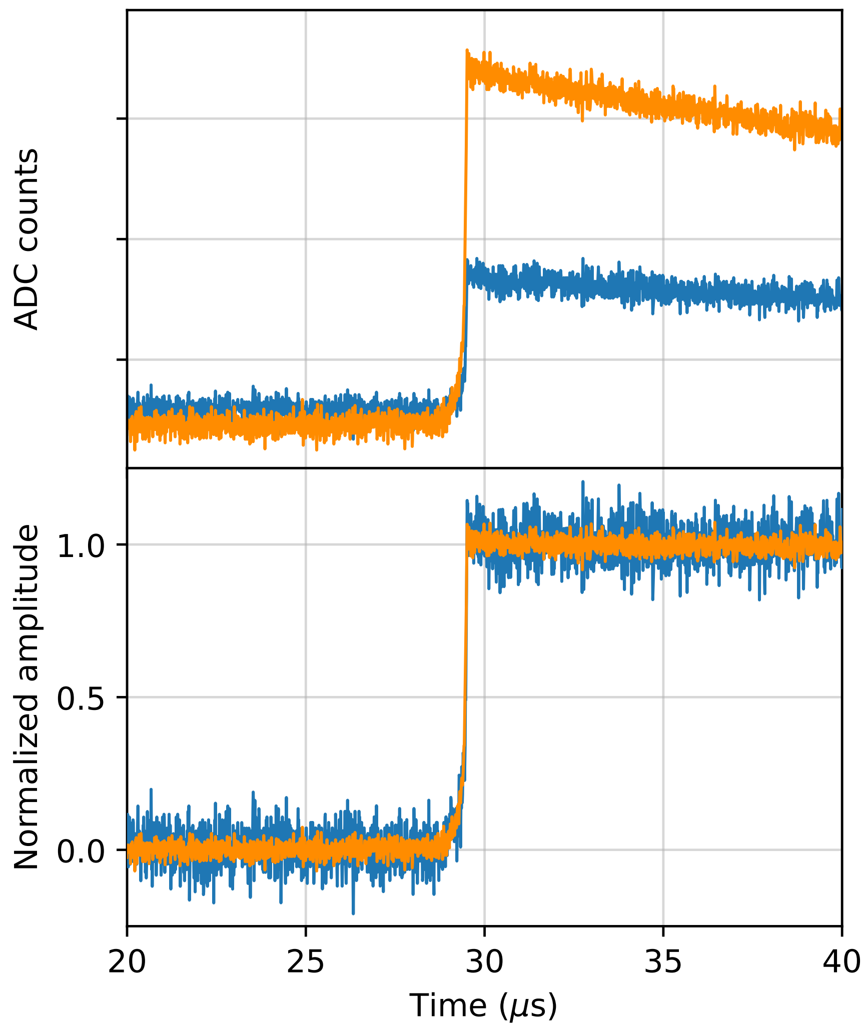


Figure 6.5: Example of two ^{241}Am data pulses before and after preprocessing. Amplitude normalization is used such that both pulses are of the same scale for the neural network. The different amplitudes, or equivalently energies, of the two pulses prior to any preprocessing (top) are reflected in the signal-to-noise ratio after preprocessing (bottom).

6.2.6 Data augmentation

A three step procedure is used to augment the simulated data and provide a large, diverse training set of noisy pulses with corresponding clean underlying pulses.

6.2.6.1 Multi-site event generation

The set of library pulses form a basis of position-dependent signals in the detector and thus contain only single-site events. Similarly, the PLS pulses are simple mathematical functions constrained by two points and are analogous to single-site events in shape. To augment the training data, artificial multi-site events are created by adding randomized combinations of simulated pulses together, without mixing between the library and PLS sets.

Events are generated with up to five sites. This upper bound is chosen because for a Poisson process with an expected rate of two, the probability of a number greater than five is less than 2 %. Furthermore, an equal number of events are generated for each number of sites, rather than being based on a physical distribution for the number of Compton scatters. The number of multi-site events is thus four times larger than the number of single-site events. This was found to be optimal for ensuring that the network does not smooth over multi-site events while also preserving the shape of single-site events.

In order to generate an n-site event, n pulses are drawn from a set of simulated pulses. A random amplitude is drawn from a uniform distribution, $\sim U(0, 1)$, for each pulse. Each pulse is also horizontally shifted by a value drawn from a discrete uniform distribution, $\sim U(0, 100)$, to account for the possible drift times from different points in the detector. The scaled and shifted pulses are then added together to create an artificial multi-site event, and the result is rescaled to have an amplitude of one.

6.2.6.2 Shifting and scaling

The process of amplitude normalization on detector signals is imperfect due to noise. Simulated pulses, in contrast, are noiseless and have perfect normalization. Preliminary results showed that the network would overfit the beginning and the end of the predicted pulses if the clean pulses started at exactly zero and ended at exactly one. To combat this issue, random shifting and scaling is applied to all simulated pulses during training. This ensures that the network sees a variety of imperfect normalizations and is able to better generalize to real detector data while avoiding overfitting. This type of artificial data augmentation can also help improve the overall performance of the network by effectively providing more data than is available. In order of application, the shifting and scaling procedure consists of:

1. **Amplitude scaling**: Each pulse is rescaled to have an amplitude drawn from the uniform distribution $\sim U(0.9, 1.1)$.
2. **Vertical shifts**: A random vertical shift, drawn from a uniform distribution $\sim U(-0.1, 0.1)$, is applied to each pulse.
3. **Horizontal shifts**: For each pulse of length 4096 samples, a number is drawn from a discrete uniform distribution $\sim U(1000, 3000)$, and the pulse is shifted such that the rise begins at this randomly chosen sample.

While vertical scaling and shifting of $\pm 10\,\%$ of the pulse amplitude is proportionally much larger than observed with typical data pulses, such a wide range of random variations further improves generalization of the autoencoder, particularly to high noise pulses and outliers such as pile-up events. As well, horizontal shifts only have a major effect near the edges of an event due to the receptive field of the convolution layers. Thus, the unrealistic range of values used in this procedure is primarily for data augmentation purposes.

6.2.6.3 Noise addition

Noise collected from the detector, described in Section 6.2.4.1, is added to the clean pulses after the previous data augmentation steps are applied. The detector noise dataset is sufficiently large such that each clean pulse has a unique, randomly chosen noise trace. Thus, noise seen in training is not seen in the validation and test phases, and noise pulses are not shared between datasets. Furthermore, the three subsets of noise collected at different periods in time are combined and treated as one when sampled from to ensure that the network is exposed to a variety of electronic noise from different detector operating conditions. All detector noise pulses are standardized to have a mean of zero and a standard deviation of σ.

In order to understand the effect of real detector noise, zero-mean normally distributed noise with no covariance over time, $\sim \mathcal{N}(0, \sigma)$, is also used, separately, to create independent datasets for comparison. For both noise types, σ is drawn randomly for each trace from a uniform distribution, $\sim U(0, 0.2)$, to simulate the effect of varying the signal-to-noise ratio.

6.2.7 Training procedure

Data files are converted from a low-level ROOT data structure to tabular formats using the NumPy [204] and HDF5 [171] file standards. As with Section 4.2.6, this is done for better compatibility with Python and its various numerical libraries which are heavily used throughout the analyses of this thesis. As well, since pulses within a set are all of fixed length, a tabular structure is much easier to work with. Each of the augmented simulated datasets are also randomly split into training, validation, and test subsets, and used according to the definitions in Section 3.5.1. The test set consists of 10 % of the overall simulated dataset. Of the remaining 90 %, 10 % of that are withheld for validation and the remainder are used for training.

Two training procedures are explored and tested in this section: a regular training procedure which requires a noisy input and clean output, and a procedure which does not require the clean version of the noisy pulse. Both procedures use the same network architecture developed in Section 6.2.3.3 – only the data and optimization procedure differ. As is done in Section 4.2.6, the networks are implemented using the Keras [173] API for TensorFlow [174]. Training and inference are conducted on NVIDIA *GeForce GTX Titan X*[1] and *GeForce RTX 3090*[2] GPUs.

6.2.7.1 Regular training

The regular training procedure consists of applying the augmentation recipe described in Section 6.2.6 to a set of simulated data d times, resulting in $N = d \times 5N_0$ pulses from an initial set of N_0 clean, single-site pulses. This is done separately with the library and PLS pulses (introduced in Section 6.2.4.2 and summarized in Table 6.4) to understand whether detailed physics simulations are required for training the algorithm. The augmentation procedure is also applied twice per base simulated set using

[1] https://www.nvidia.com/en-us/geforce/graphics-cards/geforce-gtx-titan-x/

[2] https://www.nvidia.com/en-us/geforce/graphics-cards/30-series/rtx-3090-3090ti/

Gaussian noise and real noise from the detector, for a total of four distinct datasets. A summary of these sets is shown in Table 6.5. To ensure that the sizes of each training set are comparable and that any comparisons are fair, $d = 45$ for the library pulses and $d = 35$ for the PLS pulses. This results in $\sim 3.5 \cdot 10^6$ unique pulses for each class of simulation. The exact number of events in each set after augmentation is also included in Table 6.5. Note that this number is of the overall datasets prior to the split into training, validation, and test subsets. As well, during the augmentation procedure, a copy of the pulses after the shifts and scales, but before the noise addition, is created and used for the clean targets.

Table 6.5: PPC HPGe detector augmented simulated datasets. Includes the base class of simulated data (library or PLS), the type of noise used in the augmentation procedure (detector or Gaussian), and the total number of simulated events after augmentation.

Data	**Noise**	**Number of simulated events**
Library	Detector	3 491 100
Library	Gaussian	3 491 100
PLS	Detector	3 512 250
PLS	Gaussian	3 512 250

The network is then trained to map the noisy pulses to the corresponding clean pulses. The Adam optimizer [172] with a learning rate of $3 \cdot 10^{-4}$ is used to minimize the mean squared error between the true clean pulse and the denoised pulse. The mean squared error computation is unweighted, meaning no preference is given to any specific region of the pulses. A batch size of 128 is used to balance the speed of training, the goodness of the approximation of the true gradient, and GPU memory limits.

Training is conducted over 100 epochs. At the end of every epoch of training, the network is run on the validation set and various metrics – including the loss function – are written to a file. The model at the epoch with the lowest validation loss is also saved. The validation set is used to select the best hyperparameters, including the batch size, learning rate, and architecture details, as is done in Section 4.2.6. 100 epochs of training was found to be sufficient for convergence of the network.

6.2.7.2 Noise2Noise training

The regular training procedure can only work with data where a true underlying pulse is known, and thus is limited to simulations. Here, an alternative approach for training with only noisy detector data is developed. It is based off of the Noise2Noise procedure – an approach used to train a denoising model without the need for clean target examples [205]. The central idea of the Noise2Noise method is that the mean of the target distribution minimizes the sum of squared errors, or L_2 loss[3], between the target and the prediction. In the simple case of point estimation, the L_2 loss for a set of measurements $\{x_i\}$ with $1 \leq i \leq N$ and prediction z is given by

$$L_2 = \sum_i \|z - x_i\|_2^2. \tag{6.7}$$

The best estimate z that minimizes the L_2 loss is the mean of the measurements, $z = \frac{1}{N}\sum_i x_i$. One can observe that z remains unchanged as long as the mean is unchanged. Given infinite data, the addition of zero-mean noise to the measurements would thus produce the same estimate. This logic can be applied to the denoising

[3]The mean squared error and sum of squared errors, or L_2 loss, are equivalent for the purposes of optimization and thus may be used interchangeably in that context.

task of minimizing the L_2 loss between the clean signals, x_i, and the denoised outputs from applying the autoencoder to the corrupted versions of the clean signals, $\tilde{x}_i$. Using similar notation to that of Section 3.4 and denoting the autoencoder output as $z_i = g_{\theta'}(f_\theta(\tilde{x}_i))$, the L_2 loss can be written as

$$\begin{aligned} L_2 &= \sum_i \|z_i - x_i\|_2^2 && (6.8) \\ &= \sum_i \|g_{\theta'}(f_\theta(\tilde{x}_i)) - x_i\|_2^2. && (6.9) \end{aligned}$$

If instead the target x_i is replaced with corrupted versions of itself, $\hat{x}_i$ (noting that $\hat{x}_i$ is different from $\tilde{x}_i$), drawn from a distribution or distributions with a mean of x_i, then, the optimal estimate z_i remains unchanged. Put another way, the corruption of x_i with zero-mean noise will produce the same estimate, given enough data. The loss function becomes

$$\begin{aligned} L_2 &= \sum_i \|z_i - \hat{x}_i\|_2^2 && (6.10) \\ &= \sum_i \|g_{\theta'}(f_\theta(\tilde{x}_i)) - \hat{x}_i\|_2^2. && (6.11) \end{aligned}$$

Again, it should be emphasized that $\hat{x}_i$ and $\tilde{x}_i$ are different noisy realizations of the same underlying signal. By minimizing Equation (6.11), the autoencoder should learn to predict the mean of the distribution of noisy pulses. Intuitively, the task of mapping one version of a noisy pulse to another is impossible if the model is not over-parameterized, and the best it can do is predict the mean. With finite data, the above results are only approximately true, though in practice the amount of data required for this approach to work well is reasonable, as will become clear in Section 7.1.1.

Unlike the regular training procedure, the Noise2Noise approach can be applied to *both* simulated and raw detector data. The procedure follows the regular training procedure with the amendment of adding noise to the target pulse as well. For real detector data, where no clean underlying pulse exists, noise is added to the already noisy signals. Of course, the optimal solution to the minimization problem is then the underlying noisy pulse rather than the true underlying, but unknown, clean pulse. To alleviate this issue, a simple penalty on the total variation of the denoised signal can be added to the loss function L,

$$L = L_0 + \frac{\lambda}{N} \sum_{i}^{N} \sum_{j}^{M-1} |z_{i,j+1} - z_{i,j}|, \tag{6.12}$$

where L_0 is the original loss function (here, L_0 is simply the mean squared error, or the L_2 loss scaled by the number of training samples), $z_{i,j}$ is the j^{th} sample of the i^{th} denoised output pulse, M is the number of samples in the pulse, N is the size of the training set, and λ is a scaling factor. Minimization of the total variation, with some criterion of similarity to the original signal, was introduced as a method to denoise signals in 1992 [206]. Total variation is particularly useful in retaining the important components of a signal, including sharp edges and discontinuities, while avoiding the overcompensated smoothing present in many traditional denoising techniques [207]. Since the true pulses are monotonically increasing functions, the regularizer in Equation (6.12) will evaluate to the amplitude in the case of perfect denoising. If the denoised pulse still contains components of the noise and is very jagged, the penalty will become large.

The Noise2Noise training procedure is applied to the simulated and augmented datasets described in Section 6.2.7.1 and Table 6.5, as well as the ^{60}Co dataset introduced in Section 6.2.4.1. For the simulated datasets, a different noise pulse is also added to the target pulse, though the underlying clean signals and corresponding noisy inputs remain exactly the same as in Section 6.2.7.1. In the augmentation procedure for ^{60}Co detector data, the artificial multi-site event generation step is skipped as the ^{60}Co dataset already contains a large proportion of multi-site events. As well, pulses are not horizontally shifted because the network is already equivariant to such shifts and because it is difficult to handle the bounds due to the noise. However, the remainder of the procedure is still applied because the ^{60}Co dataset consists of primarily high energy events, meaning that the shifts, scales, and noise are unrepresentative of lower energy pulses and outliers. For the noise addition, σ is instead drawn from the uniform distribution $\sim U(0.025, 0.25)$. These numbers are chosen so that at least some noise is added to the data pulses, and to minimize correlated noise between the inputs and the targets.

The network is then trained to map one version of a noisy pulse to another. The same general optimization and validation process as for the regular training procedure is used, although the learning rate is instead set to 10^{-5} as the learning rate in the regular training procedure was found to be too large. As well, the validation set of the augmented library pulses with detector noise is used to select the best model as the data pulses do not have a target pulse. The selected model is the one which minimizes the mean squared error, not the total loss, as the total variation penalty tends to dominate the validation loss.

Chapter 7

Deep Learning-Based Pulse Denoising: Results and Analysis

This chapter presents the results of applying the convolutional autoencoder developed in Chapter 6 to denoise pulses. It closely mirrors the structure of the author's publication in [1], with Section 7.1 focusing on evaluation with simulated data and Section 7.2 focusing on evaluation with real detector data. Section 7.1 investigates how well the autoencoder preserves the underlying pulse shape using different training procedures and shows comparisons to various traditional denoising methods. This section also presents expected energy resolution improvements with denoising. Additionally, Section 7.1.4 contains new results of applying this model to single-site/multi-site event discrimination not contained in [1]. Section 7.2 focuses on applying the autoencoder to data, using a statistical comparison to evaluate pulse shape preservation and determining how the energy resolution can be reduced with denoising in the context of expected improvements as from simulations. Section 7.3 concludes the chapter with discussions on the experimental implications of these results, as well as ongoing and planned work stemming from the methods developed in Part II.

7.1 Pulse denoising evaluation on simulated data

The data used for evaluation in this section are from the procedure described in Section 6.2.7, specifically the test subsets of the simulated and augmented data outlined in Table 6.5. Similar additional sets of noisy pulses – completely unseen in the training – are also generated for evaluation in Sections 7.1.3 and 7.1.4 for the specific analyses of those sections and described in more detail there.

7.1.1 Comparison between training procedures

The performance of the denoising convolutional autoencoder trained on several different datasets for the two training procedures outlined in Section 6.2.7 is first evaluated. Each model is applied to the test sets of the four available distinct simulated datasets (Table 6.5) and the mean squared error between the original and corresponding denoised pulses is calculated. The results are shown in Table 7.1. The first three columns are related to the training method and show the procedure, the dataset, and the type of additional noise used in training, respectively. The remaining columns show the mean squared error evaluated by denoising the simulated test datasets, containing either Gaussian or real detector noise on single- and multi-site events generated using either the library or the PLS pulses, for which the clean target is known. All pulses in the test sets are amplitude normalized and horizontally centred.

Each model trained on simulated data tends to perform best on the same class of pulses that it was trained on. Models trained with Gaussian noise tend to generalize slightly worse to detector noise than the other way around. As well, models trained using library and PLS simulated pulses have similar performance. However, models trained using library pulses tend to generalize better to PLS pulses than the other

Table 7.1: Mean squared error comparison of different training procedures. The procedure, dataset, and noise type used in the training are given. All results shown are evaluated on the test sets.

Training procedure and data			**Mean squared error** (10^{-5})			
			Gaussian		**Detector**	
Procedure	**Data**	**Noise**	**Library**	**PLS**	**Library**	**PLS**
Regular	Library	Detector	4.12	4.72	3.76	4.21
Regular	Library	Gaussian	3.40	3.82	4.50	4.77
Regular	PLS	Detector	5.10	4.48	4.15	3.57
Regular	PLS	Gaussian	3.93	3.36	5.02	4.31
N2N ($\lambda = 0$)	Library	Detector	3.90	4.37	3.86	4.20
N2N ($\lambda = 0$)	Library	Gaussian	3.46	3.87	4.57	4.82
N2N ($\lambda = 0$)	PLS	Detector	5.11	4.48	4.14	3.55
N2N ($\lambda = 0$)	PLS	Gaussian	3.85	3.46	4.97	4.43
N2N ($\lambda = 0$)	^{60}Co	Detector	6.54	6.30	7.78	7.40
N2N ($\lambda = 10^{-2}$)	^{60}Co	Detector	4.17	4.54	5.04	5.26

way around. This is evidenced by the difference between the performance on the test set of library and PLS pulses for a given noise type; the gap is larger for the model trained on PLS pulses.

The Noise2Noise models trained on simulated pulses perform nearly identically on the test set as the corresponding regular models in most cases. Differences in the mean squared error are typically on the order of 10^{-7} to 10^{-6}, which is expected due to statistical fluctuations in training. No total variation penalty was used as it was not found to improve the performance in the case of simulations. On simulated pulses with detector noise, neither Noise2Noise model trained on pulses from the ^{60}Co source outperforms any of the models trained with simulated data, regardless of the procedure. However, the Noise2Noise model is still effective at denoising and requires only noisy detector data. A larger set of detector data, including from other high-energy sources which produce multi-site events, could improve the performance of this model.

Furthermore, the performance of the Noise2Noise model with and without a total variation penalty is shown in Table 7.1. Unlike the models trained on simulated data, the total variation penalty has a non-negligible impact on the denoising performance in the case of real detector data, as expected. A penalty of $\lambda = 10^{-2}$ was found to have the best mean squared error on the validation set, although using any penalty in the range 10^{-4} to 10^{-1} produced similar results. In particular, the mean squared error is approximately 30 % lower by setting λ in this range. The main conclusion is that a total variation penalty is necessary to obtain optimal performance with the Noise2Noise method.

Figure 7.1 shows an example library pulse multi-site event that has been denoised by two different versions of the autoencoder. The top panel shows the pulse denoised with the regular library pulse model while the bottom panel shows the denoised pulse using the Noise2Noise model trained with ^{60}Co data with a total variation penalty.

Qualitatively, the regular library pulse model appears to fit the true underlying pulse the best, in line with the results in Table 7.1. However, the differences are subtle, and the Noise2Noise model appears to remove most of the noise without much additional distortion. This suggests that it may still work even in applications that require the pulse shape to be preserved to a high degree. Furthermore, although not shown here, visualization of the denoised pulses illustrates the impact of including a total variation penalty for the Noise2Noise method, as it removes much of the noise

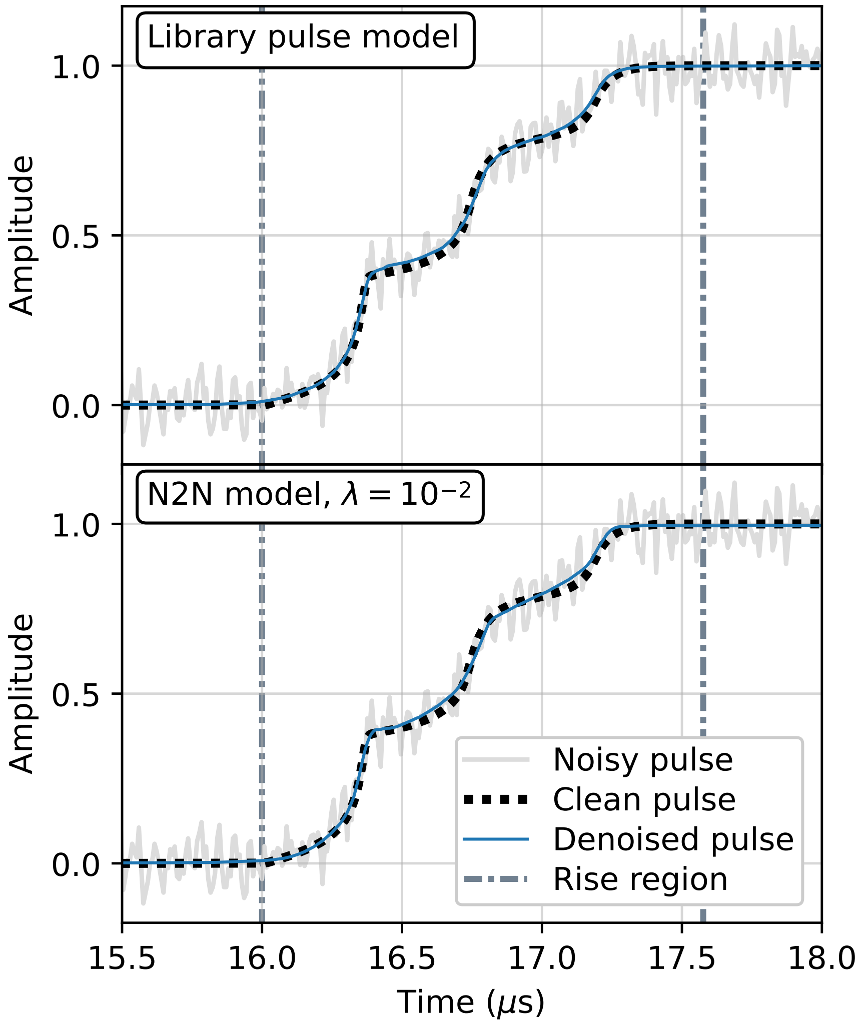


Figure 7.1: Example denoised multi-site event from the library pulse dataset. Included in each plot is the simulated pulse with artificial noise (solid light line), the clean underlying pulse (dotted line), and the corresponding denoised pulse (solid dark line) from the regular library pulse model (top) and Noise2Noise model (bottom). Figure from [1].

still present in the pulse denoised without it.

Overall, while all methods perform well, only the regular training procedure with library pulses required careful simulations of the detector. This demonstrates the power of the model at removing noise and its relative independence on the exact training procedure.

7.1.2 Comparison to traditional denoising methods

The denoising performance of the autoencoder is compared to the traditional noise removal methods described in Section 6.1. The mean squared error between the clean target and denoised output was used as the metric and evaluated on the simulated library pulse test set. The performance was evaluated over the entire pulse as well as over two distinct sections of the pulse: the rise region and flat region. The rise region begins where the simulated pulse deviates from the flat baseline and ends where the pulse reaches its maximum amplitude. The flat region is defined as the area outside of the rise region. The example pulse in Figure 7.1 includes an illustration of the region boundaries.

The mean squared error comparison between the traditional methods in Section 6.1 is shown in Figure 7.2. The performance of two autoencoder models trained with the regular procedure (one using the library set and one using the PLS set) and one model trained with the Noise2Noise procedure (using the ^{60}Co data) are included. Figure 7.2 contains two variations of each traditional method: one using the optimal parameter(s) for the rise region and one using the optimal parameter(s) for the entire pulse, both optimized on the validation set by minimizing the mean squared error. For wavelet-based methods, VisuShrink is used to determine a global threshold and soft thresholding is used to shrink the coefficients towards zero. BayesShrink was found to be too complicated for the data and ineffective, prompting the use of the simpler VisuShrink threshold method.

In addition to the four methods in Figure 7.2, a Kalman filter was implemented

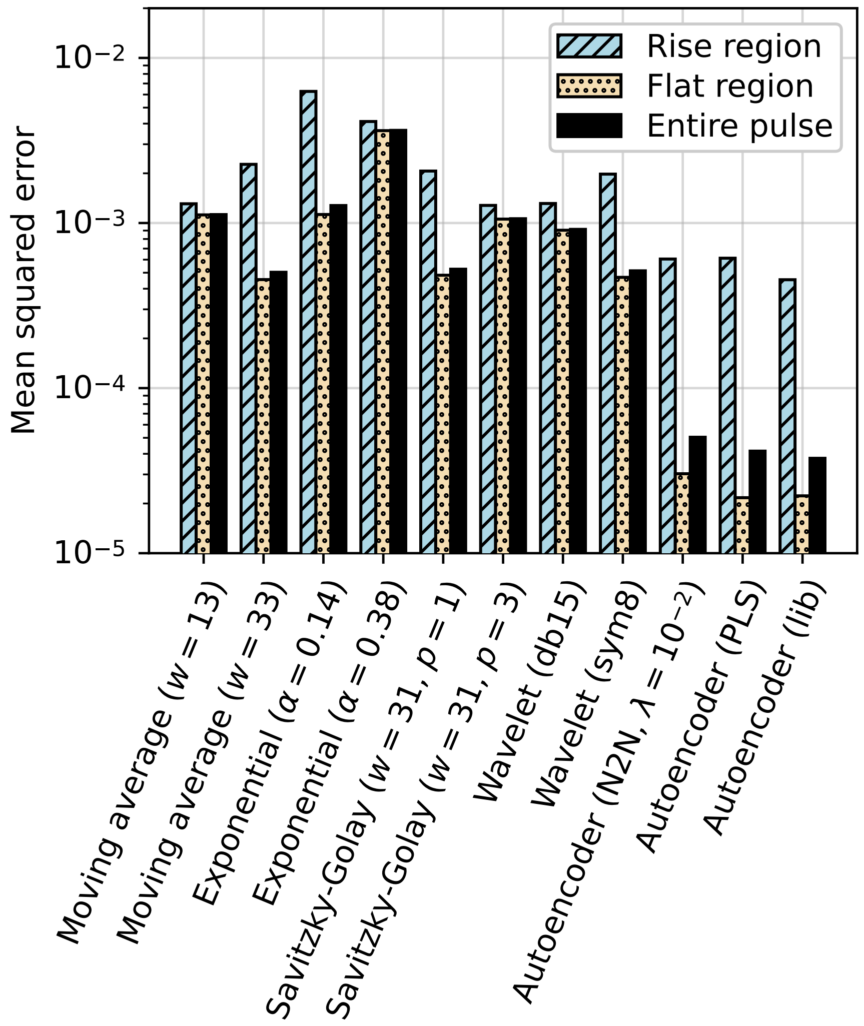


Figure 7.2: Mean squared error comparison of different noise removal methods. Results are presented for both inside and outside the rise region of the pulse (defined in the text and shown in Figure 7.1). The mean squared error inside the rise region is indicated by the slanted line hatch while outside the rise region is indicated by the dotted hatch. Solid fill corresponds to the mean squared error over the entire pulse. Figure from [1].

using the denoised output from the autoencoder as the underlying model and the baseline RMS of the noisy pulse as the measurement error. The filter was optimized by selecting the level of extrapolation between the noisy data and the underlying model that resulted in the least error. This process resulted in an optimized model with zero extrapolation, representing a copy of the autoencoder model, and so the Kalman filter is omitted from Figure 7.2.

The autoencoder outperforms all traditional methods in both the rise region and flat region of the pulses. While the method requires training to denoise a specific type of data, it does offer improvements over traditional denoising, as evaluated using the mean squared error. The structural similarity index measure (SSIM) [208] was also used to compare the performance of the traditional denoising methods as it is designed to quantify image degradation with reference to human perception. However, the relative performance of each method using SSIM is nearly identical to that using the mean squared error, and so the corresponding SSIM comparison figure is not shown here.

7.1.3 Energy resolution comparison

In this section, the energy resolution before and after denoising using the amplitude normalized library pulse model is calculated on simulated data. This serves as a baseline for a similar energy resolution analysis conducted on data in Section 7.2.3. For a given pulse, the energy is calculated from the amplitude of a trapezoidal filter with a given gap and shaping time, the latter of which is varied as described below. The energy resolution is then defined as the FWHM of the energy peak.

The set of library pulses and real detector noise, distinct from the training set, is used to create new sets of 172 400 noisy single-site event pulses for evaluation. A dataset is created for noise levels ranging from 0.02 to 0.2 in increments of 0.02. This is done so that the dependence of the energy resolution on the noise can be better quantified, which is more difficult to do with the data used in Sections 7.1.1 and 7.1.2. The energy resolution as a function of the trapezoidal filter shaping time on one such simulated dataset with a noise level of 0.1 is shown in Figure 7.3 with and without denoising. This noise level is roughly the same as we observe with ^{241}Am gamma rays. For all datasets, the gap time in the trapezoidal filter is fixed at 1.8 µs, which was found to be sufficiently large for even the slowest rise times. Clean library pulses all have an amplitude of unity before noise addition, and so the "true energy" is one, meaning that the noise level and the resolution in Figure 7.3 can both be interpreted as a fraction of the amplitude.

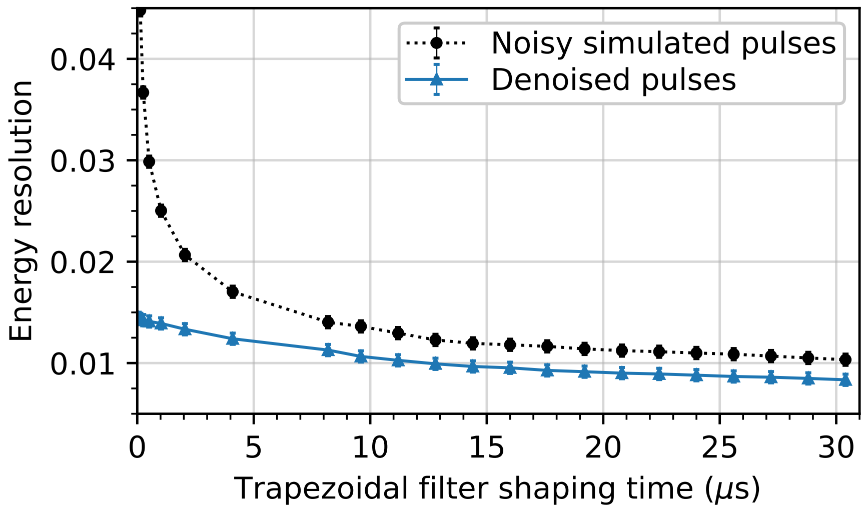


Figure 7.3: Energy resolution as a function of trapezoidal filter shaping time on simulated pulses. Calculated on library pulses with real detector noise with a baseline RMS of 0.1 before denoising (dotted line, circle markers) and after denoising with the amplitude normalized model (solid line, triangle markers). Figure from [1].

Figure 7.4 shows the relative improvement in the energy resolution after denoising as a function of the noise level. Each curve corresponds to a given shaping time, and each point on a given curve corresponds to a point from a plot such as Figure 7.3.

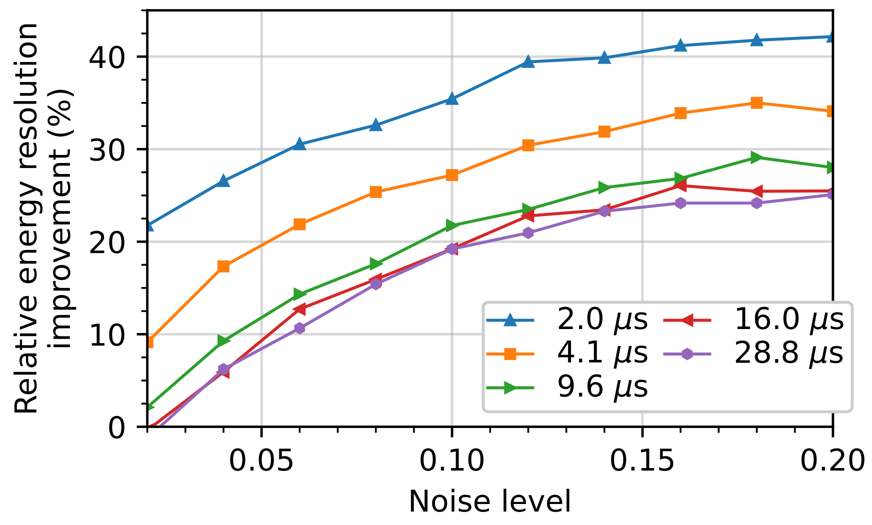


Figure 7.4: Relative improvement in the energy resolution from denoising as a function of noise level on simulated pulses. Each curve corresponds to a single trapezoidal filter shaping time. Figure from [1].

At every shaping time, the energy resolution on simulated data is lower with denoising. This is particularly prominent at shaping times less than 5 µs, where the improvement in energy resolution exceeds 20 % for all but the lowest noise levels. At higher shaping times, the averaging window of the trapezoidal filter is large enough to smooth out the noise, and so the improvement is generally smaller. Overall, the most substantial reduction in energy resolution is obtained when the pulses have a high level of noise and when the shaping time used to calculate the energy is small, which may be relevant in high rate applications such as when HPGe detectors are used in a beam facility. Still, comparable performance or improvements in the energy resolution at every noise level and shaping time evaluated are observed.

7.1.4 Single-site/multi-site event discrimination comparison

To further confirm the preservation of the pulse shape, a single-site/multi-site discrimination analysis is performed before and after applying the amplitude normalized library pulse model. Similarly to what is done in Section 7.1.3, two new sets of library pulses using real detector noise (again, distinct from the training set) are generated. Each consists of 10 000 single-site events and 10 000 multi-site events. One uses a baseline RMS of 0.1 (approximately equivalent to the 60 keV gammas from the ^{241}Am data), and the other a baseline RMS of 0.005 (approximately equivalent to the 1173 keV gammas from the ^{60}Co data). The multi-site events are generated in a similar way described in Section 6.2.6, except that the number of sites is fixed to two as that is the most difficult to discriminate from single-site events.

The standard method of single-site/multi-site discrimination is to use the amplitude over energy (A/E) ratio or variants thereof [123,209,210]. For single-site events, the maximum amplitude of the current pulse, A, relative to the energy, E, will be large. Multi-site events are more likely to have a spread out current pulse with multiple shorter peaks for the same energy due to the different arrival times of charge carriers, resulting in an A/E value that is lower and an A/E distribution that is more spread out.

The current pulse can be calculated by taking the derivative of the charge pulse. Simply taking the first difference is highly prone to noise, and so a Savitzky-Golay filter is instead used. At each point, instead of evaluating the fitted polynomial, the analytical derivative is evaluated to obtain the current pulse. The Savitzky-Golay filter produces a much smoother current pulse and a better estimate of its maximum amplitude. The Savitzky-Golay parameters were optimized on a separate validation set by calculating receiver operating characteristic (ROC) curves for values of p and w in a grid with $p \in \{1, 3, 5\}$ and $w \in \{p+2, p+4, \ldots, 63\}$. The set of parameters $p = 5$ and $w = 29$ maximized the area under the curve (AUC, in reference to the ROC curve) and are used in the results on the test set below.

Figure 7.5 shows the ROC curve for the 0.005 baseline RMS dataset, while Figure 7.6 shows the ROC curve for the 0.10 baseline RMS dataset. In both figures, the positive class is taken to be single-site, though this is arbitrary and can easily be swapped.

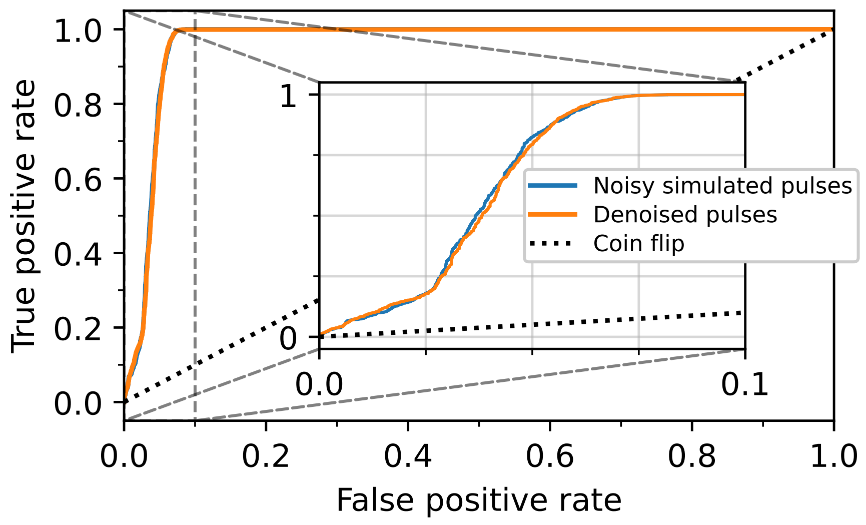


Figure 7.5: Single-site/multi-site event discrimination ROC curves before and after denoising for low noise (high energy) pulses. The positive class is defined as single-site in this figure. The ROC curve calculated before denoising is shown in blue, while the ROC curve calculated after denoising with the amplitude normalized library pulse model is shown in orange. Since the difference between the curves is difficult to see at this noise level, the inset zooms in on the most important region of the ROC curve.

For a high noise level (low energy), denoising improves single-site/multi-site event discrimination, as seen in Figure 7.6. While multi-site events are not as likely to occur at this energy range in reality, it serves as a demonstration of the autoencoder's

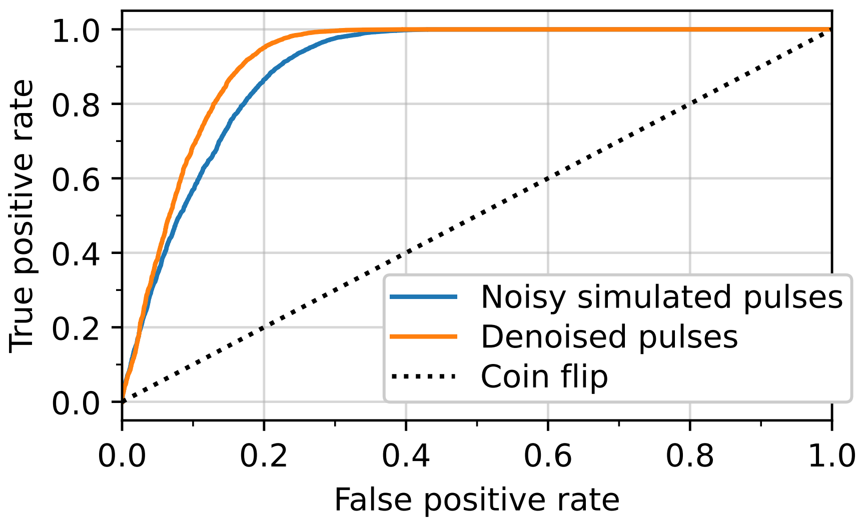


Figure 7.6: Single-site/multi-site event discrimination ROC curves before and after denoising for high noise (low energy) pulses. The positive class is defined as single-site in this figure. The ROC curve calculated before denoising is shown in blue, while the ROC curve calculated after denoising with the amplitude normalized library pulse model is shown in orange.

capabilities. Events that are erroneously determined to be single-site when using the original noisy pulses are more often correctly classified as multi-site when denoised, given a fixed true positive rate. This implies that the denoising does not distort the pulses. For a low noise level (high energy), denoising does not offer much of an improvement in the single-site/multi-site classification task, as shown in Figure 7.5. However, it also does not decrease the A/E discrimination power. This is shown more clearly in the inset of Figure 7.5. Intuitively, these results make sense – denoising a pulse with already low noise should not provide as much of a benefit in any performance metric, but should also not make it worse. This analysis also provides additional confirmation that the pulse shape is not distorted by denoising.

7.2 Pulse denoising evaluation on detector data

In this section, data collected with the ^{241}Am source (described in Section 6.2.4.1) are used to evaluate the denoising autoencoder. Gammas from the source interact in the detector and provide mono-energetic pulses with a reasonable amount of noise. Events are selected within ± 2 keV of the 60 keV peak associated with the source gamma rays, as illustrated in Figure 6.2, to obtain a pure subset of data. Any events outside of the peak are taken to be backgrounds and are not included in the evaluations that follow. The number of events in this reduced subset is 73 600, about 25 % of the data collected with the ^{241}Am source in use. Pulses with outliers in the slopes on either the baseline or end of the trace are also removed after the energy cut, reducing the subset further to 72 707 signal events.

Furthermore, the 60 keV events from the source are essentially all single-site events for which the underlying true shape is close to the ones simulated in the library dataset. This allows for a reasonable guess of the clean target to be inferred by selecting the library pulse that minimizes the χ^2 value defined in Equation (6.6). Position reconstruction using this approach is studied in [203]. When evaluating the performance on real data, one does not have a true clean signal with which to compare, as was the case with simulated pulses. The best fit library pulse can be used as a baseline for the comparisons in this section. The basis set used here was generated with a preamplifier time constant of 20 ns as that was found to best match the detector.

7.2.1 Qualitative evaluation

Figure 7.7 shows a single-site event from the ^{241}Am source data that has been denoised by two versions of the autoencoder. The top panel shows the pulse denoised with the regular library pulse model while the bottom panel shows the denoised pulse using the Noise2Noise model trained with ^{60}Co data with a total variation penalty. Each plot also includes the library pulse with the lowest χ^2 value as a best estimate of the true underlying pulse.

Qualitatively, the performance of all models on the single-site event is promising. Denoised versions of the pulse tend to better capture the shape of the signal than the corresponding best fit library pulse, which is restricted to the size of the grid used in simulating the basis set. Furthermore, while not shown here, the denoising performance of all models on the multi-site events present in the set is also very good. As was observed with the results on simulated data, the library pulse model tends to perform the best. For the Noise2Noise models, the total variation penalty is important as the model trained without it retains some of the noise.

7.2.2 Chi-squared comparison

Though meaningful event-by-event evaluations cannot be made on data pulses, statistical comparisons are possible. The χ^2 value between the data pulse and denoised pulse for events in the ^{241}Am dataset can be used to determine statistical compatibility. Although the true shape of the data pulses without noise is unknown, the χ^2 distribution can be used to determine if, statistically, the denoised pulses are consistent with their noisy progenitors. It is important to use the χ^2 instead of the mean or sum of squared errors to account for the signal-to-noise ratio of the pulses.

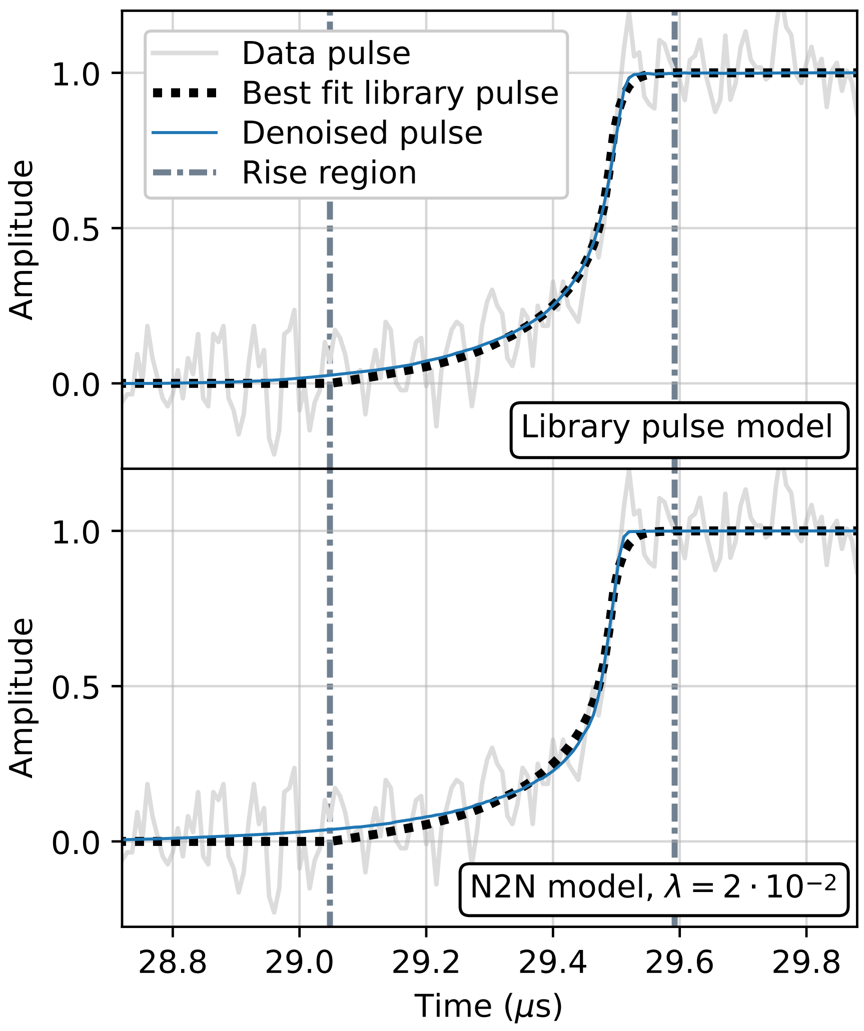


Figure 7.7: Example denoised single-site event from the ^{241}Am dataset. Included in each plot is the noisy data pulse (solid light line), the best fit library pulse (dotted line), and the corresponding denoised pulse (solid dark line) from the regular library pulse model (top) and Noise2Noise model (bottom). Figure from [1].

Figure 7.8 shows the distribution of χ^2 values between the data pulses and corresponding denoised pulses using the regular library pulse autoencoder model. It also includes the distribution of χ^2 values between the data pulses and corresponding best matching library pulses, as determined by the χ^2 minimization across all library pulses in the basis set.

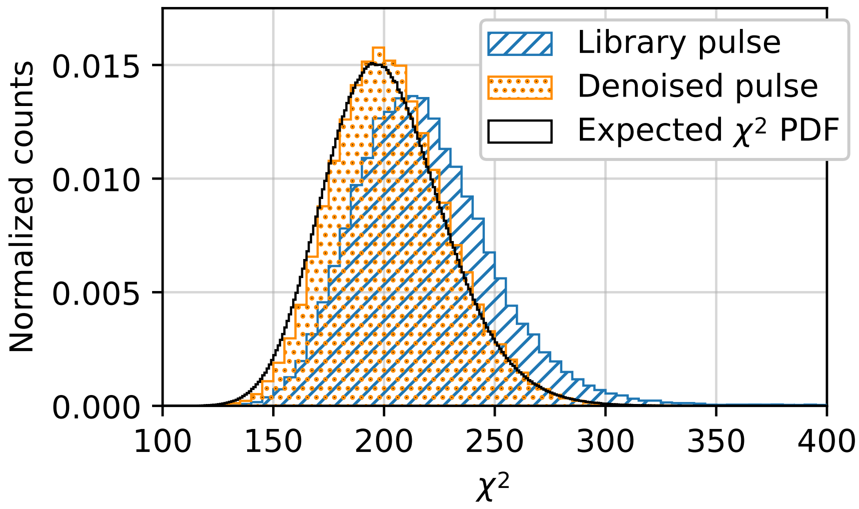


Figure 7.8: χ^2 distributions of pulses from the ^{241}Am dataset. χ^2 values are computed between the data and both the corresponding denoised pulse (orange, dotted hatch) and best fit library pulse (blue, slanted line hatch) over 200 samples containing the rise region. Figure from [1].

The χ^2 value for each pulse is computed from sample 3600 to 3800, meaning that the number of degrees of freedom (NDF) is 200. This range is chosen because the pulses are horizontally centred within this window and it is long enough to capture the important components of the pulse for the largest rise times. A modified χ^2 distribution for NDF $= 200$ expected from the detector noise is also overlayed on Figure 7.8 for comparison. This distribution is calculated directly on the detector noise described in Section 6.2.4.1, independent of any signal. Note that since the detector noise is not Gaussian, the modified χ^2 distribution is shorter and wider than the true χ^2 PDF for NDF $= 200$ which assumes normally distributed noise.

The results show that the denoised pulses fit the data better than the best matching library pulses, as indicated by the lower mean and the better fit of the distribution to the χ^2 distribution expected from the detector noise. This could indicate that there are not enough library pulses in the basis set, or that some other parameter of the detector is being modelled improperly. Additionally, while not shown here, the fit is better on *all* events, not just those in the energy range within the 60 keV peak, indicating that the model developed here is properly denoising multi-site events and low-energy events for which the pulse fitter fails to converge.

7.2.3 Energy resolution comparison

In this section, the performance of the autoencoder is evaluated by comparing the energy resolution of the 60 keV peak before and after denoising, similar to what is done in Section 7.1.3 on simulated data. The results are shown in Figure 7.9, which illustrates the energy resolution on both the original noisy and denoised pulses as a function of the trapezoidal filter shaping time. The gap time in the trapezoidal filter is again fixed at 1.8 µs. The results for two denoising models (both trained with the regular procedure and library pulses) are included: one using amplitude normalized pulses and the other using standardized pulses in the training.

While denoising does not achieve a lower energy resolution overall, it does provide a comparable energy resolution with a lower shaping time. This is especially true for the model trained with standardized pulses, where the optimal energy resolution is achieved even at shaping times under 2 µs. The discrepancy between the two denoised models at lower shaping times is surprising as the only difference is the preprocessing of the pulses. However, all data pulses are horizontally centred, meaning that the lack of robustness to horizontal shifts for standardized pulse models is not an issue here. In general, models trained with standardized pulses performed similarly or slightly worse in terms of both mean squared error (on simulations) and χ^2 (on data), and

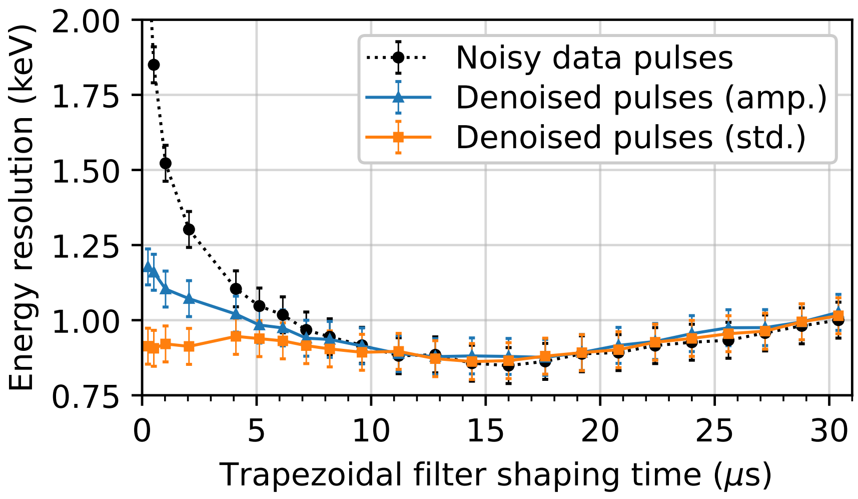


Figure 7.9: Energy resolution as a function of trapezoidal filter shaping time on ^{241}Am data pulses. Calculated on data before denoising (dotted line, circle markers) and after denoising with the amplitude normalized model (solid line, triangle markers) and standardized model (solid line, square markers). Figure from [1].

the energy resolution is the only metric where a significant performance deviation is observed.

These results are important, as the optimal shaping time required to obtain a reasonable energy resolution directly affects data collection. Specifically, a longer shaping time requires longer traces, which in turn occupies more disk space and increases the chances of event pile-up. A lower shaping time thus has many practical implications with regards to easier/more efficient data storage and analysis. This is also of interest to higher rate applications of HPGe detectors, rather than just in rare-event searches.

As shown in Section 7.1.3, an overall improvement in the energy resolution is expected from simulations under ideal conditions. However, this is not the case for real ^{241}Am data. As well, the shape of the energy resolution curve is different;

specifically, the curve in Figure 7.3 decreases as a function of the shaping time up until the largest shaping time, which is limited by the length of the noise traces. This is in contrast to Figure 7.9, where there is a clear minimum at around 15 µs. A prominent imperfection with data is that there are multiple sources of exponential decay, while the pole-zero correction only accounts for one “effective” source. This results in visually imperceptible changes to the ends of the pulses that broaden the energy resolution, particularly at high shaping times. As the models are not trained to remove this effect, it is still present to some extent even after denoising. This was confirmed by adding multiple exponential decays and then applying only one correction to a simulated test set. However, as the properties of the exponential decays present in the detector setup are unknown, this effect is not accounted for in the training data, nor is it removed from the real data.

7.2.4 Frequency spectra comparison

The frequency power spectrum of denoised pulses is analyzed for a subrun of the ^{241}Am dataset described in Section 6.2.4.1 where the source was fixed at roughly half the height of the detector. The data are denoised with some of the traditional denoising methods evaluated in Section 7.1.2, as well as with some of the autoencoder models trained with differing procedures. The discrete Fourier transform (DFT) is computed via the FFT algorithm on these denoised data in addition to the corresponding original noisy pulses in the subrun. In computing the DFT, a Hanning window [211] is used in order to remove discontinuities resulting from the step-like shape of the pulses and to lessen spectral leakage.

The resulting frequency spectra, obtained by summing the individual spectrum of each pulse for a given set of data, are shown in Figure 7.10 for the moving average, Savitzky-Golay, and wavelet denoising methods (all using the optimized parameters as in Figure 7.2). Figure 7.11 shows the frequency spectra comparisons for regular autoencoder models trained with library and PLS simulated data and the Noise2Noise model trained with data from the ^{60}Co source. It also includes the regular library pulse autoencoder trained with standardized pulses. In each figure, the frequency spectra for the noisy data and a clean simulated pulse at the position in the detector corresponding to the location of the ^{241}Am source are also shown for comparison.

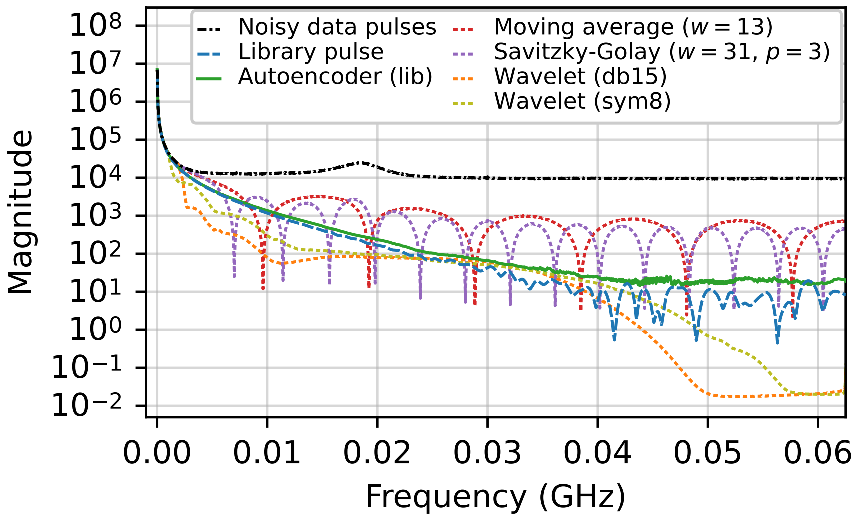


Figure 7.10: Frequency spectra comparisons for traditional denoising methods. Comparison is made between a simulated library pulse, noisy data pulses, noisy data pulses denoised with the autoencoder trained using the regular procedure and library pulses, and noisy data pulses denoised with four different traditional denoising methods. Figure from [1].

Of the methods evaluated, the pulses denoised by the autoencoders have the closest frequency spectra to that of the clean simulated pulse. In Figure 7.10, both the moving average and Savitzky-Golay method spectra have periodic artifacts from their windows. Both wavelet denoising method spectra tend to be overly aggressive

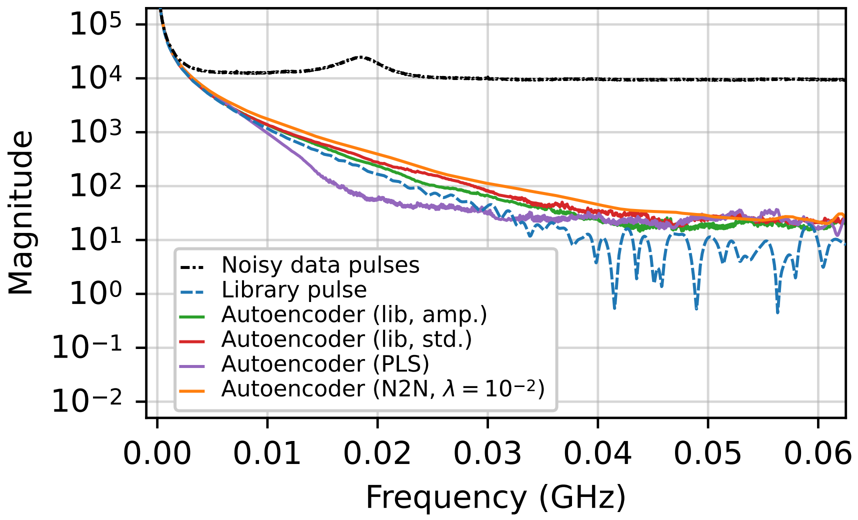


Figure 7.11: Frequency spectra comparisons for different autoencoder models. Comparison is made between a simulated library pulse, noisy data pulses, and noisy data pulses denoised with four different autoencoder models. Figure from [1].

in removing noise, specifically at lower frequencies (up to approximately 0.02 GHz) and the highest frequencies (greater than 0.05 GHz).

In Figure 7.11, both regular autoencoder models trained with library pulses and the Noise2Noise model have similar frequency spectra. The Noise2Noise model has the smoothest spectrum, but is less aggressive in removing noise at virtually all frequencies, particularly at frequencies larger than 0.04 GHz and around 0.06 GHz, indicating that it is fails to remove some of the higher frequency noise. This is consistent with visual observations of the denoised pulses. The autoencoder trained with PLS pulses has the most divergent spectrum and is removing some portion of the signal at the mid-range of frequencies. It does, however, perform similarly to the library pulse models at higher frequencies. This could be due to the fact that the PLS pulse model is more likely to distort and "smooth out" the pulse in and around the rise region, while in principle it should denoise the flat regions just as well as the library pulse models. These results are in agreement with those from Section 7.1.2 and Figure 7.2 specifically, which shows that while the Noise2Noise model is comparable or slightly better at preserving features in the rise region, it is inferior at denoising the overall pulse.

7.3 Summary and discussion

This chapter presents results of a deep, fully convolutional denoising autoencoder architecture applied to remove electronic noise from one-dimensional pulses from a PPC HPGe detector at Queen's University. Most studies are contained within the author's publication in [1] and restated here, with some additional work also included. Results show that the denoising algorithm is able to preserve the underlying pulse shape well, even in the presence of high noise. This is demonstrated via several studies on simulations, ranging from directly looking at the mean squared error between the denoised pulse and its true corresponding clean version, to applying a single-site/multi-site discrimination analysis before and after denoising. Moreover, the autoencoder is shown to work well on detector data. Statistical studies confirm that denoising with this model preserves the underlying pulse shape, and the energy resolution is shown to be improved after denoising is applied in certain circumstances.

In addition, multiple training procedures are presented and compared. Only one of the methods required detailed simulations of the detector, while the others are not limited to this stringent condition. In particular, substituting the PLS pulses as an approximation for the detailed simulated library pulses works nearly as well, demonstrating that the neural network is primarily learning to remove noise and not overfitting details about the pulse shape. Similarly, an extension of the Noise2Noise method is shown that can be trained without any underlying clean pulses as a basis of truth. Both of these methods do not outperform the regular library pulse training method. However, results are still of comparable scale and all versions of the autoencoder substantially outperform the traditional denoising methods tested. It is expected that more data, particularly from other calibration sources over a broader energy range, would improve the Noise2Noise model further.

7.3.1 Experimental implications

There are numerous implications of this work to improving the sensitivity of experiments using HPGe detectors, some of which have already been demonstrated. Simulations show that an improvement in the energy resolution can be realized with denoising. The same improvement is not observed in calibration data, though using a lower trapezoidal filter shaping time after denoising can still achieve a comparable energy resolution. If any residual exponential decays in the pulse shape can be removed, a reduction in the energy resolution should be possible on real data as well. The energy resolution of an experiment is particularly important in the $0\nu\beta\beta$ decay search for identifying and separating the peak from the $2\nu\beta\beta$ decay energy spectrum and other backgrounds. However, as the electronic noise is already quite low at $Q_{\beta\beta}$ for ^{76}Ge, improvements in the resolution at such energies are not expected to be substantial, nor is the discrimination between single- and multi-site events. With higher noise levels, however, improvements *are* expected, opening up the possibility of applying these methods to cheaper experiments or different technologies where the signal-to-noise ratio is higher at a given energy. As well, typical large germanium detector array experiments have a multifaceted physics program with simultaneous scientific goals beyond the search for $0\nu\beta\beta$ decay. Noise removal could be highly beneficial for rare event searches at low energies, such as certain Standard Model-forbidden decays, leading to sensitivity improvements.

In terms of practical implications, denoising allows one to reach the optimal energy resolution with a trapezoidal filter at significantly reduced shaping times for low energy pulses. Presumably, this would extend to other pulse processing methods such as Gaussian or cusp filters. A lower shaping time requisite is important for more efficient data collection, storage, and analysis as it reduces the need to collect longer traces without loss in energy resolution. The trained denoiser is also quick to evaluate, making integration into the data processing pipeline feasible. Similar to what was done with RAT, a framework for interfacing with TensorFlow has been partially integrated into QUADIS with major contributions from the author. Finally, the training procedures which do not require detailed simulations are still easily implemented if a basis set of pulses does not exist or fails to model the detector completely. Some of the following discussion concerns applications where this would be particularly useful.

7.3.2 Future work and prospects

The work developed here is easily scaled to a full experiment utilizing HPGe detectors. The most obvious candidate is the future tonne-scale LEGEND and its operational 200 kg demonstrator. Currently, the architecture and training procedure presented here and in [1] is being applied to data from LEGEND-200 by another student in the group at Queen's University.

As well, any experiment or application where noisy one-dimensional pulses are collected can benefit from these methods. The flexible nature of the architecture allows for any number of input channels and arbitrary length input pulses to be used, making this extension straightforward. The absence of the requirement for highly detailed simulations further makes this easy to apply elsewhere. In [212], modifications of this architecture were applied to data from NEWS-G (New Experiment with Spheres – Gas). As a dark matter search experiment based on spherical proportional counters which contain light gases as the detection medium, this technology is substantially different from the PPC HPGe detector presented earlier. The work in [212] demonstrated the application of the denoising autoencoder to two-channel correlated pulses from NEWS-G detector configurations and presented results showings improvements in energy reconstruction and primary electron counting. It was also shown through a simple study that dark matter exclusion limits could be improved with triggering on the denoised traces, rather than the original noisy ones. Work is ongoing on the NEWS-G experiment in continuing to apply and extend these methods. The group at Queen's University has also studied the feasibility of applying this architecture to signals from bubble chambers (yet another distinct detector technology used primarily for dark matter detection) with promising results.

The encoder portion of the network produces a compressed and efficient representation of the input data via its learned mapping. This opens up the possibility of using the encoded representation for other tasks, as it should contain less redundant information and lead to better outcomes. Transfer learning [213,214] can also be used to freeze some of the network parameters, retraining only those in the last layer(s) for better suitability to a specific task. This is demonstrated in [212] for direct predictions of energy and electron counting, as well as triggering. It is also still being actively explored for other projects including clipping restoration, peak finding, and single-site/multi-site event discrimination.

The group at Queen's University is also investigating new architectures for denoising to try to improve on the results presented here. For example, DualGAN [215], CycleGAN [216], and diffusion probabilistic models [217] are now being studied. These techniques may have a better chance at modelling the multiple exponential decays present in data and lead to further improvements in the energy resolution that are closer to that expected from simulations. Some of this work was proposed and initiated by the author. Investigations of continuous inline denoising before triggering to reduce thresholds and more reliably trigger on low-energy signal events are also in progress.

Overall, the work presented here is highly flexible and broadly applicable to the particle astrophysics community and beyond. Several applications of this work (even to completely different detector technologies) have already been demonstrated to be successful, and interest is being taken by collaborators on a number of different experiments.

Conclusion and Endmatter

Chapter 8

Conclusion

A large global effort in the search for neutrinoless double-beta decay is underway. As the boundaries of exclusion limits are being pushed, modern experiments hoping to detect this process are becoming much larger, more complex, and highly sophisticated. Such physical advances must be accompanied by analytical ones in order to efficiently process and analyze the massive amounts of data being collected.

This thesis presents two distinct projects, each applied to a completely different rare event search detector technology. The common theme between them is the development of novel, highly flexible neural network architectures for application to neutrinoless double-beta decay searches. Both projects contain studies to demonstrate how these methods can be used to improve sensitivities of modern experiments to rare events, in addition to the practical benefits with regards to big data processing that are provided.

The first part of this dissertation presented a new method for event reconstruction in the SNO+ detector. Results indicate that improvements to probability density functions in fits and optimizations can be realized with better position reconstruction from the neural network. The reduction in drive bias from the neural network should also have an impact on directionality estimation in liquid scintillator and is expected to improve the performance of the likelihood-based approach. Furthermore, direction reconstruction is shown to be successful using an extension of the position neural network architecture and training procedure, which will in turn help improve the sensitivity of the SNO+ experiment to the neutrinoless double-beta decay process.

The second part of this dissertation presented a deep fully convolutional denoising autoencoder developed for and applied to a local p-type point contact high purity germanium detector as an example to motivate the method. A number of studies conducted demonstrate that in addition to preserving the underlying pulse shape while simultaneously removing most of the electronic noise, denoising helps to improve the energy resolution at lower trapezoidal filter shaping times. The largest improvements seen occur at the highest noise levels, indicating that denoising is more useful in low energy rare event searches than the relatively high energies of neutrinoless double-beta decay. As well, two methods that do not require detailed simulations of the detector are shown, and comparable performance is obtained. Past and ongoing direct applications of this work to several different experiments, including NEWS-G and LEGEND-200, highlight the success of the denoising approach developed.

These methods, while applied to the specific detector configurations for which they were developed, are limited neither to the particular detection technology nor the search for neutrinoless double-beta decay. Both techniques are designed to be highly flexible with regards to the desired task, allowing for arbitrary input and output shapes as well as straightforward changes to the model hyperparameters. Explicit suggestions are made for how the architectures can be modified to accommodate a different detector, different data, and different regression or classification problems.

Numerous avenues of ongoing and potential future work are explored to demonstrate the broad applicability of the methods developed here and their relevance to the particle astrophysics community.

Bibliography

[1] M. R. Anderson *et al.*, "Performance of a convolutional autoencoder designed to remove electronic noise from p-type point contact germanium detector signals," *Eur. Phys. J. C*, vol. 82, no. 12, p. 1084, 2022.

[2] J. Chadwick, "Intensitätsverteilung im magnetischen spectrum der β-strahlen von radium B + C," *Verhandl. Dtsc. Phys. Ges.*, vol. 16, p. 383, 1914, written in German.

[3] J. Chadwick and C. D. Ellis, "A preliminary investigation of the intensity distribution in the β-ray spectra of radium B and C," in *Proc. Camb. Philos. Soc.*, vol. 21, 1922, pp. 274–280.

[4] C. D. Ellis and W. A. Wooster, "The average energy of disintegration of radium E," *Proc. R. Soc. Lond., Ser. A*, vol. 117, no. 776, pp. 109–123, 1927.

[5] C. D. Ellis and N. F. Mott, "Energy relations in the β-ray type of radioactive disintegration," *Proc. R. Soc. Lond., Ser. A*, vol. 141, no. 845, pp. 502–511, 1933.

[6] W. Pauli, "Offener brief an die gruppe der radioaktiven bei der Gau-Vereinigung zu Tübingen," 1930, written in German, translated by Kurt Riesselmann [7].

[7] K. Riesselmann, "Open letter to the group of radioactive people at the Gauverein meeting in Tübingen," 2015, translation of [6]. [Online]. Available: https://www.math.utah.edu/~beebe/talks/2015/qtm/pdf/pauli-1930-ltc.pdf

[8] E. Fermi, "Tentativo di una teoria dei raggi β," *Nuovo Cim.*, vol. 11, no. 1, pp. 1–19, 1934, written in Italian.

[9] E. Fermi, "Versuch einer theorie der β-strahlen. I," *Z. Phys.*, vol. 88, no. 3–4, pp. 161–177, 1934, written in German, translated by Fred L. Wilson [10].

[10] F. L. Wilson, "Fermi's theory of beta decay," *Am. J. Phys.*, vol. 36, no. 12, pp. 1150–1160, 1968.

[11] C. L. Cowan, F. Reines, F. B. Harrison, H. W. Kruse, and A. D. McGuire, "Detection of the free neutrino: a confirmation," *Science*, vol. 124, no. 3212, pp. 103–104, 1956.

[12] B. Pontecorvo, "Mesonium and antimesonium," *Sov. Phys. J. Exp. Theor. Phys.*, vol. 6, no. 2, pp. 429–431, 1957.

[13] Z. Maki, M. Nakagawa, and S. Sakata, "Remarks on the unified model of elementary particles," *Prog. Theor. Phys.*, vol. 28, no. 5, pp. 870–880, 1962.

[14] B. Pontecorvo, "Neutrino experiments and the problem of conservation of leptonic charge," *Sov. Phys. J. Exp. Theor. Phys.*, vol. 26, no. 5, pp. 984–988, 1968.

[15] V. Gribov and B. Pontecorvo, "Neutrino astronomy and lepton charge," *Phys. Lett. B*, vol. 28, no. 7, pp. 493–496, 1969.

[16] R. Davis, D. S. Harmer, and K. C. Hoffman, "Search for neutrinos from the Sun," *Phys. Rev. Lett.*, vol. 20, no. 21, pp. 1205–1209, 1968.

[17] J. N. Bahcall, N. A. Bahcall, and G. Shaviv, "Present status of the theoretical predictions for the ^{37}Cl solar-neutrino experiment," *Phys. Rev. Lett.*, vol. 20, no. 21, pp. 1209–1212, 1968.

[18] J. N. Bahcall, "Solar neutrinos. I. theoretical," *Phys. Rev. Lett.*, vol. 12, no. 11, pp. 300–302, 1964.

[19] R. Davis, "Solar neutrinos. II. experimental," *Phys. Rev. Lett.*, vol. 12, no. 11, pp. 303–305, 1964.

[20] B. T. Cleveland *et al.*, "Measurement of the solar electron neutrino flux with the Homestake chlorine detector," *Astrophys. J.*, vol. 496, no. 1, pp. 505–526, 1998.

[21] Y. Fukuda *et al.*, "Solar neutrino data covering solar cycle 22," *Phys. Rev. Lett.*, vol. 77, no. 9, pp. 1683–1686, 1996.

[22] W. Hampel *et al.*, "GALLEX solar neutrino observations: Results for GALLEX IV," *Phys. Lett. B*, vol. 447, no. 1–2, pp. 127–133, 1999.

[23] J. N. Abdurashitov *et al.*, "Solar neutrino flux measurements by the Soviet-American Gallium Experiment (SAGE) for half the 22-year solar cycle," *J. Exp. Theor. Phys.*, vol. 95, no. 2, pp. 181–193, 2002.

[24] Y. Fukuda *et al.*, "Evidence for oscillation of atmospheric neutrinos," *Phys. Rev. Lett.*, vol. 81, no. 8, pp. 1562–1567, 1998.

[25] Q. R. Ahmad *et al.*, "Direct evidence for neutrino flavor transformation from neutral-current interactions in the Sudbury Neutrino Observatory," *Phys. Rev. Lett.*, vol. 89, no. 1, p. 011301, 2002.

[26] M. Aker *et al.*, "Direct neutrino-mass measurement with sub-electronvolt sensitivity," *Nat. Phys.*, vol. 18, no. 2, pp. 160–166, 2022.

[27] N. Aghanim *et al.*, "Planck 2018 results - VI. cosmological parameters," *Astron. Astrophys.*, vol. 641, p. A6, 2020.

[28] S. Abe *et al.*, "Precision measurement of neutrino oscillation parameters with KamLAND," *Phys. Rev. Lett.*, vol. 100, no. 22, p. 221803, 2008.

[29] R. L. Workman *et al.*, "Review of particle physics," *Prog. Theor. Exp. Phys.*, vol. 2022, no. 8, p. 083C01, 2022.

[30] R. B. Patterson, "Prospects for measurement of the neutrino mass hierarchy," *Ann. Rev. Nucl. Part. Sci.*, vol. 65, no. 1, pp. 177–192, 2015.

[31] M. Goldhaber, L. Grodzins, and A. W. Sunyar, "Helicity of neutrinos," *Phys. Rev.*, vol. 109, no. 3, pp. 1015–1017, 1958.

[32] F. T. Avignone, S. R. Elliott, and J. Engel, "Double beta decay, Majorana neutrinos, and neutrino mass," *Rev. Mod. Phys.*, vol. 80, no. 2, pp. 481–516, 2008.

[33] E. Majorana, "Teoria simmetrica dell'elettrone e del positrone," *Nuovo Cim.*, vol. 14, no. 4, pp. 171–184, 1937, written in Italian, translated by Luciano Maiani [34].

[34] L. Maiani, "A symmetric theory of electrons and positrons," in *Ettore Majorana Scientific Papers*, G. Franco Bassani and The Council of the Italian Physical Society, Eds. Berlin, Berlin, Germany: Springer, 2006, pp. 113–128, translation of [33].

[35] T. Yanagida, "Horizontal gauge symmetry and masses of neutrinos," in *Proc. Workshop Baryon Number Universe Unified Theor.*, 1979, pp. 95–99.

[36] L. Canetti, M. Drewes, and M. Shaposhnikov, "Matter and antimatter in the Universe," *New J. Phys.*, vol. 14, no. 9, p. 095012, 2012.

[37] A. D. Sakharov, "Violation of CP-invariance, C-asymmetry, and baryon asymmetry of the Universe," *J. Exp. Theor. Phys.*, vol. 5, pp. 24–27, 1967.

[38] M. Fukugita and T. Yanagida, "Baryogenesis without grand unification," *Phys. Lett. B*, vol. 174, no. 1, pp. 45–47, 1986.

[39] B. Jones, "The physics of neutrinoless double beta decay: A beginners guide," in *Proc. Theor. Adv. Stud. Inst.*, vol. 388, 2021, p. 007.

[40] V. I. Tretyak and Y. G. Zdesenko, "Tables of double beta decay data–an update," *At. Data Nucl. Data Tables*, vol. 80, no. 1, pp. 83–116, 2002.

[41] A. Barabash, "Precise half-life values for two-neutrino double-β decay: 2020 review," *Universe*, vol. 6, no. 10, p. 159, 2020.

[42] M. Agostini, G. Benato, J. A. Detwiler, J. Menéndez, and F. Vissani, "Toward the discovery of matter creation with neutrinoless ββ decay," *Rev. Mod. Phys.*, vol. 95, no. 2, p. 025002, 2023.

[43] J. Engel and J. Menéndez, "Status and future of nuclear matrix elements for neutrinoless double-beta decay: a review," *Rep. Prog. Phys.*, vol. 80, no. 4, p. 046301, 2017.

[44] National Nuclear Data Center, "NuDat database." [Online]. Available: https://www.nndc.bnl.gov/nudat3/

[45] J. Meija *et al.*, "Isotopic compositions of the elements 2013," *Pure and Appl. Chem.*, vol. 88, no. 3, pp. 293–306, 2016.

[46] Commission on Isotopic Abundances and Atomic Weights, "Isotopic compositions of the elements 2021." [Online]. Available: https://www.ciaaw.org/

[47] R. Arnold *et al.*, "Measurement of the double-beta decay half-life and search for the neutrinoless double-beta decay of ^{48}Ca with the NEMO-3 detector," *Phys. Rev. D*, vol. 93, no. 11, p. 112008, 2016.

[48] S. Umehara *et al.*, "Neutrino-less double-β decay of ^{48}Ca studied by CaF$_2$(Eu) scintillators," *Phys. Rev. C*, vol. 78, no. 5, p. 058501, 2008.

[49] I. J. Arnquist *et al.*, "Final result of the Majorana Demonstrator's search for neutrinoless double-β decay in ^{76}Ge," *Phys. Rev. Lett.*, vol. 130, no. 6, p. 062501, 2023.

[50] M. Agostini *et al.*, “Final results of GERDA on the search for neutrinoless double-β decay,” *Phys. Rev. Lett.*, vol. 125, no. 25, p. 252502, 2020.

[51] R. Arnold *et al.*, “Final results on ^{82}Se double beta decay to the ground state of ^{82}Kr from the NEMO-3 experiment,” *Eur. Phys. J. C*, vol. 78, no. 10, p. 821, 2018.

[52] O. Azzolini *et al.*, “Final result on the neutrinoless double beta decay of ^{82}Se with CUPID-0,” *Phys. Rev. Lett.*, vol. 129, no. 11, p. 111801, 2022.

[53] J. Argyriades *et al.*, “Measurement of the two neutrino double beta decay half-life of Zr-96 with the NEMO-3 detector,” *Nucl. Phys. A*, vol. 847, no. 3, pp. 168–179, 2010.

[54] R. Arnold *et al.*, “Results of the search for neutrinoless double-β decay in ^{100}Mo with the NEMO-3 experiment,” *Phys. Rev. D*, vol. 92, no. 7, p. 072011, 2015.

[55] C. Augier *et al.*, “Final results on the 0νββ decay half-life limit of ^{100}Mo from the CUPID-Mo experiment,” *Eur. Phys. J. C*, vol. 82, no. 11, p. 1033, 2022.

[56] R. Arnold *et al.*, “Measurement of the 2νββ decay half-life and search for the 0νββ decay of ^{116}Cd with the NEMO-3 detector,” *Phys. Rev. D*, vol. 95, no. 1, p. 012007, 2017.

[57] A. S. Barabash *et al.*, “Final results of the Aurora experiment to study 2β decay of ^{116}Cd with enriched ^{116}CdWO$_4$ crystal scintillators,” *Phys. Rev. D*, vol. 98, no. 9, p. 092007, 2018.

[58] D. Q. Adams *et al.*, "Search for Majorana neutrinos exploiting millikelvin cryogenics with CUORE," *Nature*, vol. 604, no. 7904, pp. 53–58, 2022.

[59] G. Anton *et al.*, "Search for neutrinoless double-β decay with the complete EXO-200 dataset," *Phys. Rev. Lett.*, vol. 123, no. 16, p. 161802, 2019.

[60] S. Abe *et al.*, "Search for the Majorana nature of neutrinos in the inverted mass ordering region with KamLAND-Zen," *Phys. Rev. Lett.*, vol. 130, no. 5, p. 051801, 2023.

[61] R. Arnold *et al.*, "Measurement of the $2\nu\beta\beta$ decay half-life of ^{150}Nd and a search for $0\nu\beta\beta$ decay processes with the full exposure from the NEMO-3 detector," *Phys. Rev. D*, vol. 94, no. 7, p. 072003, 2016.

[62] C. Arnaboldi *et al.*, "CUORE: a cryogenic underground observatory for rare events," *Nucl. Instrum. Meth. A*, vol. 518, no. 3, pp. 775–798, 2004.

[63] D. R. Artusa *et al.*, "Searching for neutrinoless double-beta decay of ^{130}Te with CUORE," *Adv. High Energy Phys.*, vol. 2015, p. 879871, 2015.

[64] V. Alenkov *et al.*, "Technical design report for the AMoRE $0\nu\beta\beta$ decay search experiment," *arXiv preprint arXiv:1512.05957*, 2015.

[65] H. S. Jo *et al.*, "Status of the AMoRE experiment searching for neutrinoless double beta decay using low-temperature detectors," *J. Low Temp. Phys.*, vol. 193, pp. 1182–1189, 2018.

[66] K. Alfonso *et al.*, "CUPID: The next-generation neutrinoless double beta decay experiment," *J. Low Temp. Phys.*, vol. 211, no. 5–6, pp. 375–383, 2023.

[67] A. Armatol *et al.*, "CUPID pre-CDR," *arXiv preprint arXiv:1907.09376*, 2019.

[68] A. Armatol *et al.*, "Toward CUPID-1T," *arXiv preprint arXiv:2203.08386*, 2022.

[69] R. Arnold *et al.*, "Technical design and performance of the NEMO 3 detector," *Nucl. Instrum. Meth. A*, vol. 536, no. 1–2, pp. 79–122, 2005.

[70] R. Arnold *et al.*, "Probing new physics models of neutrinoless double beta decay with SuperNEMO," *Eur. Phys. J. C*, vol. 70, no. 4, pp. 927–943, 2010.

[71] M. Auger *et al.*, "The EXO-200 detector, part I: detector design and construction," *J. Instrum.*, vol. 7, no. 05, p. P05010, 2012.

[72] G. Adhikari *et al.*, "nEXO: neutrinoless double beta decay search beyond 10^{28} year half-life sensitivity," *J. Phys. G*, vol. 49, no. 1, p. 015104, 2021.

[73] F. Granena *et al.*, "NEXT, a HPGXe TPC for neutrinoless double beta decay searches," *arXiv preprint arXiv:0907.4054*, 2009.

[74] P. Novella *et al.*, "Demonstration of neutrinoless double beta decay searches in gaseous xenon with NEXT," *J. High Energ. Phys.*, vol. 2023, no. 9, p. 190, 2023.

[75] C. Adams *et al.*, "Sensitivity of a tonne-scale NEXT detector for neutrinoless double-beta decay searches," *J. High Energ. Phys.*, vol. 2021, no. 8, p. 164, 2021.

[76] S. Ajimura *et al.*, "Low background measurement in CANDLES-III for studying the neutrinoless double beta decay of ^{48}Ca," *Phys. Rev. D*, vol. 103, no. 9, p. 092008, 2021.

[77] J. B. Benziger *et al.*, "A scintillator purification system for a large scale solar neutrino experiment," *Nucl. Instrum. Meth. A*, vol. 417, no. 2, pp. 278–296, 1998.

[78] J. Benziger *et al.*, "A scintillator purification system for the Borexino solar neutrino detector," *Nucl. Instrum. Meth. A*, vol. 587, no. 2, pp. 277–291, 2008.

[79] T. Mitsui, "Low-energy neutrino physics with KamLAND," in *Proc. Neutrino Oscil. Workshop*, ser. Nuclear Physics B - Proceedings Supplements, vol. 217, no. 1, 2011, pp. 89–94.

[80] R. J. Ford, "A scintillator purification plant and fluid handling system for SNO+," in *Proc. Int. Workshop Low Radioact. Tech.*, ser. American Institute of Physics Conference Proceedings, vol. 1672, no. 1, 2015, p. 080003.

[81] M. R. Anderson *et al.*, "Development, characterisation, and deployment of the SNO+ liquid scintillator," *J. Instrum.*, vol. 16, no. 05, p. P05009, 2021.

[82] J. Ye *et al.*, "Development of water extraction system for liquid scintillator purification of JUNO," *Nucl. Instrum. Meth. A*, vol. 1027, p. 166251, 2022.

[83] Z. Zhu *et al.*, "Optical purification pilot plant for JUNO liquid scintillator," *Nucl. Instrum. Meth. A*, vol. 1048, p. 167890, 2023.

[84] V. Albanese *et al.*, "The SNO+ experiment," *J. Instrum.*, vol. 16, no. 08, p. P08059, 2021.

[85] A. Gando *et al.*, “Measurement of the double-β decay half-life of ^{136}Xe with the KamLAND-Zen experiment,” *Phys. Rev. C*, vol. 85, no. 4, p. 045504, 2012.

[86] F. An *et al.*, “The detector system of the Daya Bay reactor neutrino experiment,” *Nucl. Instrum. Meth. A*, vol. 811, pp. 133–161, 2016.

[87] J. Park *et al.*, “Production and optical properties of Gd-loaded liquid scintillator for the RENO neutrino detector,” *Nucl. Instrum. Meth. A*, vol. 707, pp. 45–53, 2013.

[88] E. E. Haller, W. L. Hansen, G. S. Hubbard, and F. S. Goulding, “Origin and control of the dominant impurities in high-purity germanium,” *IEEE Trans. Nucl. Sci.*, vol. 23, no. 1, pp. 81–87, 1976.

[89] G. S. Hubbard, E. E. Haller, and W. L. Hansen, “Zone refining high-purity germanium,” *IEEE Trans. Nucl. Sci.*, vol. 25, no. 1, pp. 362–370, 1978.

[90] G. Yang *et al.*, “Zone refinement of germanium crystals,” in *J. Phys. Conf. Ser.*, vol. 606, no. 1, 2015, p. 012014.

[91] M. Agostini *et al.*, “Limits on uranium and thorium bulk content in GERDA Phase I detectors,” *Astropart. Phys.*, vol. 91, pp. 15–21, 2017.

[92] P. N. Luke, F. S. Goulding, N. W. Madden, and R. H. Pehl, “Low capacitance large volume shaped-field germanium detector,” *IEEE Trans. Nucl. Sci.*, vol. 36, no. 1, pp. 926–930, 1989.

[93] P. Barbeau, J. Collar, and O. Tench, “Large-mass ultralow noise germanium detectors: performance and applications in neutrino and astroparticle physics,” *J. Cosmol. Astropart. Phys.*, vol. 2007, no. 09, p. 009, 2007.

[94] N. Abgrall *et al.*, "LEGEND-1000 preconceptual design report," *arXiv preprint arXiv:2107.11462*, 2021.

[95] R. Deckert *et al.*, "The LEGEND-200 liquid argon instrumentation: from a simple veto to a full-fledged detector," in *Proc. Int. Conf. Top. Astropart. Underground Phys.*, ser. Proceedings of Science, vol. 441, 2024, p. 256.

[96] R. J. Cooper, D. C. Radford, P. A. Hausladen, and K. Lagergren, "A novel HPGe detector for gamma-ray tracking and imaging," *Nucl. Instrum. Meth. A*, vol. 665, pp. 25–32, 2011.

[97] J. Ebert *et al.*, "The COBRA demonstrator at the LNGS underground laboratory," *Nucl. Instrum. Meth. A*, vol. 807, pp. 114–120, 2016.

[98] J. Ebert *et al.*, "Results of a search for neutrinoless double-β decay using the COBRA demonstrator," *Phys. Rev. C*, vol. 94, no. 2, p. 024603, 2016.

[99] J.-H. Arling *et al.*, "Commissioning of the COBRA extended demonstrator at the LNGS," *Nucl. Instrum. Meth. A*, vol. 1010, p. 165524, 2021.

[100] F. Duncan, A. Noble, and D. Sinclair, "The construction and anticipated science of SNOLAB," *Ann. Rev. Nucl. Part. Sci.*, vol. 60, no. 1, pp. 163–180, 2010.

[101] N. J. T. Smith, "The SNOLAB deep underground facility," *Eur. Phys. J. Plus*, vol. 127, no. 9, p. 108, 2012.

[102] B. Aharmim *et al.*, "Measurement of the cosmic ray and neutrino-induced muon flux at the Sudbury Neutrino Observatory," *Phys. Rev. D*, vol. 80, no. 1, p. 012001, 2009.

[103] S. Andringa *et al.*, “Current status and future prospects of the SNO+ experiment,” *Adv. High Energy Phys.*, vol. 2016, p. 6194250, 2016.

[104] P. Antonioli *et al.*, “SNEWS: the SuperNova Early Warning System,” *New J. Phys.*, vol. 6, no. 1, p. 114, 2004.

[105] S. Al Kharusi *et al.*, “SNEWS 2.0: a next-generation supernova early warning system for multi-messenger astronomy,” *New J. Phys.*, vol. 23, no. 3, p. 031201, 2021.

[106] M. Anderson *et al.*, “Measurement of the ^{8}B solar neutrino flux in SNO+ with very low backgrounds,” *Phys. Rev. D*, vol. 99, no. 1, p. 012012, 2019.

[107] M. Anderson *et al.*, “Search for invisible modes of nucleon decay in water with the SNO+ detector,” *Phys. Rev. D*, vol. 99, no. 3, p. 032008, 2019.

[108] M. R. Anderson *et al.*, “Measurement of neutron-proton capture in the SNO+ water phase,” *Phys. Rev. C*, vol. 102, no. 1, p. 014002, 2020.

[109] A. Allega *et al.*, “Evidence of antineutrinos from distant reactors using pure water at SNO+,” *Phys. Rev. Lett.*, vol. 130, no. 9, p. 091801, 2023.

[110] G. Alimonti *et al.*, “A large-scale low-background liquid scintillation detector: the counting test facility at Gran Sasso,” *Nucl. Instrum. Meth. A*, vol. 406, no. 3, pp. 411–426, 1998.

[111] A. Allega *et al.*, “Initial measurement of reactor antineutrino oscillation at SNO+,” *arXiv preprint arXiv:2405.19700*, 2024.

[112] A. Allega *et al.*, “Event-by-event direction reconstruction of solar neutrinos in a high light-yield liquid scintillator,” *Phys. Rev. D*, vol. 109, no. 7, p. 072002, 2024.

[113] D. J. Auty *et al.*, “A method to load tellurium in liquid scintillator for the study of neutrinoless double beta decay,” *Nucl. Instrum. Meth. A*, vol. 1051, p. 168204, 2023.

[114] J. B. Birks, “Scintillations from organic crystals: Specific fluorescence and relative response to different radiations,” *Proc. Phys. Soc. A*, vol. 64, no. 10, p. 874, 1951.

[115] J. B. Birks, *The Theory and Practice of Scintillation Counting.* Oxford, Oxfordshire, UK: Pergamon Press, 1964.

[116] R. A. Weldon *et al.*, “Characterization of stilbene’s scintillation anisotropy for recoil protons between 0.56 and 10 MeV,” *Nucl. Instrum. Meth. A*, vol. 977, p. 164178, 2020.

[117] P. Belli *et al.*, “The future role of inorganic crystal scintillators in dark matter investigations,” *Instruments*, vol. 5, no. 2, p. 16, 2021.

[118] P. A. Cherenkov, “Visible radiation produced by electrons moving in a medium with velocities exceeding that of light,” *Phys. Rev.*, vol. 52, no. 4, pp. 378–379, 1937.

[119] S. Agostinelli *et al.*, “GEANT4 – a simulation toolkit,” *Nucl. Instrum. Meth. A*, vol. 506, no. 3, pp. 250–303, 2003.

[120] G. Horton-Smith, "GLG4sim," 2007. [Online]. Available: https://www.phys.ksu.edu/personal/gahs/GLG4sim/

[121] R. Brun and F. Rademakers, "ROOT – an object oriented data analysis framework," *Nucl. Instrum. Meth. A*, vol. 389, no. 1, pp. 81–86, 1997.

[122] "ORTEC/AMETEK," 2020. [Online]. Available: https://www.ortec-online.com/

[123] S. I. Alvis *et al.*, "Multisite event discrimination for the MAJORANA DEMONSTRATOR," *Phys. Rev. C*, vol. 99, no. 6, p. 065501, 2019.

[124] "Struck Innovative Systems," 2022. [Online]. Available: https://www.struck.de/

[125] D. C. Radford, "siggen," 2023. [Online]. Available: https://github.com/radforddc/icpc_siggen

[126] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[127] G. V. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control Signals Syst.*, vol. 2, pp. 303–314, 1989.

[128] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient BackProp," in *Neural Networks: Tricks of the Trade*, G. Montavon, G. B. Orr, and K.-R. Müller, Eds. Berlin, Berlin, Germany: Springer, 2012, pp. 9–48.

[129] K. Fukushima, "Visual feature extraction by a multilayered network of analog threshold elements," *IEEE Trans. Syst. Sci. Cybern.*, vol. 5, no. 4, pp. 322–333, 1969.

[130] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. Int. Conf. Artif. Intell. Stat.*, ser. Proceedings of Machine Learning Research, vol. 15, 2011, pp. 315–323.

[131] M. A. Nielsen, *Neural Networks and Deep Learning.* San Francisco, California, United States: Determination Press, 2015, http://neuralnetworksanddeeplearning.com.

[132] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* Cambridge, Massachusetts, United States: MIT Press, 2016, https://www.deeplearningbook.org.

[133] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[134] S. Linnainmaa, "The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors," Master's thesis, University of Helsinki, 1970, written in Finnish.

[135] S. Linnainmaa, "Taylor expansion of the accumulated rounding error," *BIT Numer. Math.*, vol. 16, no. 2, pp. 146–160, 1976.

[136] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Comput. Math. Math. Phys.*, vol. 4, no. 5, pp. 1–17, 1964.

[137] J. Martens, “Deep learning via Hessian-free optimization,” in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 735–742.

[138] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proc. Int. Conf. Artif. Intell. Stat.*, ser. Proceedings of Machine Learning Research, vol. 9, 2010, pp. 249–256.

[139] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification,” in *IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1026–1034.

[140] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, “On the importance of initialization and momentum in deep learning,” in *Proc. Int. Conf. Mach. Learn.*, ser. Proceedings of Machine Learning Research, vol. 28, no. 3, 2013, pp. 1139–1147.

[141] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *Proc. Int. Conf. Mach. Learn.*, ser. Proceedings of Machine Learning Research, vol. 37, 2015, pp. 448–456.

[142] G. Montavon, G. B. Orr, and K.-R. Müller, Eds., *Neural Networks: Tricks of the Trade.* Berlin, Berlin, Germany: Springer, 2012.

[143] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *arXiv preprint arXiv:1207.0580*, 2012.

[144] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014.

[145] S. J. Hanson, "A stochastic version of the delta rule," *Physica D*, vol. 42, no. 1–3, pp. 265–272, 1990.

[146] N. Frazier-Logue and S. J. Hanson, "The stochastic delta rule: Faster and more accurate deep learning through adaptive weight noise," *Neural Comput.*, vol. 32, no. 5, pp. 1018–1032, 2020.

[147] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," University of California, San Diego, Tech. Rep., 1985, ICS Report 8506.

[148] G. E. Hinton, D. E. Rumelhart, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, J. A. Feldman, P. J. Hayes, and D. E. Rumelhart, Eds. Cambridge, Massachusetts, United States: MIT Press, 1986, pp. 318–362.

[149] D. H. Ballard, "Modular learning in neural networks," in *Proc. Natl. Conf. Artif. Intell.*, vol. 1, 1987, pp. 279–284.

[150] M. A. Kramer, "Autoassociative neural networks," *Comput. Chem. Eng.*, vol. 16, no. 4, pp. 313–328, 1992.

[151] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proc. Workshop Mach. Learn. Sens. Data Anal.*, 2014, pp. 4–11.

[152] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent.*, 2014.

[153] P. Holl, L. Hauertmann, B. Majorovits, O. Schulz, M. Schuster, and A. J. Zsigmond, "Deep learning based pulse shape discrimination for germanium detectors," *Eur. Phys. J. C*, vol. 79, no. 6, p. 79, 2019.

[154] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 110, pp. 3371–3408, 2010.

[155] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.

[156] M. J. D. Powell, "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *Comput. J.*, vol. 7, no. 2, pp. 155–162, 1964.

[157] A. Boehnlein *et al.*, "Colloquium: Machine learning in nuclear physics," *Rev. Mod. Phys.*, vol. 94, no. 3, p. 031003, 2022.

[158] G. Karagiorgi, G. Kasieczka, S. Kravitz, B. Nachman, and D. Shih, "Machine learning in the search for new fundamental physics," *Nat. Rev. Phys.*, vol. 4, no. 6, pp. 399–412, 2022.

[159] P. Shanahan, K. Terao, and D. Whiteson, "Snowmass 2021 Computational Frontier CompF03 topical group report: Machine learning," *arXiv preprint arXiv:1207.0580*, 2022.

[160] R. Acciarri *et al.*, "Convolutional neural networks applied to neutrino events in a liquid argon time projection chamber," *J. Instrum.*, vol. 12, no. 03, p. P03011, 2017.

[161] P. Abratenko *et al.*, "Semantic segmentation with a sparse convolutional neural network for event reconstruction in MicroBooNE," *Phys. Rev. D*, vol. 103, no. 5, p. 052012, 2021.

[162] P. Abratenko *et al.*, "Convolutional neural network for multiple particle identification in the MicroBooNE liquid argon time projection chamber," *Phys. Rev. D*, vol. 103, no. 9, p. 092003, 2021.

[163] A. Aurisano *et al.*, "A convolutional neural network neutrino event classifier," *J. Instrum.*, vol. 11, no. 09, p. P09001, 2016.

[164] P. Baldi, J. Bian, L. Hertel, and L. Li, "Improved energy reconstruction in NOνA with regression convolutional neural networks," *Phys. Rev. D*, vol. 99, no. 1, p. 012011, 2019.

[165] F. Psihas *et al.*, “Context-enriched identification of particles with a convolutional network for neutrino events,” *Phys. Rev. D*, vol. 100, no. 7, p. 073005, 2019.

[166] A. Li, A. Elagin, S. Fraker, C. Grant, and L. Winslow, “Suppression of cosmic muon spallation backgrounds in liquid scintillator detectors using convolutional neural networks,” *Nucl. Instrum. Meth. A*, vol. 947, p. 162604, 2019.

[167] A. Li *et al.*, “KamNet: An integrated spatiotemporal deep neural network for rare event searches in KamLAND-Zen,” *Phys. Rev. C*, vol. 107, no. 1, p. 014323, 2023.

[168] P. Harvey *et al.*, “X-Windows SNO(+) Event Display (XSNOED),” 2022, repository is private; see [169] for a very similar public version of the code with fewer dependencies and less capabilities. [Online]. Available: https://github.com/snoplus/xsnoed

[169] P. Harvey *et al.*, “Standalone XSNOED for SNO+,” 2020. [Online]. Available: https://github.com/boardhead/xsnoed_snoplus

[170] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, “Deep sets,” in *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 3391–3401.

[171] The HDF Group and Q. Koziol, “HDF5,” 2020. [Online]. Available: https://github.com/HDFGroup/hdf5

[172] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015.

[173] F. Chollet *et al.*, "Keras," 2015. [Online]. Available: https://keras.io

[174] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: https://www.tensorflow.org/

[175] J. Paton, "Directional reconstruction in liquid scintillator neutrino detectors," Ph.D. dissertation, University of Oxford, 2023.

[176] H. C. Griffin, "Natural radioactive decay chains," in *Handbook of Nuclear Chemistry*, A. Vértes, S. Nagy, Z. Klencsár, R. G. Lovas, and F. Rösch, Eds. Boston, Massachusetts, United States: Springer, 2011, pp. 667–687.

[177] W. Heintzelman, "The radial distribution of interactions of γ-rays from PMT background events," University of Pennsylvania, Tech. Rep., 2013, SNO+ internal document (DocDB 1708).

[178] W. Heintzelman, "Angular distribution of surviving γs from PMT β-γ events," University of Pennsylvania, Tech. Rep., 2013, SNO+ internal document (DocDB 1712).

[179] S. Naugle, "Drive studies and rudimentary directionality," University of Pennsylvania, Tech. Rep., 2023, SNO+ internal document (DocDB 8075).

[180] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. Int. Conf. Mach. Learn.*, ser. Proceedings of Machine Learning Research, vol. 48, 2016, pp. 1050–1059.

[181] D. J. C. MacKay, “A practical Bayesian framework for backpropagation networks,” *Neural Comput.*, vol. 4, no. 3, pp. 448–472, 1992.

[182] E. Goan and C. Fookes, “Bayesian neural networks: An introduction and survey,” in *Case Studies in Applied Bayesian Data Science*, K. L. Mengersen, P. Pudlo, and C. P. Robert, Eds. Cham, Zug, Switzerland: Springer, 2020, pp. 45–87.

[183] M. Gori, G. Monfardini, and F. Scarselli, “A new model for learning in graph domains,” in *Proc. IEEE Int. Joint Conf. Neural Networks*, vol. 2, 2005, pp. 729–734.

[184] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, “The graph neural network model,” *IEEE Trans. Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[185] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *Proc. Int. Conf. Learn. Represent.*, 2016.

[186] S. Cheng, “Event reconstruction in the SNO+ detector with graph neural networks,” Master’s thesis, Queen’s University, 2024.

[187] R. Brown, *Exponential Smoothing for Predicting Demand.* Cambridge, Massachusetts, United States: Arthur D. Little Inc., 1956.

[188] A. Savitzky and M. J. E. Golay, “Smoothing and differentiation of data by simplified least squares procedures,” *Anal. Chem.*, vol. 36, no. 8, pp. 1627–1639, 1964.

[189] C. Taswell, “The what, how, and why of wavelet shrinkage denoising,” *Comput. Sci. Eng.*, vol. 2, no. 3, pp. 12–19, 2000.

[190] R. Polikar, “The story of wavelets,” in *Proc. Conf. Circuits Syst. Commun. Comput.*, vol. 1, 1999, pp. 5481–5486.

[191] D. L. Donoho and J. M. Johnstone, “Ideal spatial adaptation by wavelet shrinkage,” *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[192] S. G. Chang, B. Yu, and M. Vetterli, “Adaptive wavelet thresholding for image denoising and compression,” *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1532–1546, 2000.

[193] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, 1960.

[194] N. Abgrall *et al.*, “New limits on bosonic dark matter, solar axions, Pauli Exclusion Principle violation, and electron decay from the MAJORANA DEMONSTRATOR,” *Phys. Rev. Lett.*, vol. 118, no. 16, p. 161801, 2017.

[195] M. Agostini *et al.*, “First search for bosonic superweakly interacting massive particles with masses up to $1\,\mathrm{MeV}/c^2$ with GERDA,” *Phys. Rev. Lett.*, vol. 125, no. 1, p. 011801, 2020.

[196] I. J. Arnquist *et al.*, “Search for solar axions via axion-photon coupling with the MAJORANA DEMONSTRATOR,” *Phys. Rev. Lett.*, vol. 129, no. 8, p. 081803, 2022.

[197] I. J. Arnquist *et al.*, “Exotic dark matter search with the MAJORANA DEMONSTRATOR,” *Phys. Rev. Lett.*, vol. 132, no. 4, p. 041001, 2024.

[198] E. Aguayo *et al.*, “Characteristics of signals originating near the lithium-diffused N+ contact of high purity germanium p-type point contact detectors,” *Nucl. Instrum. Meth. A*, vol. 701, pp. 176–185, 2013.

[199] C. Wiseman, “A low energy rare event search with the Majorana Demonstrator,” in *J. Phys. Conf. Ser.*, vol. 1468, no. 1, 2020, p. 012040.

[200] V. T. Jordanov and G. F. Knoll, “Digital synthesis of pulse shapes in real time for high resolution radiation spectroscopy,” *Nucl. Instrum. Meth. A*, vol. 345, no. 2, pp. 337–345, 1994.

[201] M. Agostini *et al.*, “Improvement of the energy resolution via an optimized digital signal processing in GERDA Phase I,” *Eur. Phys. J. C*, vol. 75, no. 6, p. 255, 2015.

[202] V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning,” *arXiv preprint arXiv:1603.07285*, 2016.

[203] Vasundhara, “Pulse fitting for event localization in high purity germanium point contact detectors,” Master’s thesis, Queen’s University, 2020.

[204] C. R. Harris *et al.*, “Array programming with NumPy,” *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.

[205] J. Lehtinen *et al.*, “Noise2Noise: Learning image restoration without clean data,” in *Proc. Int. Conf. Mach. Learn.*, ser. Proceedings of Machine Learning Research, vol. 80, 2018, pp. 2965–2974.

[206] L. I. Rudin, S. Osher, and E. Fatemi, “Nonlinear total variation based noise removal algorithms,” *Physica D*, vol. 60, no. 1–4, pp. 259–268, 1992.

[207] D. Strong and T. Chan, "Edge-preserving and scale-dependent properties of total variation regularization," *Inverse Probl.*, vol. 19, no. 6, p. S165, 2003.

[208] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[209] M. Agostini *et al.*, "Pulse shape discrimination for GERDA Phase I data," *Eur. Phys. J. C*, vol. 73, no. 10, p. 2583, 2013.

[210] M. Agostini *et al.*, "Pulse shape analysis in GERDA Phase II," *Eur. Phys. J. C*, vol. 82, no. 4, p. 284, 2022.

[211] R. B. Blackman and J. W. Tukey, "The measurement of power spectra from the point of view of communications engineering – Part I," *Bell Syst. Tech. J.*, vol. 37, no. 1, pp. 185–282, 1958.

[212] N. Rowe, "Machine learning applications for the NEWS-G dark matter search experiment," Master's thesis, Queen's University, 2023.

[213] S. Bozinovski and A. Fulgosi, "Utjecaj slicnosti likova i transfera ucenja na obucavanje baznog perceptrona," in *Proc. Symp. Informatica*, vol. 3, 1976, pp. 121–126, written in Croatian, reviewed in English by Stevo Bozinovski [214].

[214] S. Bozinovski, "Reminder of the first paper on transfer learning in neural networks, 1976," *Informatica*, vol. 44, no. 3, 2020.

[215] Z. Yi, H. Zhang, P. Tan, and M. Gong, "DualGAN: Unsupervised dual learning for image-to-image translation," in *IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2868–2876.

[216] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2242–2251.

[217] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 6840–6851.