SuperCDMS Event Reconstruction Using Convolutional Neural Networks

by

Lucas Valenca Soares Bezerra

B.A., Dartmouth College, 2016

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

The Faculty of Graduate and Postdoctoral Studies

(Physics)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

August 2020

The following individuals certify that they have read, and recommend to the Faculty of Graduate and Postdoctoral Studies for acceptance, the thesis entitled:

SuperCDMS Event Reconstruction Using Convolutional Neural Networks

submitted by **Lucas Valenca Soares Bezerra** in partial fulfillment of the requirements for the degree of **Master of Science** in **Physics**.

Examining Committee:

Scott Oser, Physics and Astronomy
Supervisor

Jess McIver, Physics and Astronomy
Additional Examiner

Abstract

The SuperCDMS experiment uses cryogenic silicon and germanium detectors to search for dark matter candidates such as WIMPs (Weakly Interacting Massive Particles) streaming through the Earth. Collisions in the silicon and germanium crystals are expected to produce phonons whose thermal signatures can be measured.

This thesis first describes the integration of a new Signal Distribution Unit (SDU) to the SuperCDMS data acquisition system, which allows for synchronization of multiple detectors and electronic/mechanical noise characterization via accelerometer, antenna, and AC phase measurements.

From SuperCDMS detector data it is necessary to reconstruct the energies of the particle events. This thesis explores the use of Convolutional Neural Networks (CNNs) to perform this reconstruction and finds that, although they perform well, changing the noise model breaks the model and requires the neural network to be retrained. In order to mitigate this issue, a new CNN model is proposed which includes the noise Power Spectral Density (PSD) of the data as an additional input to the CNN. While it proves to be effective as a denoising algorithm, it still fails for data with a different noise model. However, including data from multiple PSDs in the neural network training sample allows it to handle data with different types of noise while still maintaining the quality of the reconstruction. Nevertheless, neural networks trained even on multiple PSDs do not robustly handle data taken with PSDs dissimilar to those in the training sample, suggesting that CNNs may need to be retrained whenever the noise environment changes in a significant way.

Lay Summary

Astrophysical observations have established that there exists a form of invisible matter—named dark matter—which makes up the majority of the matter of the universe. Dark matter has been detected indirectly through its gravitational presence, but little is known about its composition. Leading theories suggest that dark matter consists of heavy particles which interact very little with normal matter. This thesis describes my contributions to an experiment designed to find collisions between these particles and normal matter. I create an algorithm which can observe tens of thousands of particle collision events, teaching itself to determine their energies, and test how well it predicts energies when subjected to different types of noise.

Preface

The works in this thesis were developed based on analyses and functionalities required by the Super Cryogenic Dark Matter Search (SuperCDMS) Collaboration. While the discussions are my own, the code described in Chapter 2 was developed alongside University of British Columbia postdoc Emanuele Michielin, with aid from TRIUMF scientific programmer Ben Smith. Chapters 3-4 describe my work with artificial neural networks which was inspired by To Chin Yu, a Stanford University Ph. D. student. Throughout this work, sources are cited as appropriate, particularly in Chapter 1 which introduces dark matter and the motivations of the SuperCDMS collaboration.

Table of Contents

List of Tables

List of Figures

Acknowledgements

I'd like to thank my advisor, Scott Oser, for his guidance and in particular for his consistently constructive and honest feedback. I'd also like to thank Emanuele Michielin for helping me extensively over the past two years and encouraging me in my moments of frustration. It has been a privilege learning from both of you.

Chapter 1

Direct Dark Matter Detection

1.1 Motivation

1.1.1 Galactic Inconsistencies

Gravity is the fundamental interaction which dictates the motion of astronomical objects through space. The theory of gravity, initially developed by Sir Isaac Newton in 1687 and further refined by Einstein's General Relativity in 1916 [1], came to be challenged in the early 20th century. Fritz Zwicky in 1933 estimated the mass of the Coma cluster based on measurements made by Hubble and Humason of the velocity dispersion of galaxies in the cluster [2], and claimed the cluster had approximately 400 times more mass than was accountable by observable, luminous matter [3]. Similar mass/light discrepancies occur on smaller scales—galaxy rotation curves suggest that the galactic mass-to-luminosity ratio should increase radially.

From Newtonian gravity, the orbital velocity of stars in a galaxy is predicted to be

$$v(r) = \sqrt{\frac{GM(r)}{r}}. \tag{1.1}$$

Here $M(r)$ denotes the mass within the enclosed radius. The theoretical prediction of the rotational velocities of objects in galaxies, then, based on studies of the luminous mass is that at large radii beyond the bulk of the luminous matter distribution, where $M(r)$ is constant, the velocity should fall off as $r^{-1/2}$. In the 1970s, Vera Rubin and Kent Ford demonstrated that rotation curves flatten out away from the central galactic bulge by analyzing the Doppler shift of the hydrogen-α emission line in M31 [4]. An example of their results is shown in Fig. 1.1. Fig. 1.2 shows a comparison of the measured rotation curve and the expected rotation curve based on stellar mass for galaxy NGC 1090.

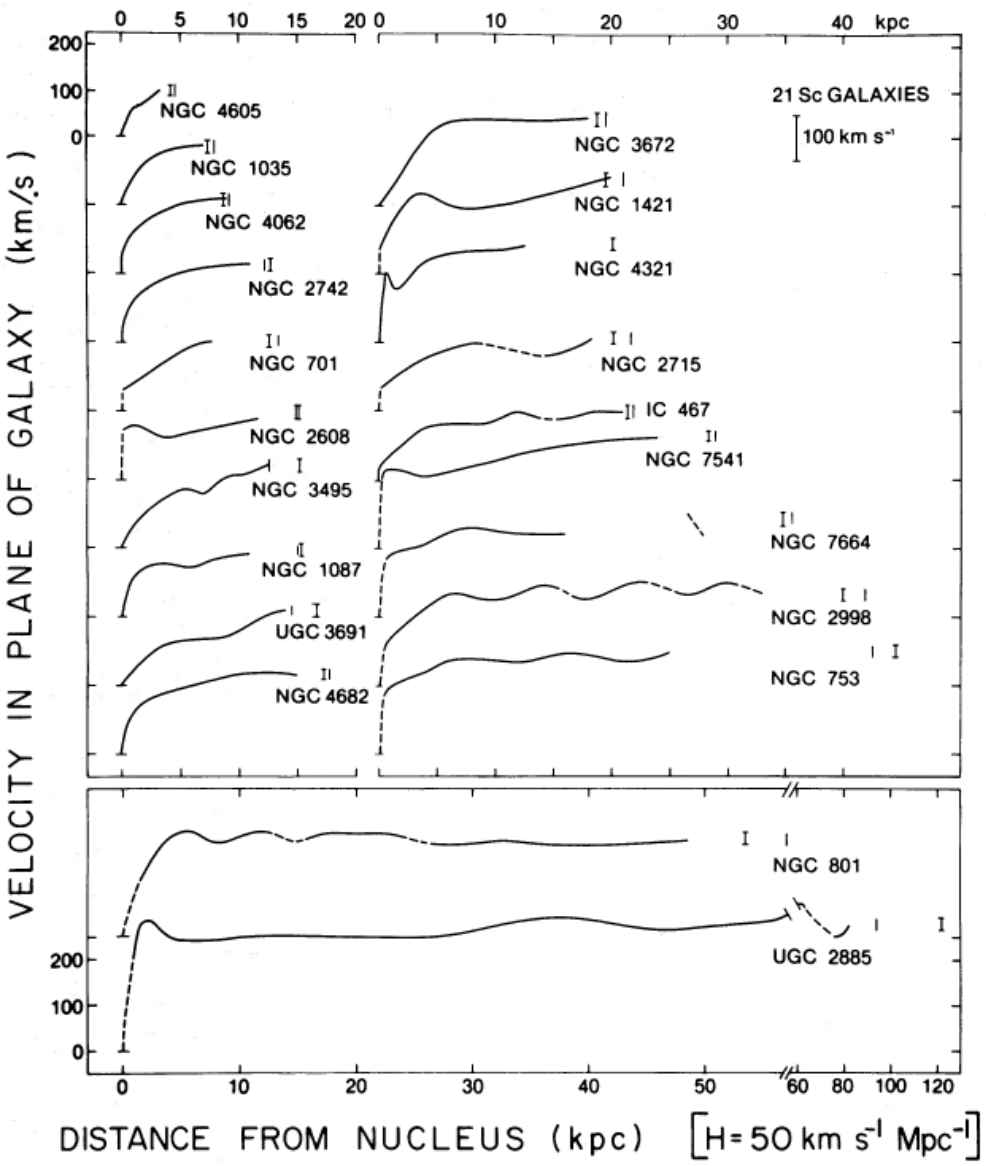

Figure 1.1: Measurements of average velocities in galactic planes for 21 spiral (Sc) galaxies ordered by radial size, showing flattened and even radially increasing rotation curves. From [5] ©American Astronomical Society, provided by the NASA Astrophysics Data System, with permission.

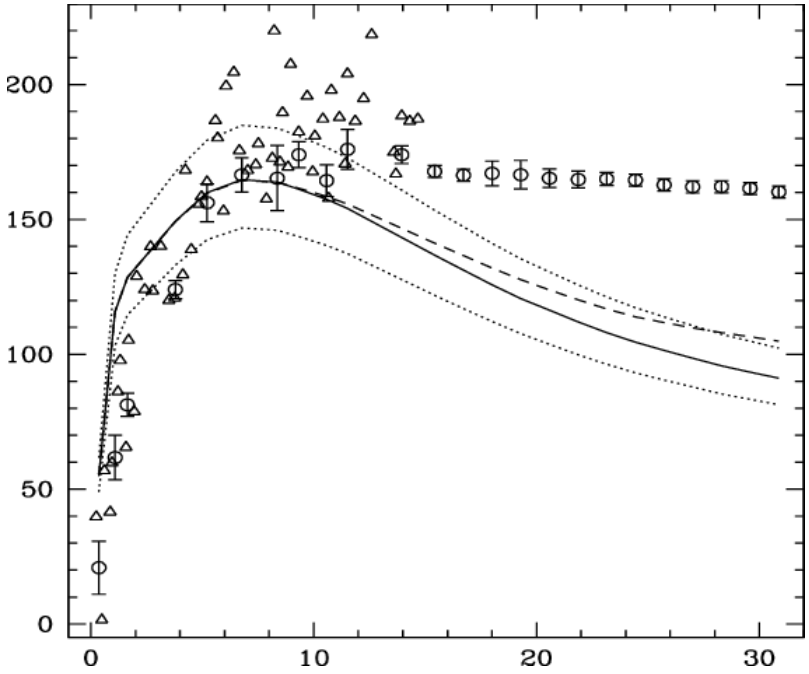

Figure 1.2: Observed rotation curves from NGC 1090 (circles are hydrogen-alpha measurements, triangles are neutral hydrogen rotation measurements) compared to predicted curves based on visible mass. The solid line is calculated from stellar mass (with an uncertainty band) and the dashed line is calculated from stellar mass and neutral hydrogen gas. X-axis is the radial distance from centre, in arcsec, and y-axis is the radial velocity in km s^{-1}. From [6], with permission.

These measurements have since been confirmed and further refined. For these observations to be consistent with Keplerian motion, it is necessary for the mass to increase radially. Additionally, galaxies must contain approximately six [5] times as much "dark" or unseen mass as visible mass. The term "dark matter" refers to this unseen, invisible mass which accounts for the majority of matter in the Universe.

1.1.2 Lensing measurements

The existence of dark matter is further evidenced by gravitational lensing measurements of galaxies and galaxy clusters. As a consequence of General Relativity, massive objects warp light between a source and an observer in line with it; the strength of the lensing effect is proportional to the object's mass. As such, it is possible to directly measure the presence of mass via the distortion of light. Measurements such as these track the distribution of dark vs luminous matter in galaxy clusters.

One of the most famous lensing observations of galactic dark matter halos is the Bullet Cluster, a collision of two galaxies observed and measured by the Hubble Space Telescope and the Chandra Observatory [7], shown in Figure 1.3. Gravitational lensing demonstrates the distribution of mass (green contours) overlaid on the image from the Chandra observatory (X-ray emissions from hot gas). The Bullet Cluster matter distributions show two gravitational spheres which have detached themselves from the luminous matter; this phenomenon is consistent with a non-interacting dark matter galactic halo, while the luminous matter in the galaxies experiences collision effects slowing down its motion.

1.1.3 Big Bang Nucleosynthesis

Observations related to the abundance of light elements in the universe constrain the amount of baryonic matter in the universe. Deuterium, helium-3, helium-4, and lithium are synthesized minutes after the Big Bang, and their abundances depend on the baryon-to-photon ratio of the early universe. Deuterium measurements in particular allow us to set an upper limit on the baryon density since its abundance decreases with time as it is destroyed easily [8]. Measurements of the deuterium abundance limit the baryonic mass density parameter in the Universe to approximately $\Omega_b = 0.04$, which is not enough to explain the increase in galactic mass due to the rotation curves or velocity dispersion measurements. There is not enough baryonic matter in the Universe to account for dark matter effects; thus, dark matter

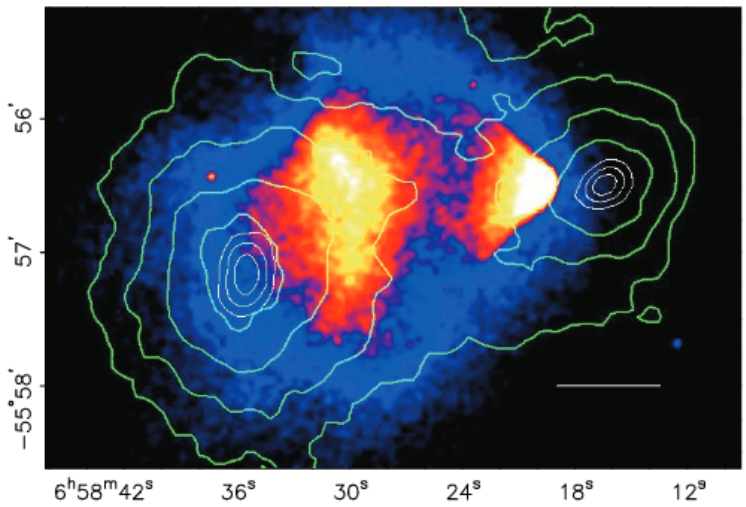

Figure 1.3: Bullet Cluster image—the green contours show the lensing mass measurements and color shows the X-ray visible matter measurements. From [7]. ©2004 X-ray: NASA/CXC/CfA/M.Markevitch et al.; Optical: NASA/STScI; Magellan/ U.Arizona/ D.Clowe et al.; Lensing Map: NASA/STScI; ESO WFI; Magellan/ U.Arizona/ D.Clowe et al., by permission

must be something else.

1.1.4 Cosmic Microwave Background

The early universe was a hot plasma with black body radiation. As it expanded and cooled enough, it became energetically favorable to form neutral hydrogen. Thus protons and electrons became transparent to primordial radiation, and it no longer scattered. This relic radiation, which permeates the entire universe, is known as the Cosmic Microwave Background (CMB), and it reflects the state of the universe at that time. The CMB has decreased in temperature with the expansion of the universe, and today is consistent with a blackbody spectrum of present temperature 2.7 K. The CMB is incredibly isotropic, making measurements of its anisotropies a source of valuable information about the universe. These anisotropies allow us to determine the density of non-baryonic matter in the universe. Even though dark matter does not interact with electromagnetic radiation, it still left a gravitational signature in the CMB from the early universe.

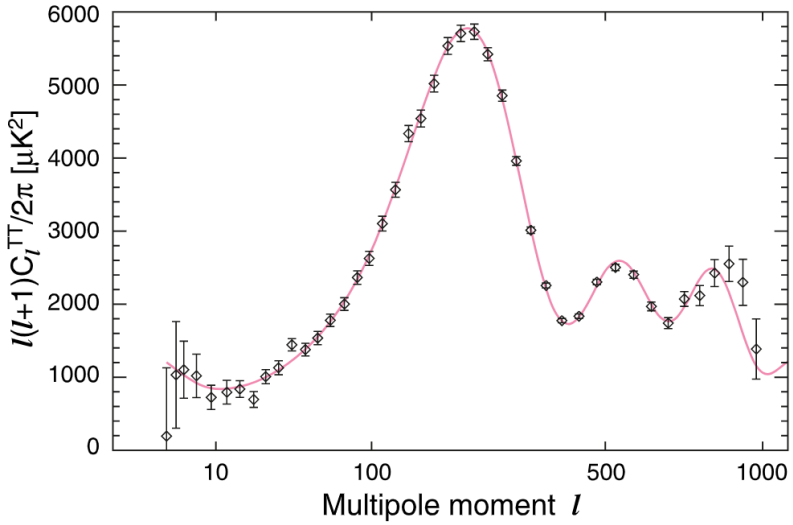

Figure 1.4: WMAP 5-year CMB Angular power spectrum. The red curve is the best-fit theory spectrum and uncertainties include both systematic (instrumental) noise and cosmic variances. Copied from [9] with permission.

By transforming the spatial anisotropies into an angular power spectrum via spherical harmonics expansion, we can obtain acoustic peaks which demonstrate the scales at which luminous matter clumped together before

recombination [9–11], as shown in Fig. 1.4. These clumps would be the product of gravitational wells generated by density fluctuations that would eventually turn into the large-scale structure of the universe [12]. In the case of baryonic matter, since excess photon emissions from these overdensities would result in an outward radiative pressure, there are resulting sound waves named Baryon Acoustic Oscillations (BAO). The center of these perturbations, however, would consist mostly of dark matter. As such, the oscillations are dependent on the dark matter density, and the amplitude of the power spectrum peaks depends on the dark matter density of the universe.

Measurements of the cosmological density parameters of the universe from the CMB currently place its composition at 26.8% matter, of which 4% is baryonic matter, with the remaining 73% consisting of dark energy. [13].

1.2 Dark Matter Candidates

Dark matter maintains its status as a major unresolved cosmological problem, as the actual composition of dark matter is currently unknown. The following subsections will present some of the leading candidates for dark matter particles.

1.2.1 WIMPs

The leading hypothesis over the last 30 years [14] for the composition of the elusive dark matter particle has been the Weakly Interacting Massive Particle (WIMP). Finding WIMPs is the primary goal of the SuperCDMS experiment and its competitors. The WIMP refers to a generic class of particle with coupling strengths characteristic of the weak interaction [15]. The WIMP is an attractive candidate because it satisfies conditions from both cosmology and particle physics. In cosmology, cold dark matter (CDM) simulations obtain large-scale structures similar to those we observe [16].

If we assume the existence of a massive particle in thermal equilibrium with the early universe, that particle would experience constant creation and annihilation until the universe cooled enough such that its number density were suddenly exponentially suppressed. At that point, the annihilation rate for WIMPs would drop below the expansion rate, and the particles would fall out of thermal equilibrium with the universe. The WIMP relic density after this “freeze out” is dependent on its annihilation cross-section. Calculating the solutions to the Boltzmann equation in an expanding universe

yields the comoving WIMP number density for different cross-sections (see [17] for a more detailed discussion). To account for the observed dark matter abundance, it is necessary for WIMPs to have annihilation cross sections on the order of the weak interaction. This coincidence between cosmological and particle interactions increased the popularity of WIMPs, as the existence of non-Standard Model particles on the weak scale is among potential solutions to the hierarchy problem of particle physics.

1.2.2 Axions and Axion-Like Particles

The quantum chromodynamics Lagrangian contains a CP-violating term which would have observable consequences such as giving the neutron an electric dipole moment [18]. However, the neutron electric dipole moment has never been observed. The standard solution [19] to this "CP problem" is a new, spontaneously broken $U(1)$ symmetry. The axion, then, is the particle associated with this spontaneous symmetry breaking. If axions exist, they are produced in the interior of stars, providing an energy-loss mechanism which allows astronomical observations of neutron stars and supernovae to set an upper limit of $\sim 10^{-2}$ eV/c^2 on their mass [20]. Since its conceptualization, the axion has become a popular dark matter candidate.

Additionally, other extensions to the Standard Model contain spontaneously broken symmetries which give rise to axion-like particles (ALPs). Both axions and ALPs are capable of coupling to electrons [21]—this interaction is named axioelectric coupling, and it leads to the absorption of the ALP, ejecting the electron and giving it the excess energy corresponding to the ALP mass. SuperCDMS has published a recent analysis [22] which sets the strongest laboratory constraints on the axioelectric coupling parameter in the mass range 40-186 eV/c^2.

1.2.3 Dark Sector

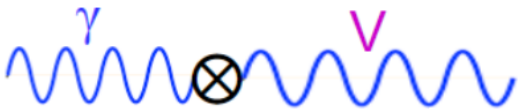

Figure 1.5: Kinetic mixing of a dark photon (V) with a SM photon (γ) via loop interactions. Further discussion can be found in [23].

Dark sector theories suggest the existence of a hidden or "dark" sector of particles with new kinds of charge, analogous to electric charge, some of

which are dark matter candidates. One theory in particular postulates a force carrier, referred to as a dark photon, with the same quantum numbers as the SM photon, but a nonzero mass [24]. The dark photon is able to kinetically mix with the SM photon and can thus interact with other SM particles. The existence of this interaction allows the dark photon (and other such dark particles) to be searched for by experiments searching for WIMP interactions. In [22] SuperCDMS sets world-leading or competitive limits on the kinetic mixing parameter for dark photons in the mass range of 40-186 eV/c^2.

1.3 Dark Matter Detection

1.3.1 Direct detection

Direct detection of WIMPs and other cold dark matter candidates relies on the interaction between baryonic and non-baryonic matter. As the Earth moves through the Milky Way's dark matter halo, it passes through what is referred to as the "WIMP wind" or "dark matter wind". Thus, there is a probability (albeit it is extremely low) of elastic collisions between WIMPs and atoms, and thus a sufficiently sensitive detector would be able to pick up these collisions and their energy signatures.

The predicted recoil energy for a WIMP interaction is given by [25]:

$$E_{\text{recoil}} = \left(\frac{m_\chi m_T}{m_\chi + m_T}\right)^2 \frac{v^2}{m_T}(1 - \cos\theta). \tag{1.2}$$

Here m_χ is the WIMP mass, m_T is the target particle mass, v is the WIMP velocity, and θ is the scattering angle. The WIMP velocity is on the order of magnitude of the Earth's velocity as it travels around the Milky Way's center; as such, these velocities are expected to have values $v \sim 300$ km/s[15]. We can use equation 1.2 to calculate the expected recoil energies for WIMPs of certain masses. If we average over the possible recoil angles ($\langle\cos\theta\rangle = 0$) and take, for example, an electron as a target (where $m_e \ll m_\chi$):

$$E_r \approx m_e v^2 \tag{1.3}$$

We can estimate an average electron recoil energy of ≈ 0.125 eV, which is (for high mass WIMPs) currently smaller than background noise levels for state-of-the-art particle detectors. However, projected sensitivities for future detectors show the possibility of detecting WIMP-electron scattering. If the

target instead is a larger nucleus of approximately the same mass as the WIMP ($m_T \approx m_\chi$), however, we achieve recoil energy values ~ 10 keV, which are detectable with current technologies. In general, the recoil energy is proportional to the WIMP mass if the nucleus is not too light.

The event rate for nuclear recoils from simple WIMP models is featureless and quasiexponential [26]. The differential nuclear recoil rate is:

$$\frac{dR}{dE_r} = \frac{N_T m_T}{2 m_\chi \mu_T^2} \left[\sigma_0^{\mathrm{SI}} F_{\mathrm{SI}}^2(E_r) + \sigma_0^{\mathrm{SD}} F_{\mathrm{SD}}^2(E_r)\right]. \tag{1.4}$$

Here N_T quantifies the number of nuclei per target mass, m_T is the target nucleon mass, $\mu_T = m_\chi m_T/(m_\chi + m_T)$ is the reduced mass of the WIMP-target system, σ are the cross-sections for the spin-independent (SI) and spin-dependent (SD)[1] models, and F is the nuclear form factor.

Figure 1.6 shows the calculated recoil rates for WIMPs of different masses, with standard astrophysical parameters (subject to some variation). The curves in the figure clearly show a significant increase in event rate at lower energies; as such, it is reasonable to attempt to run detectors sensitive to those energies.

1.3.2 Indirect Detection

Along with the direct detection efforts, it is also possible to look for astronomical signatures of WIMP annihilation in the form of secondary particles that would arise. These searches are referred to as indirect detection experiments.

Gamma rays are a promising dark matter signature. They are easy to detect and retain their directionality over extragalactic distances [15]. As such, a gamma ray detector could map out the annihilation signature of dark matter throughout the universe. However, this requires the WIMP to annihilate to a final state containing a photon, a process which has a low branching ratio and is difficult to detect.

An alternative is to focus on locations where dark matter may have accumulated, leading to higher rates of annihilation. One such example is the centre of the galaxy, although whether or not dark matter has actually

[1] The spin-independent interactions refer to zero-spin WIMPs interacting with zero-spin nuclei. The spin-independent term in the cross-section scales as the number of nucleons squared, which makes the spin-independent cross-section significantly higher than the spin-dependent version, which does not scale accordingly. For this reason, and for the fact that spin-sensitive heavy isotopes are rare, most direct detection experiments focus on spin-independent interactions[15].

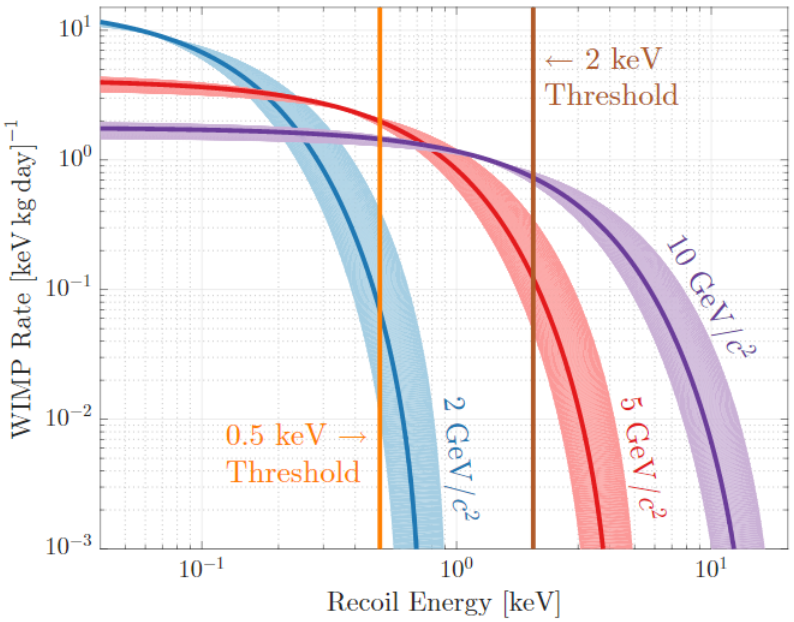

Figure 1.6: Expected recoil event rates for 2, 5 and 10 GeV/c^2 WIMPs on a Germanium target. The bands around each curve are produced by varying the astrophysical parameters of the dark matter halo, and the WIMP-nucleon cross-section considered is 10^{-41} cm^2. Image taken from [26] with permission.

clumped there is contested [27]. However, dark matter is also expected to accumulate and annihilate inside the Sun due to being trapped in its gravitational well. It is possible, in fact, to reach a state of equilibrium between annihilation and accumulation.

Most annihilation products will be captured rapidly by the material enclosing the dark matter, which makes neutrinos a viable candidate for an annihilation tracer, as their extremely low cross section allows many of them to pass through unimpeded.

Direct and indirect detection experiments are complementary to each other, helping to constrain dark matter masses and properties.

1.4 The SuperCDMS Experiment

The Super Cryogenic Dark Matter Search is one of the world-leading direct detection dark matter experiments. SuperCDMS's focus is to use semiconducting detectors to achieve very low energy thresholds, giving sensitivity to energy deposition from light dark matter candidates that deposit less energy than heavier candidates.

1.4.1 Operating Principle

SuperCDMS uses cylindrical semiconducting germanium and silicon crystals with two different designs—iZIPs (interleaved Z-Sensitive Ionization and Phonon detectors) and HV (high voltage)—for dark matter detection.

Phonon Signal

When a particle collides inside of the crystal, it generates phonons in the lattice. The detectors are held at temperatures ∼10 mK. Since the crystal heat capacity $\propto T^3$, the low temperature makes the detector more sensitive to minute temperature changes (additionally, it reduces electronic noise [25]). There is additionally a bias voltage applied across the z-axis of the detector, which allows CDMS to take advantage of the Luke-Neganov effect. The initial particle collision liberates electron/hole pairs from the valence band into the conduction band. This is one of the reasons for the choice of germanium and silicon crystals—both have small ($\sim$ 1 eV) band gaps. The liberated charges then drift across the detector due to the applied electric field. As they move, they collide continuously with atoms in the crystal structure, dissipating the energy supplied by the electric field in the form of Neganov-Trofimov-Luke (NTL) phonons, or Luke phonons for short. The accelerated

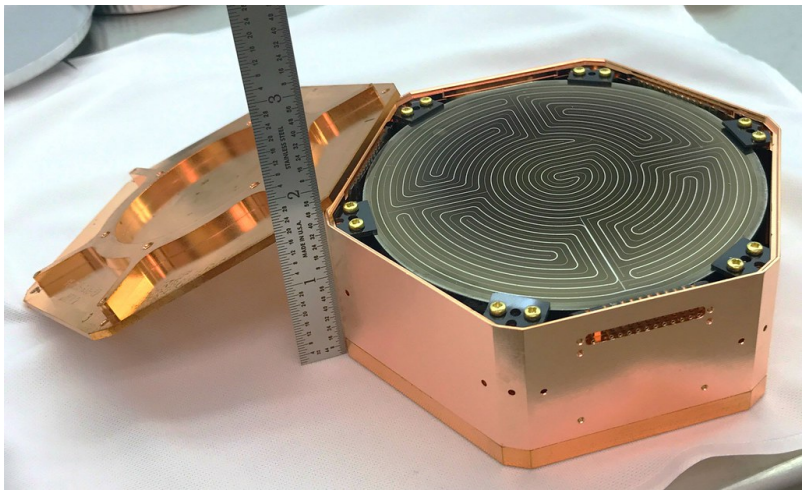

Figure 1.7: SuperCDMS iZIP detector. Figure from SuperCDMS Collaboration approved public plots, taken with permission.

electron-hole pairs relax to the Fermi sea near detector boundaries; their excess energies are released as phonons, called relaxation or recombination phonons [26]. The phonon channels are equipped with thousands of tungsten Transition Edge Sensors (TES). These are superconducting sensors which have a cryogenic critical temperature higher than the operating temperature of the detector. By applying a voltage bias to the TESs, holding it within the range of its superconducting transition, it becomes highly sensitive to temperature fluctuations, as a minute change in temperature will produce a huge change in resistance. As such, heat from the phonons decreases the current through the TES, yielding pulse (current) shapes such as the one shown in Figure 1.8.

Charge Signal

If a particle interacting in a semiconducting lattice deposits much more energy than the material's band gap, valence electrons can be excited onto the conducting band. These electrons can then liberate other electrons. As such, with collisions in detector crystals there is a population of electron-hole pairs that are liberated; since we apply a voltage across the detector, those charges move towards ionization electrodes on the bottom face of the detector[2], shown in Figure 1.9. This drift also keeps electron-hole pairs from

[2]Only the iZIP detectors have the ionization electrodes—HV only has phonon channels.

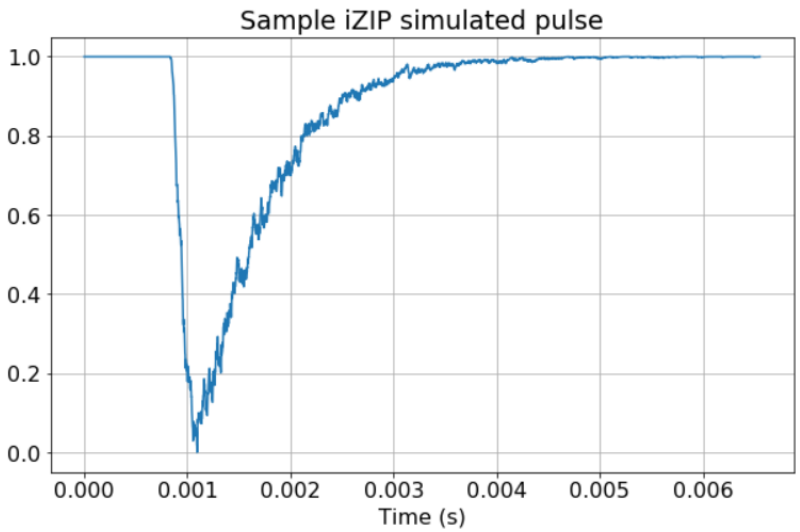

Figure 1.8: Simulated iZIP WIMP pulse. Y-axis has been rescaled to fit values between 0 and 1.

recombining at the recoil site [15]. We then obtain an ionization signal.

1.4.2 Nuclear and Electron Recoils

A particle colliding in a crystal lattice can deposit energy directly by exciting valence electrons to the conduction band, or by collision with atomic nuclei, giving them kinetic energy—these are referred to as electron recoils and nuclear recoils respectively. It is important to discriminate between these two types of signals, as most radioactive backgrounds generate electron recoil events. In addition, WIMPs are expected to have a much higher cross-section for scattering on nuclei than on electrons, so WIMPs will produce nuclear recoils. Typical CDMS analyses involve event selection criteria that cut electron recoils; however, some atypical [22, 28] analyses look for electron recoil signatures to set limits on light dark matter candidates such as dark photons and ALPs.

By measuring both charge and phonon signals, we can then compute the ionization yield:

$$Y(E_r) = \frac{E_Q}{E_r} \tag{1.5}$$

Here E_Q is the energy used in producing electron/hole pairs. For electron

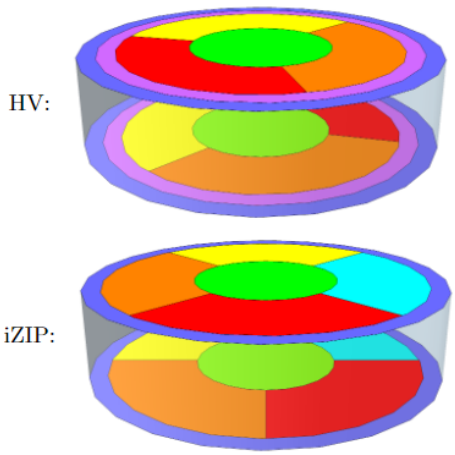

Figure 1.9: Channel layout for the HV (top) and iZIP (bottom) detectors. The HV detectors have six phonon channels arranged as an inner "core" channel, three wedge-shaped channels and two rings intended for edge event rejection. The iZIP detectors have six phonon channels as well, but four wedge shaped channels and one edge rejection ring. In addition, iZIPs have an outer ionization channel interleaved with the outermost phonon ring, and an inner ionization channel interleaved with the rest of the channels. Figure from SuperCDMS Collaboration [26], and is used with permission.

recoils, $E_Q = E_r$, whereas the yield tends to be lower (~ 0.3) for nuclear recoils above 10 keV [26]. This yield discrimination allows us to effectively remove all electron recoil backgrounds in the bulk of the detector.

1.4.3 HV vs iZIP

Separation between electron recoils and nuclear recoils is an essential feature of CDMS analyses, but it is only possible with iZIP detectors, since HV detectors do not have phonon channels. The advantage of HV, however, is its sensitivity to lower energy events due to its high voltage amplification of lower-energy events, taking advantage of the Luke-Neganov effect described in section 1.4.1. The two detector types are intended to work in tandem and are complementary to each other. While HV can detect lower energy events, iZIP can discriminate recoil types in a higher energy range, quantifying background sources which can then be eliminated from HV measurements via background reduction techniques.

1.4.4 SNOLAB Facility

SuperCDMS's last data runs took place at the Soudan Underground Mine in Minnesota, from 2011 to 2015. The next generation of the SuperCDMS experiment will take place in the SNOLAB underground laboratory in Sudbury, Ontario. This new installation will have greater shielding against high energy cosmic rays and radioactive decay byproducts, which will reduce backgrounds and refine the search for dark matter particles.

The experiment will consist of 4 detector towers with larger detectors yielding a much better energy resolution than the Soudan experiment's, with an anticipated 5 years of operation with 80% live time. SNOLAB is also a cleaner and deeper (2km underground) site compared to the Soudan mine (CDMS's last site), resulting in significant background reductions. As figure 1.10 shows, SuperCDMS will have world-leading sensitivity to WIMP-nucleon scattering in the 0.5-5 GeV/c^2 mass range. Installation completion is expected for 2021.

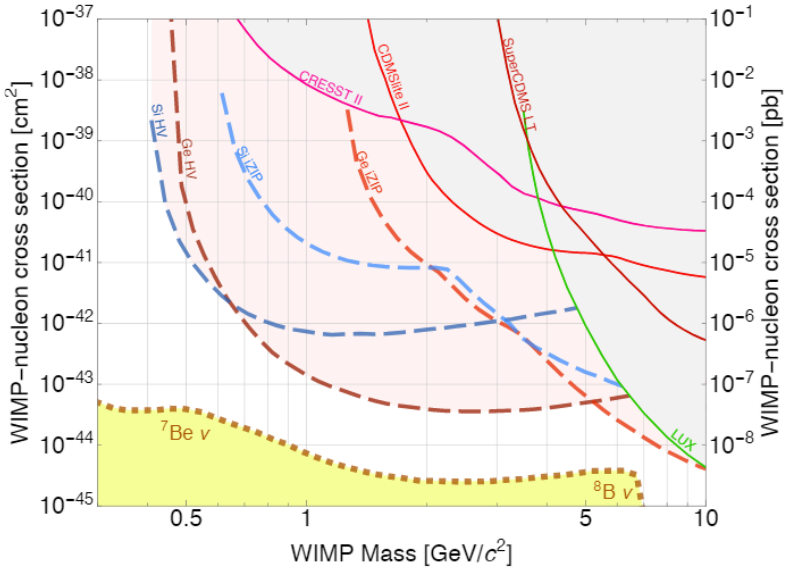

Figure 1.10: Projected sensitivities for SuperCDMS SNOLAB. Figure from SuperCDMS Collaboration approved public plots.

Chapter 2

Signal Distribution Unit

This chapter describes generally the readout electronics of SuperCDMS SNOLAB, as well as the the development of the software to operate the newly-built Signal Distribution Unit (SDU).

2.1 Readout Electronics

Each SuperCDMS detector at SNOLAB will be equipped with a Detector Control and Readout Card (DCRC). DCRCs are responsible for continuously digitizing detector analog signals from the phonon and charge channels described in section 1.4.1. The DCRC is the fundamental piece of readout hardware for SuperCDMS readout electronics, as it is capable of simultaneously triggering and digitizing the detector data stream (a feature previous iterations of CDMS did not have), resulting in trigger and readout functions without deadtime.

The DCRCs continuously write digitized signals from their channels onto a circular memory buffer which can store several seconds of data bins. They also record where triggers occur within their circular buffer (in the form of a pointer) and triggers and associated waveforms are read out before the circular buffer resets and overwrites the data [29].

2.2 MIDAS and Readout

In order to read data from the DCRCs, we have developed software which passes "level 1" (L1) trigger information from the DCRCs to a global "level 2" trigger (described in section 2.5) which determines what to do with the trigger information. This software is referred to as the readout front-end. After passing the L1 trigger information, the readout front-end for each DCRC is then responsible for reading waveform data and sending it to an event builder (EB) program, which combines data from multiple front-ends into complete events [30].

To program these front-ends, we rely heavily on the C++ MIDAS (Maximum Integrated Data Acquisition System) package, since it comes with a host of built-in data acquisition features which are highly relevant to our experiment[31].

2.3 SDU Purpose and Functionality

In order to maximize sensitivity it is important to remove background events and make sure the DCRCs are properly synchronized in order to avoid unwanted coincidences. For that purpose SuperCDMS SNOLAB will utilize a Signal Distribution Unit (SDU) which has been designed and built by Sten Hansen, an engineer at Fermi National Accelerator Laboratory (FNAL). The SDU is a piece of custom electronics designed as a type of synchronizer and noise monitor for the DCRCs and the experiment in general. It controls the clocks on all of the DCRCs through daisy chains which connect up to six DCRCs in a row with CAT-5 cables (as shown in figure 2.1). When the SDU is enabled and acquiring waveforms, fast signals sent down the daisy chain cause all of the DCRCs to reset their waveform pointer, which acts as a clock, to zero, and then to begin acquiring data. This allows us to align data taken by different DCRCs in time.

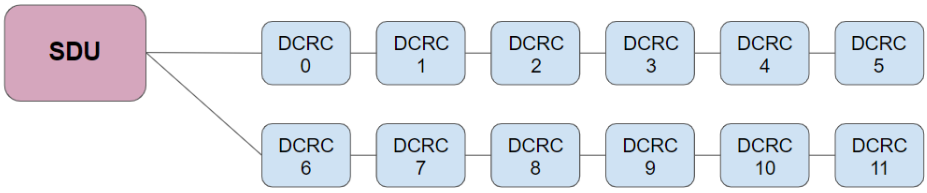

Figure 2.1: The SDU connects up to six DCRCs in series per daisy chain; this allows it to synchronize all of the DCRCs in the experiment. Here only 12 are shown, but SuperCDMS SNOLAB will have 36 DCRCs.

It is sensitive to electronic and mechanical noise, as it is equipped with:

- Four accelerometer channel inputs, digitized at 625 kHz with 8-bit digitizers;
- Two RF antenna inputs, digitized at 2.5 MHz with 8-bit digitizers;
- One AC power digitizer, digitized at 19.53 kHz with an 8-bit digitizer.

The SDU has an internal trigger firmware similar to the DCRC L1 trigger which allows it to form triggers on accelerometer, antenna, and AC phase channels. The trigger firmware operates separately on each of the accelerometer channels, but it combines the input from the antenna channels. If a given input is higher than a trigger threshold, a trigger primitive is recorded and sent to the L2 trigger by the readout front-end (described in Section 2.6). We employ a series of decision-making steps with the SDU triggers; these are explained in Section 2.5. With this functionality, CDMS events can be ruled out in cases of environmental electronic or mechanical noise picked up by the SDU. Additionally, data from the accelerometers, antenna, and AC phase can be characterized to help with background reduction. Samples of data from the accelerometers and antenna are shown in figure 2.2.

The SDU also has four LEMO inputs and two LEMO outputs. The inputs allow us to accept trigger inputs from external equipment, and the outputs allow us to generate digital output pulses. A pulse on one of the input channels will cause an entry to be made in the SDU's trigger firmware (if that channel is enabled—see the ODB subsection in 2.4). We use these as selectable triggers—these will also be explained in 2.5.

Finally, the SDU is capable of sending messages to all the DCRCs. The primary function of these messages is as a synchronization trigger, in which we read the internal clock pointers for all of the DCRCs and SDU, to make sure they are synchronized. If the DCRCs and SDU are out-of-sync we are unable to group simultaneous events (coincidences) properly. Therefore, the synchronization trigger is sent regularly and can also be issued manually by the user.

The SDU Message can also set or clear a trigger inhibit, which keeps the DCRCs from triggering, or simultaneously send a synchronization trigger and set/clear a trigger inhibit. The inhibit messages could potentially be used as event vetoes, but for now they have not been incorporated into the SDU/DCRC software.

The following sections describe the software used to operate the SDU. I was responsible for creating and testing the SDU Driver, and programming the interface between the SDU and the L2 trigger (described in Section 2.5).

2.4 SDU Driver

The SDU Driver is the program responsible for controlling the settings of the SDU. MIDAS allows us to create an Online Database (ODB) which allows

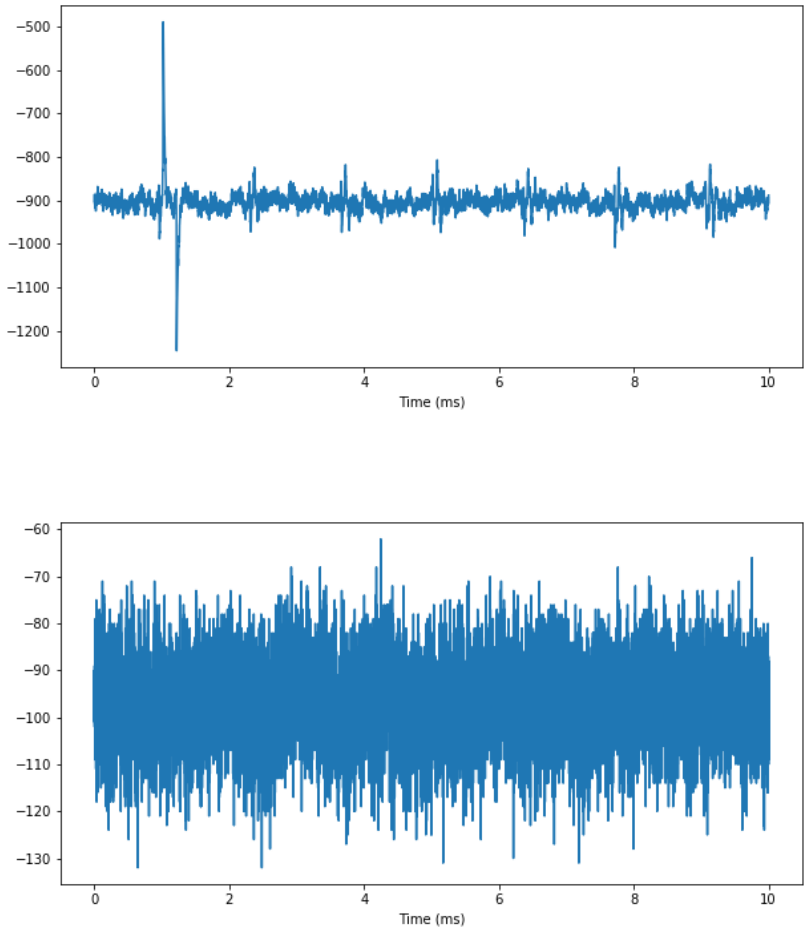

Figure 2.2: Plots of data from one accelerometer channel (top) and one antenna channel (bottom), taken during testing of the SDU readout front-end.

users to manually specify settings. In the ODB we have created a section called "Readback" which also allows users to read all of the current SDU settings. The primary purpose of the SDU Driver is to propagate the changes made by the users in the ODB to the SDU, and read back the information from the SDU. The SDU Driver achieves this by constantly monitoring the variables created in the ODB. If one of them changes, that change is first checked—for many of the variables, we have specified allowed values and if the user input does not fall within those values, the variable will remain unchanged and an error message will be displayed to the user. Then, the allowed change will be linked to a specific function call by a MIDAS "hotlink" functionality. The MIDAS ODB is extremely user-friendly, and one of its advantages is it allows the user to put in real values which can then be converted into values the SDU understands. Registers in the SDU may take hexadecimal values from 0 to 0xFFFF (65535 in denary). As an example, if there were a setting which regulated a voltage and it could range from 0 to 40 V, the user may enter the voltage directly (ie: 20 V) and the SDU driver automatically calculates the correct hexadecimal value to pass to the register. Thus, users can put in and read back direct, real values for settings instead of hexadecimal values which then have to be converted.

ODB

In the ODB we have created an SDU category with all of the required settings. The SDU category then is divided into:

1. **Control and Status Settings:** allows us to reset the DRAM (dynamic random access memory) pointer. This resets the internal time on the SDU and is done in principle at the beginning of every data run. It also causes the internal time on all DCRCs to reset.

2. **Antenna:** Controls the antenna settings. These include:

   - The number of samples to be averaged for the antenna baseline and signal (separately),
   - The antenna trigger threshold in ADC counts,
   - The trigger deadtime—minimum time between antenna triggers,
   - A weighting factor applied to the antenna inputs before being added to the trigger function and applied to the trigger threshold.

3. **DAC Settings:** controls the Digital to Analog Converters (DACs) used to power the accelerometers.

4. **Accelerometer:** controls the settings for the four accelerometer channels, including:
   - The number of samples to be averaged for the accelerometers' baseline and signal (separately),
   - The trigger threshold in ADC counts for each accelerometer channel
   - Accelerometer trigger deadtime,
   - The gain on the programmable amplifiers for the each of the accelerometer inputs.
5. **Trigger:** This subcategory contains only one single functionality, which is to enable or disable the SDU's triggering on any of its channels including the accelerometers, antenna inputs, AC phase, LEMO inputs, and synchronization.
6. **SDU Message:** is responsible for sending a message to all the DCRCs. This message is mainly intended as the synchronization trigger—by setting the message to a specific value, it will cause all the DCRCs and the SDU to issue a synchronization trigger in their trigger functions. The other possible messages are to set or clear a trigger inhibit, and to set a trigger inhibit and send a synchronization trigger simultaneously.
7. **LEMO Outputs:** controls the two front panel LEMO Outputs, allowing us to:
   - Enable/disable each of them,
   - Select between latched and pulse outputs, and
   - Determine the length of the pulses, if in pulse mode.
8. **Sync Trigger:** The SDU Driver has been programmed to automatically send a synchronization trigger periodically. Here the user can determine the time in between sync triggers.

2.5 Interface with the L2 Trigger

The L2 trigger is, unlike the DCRC/SDU trigger firmware described so far, a program which combines trigger primitive information from the DCRCs

and SDU and makes decisions on which detectors to issue readout requests to. Additionally, it is able to issue experiment-wide random triggers.

The L2 communicates with the DCRC/SDU readout front-ends directly via a socket connection. Each front-end periodically sends trigger information to the L2. Then the L2 sends readout requests back to the front-ends.

2.5.1 Types of Triggers

There are several possible types of triggers the L2 can encounter.

Primitive Triggers

Primitives are the primary, "physics" triggers the L2 receives from the DCRCs/SDU. When the L2 receives physics triggers, it sorts them by time and checks for events within short time windows of each other. If events from the same detector fall within this window, they are combined and labeled as piled-up or pileup events. If they come from different detectors, they are referred to as coincidence events and the triggers and readout requests are collected within a single event. The time window for piled-up events and coincidence events can be modified separately by the users in the ODB. The L2 may also combine SDU triggers into pileup events; there is, however, no coincidence logic in place for the SDU.

After pileup and coincidence logic, the L2 decides what to read out, based on what type of trigger it has received. For any type of DCRC trigger, the L2 is capable of reading the triggering detectors, all detectors on the same horizontal plane across the detector towers, all detectors in the same tower (there will be four detector towers at SNOLAB, each with six detectors), or the entire experiment. Normally all triggers are read, but if the event rate is too high, we can choose to prescale them.

With the inclusion of the SDU, there have been a few additions to the L2's trigger logic. Firstly, for all types of events (singles, pileup, coincidence) there is now an SDU prescale which determines whether or not the SDU should be read out in the case of a DCRC trigger. In principle for dark matter detection runs, the SDU data should always be read out in order to exclude noise events; however, this poses a data volume problem as will be shown in section 2.5.3. Therefore there is now an SDU prescale as well. Furthermore, when the SDU receives an accelerometer/antenna trigger it is possible to issue a readout request only to the SDU (the default), or to all the detectors in the experiment.

Even though for the most part, the SDU is treated as another DCRC

with a special ID, SDU trigger primitives are seen as different to DCRC primitives. This was done to allow us to separate out the SDU information and make trigger decisions more easily. Thus when the L2 receives trigger primitives, it first deals with the SDU triggers before dealing with the DCRC triggers.

When the L2 receives SDU triggers it immediately categorizes the trigger type. In the case of accelerometer/antenna trigger, the readout request is sent and the event is labeled as an SDU event.

LEMO trigger

In case the trigger comes from one of the LEMO inputs, then the L2 looks at the ODB. We have created, for each one of the LEMO inputs, a list of booleans, each one corresponding to a detector number. If any booleans in a list are set to 1, that detector will be read out if the L2 receives a trigger from that LEMO input. Thus we use the LEMO as an experiment-wide selectable triggers, allowing users to trigger specific detectors simultaneously and on command. In that case, these events are labeled LEMO events to keep them separate from physics events.

Synchronization trigger

Lastly, the L2 may receive synchronization triggers from either the DCRCs or SDU. When the L2 receives a list of triggers, it immediately filters out the synchronization triggers into a separate list to be dealt with separately. This is done so that the synchronization can be grouped together into a single event. Sync triggers do not require any waveforms to be read out—all the L2 does is save the trigger primitives together into a single event, label it a synchronization event, and send the information to the event builder.

These are all the types of primitive triggers the L2 may receive from the DRCs/SDU. The following section will describe the other potential trigger types the L2 deals with.

2.5.2 Other types of triggers

The L2 has a few other trigger sources:

- **Random triggers** are issued at random times with an average rate specified by the user. In the case of a random trigger (created directly by the L2), the entire experiment (including the SDU) is read out.

- **Test Signal triggers** occur when the DCRCs generate test signals for themselves and then trigger on those signals. These are simple triggers with no pileup/coincidence logic.
- **Baseline Control triggers** are special random triggers that activate when phonon/charge channel baselines go out of threshold for the DCRCs. It is a trigger with a high rate so the Baseline Control program can tell when the baselines have returned to normal.
- **Salting Stub Random triggers** inject dummy events into the triggers to later substitute for fake events, while keeping the ordering of real events consistent.

The L2 makes trigger decisions every second, with triggers sorted into time-ascending order. It also saves trigger history information, including the rates of each type of trigger.

2.5.3 Data Volume

With the inclusion of the SDU data in CDMS events, there has been an increase in the CDMS data volume. CDMS will employ a "hybrid readout" scheme for the DCRCs, wherein pre-pulse and post-pulse samples of event data are downsampled to reduce data cost. We do not employ downsampling for the SDU, and the antenna is digitized at four times the rate of the DCRC data, so SDU information takes up a significant amount of space. Figures 2.3a and 2.3b show the increase in the data volume estimate due to the SDU.

Due to the high triggering rate for barium calibration measurements, the data volume becomes extremely high if the SDU is included. Hence, the decision was made to exclude a high amount of SDU data from the Barium calibration data (only keep one in ten events).

2.6 SDU Readout Front-End

The SDU has its own readout front-end program, separate from the DCRC readout front-ends. It is responsible for fetching SDU trigger information, passing it to the L2 trigger, and reading any waveforms requested by the L2 trigger. A thread of the program regularly checks the SDU for new triggers through a linux socket. When a trigger primitive is found, it is sent to the L2 trigger along with the pointer to its location on the SDU dynamic memory, the amplitude of the trigger, and a "trigger word" which depends on the

Data Type	Trigger Rate	Event Size	Max Data Rate	Live Fraction	Average Uncompressed Rate
Ba Calibration	120 Hz (5x24 det.)	0.14 MB	16.8 MB/s	0.035	0.34 TB/wk 17.6 TB/yr
WIMP Search	0.03 Hz (low noise) 1 Hz (high noise)	3.36 MB	0.10 MB/s (low) 3.36 MB/s (high)	0.8	0.046 TB/wk 2.4 TB/yr (low) 1.54 TB/wk 80 TB/yr (high)
Noise traces	0.1 Hz	43.2 MB	4.32 MB/s	0.835	2.08 TB/wk 108 TB/yr
TOTAL				**0.835**	**128-206 TB/yr**

(a) Data volume estimates (no SDU).

Data Type	Trigger Rate	Event Size	Max Data Rate	Live Fraction	Average Uncompressed Rate
Ba Calibration	120 Hz (5x24 det.)	**0.92 MB**	**110.4 MB/s**	0.035	**2.23 TB/wk** **115.96 TB/yr**
WIMP Search	0.03 Hz (low noise) 1 Hz (high noise)	**4.14 MB**	**0.12 MB/s (low)** **4.14 MB/s (high)**	0.8	**0.055 TB/wk** **2.86 TB/yr (low)** **1.91 TB/wk** **99.32 TB/yr** **(high)**
Noise traces	0.1 Hz	**43.98 MB**	**4.398 MB/s**	0.835	**2.107 TB/wk** **109.7 TB/yr**
TOTAL				**0.835** **0.835**	**229-324 TB/yr** **130-226 TB/yr** **(without Ba SDU)**

(b) Data volume estimates with SDU.

Figure 2.3: Data volume calculations. Top: estimates without the SDU.

source of the trigger (accelerometer, antenna, LEMO, etc). The L2 then uses the trigger word to execute its trigger decisions.

A separate thread listens for L2 waveform readout requests every 20 ms. Since the DCRCs and the SDU rotate through a dynamic memory, it is possible for the whole data acquisition system to become overloaded and not able to keep up. If that happens, triggers may correspond to data that has already been overwritten. To mitigate this possibility, the readout front-end checks the current pointer time against the time at the end of the waveform to be read. If the difference is too large, these events are marked as "stale" and not read out.

Chapter 3

Energy Reconstruction with Neural Networks

3.1 Motivation

Chapter 2 discussed how to obtain data from SuperCDMS detectors, following the explanation in Chapter 1 regarding the production of phonon and charge signals based on particle events. This chapter will discuss the standard techniques used to extract event information (such as energy and position) from data. Following that, I will describe the potential of using deep learning algorithms to extract the same event information, and report preliminary results from my first analysis.

3.1.1 Optimal Filter

In order to extract signal from background, we employ an optimal filter (OF), which is essentially a fit of a template or pulse model to data. Using an optimal filter relies on a particular set of circumstances—firstly, the template describes the shape of the underlying signal. Second, the noise is a Gaussian random process with a known power spectral density (PSD)—details on how to calculate a noise PSD will be given in section 4.1. This means the signal takes the form of $S(t) = aA(t) + n(t)$, where A is the template and n is the noise (which can be characterized by the noise PSD $J(f)$), and we want to estimate the signal amplitude a. Even if these conditions are not completely upheld, optimal filters still perform well.

Because in general detector noise at different points in time is correlated (except for the special case of white noise, aka a flat PSD), fitting in the time domain is complicated. Instead optimal filtering is done in the frequency domain, where the noise is uncorrelated between frequencies. Thus the first step of an optimal filter is to take a Fourier transform of the signal and template. From now on a $\sim$ symbol above a variable will denote Fourier transforms, such that the transform of the template, for example, will be $\tilde{A}$ and so on.

Thus the χ-squared fit for the signal amplitude a is given by:

$$\chi^2(a) = \sum_n^N \frac{|\tilde{S}_n - \tilde{A}_n|^2}{J_n}. \tag{3.1}$$

This is a sum over the N independent frequencies of the data in its discrete Fourier transform. We can minimize this to get an estimator for a: [15]:

$$\hat{a} = \sum_n \frac{\tilde{A}_n^* \tilde{S}_n}{J_n} \Big/ \sum_n \frac{|\tilde{A}_n|^2}{J_n}. \tag{3.2}$$

This minimization relies on the frequency-domain noise being uncorrelated. This assumption is true for stationary noise, which has a probability distribution which is not a function of time. This estimator can be viewed as applying a Fourier domain filter to the data (seen in the numerator, as the denominator is independent of the signal):

$$\tilde{\phi}_n = \frac{\tilde{A}_n^*}{J_n} \tag{3.3}$$

The OF effectively is a frequency-domain fit of the pulse shape to the data in which frequencies with larger noise are deweighted relative to those with lower noise. If the previously specified conditions are met, this method gives the best possible resolution of a linear estimator calculable from the data.

3.2 Neural networks

A neural network (NN) is a machine learning algorithm which employs a layer hierarchy inspired by biological neural networks. Neural networks can be trained to learn a task without having a solution programmed directly.

There are many different types of NNs. Generally the choice of parameters in a neural network is referred to as its architecture. In the following sections we will go over the most basic kinds of architecture, which will help explain the operating principle behind NNs, and then section 3.2.2 will introduce convolutional NNs, which are the backbone of my analysis.

3.2.1 Feedforward neural networks

Feedforward neural networks (FNNs) are the simplest and most basic of neural networks, as well as being the first to be devised [32]. An FNN

consists of an input layer (the data to be studied), a number of hidden layers ranging anywhere from zero to infinity, and an output layer. Each layer has a certain number of "neurons"—as an example, if we were trying to develop a neural network to recognize an image with 800 pixels and determine whether the picture is a cat or a dog, our input layer would have 800 neurons (one for each pixel), each of which corresponded to an RGB value, and our output layer would have two neurons—one for "cat" and one for "dog". In between, we may have several hidden layers, also known as fully connected layers, and the number of neurons in each layer is a hyperparameter—an NN property defined before learning happens–which may affect the quality of the NN. In a fully connected network, every neuron in a layer is connected to every neuron in the next layer and the previous layer; thus, every input affects every subsequent output.

Neurons are connected by weights, similar to axons in a biological neuron. The value of any neuron is the sum of the values of all the previous neurons connected to it multiplied by the weights. As such, for the second layer of a neural network, the value stored in one of the neurons would be:

$$y_j = A\left(\sum_{i=0}^{n} w_{i,j}x_i + b_j\right) \tag{3.4}$$

Here y_j is the value of the j-th neuron in the second layer, x_i is the value of the i-th neuron in the input layer, $w_{i,j}$ is its attached weight, and b_j is a bias value added (outside of the sum).

In equation 3.4, A refers to the activation function. After the weights and inputs are multiplied and added, the final value of the neuron is calculated via an activation function which determines whether or not it fires. The activation function ensures non-linearity for the neural network and is the link between biological neural networks and artificial neural networks—in a biological system, the activation determines whether or not a neuron fires. Common activation functions include sigmoid, tanh, and ReLU functions shown in table 3.1.

FNNs repeat this process for additional layers, culminating in an output layer directly connected to every input via a complex neural structure. Figure 3.1 shows a basic layout of how the values propagate forward.

Training

In order for a neural network to learn, it requires a training set. This is a (preferably large) set of data for which the correct ("true") output values

Table 3.1: Common neural network activation functions.

Name	Functional Form
Sigmoid	$A(x) = \frac{1}{1+e^{-x}}$
tanh	$A(x) = \tanh x$
ReLU	$A(x) = x$ if $x > 0$; 0 otherwise

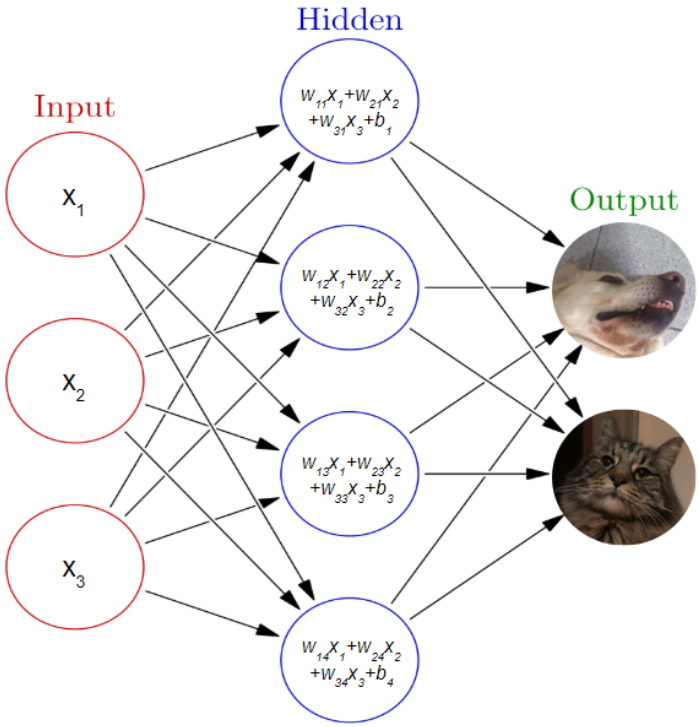

Figure 3.1: Visualization of a feedforward neural network with the values of the hidden layers' neurons written as per equation 3.4. Figure adapted from [33] with permission.

are known. At first neural networks will initialize all weight and bias values randomly, and then select a small batch of the training sample and calculate output values on it. In our previous example, the NN would take a set of, say, 100 out of the 10,000 pictures of cats and dogs we provided it, and for each of them it would produce an output value for the "cat" (C) and "dog" (D) neurons. If the C value is higher than the D value, it will determine the picture is, in fact, of a cat.

For each of those pictures, the neural network will compare its prediction with the true value via a loss function. The loss function is a quantification of wrongness. The most simple loss function is the mean squared error (MSE):

$$MSE = \frac{1}{N}\sum_{n=1}^{N}(Y_{p,n} - Y_{t,n})^2 \tag{3.5}$$

Here N is the number of pictures being trained on, Y_p is the predicted output, and Y_t is the true output. Thus the neural network's aim is to minimize the loss function by altering its weights and bias values attempting to find a loss minimum. For a picture which is indeed a cat, the true C value should be 1 and the true D value should be 0. By altering itself slightly, the neural network can achieve an improved performance on the first batch of data, then move on to the second batch, and so on until the loss function reaches a minimum.

The most elementary mathematical mechanism for minimizing the loss function is referred to as gradient descent. By calculating the gradient of the loss function with respect to the weights and biases, the neural network can find the direction of maximum change towards the minimum. If we refer to $\vec{w}$ as a vector containing all the weights in the network, the gradient descent calculation after the l-th training iteration ("step") yields:

$$\vec{w}_{l+1} = \vec{w}_l - \gamma\frac{1}{N}\sum_{i=1}^{N}\nabla_w\mathcal{L}(x_i, \vec{w}_l) \tag{3.6}$$

Here N is the number of events in the current batch the neural network is training on, $\mathcal{L}(x_i, w_l)$ is the loss as a function of the weight vector and all the data points x_i, and γ is a gain function which determines the maximal amount a neural network's weight can be changed per training step.

This algorithm is computationally inefficient. Modern neural networks use more efficient algorithms such as stochastic gradient descent (SGD) which, instead of calculating the gradient on the entire data set, picks one

data point at random and calculates the gradient for that. Although this procedure introduces noise [34], the hope is that SGD behaves like equation 3.6. The algorithms used to calculate the next iteration of weights on a neural network are referred to as optimizers.

Overtraining

Selecting and preparing the training sample is a crucial element of neural network analysis. If the image recognition neural network described previously was trained on pictures of exclusively black cats, and exclusively white dogs, upon seeing a black dog it would probably mislabel it as a cat. This is referred to as overtraining.

There is a wide variety of methods available to prevent overtraining. This is referred to as regularization. I list some below [35]:

- The γ in equation 3.6, referred to as the learning rate, is a tunable hyperparameter which limits the size of the steps taken in training.
- Similarly, it is possible to introduce what is known as "weight decay." This means the learning rate parameter gets smaller after each training step. The idea is that for early training steps, neural networks are random and not likely to overtrain, but in later iterations as the loss function approaches a minimum it is better to have a small learning rate to avoid overtraining.
- Often NNs will have a training set and a validation set of data. After each training step, the NN can check its performance on the validation set via a loss function calculation. It can then stop training if the loss function is changing too little (i.e. less than a predetermined cutoff value), or if it has gone up. This is referred to as early stopping.
- Penalizing neurons with large weights in the loss function is known as weight constraining. It avoids overfitting on a single parameter and ensures more simplicity in the NN.
- Adding a small amount of statistical noise (such as Gaussian noise) can prevent overfitting.
- Randomly removing neurons from the network during the training steps (then inserting them back in), a technique known as dropout, helps avoid overtraining.

Applying regularization is a crucial element of any NN work, and in my studies I attempted all of the regularization techniques listed above, obtaining the most success with a combination of weight decay and early stopping.

3.2.2 Convolutional neural networks

A convolutional neural network (CNN) is a class of NN which is highly popular and successful in image recognition and analysis. CNNs are efficient at distinguishing features of short length within an overall data set, where the feature's position is not of high relevance. The FNN described in the section above is not translation-invariant—the location of features is something the network learns, whereas CNNs do not. The earlier example for an FNN would actually be terrible at image recognition, since it would attempt to learn not only what cats and dogs look like, but where they are in the picture as well. Therefore for time-domain particle data where features will have a time shift, a CNN is a potentially useful tool for event reconstruction: hence the idea of applying CNNs to CDMS data.

Operating Principle

The backbone of the CNN is the convolutional layer. A convolutional layer consists of a set of convolution kernels, or filters, which scan through the data and pick out its features. Kernels take advantage of the fact that in an image, data points close to each other are generally more strongly related than distant ones [36].

A kernel is a matrix. Its values are the weights in the neural network—they are optimized during training. As an example, imagine a two-dimensional data set of 4x4 matrices. We choose a kernel size—in this case, let's say 3x3—and the neural network overlays the kernel along the 4x4 matrix, multiplying every element in the subsection of the data matrix by every element in the kernel, resulting in a new, smaller matrix of dimensions 2x2. An example is shown below, using an asterisk to represent the convolution, and not multiplication:

$$\textit{Convolution} \rightarrow \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 2 & 2 & 2 \\ 3 & 2 & 3 & 2 \\ 4 & 3 & 2 & 1 \end{bmatrix} * \begin{pmatrix} 1 & 0 & 1 \\ 0 & -1 & 1 \\ 3 & 1 & 0 \end{pmatrix}$$

$$\text{1st element} \rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 2 & 2 & 2 \\ 3 & 2 & 3 \end{bmatrix} * \begin{pmatrix} 1 & 0 & 1 \\ 0 & -1 & 1 \\ 3 & 1 & 0 \end{pmatrix} = 15$$

$$\text{Final result} \rightarrow \begin{bmatrix} 15 & 15 \\ 20 & 14 \end{bmatrix}$$

A more visual representation is shown in figure 3.2.

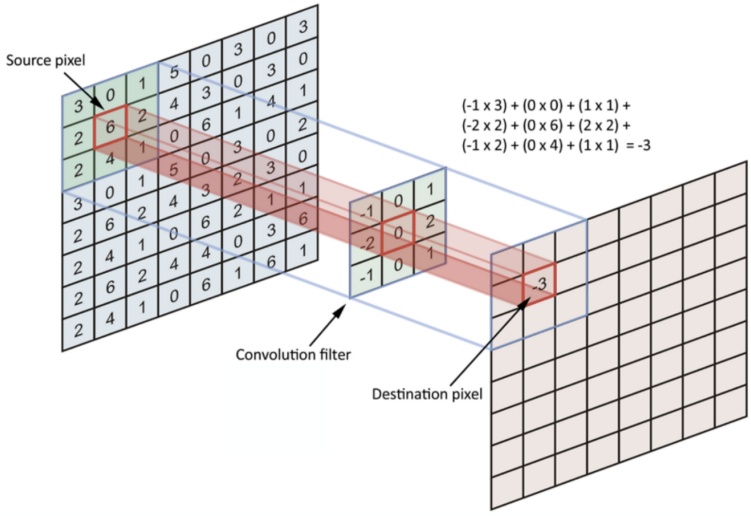

Figure 3.2: A visual representation of a single convolution filter being applied to an image. Taken from [36] with permission.

Note that this is not matrix multiplication—elements in one matrix are multiplied by their respective elements in the other matrix. Each layer in a CNN may have several filters, and their sizes are hyperparameters in the network.

Applying a filter to the image generates the neural network's second layer, which is referred to as a feature map and is an image with reduced dimensionality. Each pixel in the feature map is then put through an activation function which will select out the features in the data. However, as stated

previously, we might have multiple filters in the same layer, which means we will have multiple feature maps (all with the same size). This means after a convolution, a 2D image becomes a 3D feature map (a set of 2D feature maps, one for each filter). See figure 3.3 for a more visual representation. At this point it is possible to apply more convolutional layers, extracting more features and making them more abstract. But in order to produce an output from a CNN, we require what is known as a pooling layer. A pooling layer combines values across several feature maps, either by averaging or, more commonly, selecting the maximum value for each point, which allows us to then introduce a fully-connected layer of FNN-type neurons, which produce an output layer.

3.3 First results

This section describes my preliminary results obtained in applying CNNs to CDMS data.

3.3.1 CDMSlite Simulated Data

To run a neural network on particle detector data, it is necessary to have a set of data with the true event properties known. This can be either a set of simulated data that closely resembles the particle events in the detector, or a set of real data taken from sources with known energy distributions.

SuperCDMS ran a previous iteration of the experiment at the Soudan Underground Laboratory in Minnesota. One of the operating modes used at Soudan was the CDMS low ionization threshold experiment (CDMSlite), which reduces the experiment's energy threshold and increases sensitivity to low-mass WIMPs [26]. For the CDMSlite data runs, the SuperCDMS simulations group has produced 100,000 Monte Carlo simulated WIMP events with energies between 100 and 3500 eV. I chose to use the simulated data for the ease in having the true energy and position of the events.

Properties of the Dataset

From the 100,000 simulated events, I randomly sampled 39,280 to create a dataset. I split those events up into a training set (25,000 events), a validation set (5,000 events), and a testing set (9,280 events). Each event consists of 8 time-domain pulses (one for each phonon channel in the simulated detector) with 4,096 bins each. Figure 1.8 shows a CDMSlite phonon pulse before the preprocessing described in the next section.

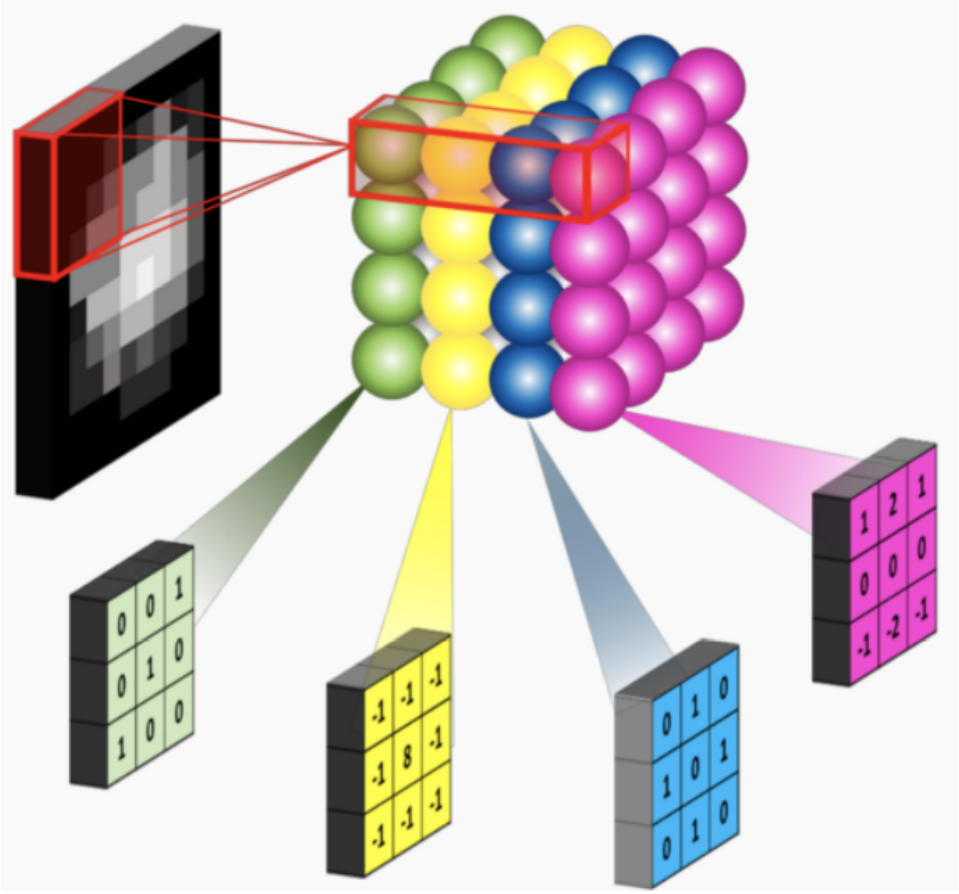

Figure 3.3: Four filters are applied to a 2D image. Each filter produces a 2D feature map, and the feature maps are overlayed to produce a 3D map. A max pooling layer would select the highest value across the feature maps (in the z axis) to reduce the dimensionality and produce outputs. Adapted from [36] with permission.

Preprocessing

In order to reduce computation time and increase efficacy, I made a few preprocessing decisions. Firstly, the first 500 elements of each pulse are removed since the CDMSlite simulated pulses all start at the same time and the first 500 points are just baseline. Secondly, the dataset is downsampled by only keeping every fourth data point. Thirdly, the events are inverted so that the pulse is a peak instead of a trough. Additionally, all the energies are scaled down linearly so that the values lie between 0 and 1, keeping the shape of the distribution the same. Lastly, some of the events happen close to the detector edge. These produce fewer phonons and result in smaller amplitude pulses which seem like lower energy events. For simplicity, these events were excluded. Figure 3.4 shows a sample phonon pulse and the energy distribution after the preprocessing.

After all the cuts, the training sample now has 8805 events, the validation sample has 1775 events, and the test sample has 3376 events.

Network architecture

For the preliminary tests, I used an exceedingly simple CNN architecture. After testing several variations, the most successful neural network had only a single one-dimensional convolutional layer, followed by a pooling layer, and lastly a fully connected layer.

The convolutional layer has 8 filters, each with size 1x899. Our input "image" is an 8x899 set of data points. I then apply 8 filters, each producing an 8x1 feature map. It is helpful to think of this feature map as being in the x-y plane. After the convolutional layer, we have an 8x1x8 (x-y-z) feature cube. With a more layered network architecture it is possible to achieve better results for more complex datasets; however, it is generally good practice to keep the architecture as simple as possible. The pooling layer reduces the dimensionality to 8x1 by selecting the maxima of each of the feature maps (across the z-axis), and the final fully-connected layer has one neuron only, since this network aims to only predict the energy of each event. Initially I had networks with four neurons, to predict the energy and each of the positional variables, but I reduced this to energy only for computational efficiency and simplicity.

Results

Figure 3.5 shows the absolute and fractional errors for the first energy predictions on CDMSlite data. As the figure shows, the CNN was able to

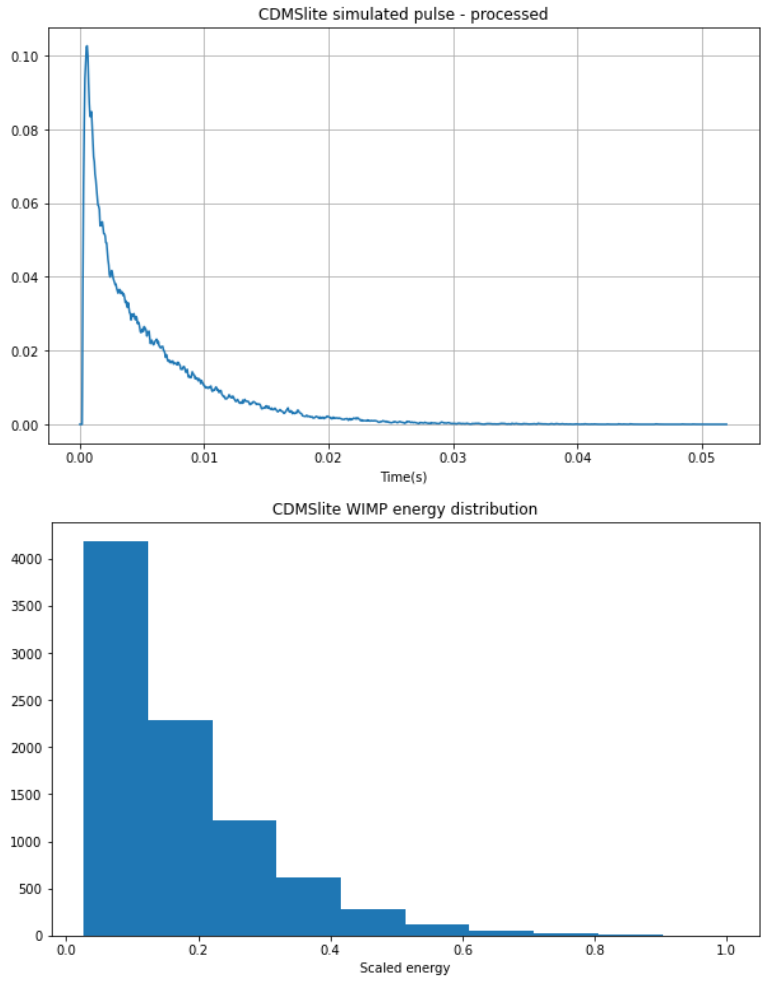

Figure 3.4: Top: sample simulated pulse after pre-processing. Bottom: energy histogram after rescaling to keep values between 0 and 1.

adequately learn the pulse shapes and predict the energy to within 10% error for a vast majority of cases. One feature of note is that for the fractional error in figure 3.5, there is a tail on the positive side of the graph. This feature seemed curious so upon further inspection, I discovered that these were the bottom-end (lowest energy) events. Upon closer inspection, the neural network was overpredicting low energy events, and underpredicting high-energy events to get closer (i.e. predicting around the mean). Prediction around the mean is somewhat expected since this neural network utilizes the MSE loss function. Figure 3.6 shows a histogram comparison between the true energy distribution and the predicted energy distribution. The visible discrepancy at the low end shows overpredicted low-energy events, and there is a less-obvious discrepancy for events above the mean as well—they are underpredicted up until the highest-energy events, which are again overpredicted. This is corroborated by figure 3.7, which shows the absolute and fractional errors as functions of the true energy of each event.

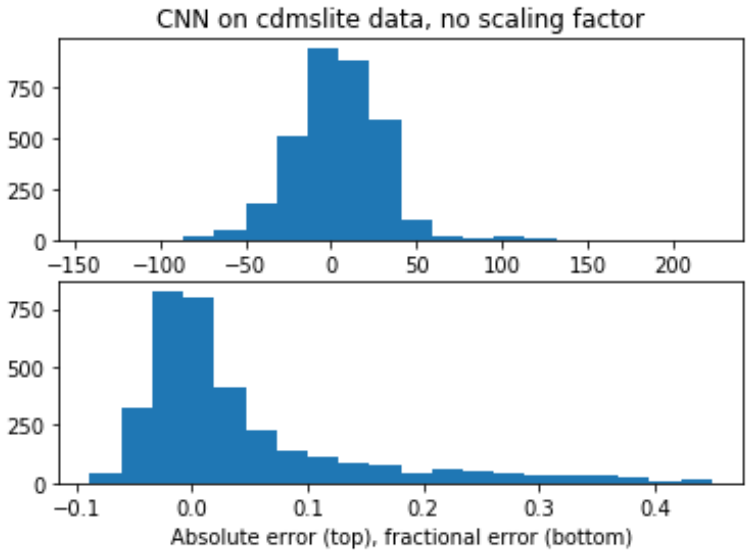

Figure 3.5: Absolute error $E_{\text{pred}} - E_{\text{true}}$ vs fractional error $(E_{\text{pred}} - E_{\text{true}})/E_{\text{true}}$ for the first iteration of tests with CDMSlite data. The top axis is in units of eV.

Lastly, to produce more meaningful statistics I calculated the RMS and fractional RMS for the predicted energies with five different energy bins. These are shown in figure 3.8. The calculations are as follows:

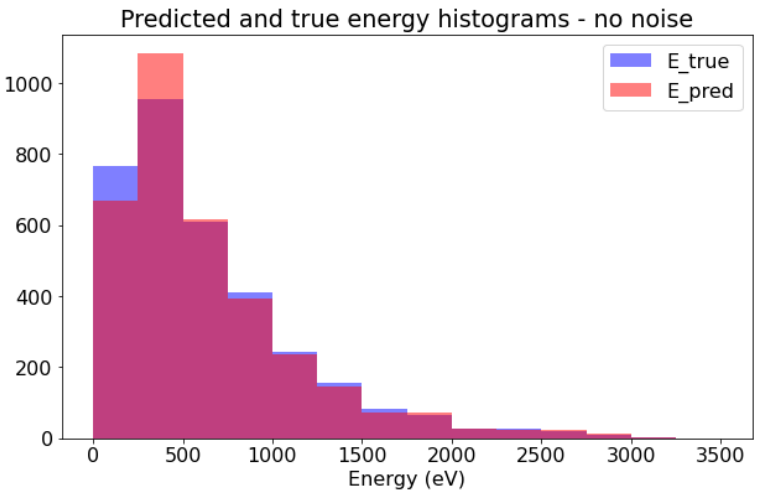

Figure 3.6: Overlaid histograms of predicted energy and true energy for the first iteration of applying CNNs on CDMSlite data. The discrepancy for low-energy events shows a tendency to predict closer to the mean, as low-energy events are being overpredicted.

$$\text{rms}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(Y_{P,i} - Y_{T,i})^2 \tag{3.7}$$

$$\text{fractional rms}^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(\frac{Y_{P,i} - Y_{T,i}}{Y_{T,i}}\right)^2 \tag{3.8}$$

Here N is the total number of events in each bin, and Y_P and Y_T are the predicted and true energies respectively. The error bars are calculated by dividing each of the RMS or fractional RMS values by $\sqrt{2N-2}$.

This preliminary study was promising. An exceedingly simple CNN was able to predict the energy well, to within 10% error for a majority of events. In the next chapter, I describe how I added noise to the CDMSlite data and tested the network's robustness under changing noise models.

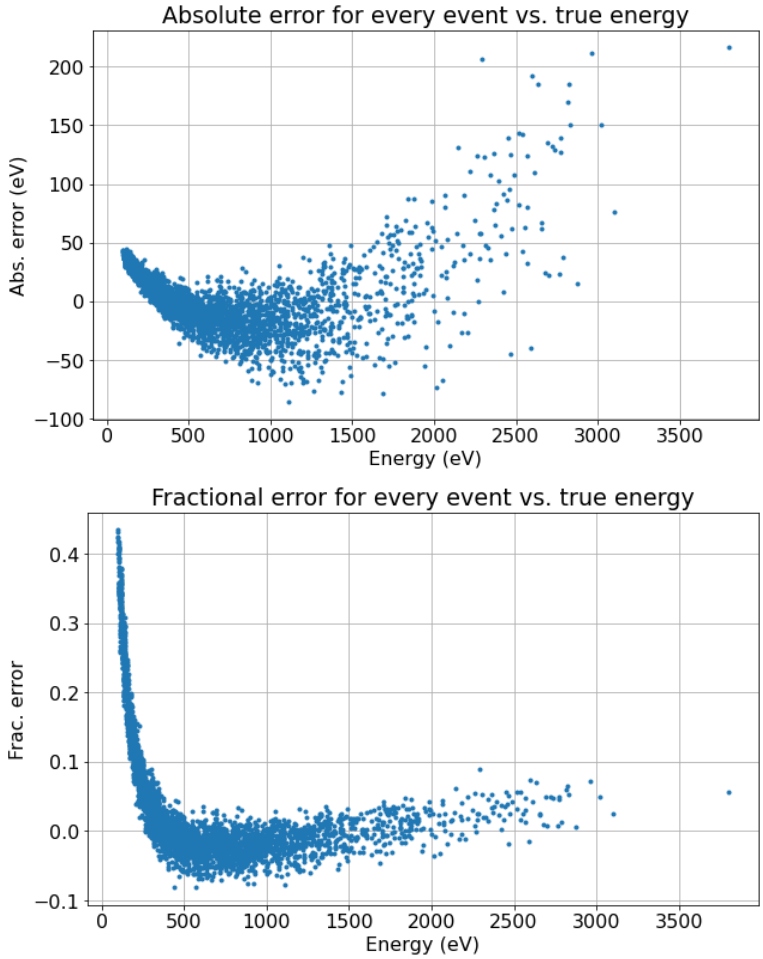

Figure 3.7: Plots of the absolute error and fractional error as functions of the true energy, which show the low-energy events being overpredicted. The events immediately above the mean are underpredicted, but the highest-energy events are overpredicted.

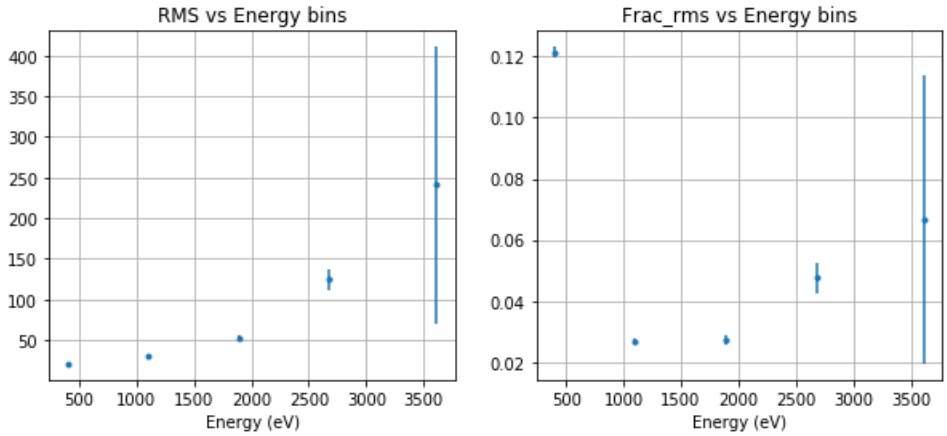

Figure 3.8: Absolute and fractional rms calculated for testing data, split into five energy bins. Values and error bars calculated by equations 3.7 and 3.8. The large error bars in the rightmost bin is due to poor statistics (few events) for that bin.

Chapter 4

Adding noise to the Equation

One issue with applying a neural network approach to SuperCDMS is that our detectors' noise profile is not constant. This begs the question—would a neural network trained on one noise profile be able to still robustly predict on data with a different noise profile? In this chapter, I first test the accuracy of neural network predictions for different types of noise in training and testing.

4.1 Noise PSDs

I generated noise for the simulated CDMSlite data using a noise power spectral density (PSD). A noise PSD describes how the power of the noise in a series is distributed over different frequencies; for example, in the presence of noise from an alternating current source, a noise PSD might see a peak at the 60 Hz noise level. A PSD is generally given in units of V^2/Hz, but throughout the rest of this chapter all the PSDs have been normalized so values lie between 0 and 1, for simplicity.

4.1.1 Generating Noise from a PSD

In order to generate noise from a PSD I use the NoiseSim package which was developed for SuperCDMS by the Toback group at Texas A&M [37].

The method is quite simple. The PSD is in frequency space. Each time we wish to generate a time-domain noise series, we generate a complex coefficient for each frequency in the PSD by drawing values for the real and imaginary parts of the coefficient from a Gaussian with mean zero and variance equal to the PSD's value; thus, frequencies with higher values in the PSD will have higher likelihood of a greater coefficient. This also incorporates randomness into the process—otherwise noise subtraction would be trivial. Then, we apply a reverse discrete Fourier transform to this list of coefficients to generate a time-domain noise trace. Then, we add that noise trace to the noiseless signal template.

4.2 New Data

I generated three new data sets using the routine described in section 4.1.1, using a single PSD scaled by different values. This PSD was drawn from a test run done for one of the CDMS test detectors. I have a "high" noise dataset, a "mid" noise dataset (PSD values divided by 3), and a "low" noise dataset (PSD values divided by 5). The dataset used in the previous chapter is referred to as the "no" noise dataset (and its PSD is simply an array of zeros). The PSDs are shown in figure 4.1.

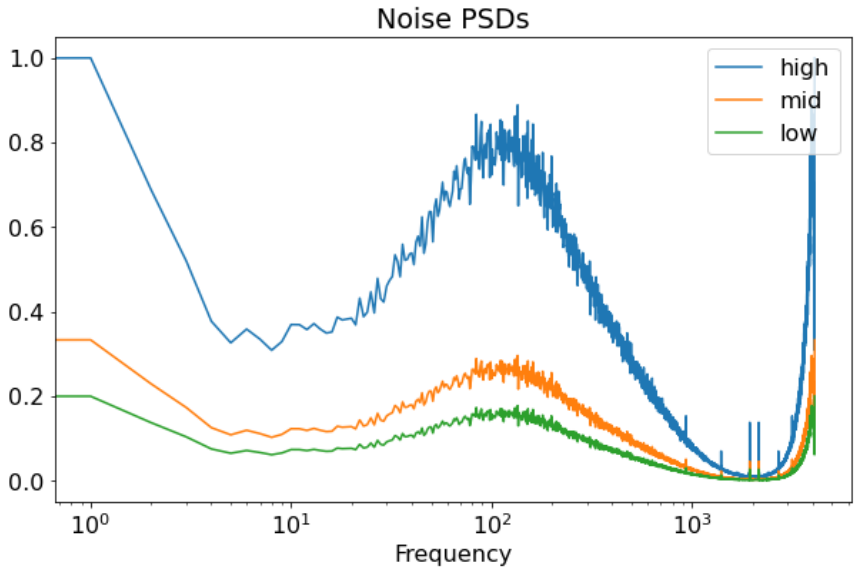

Figure 4.1: Noise PSDs for the new datasets.

Figure 4.2 shows sample pulses from the new datasets. With this new data, I first applied the same neural network architecture from the previous chapter. I trained, however, three iterations of the neural network:

- CNN trained on high noise;
- CNN trained on low noise;
- CNN trained on no noise (same as previous chapter's).

I did not train a CNN on mid level noise in order to do a future test of how a CNN trained simultaneously on all noise levels would react to a noise level it had never seen before; more on that in section 4.3. Each of

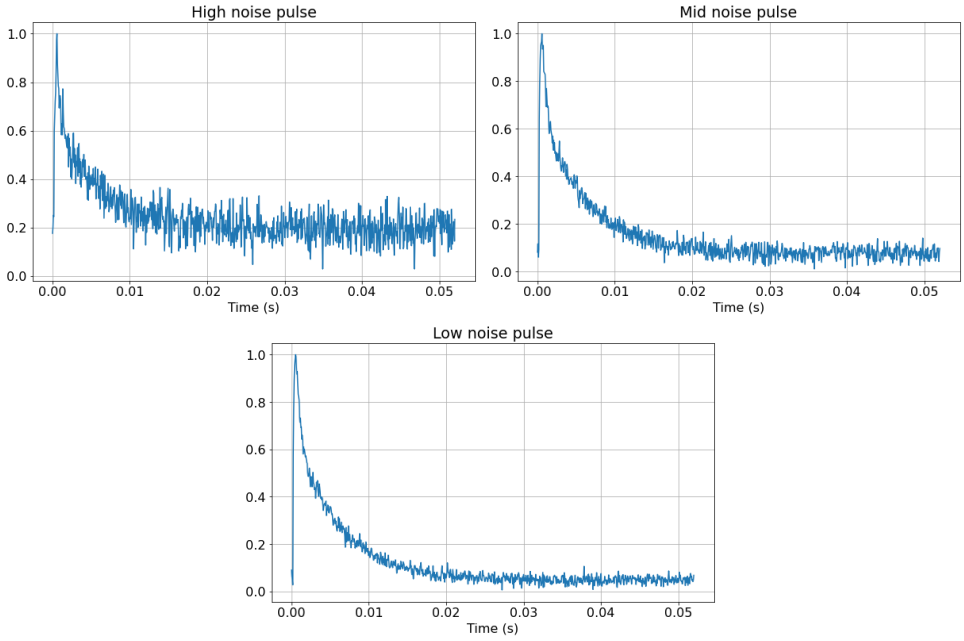

Figure 4.2: Example of pulses from the high noise, mid noise, and low noise datasets.

these networks was then tested on all other noise levels. This allowed me to produce nine different plots comparing the performance of each neural network (high, low, no noise) on each data set (high, low, no noise).

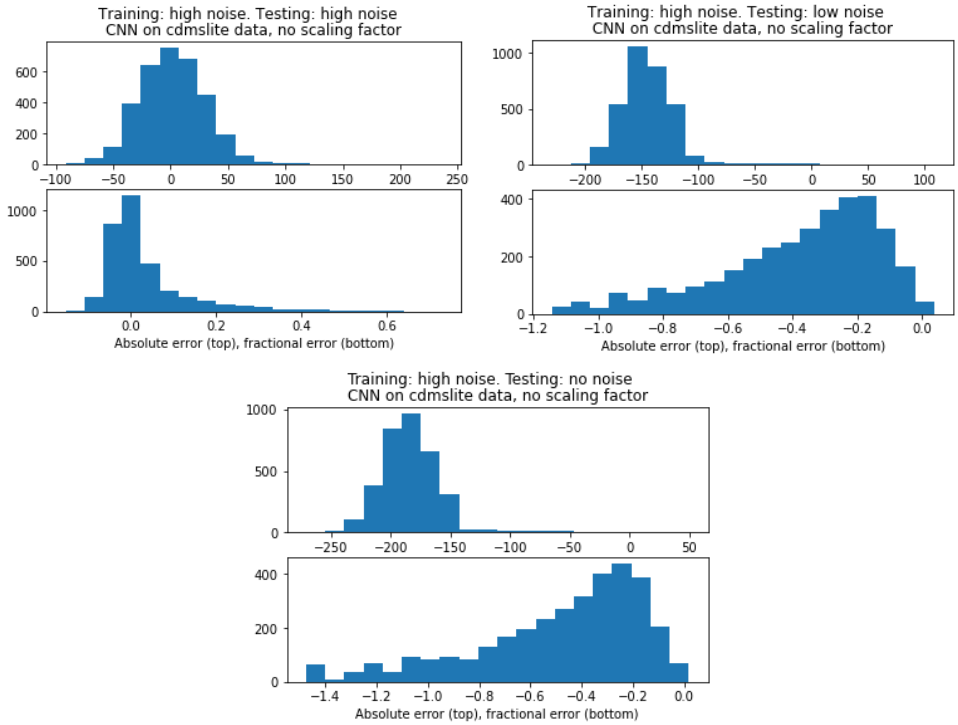

Figure 4.3: Absolute and fractional error histograms for neural network trained on high noise. When testing the high-noise network on low- or no-noise data, it consistently underpredicts the energy.

While increasing the amount of noise did not significantly affect the performance of the neural network when it is tested on the same type of data that it was trained on, it becomes quite hopeless when the noise model changes. The simple change of noise PSD amplitude threw the network's predictions off-center and increased their variance as well (shown in figures 4.3, 4.4, and 4.5, and summarized in table 4.1).

One of the difficulties with using neural networks is understanding the underlying logic of what is happening. My hypothesis to explain why the under- and over-predictions happens is that to first-order, this exceedingly simple neural network learns to correlate the height of the highest peak with the energy of the event. With noise, fluctuations increase the value

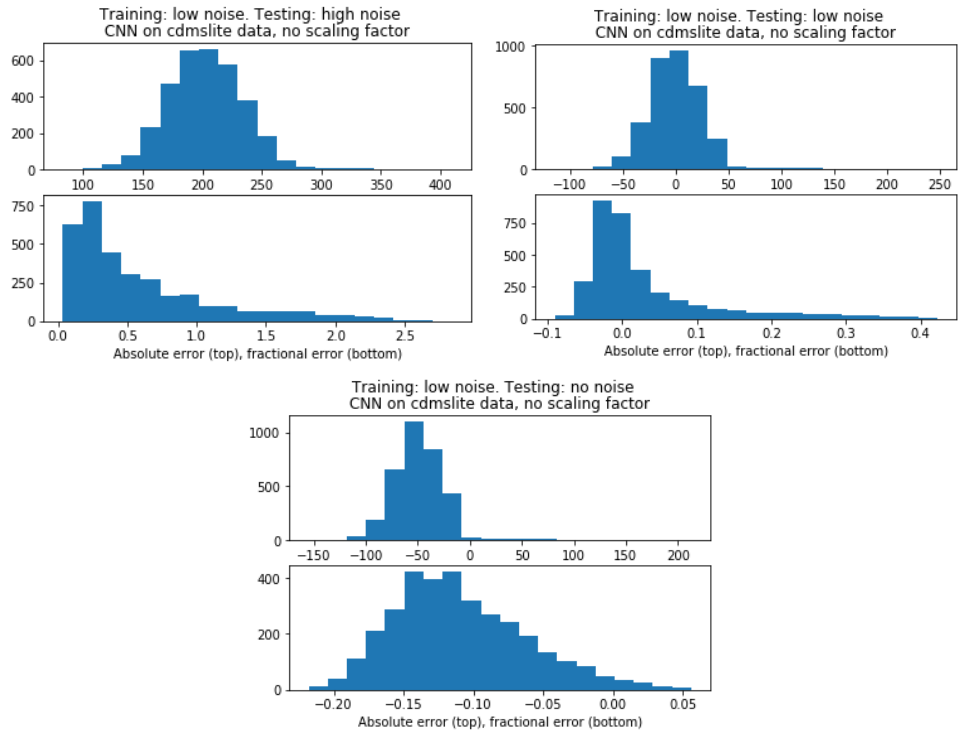

Figure 4.4: Absolute and fractional error histograms for neural network trained on low noise. The low-noise network underpredicts energies on no-noise data and overpredicts energies on high-noise data.

Average Fractional bias / Fractional RMS Calculations

	No Noise Test	Low Noise Test	High Noise Test
No-noise network	0.03/0.06	0.08/0.10	0.28/0.32
Low-noise network	-0.04/0.07	0.02/0.04	0.29/0.34
High-noise network	-0.17/0.20	-0.14/0.16	0.03/0.06

Table 4.1: Calculations for average fractional RMS and average fractional bias for each neural network tested on each dataset. There is a significant increase in the RMS and bias when a neural network encounters data with a noise model it has not seen before.

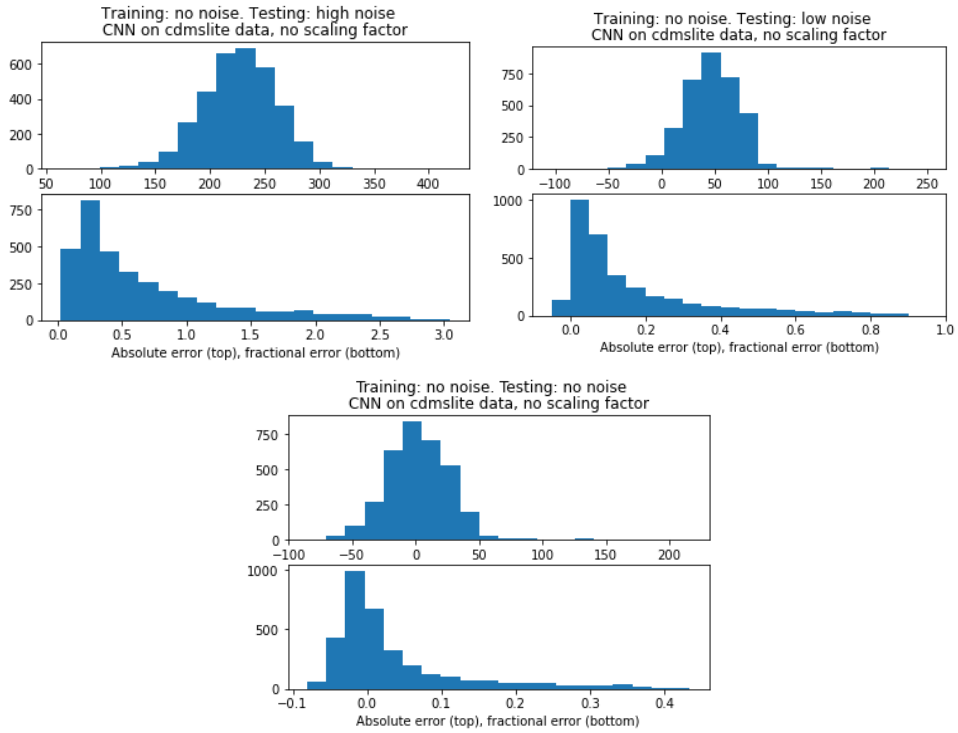

Figure 4.5: Absolute and fractional error histograms for neural network trained on no noise. The no-noise network overpredicts energies when tested on noisy data.

of the peak amplitude, and the neural network fails to adjust for these fluctuations and understands this event as a higher-amplitude event. In general, increasing the complexity of the neural network should work to reduce this reliance on the peak amplitude. However, before increasing the layered complexity of the neural network, I attempted to include the noise PSD into the training to determine whether the CNN was able to learn to distinguish between different types of noise.

4.3 Training with the PSD

I created a merged dataset consisting of the pulses from the high, low, and no-noise datasets. However, this time, each event came with its noise PSD attached. I then created a new neural network architecture to accept the PSD as a separate input from the event. The new network contains two separate convolutional layers, one of which operates on the traces, and one of which operates on the PSD. The layers are then added together, and attached to a pooling layer followed by a fully-connected layer which outputs the energy. A diagram of the network architecture is shown in Figure 4.6.

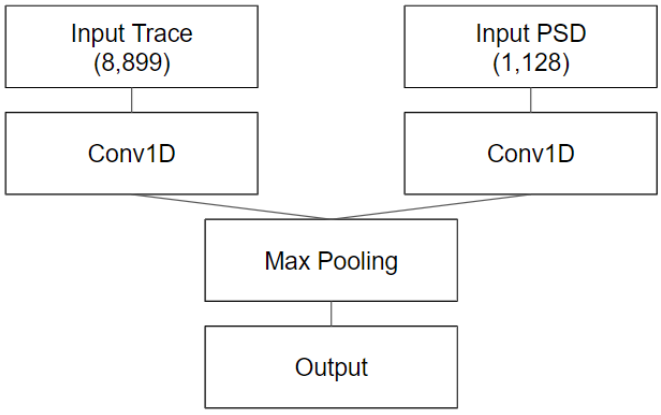

Figure 4.6: Schematic diagram of a CNN which includes the noise PSD as an input layer.

4.3.1 Results

Including the PSD in the training regime yielded far better results. The neural network correctly accounted for the changing noise models, and was

able to correct the over-prediction/under-prediction issue from the previous iteration. Results are shown in Figure 4.7.

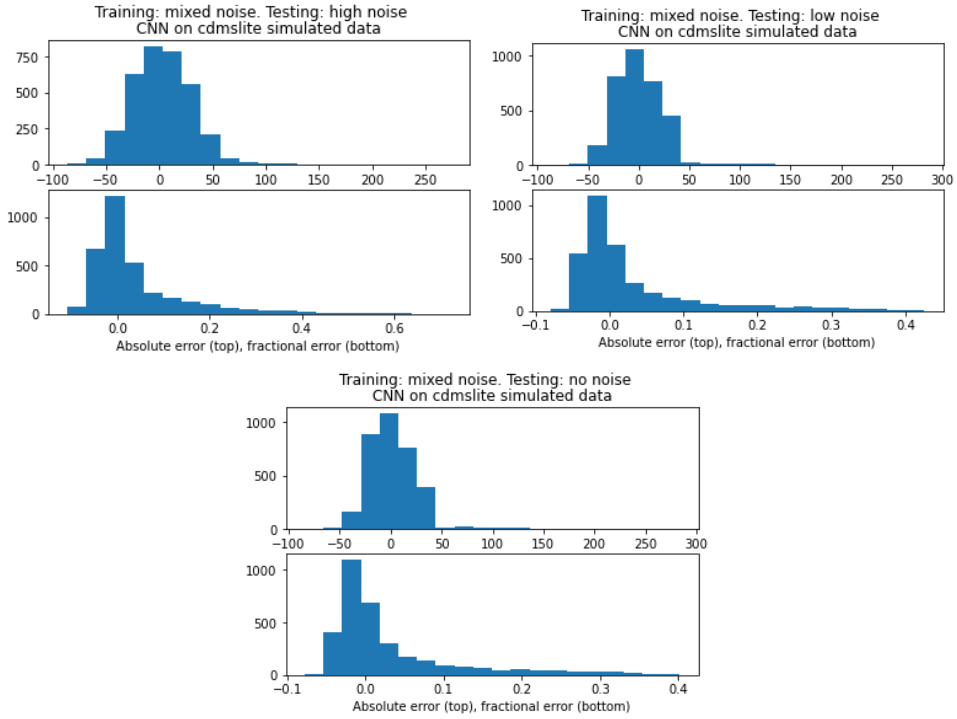

Figure 4.7: Absolute and fractional error histograms for neural network trained on different types of noise, with the inclusion of the noise PSD in the training.

I previously described that there was another dataset, trained on "mid" noise, which I hadn't used up to this point. This dataset was made with the intention to test this neural network architecture with a noise model it had never seen before—whether or not it would be able to understand a new noise level. The results are shown in figure 4.8.

This new iteration of neural network was able to parse through the noise even for a noise level it was not trained on.

Having a higher noise level on a dataset still results in a higher spread in the predictions, however, shown in Table 4.2. The RMS of the absolute error increases as the noise increases. The mean error also increases, but it is actually lowest for the low-noise predictions. Additionally, comparing these results to table 4.1 shows a vast improvement in the reconstruction

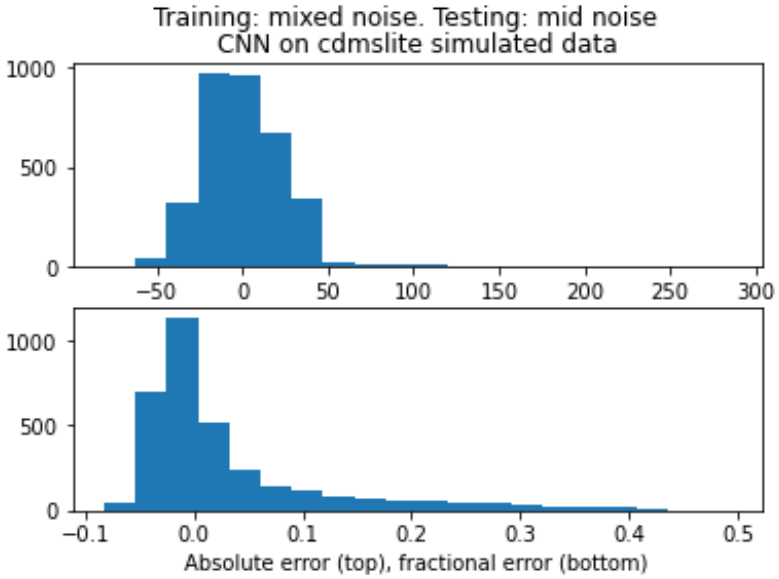

Figure 4.8: Results of predicting on mid-level noise, which the neural network was not trained on.

Noise Level	Mean Error (eV)	RMS	Frac. Bias	Frac. RMS
High	3.4	28.6	-0.004	0.033
Mid	0.9	25.4	-0.002	0.026
Low	0.8	25.0	-0.002	0.026
No	1.9	24.5	0.000	0.025

Table 4.2: Mean absolute error, RMS, fractional bias, and fractional RMS for each noise level.

by including multiple types of noise in the training. The next section will delve into the RMS calculations for these results to show how the noise level affects the reconstruction overall, and the shortcomings of this method.

4.4 RMS measurements and biases

For the CNN trained on a mixed dataset, with noise PSDs as part of the input, I split up the predicted data for each noise level into bins based on true energy, and then calculated the RMS and fractional RMS (as per equations 3.7 and 3.8) per energy bin.

Figures 4.7 and 4.8 show the same tail on the fractional RMS described in section 3.3.1, which was caused by low-energy events and high-energy events being overpredicted (and "middle" energy events beign underpredicted). As such, for each dataset I also calculated a bias $E_{\text{pred}} - E_{\text{true}}$ and a fractional bias $(E_P - E_T)/E_T$ for each event, and found the average of all the events in the bin. These are shown in figures 4.9, 4.10, 4.11, and 4.12.

The first artefact to be noticed from this method is the huge error bar in the RMS measurement for the highest-energy bin—this is due, as before, to low statistics (the bins were made to be equidistant in energy).

Looking at how the RMS changes between the different datasets, it becomes clear that there is indeed an increase in the RMS for the higher-noise datasets. This is to be expected. Interestingly, however, there seems to be virtually no difference between the low noise and no-noise datasets—I've plotted the RMS for those two side-by-side to showcase this in figure 4.13. This is consistent with the idea presented in section 3.2.1 that a small amount of noise can help improve reconstruction and prevent overtraining.

The bias plots show that there is still a positive bias away from the 1000-1500 eV range where events are being, on average, overpredicted, and underpredicted otherwise. Fractionally, my neural network is overpredicting low-energy and high-energy events the most. To investigate this, I plotted the value of the highest amplitude in the traces for each event (without noise) vs the true energy, and then made a density colormap to show where a high density of events exists. This is shown in figure 4.14. As stated previously, to first order the peak amplitude is a reasonable energy estimator; looking at the colormap, then, we can see that the scatter in the peak amplitude increases as the energy grows. This is to be expected. Additionally, there are more outlier events with peak amplitude much higher than the trend as the energy gets higher; this explains somewhat why the higher-energy events are being overpredicted. Increasing the complexity of the neural

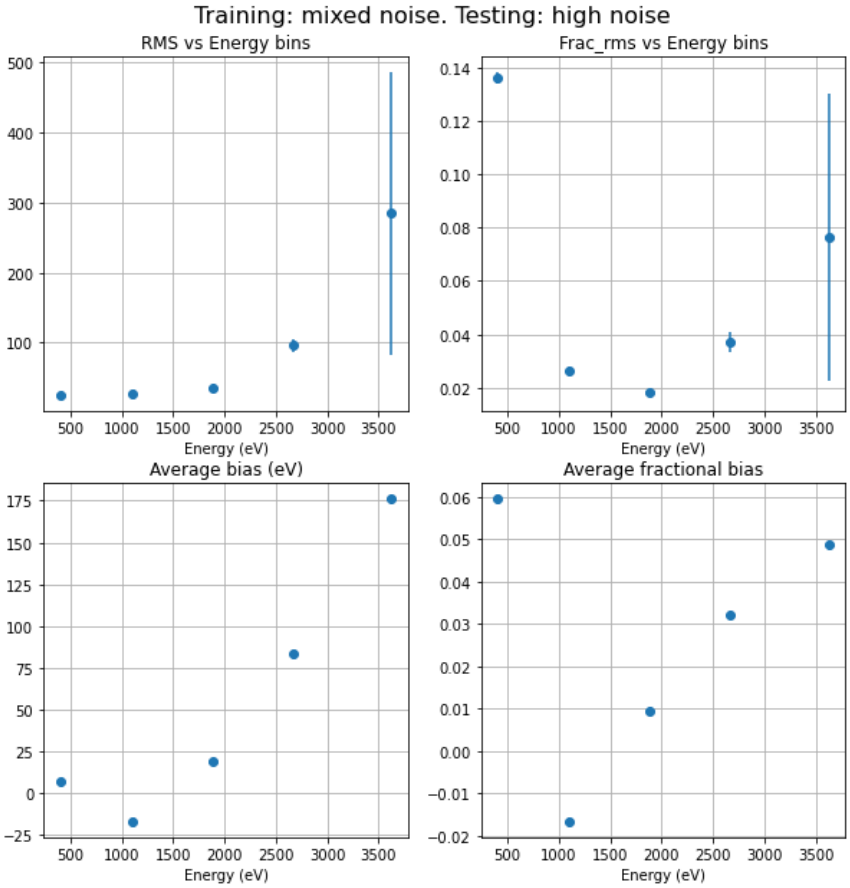

Figure 4.9: RMS, fractional RMS, average bias, and average fractional bias per energy bin for CNN predicting on data with high-level noise.

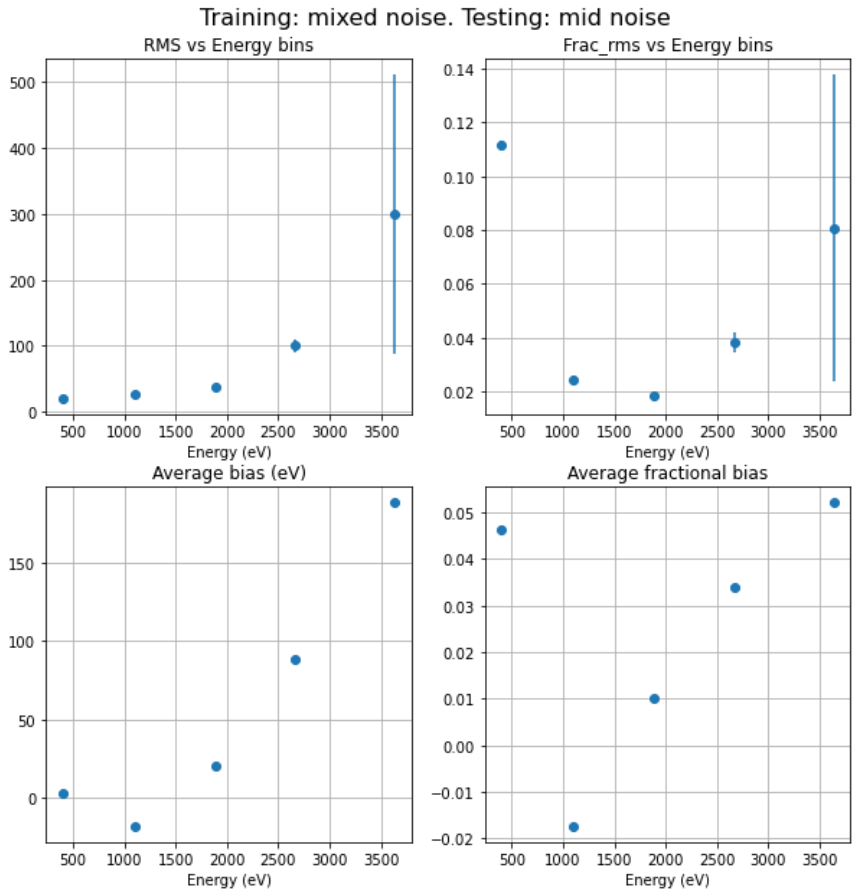

Figure 4.10: RMS, fractional RMS, average bias, and average fractional bias per energy bin for CNN predicting on data with mid-level noise.

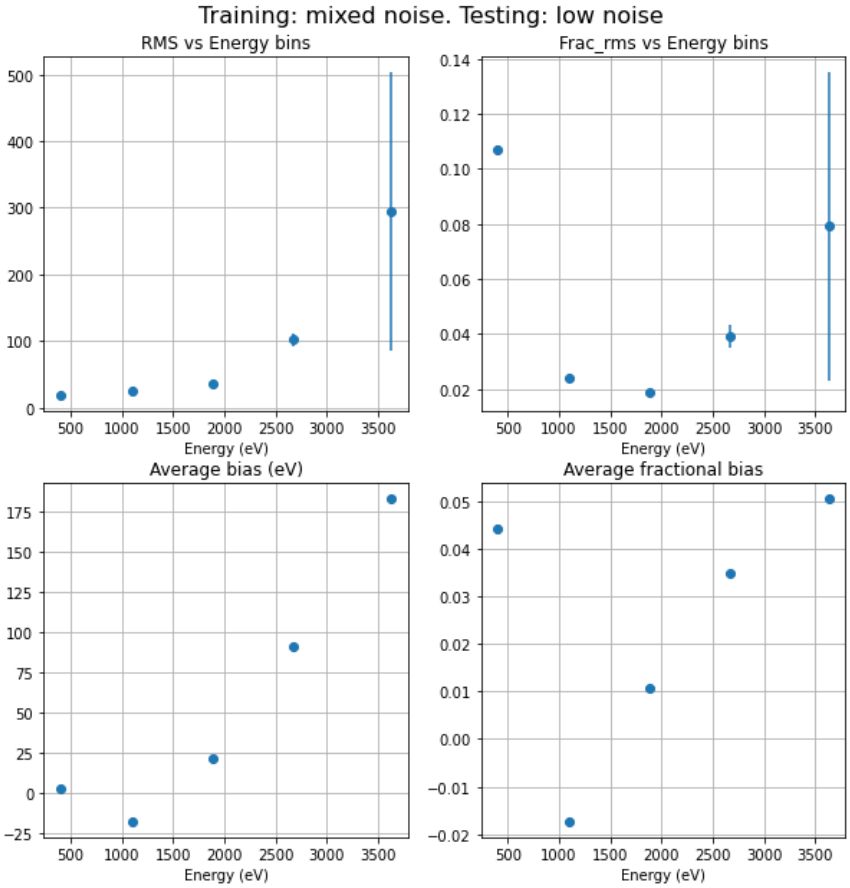

Figure 4.11: RMS, fractional RMS, average bias, and average fractional bias per energy bin for CNN predicting on data with low-level noise.

network architecture, and increasing the number of high-energy events in the training sample, would probably help to mitigate this bias. Additionally, increasing the duration of training would probably help, but it can also incur overtraining.

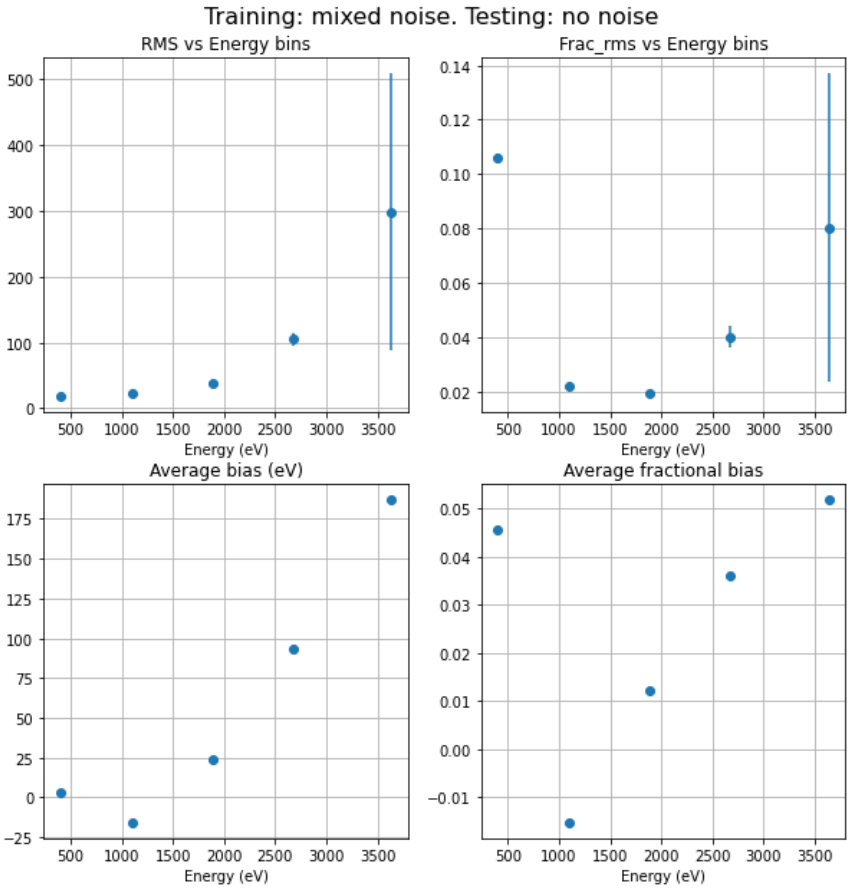

Figure 4.12: RMS, fractional RMS, average bias, and average fractional bias per energy bin for CNN predicting on data without noise.

To investigate the aforementioned peak amplitude effect, I went back to the CNN trained on low noise which was previously used in section 4.2, and had it predict on a random sample of 1000 events from the low noise data (I reduced the size of the dataset for expediency).

For 50% of those events (at random), I increased the value of the highest points in each of the 8 phonon channel traces in the data by a random value between 0 and 0.25 (the trace values are normalized to between 0 and 1).

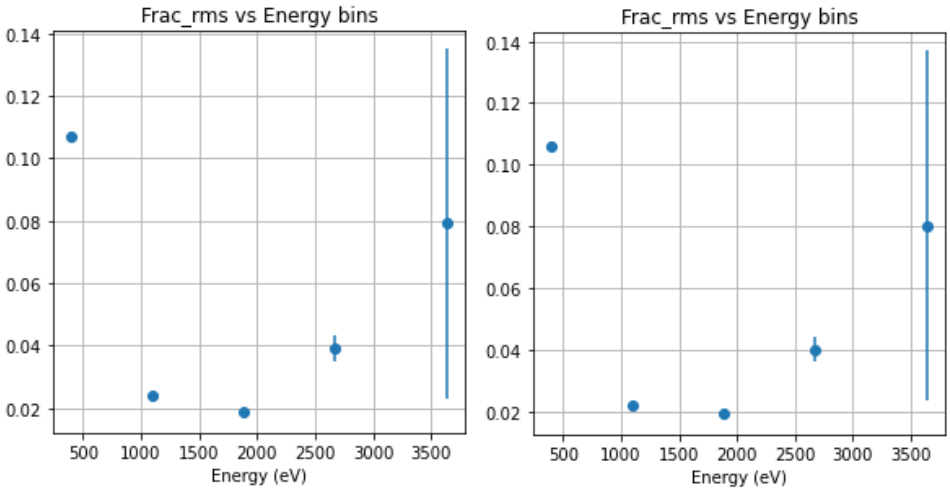

Figure 4.13: Side-by-side comparison of the binned fractional RMS for predictions on low noise (left) and no noise (right) data.

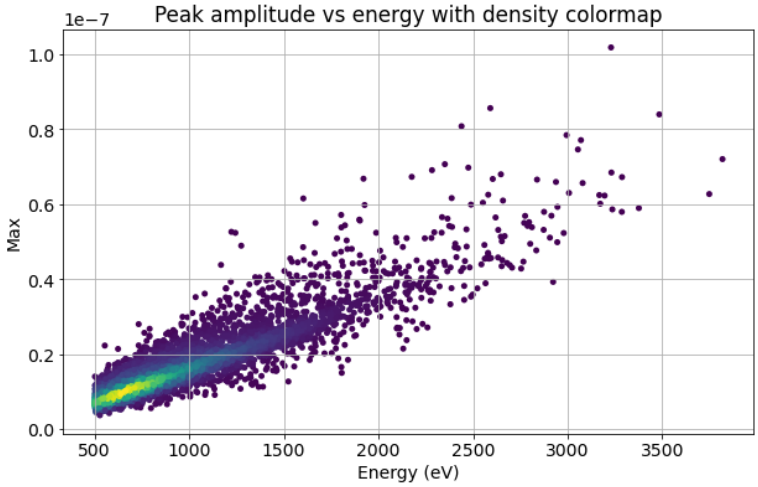

Figure 4.14: Density colormap of the peak amplitude value in the trace vs the true energy in eV. Warmer colors show where the density of points is higher.

This new dataset, then, contains events with increased peak amplitudes.

I then used the low-noise CNN to make energy predictions on this set of events, both before adding to the peaks and after adding to the peaks. From these predictions, I removed the events that had not been changed (since the difference in the predictions is 0), then subtracted the energy predictions to find a peak addition bias effect. The results of this study are shown in figure 4.15. Increasing the peak values has for the most part increased the energy prediction, but surprisingly, the neural network predicted a lower value for the energy of other events. This is not a negligible effect—of the 503 events which had their peaks modified, 124 ended up with a lower-energy prediction compared to the unmodified event.

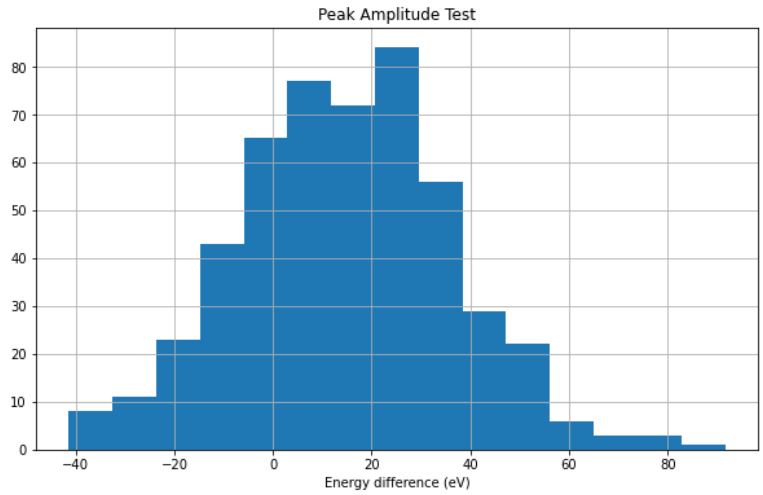

Figure 4.15: Histogram of the difference in predicted energy after increasing the peak values.

Although the neural network architectures used so far are exceedingly simple, they have accomplished powerful results—including the noise PSD in the training shows promise as a denoising method. In the next sections, I generate a new data set and then test the mixed-training network for data generated from differently-shaped noise PSDs, and compare its effectiveness with and without the PSD as an input.

4.5 Complicating Things

After proving that a simple CNN could be successful in distinguishing between different levels of noise, the next step was to compare its performance given a different noise profile entirely. Thus I created a new dataset generated using 8 different PSDs. These PSDs were taken from alterations of the previously-used PSDs (with peaks added randomly), and new ones with random peaks, based off of a logarithmic function with some added random noise. The data from the original no-noise dataset was randomly sampled, and noise was generated from a randomly-chosen PSD. The traces, true energy, and PSD were then saved as an event and were passed to the neural networks as inputs. The normalized (amplitude between 0 and 1) PSDs are shown in figure 4.16.

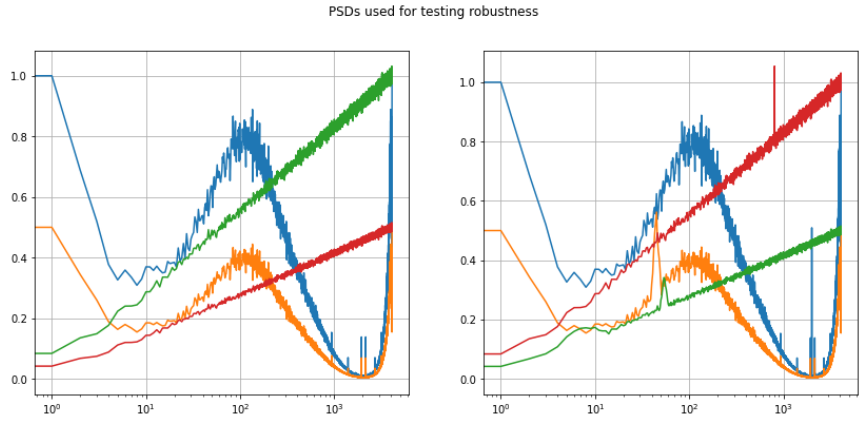

Figure 4.16: Set of 8 PSDs used to generate more complex-noise dataset. Plotted on two separate plots for greater clarity.

From these PSDs I generated a dataset with 10,000 training events, 3,000 validation events, and 5,000 testing events.

The first question is—how does the previous CNN fare on this dataset in its entirety? Unfortunately, not well. Figure 4.17 shows the RMS and bias calculations, and the RMS is consistently higher than 15% with pretty significant biases. Clearly the neural network has not adapted well to a mixture of PSDs, so I decided to test it on a 5000-event robustness test sample generated from only one of the new PSDs (shown in figure 4.18).

The CNN trained on the 3 PSDs from earlier did not perform well on this 5000-event robustness-test dataset. Results are shown in figure 4.19. The high RMS and bias value lead to the conclusion that for the purposes

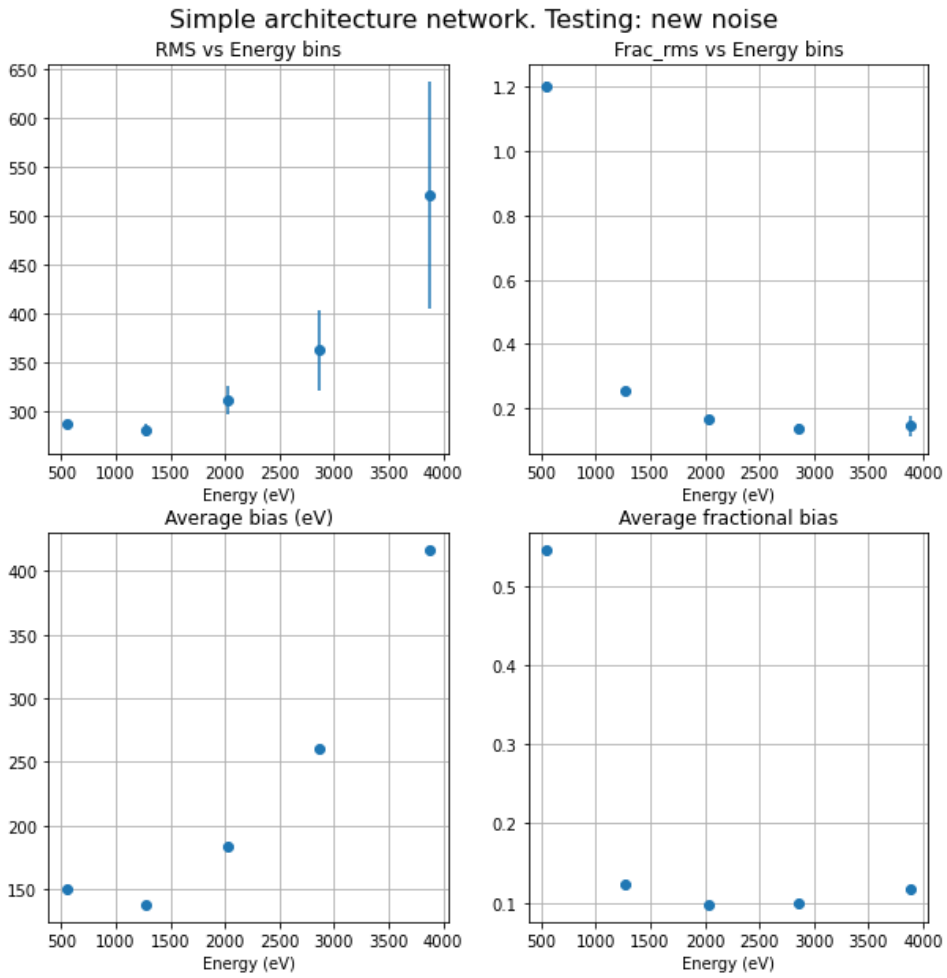

Figure 4.17: CNN performance once the dataset is changed.

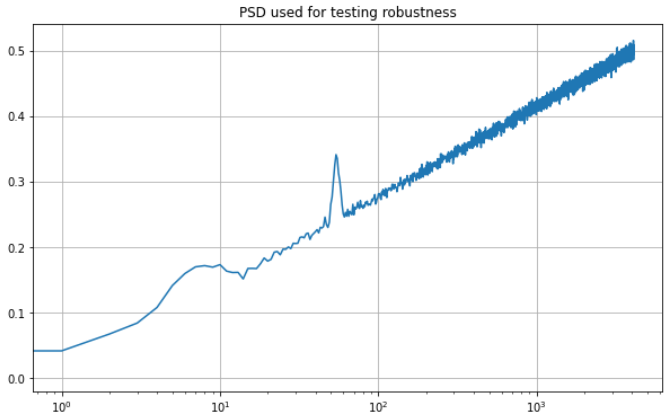

Figure 4.18: Logarithmic PSD with randomized noise. From this PSD 5000 events were created to test CNN robustness under a change of noise model.

of a CDMS analysis, this CNN would not maintain accuracy in its energy predictions under a changing noise model, and thus would need to be fed the noise PSD and retrained in order to produce satisfying results in energy reconstruction.

4.5.1 Improving training

In order to improve reconstruction, I trained my CNN on this dataset generated from 8 different PSDs and then tested it on the robustness set. I also trained a neural network without the PSD as an input to show the difference between including the PSD in the training and training with no PSD.

Figure 4.20 shows the results from training with the PSD, and the results for a CNN which does not take the PSD as an input are shown in figure 4.21.

Figure 4.22 shows the fractional RMS for the two neural networks tests side by side. We can see from it figures that including the PSD in the training has resulted in a reconstruction improvement even for vastly different PSDs, which is an indicator that this denoising technique can indeed work under a controlled set of conditions. It would take a significantly more complex neural network to be able to handle seeing completely new PSDs it has never seen before. Since one of the 8 PSDs used in this new dataset, however, is the "high" noise PSD from the previous dataset, when the CNN trained on this 8-PSD data sees the "high" noise data it predicts incredibly well—this is shown in Figure 4.23. In fact, in comparison with figure 4.9,

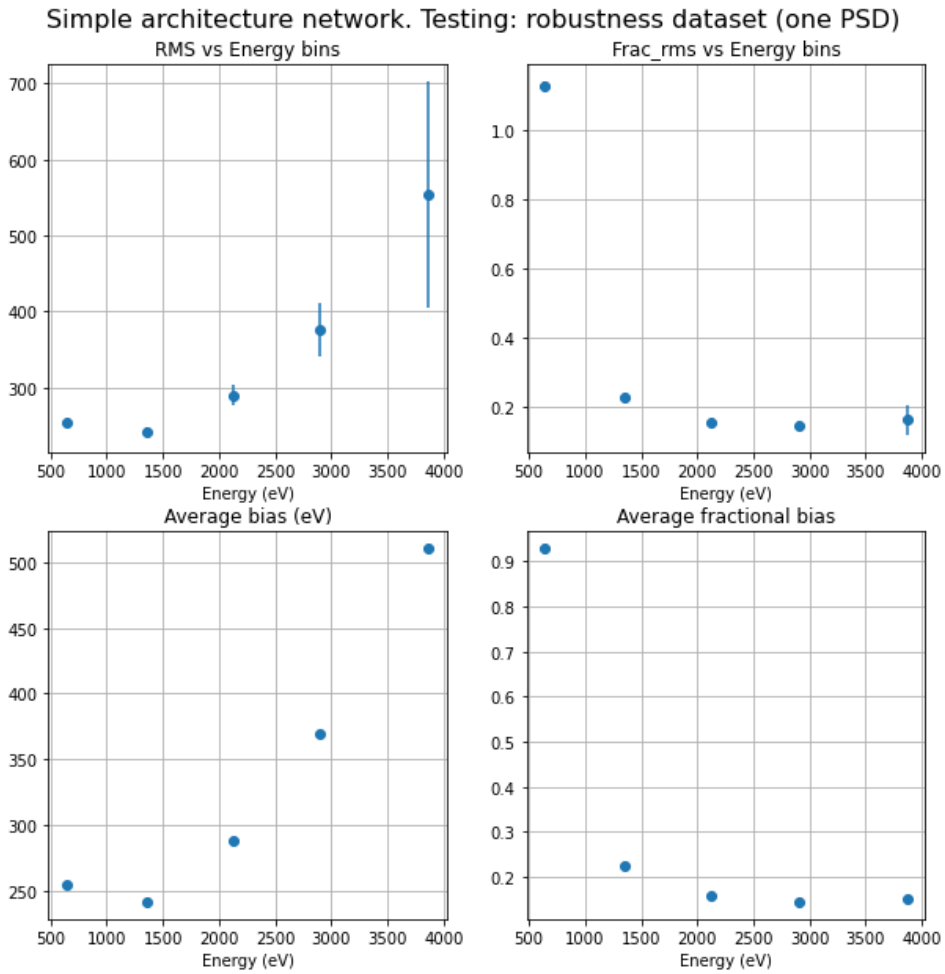

Figure 4.19: Neural network trained on "mixed" noise data (3 PSDs) from before, testing on robustness dataset. Changing the noise model leads to the neural network being unable to predict energies well.

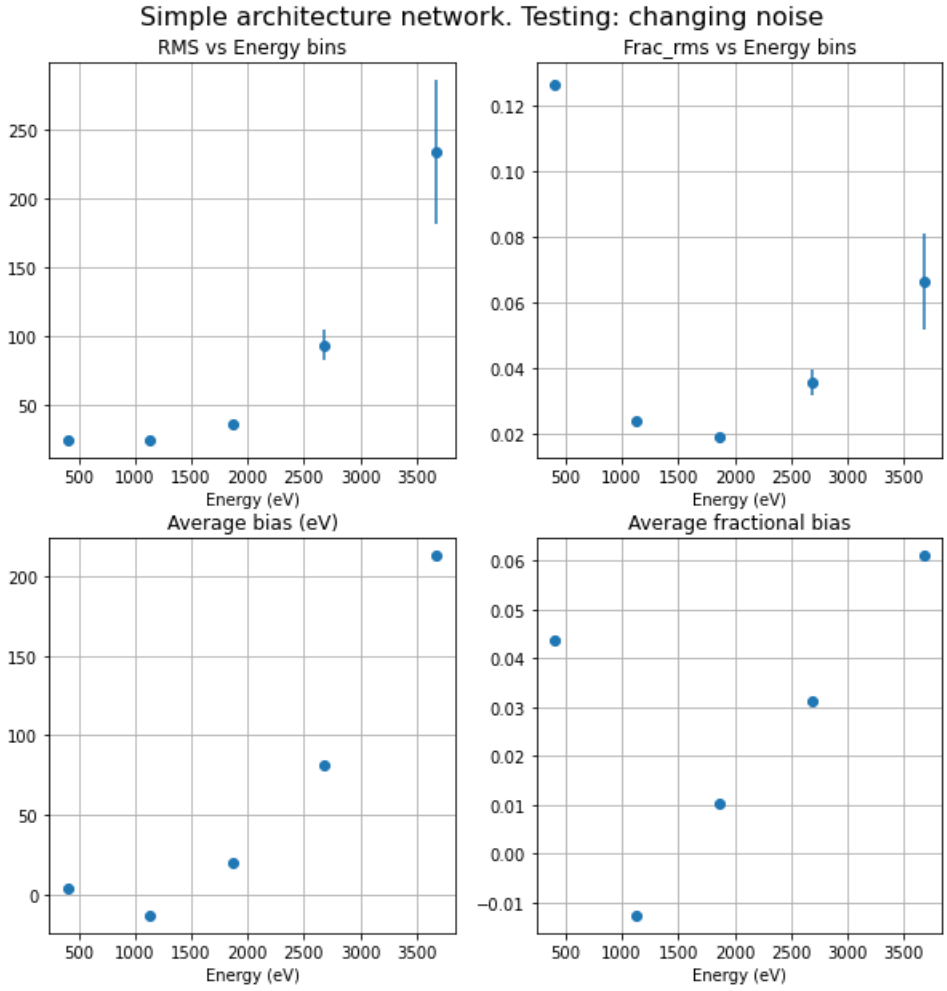

Figure 4.20: RMS, fractional RMS, average bias, and average fractional bias for simple architecture network **including** the PSD as an input. Trained on 8-PSD noise, tested on 8-PSD noise.

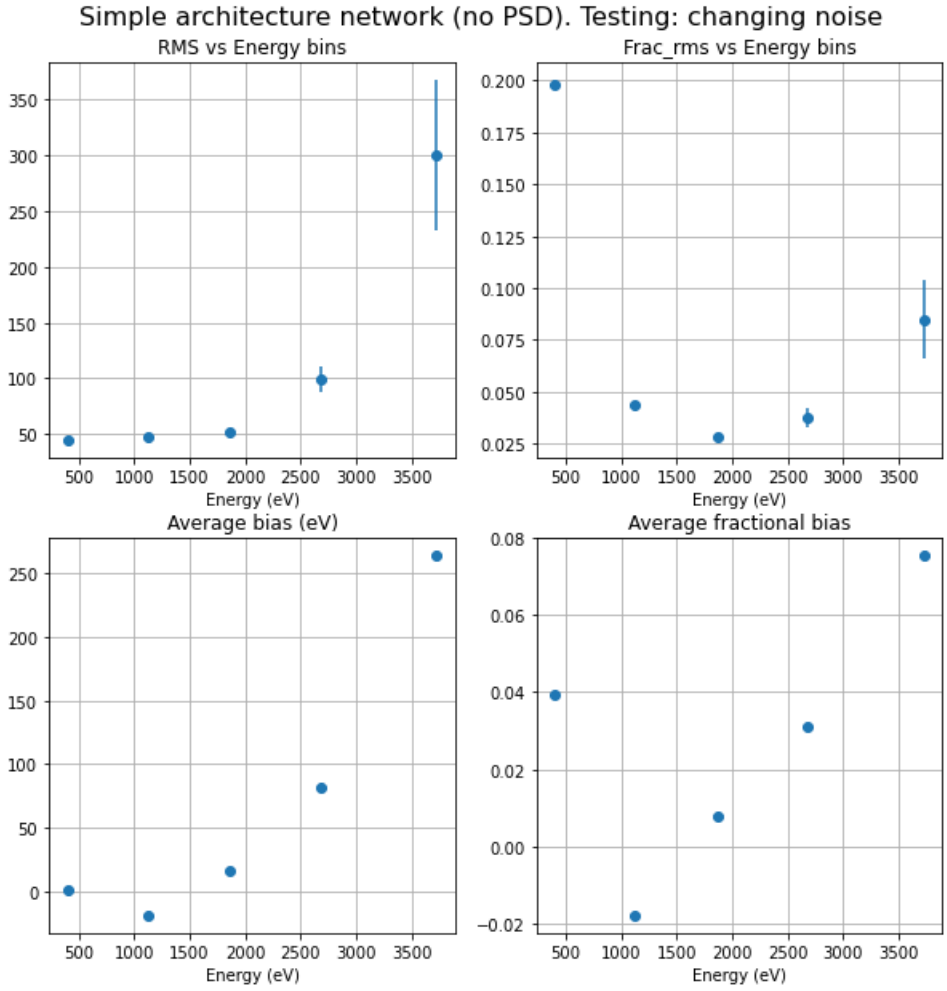

Figure 4.21: RMS, fractional RMS, average bias, and average fractional bias for simple architecture network **without** the PSD as an input. Trained on 8-PSD noise, tested on 8-PSD noise.

its reconstruction is equally powerful. Figure 4.24 shows a side-by-side fractional RMS comparison, and it shows that although the fractional RMS has increased slightly (primarily in the low-energy range), the reconstruction is still strong. Thus, it seems that including more PSDs in the reconstruction has made the neural network more versatile without sacrificing the quality of prediction for any individual noise model.

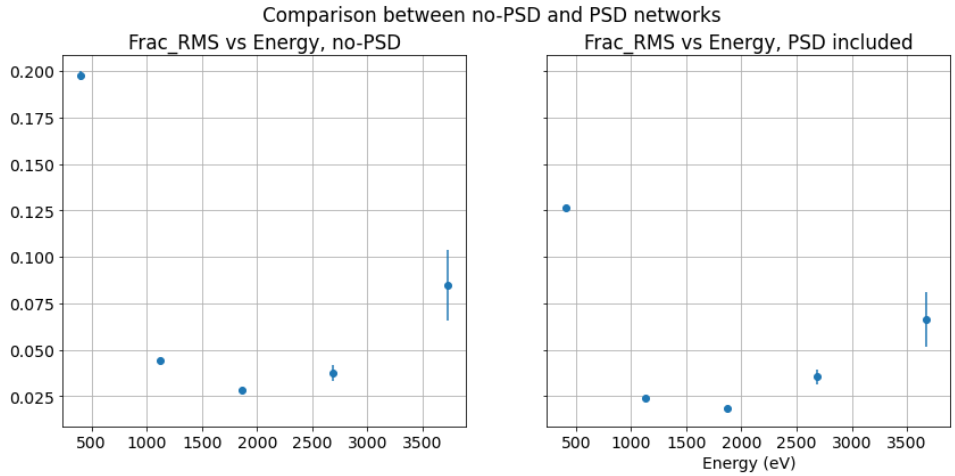

Figure 4.22: Side-by-side comparison of the fractional RMS for the NN without (left) and with (right) the PSD as an input.

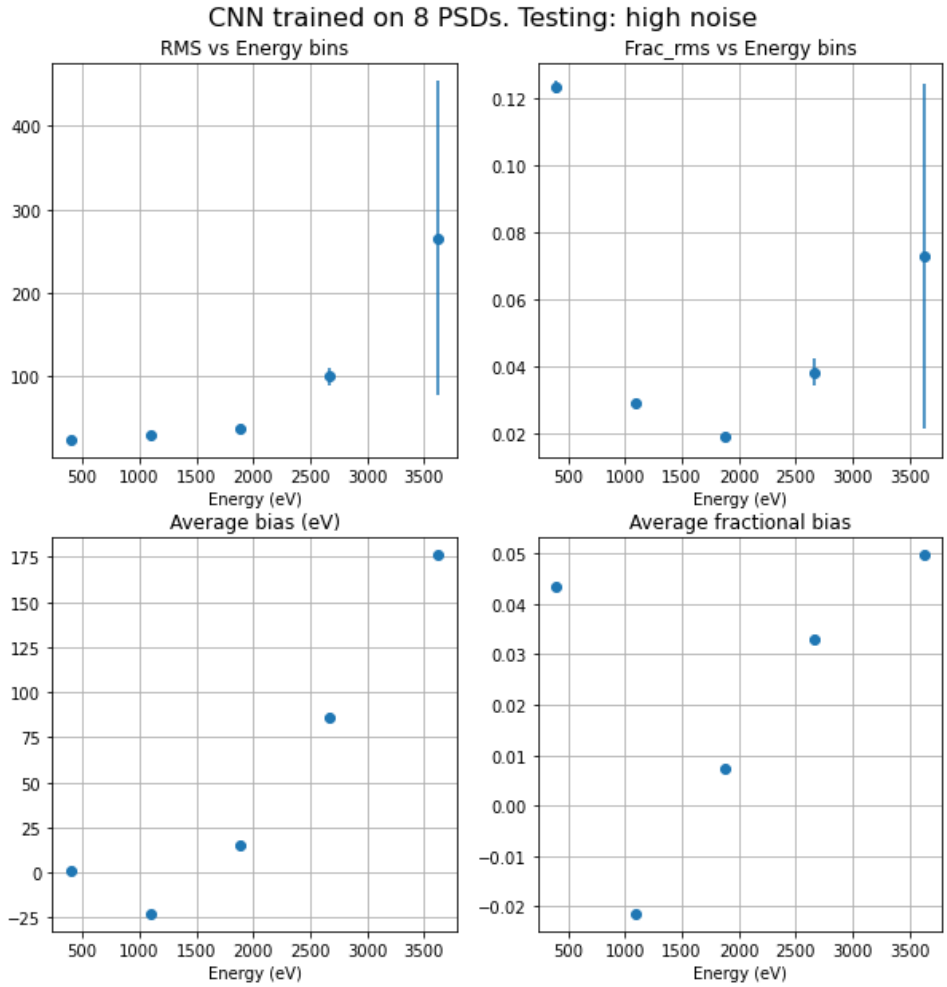

Figure 4.23: CNN trained on 8 PSDs tested on "high" noise.

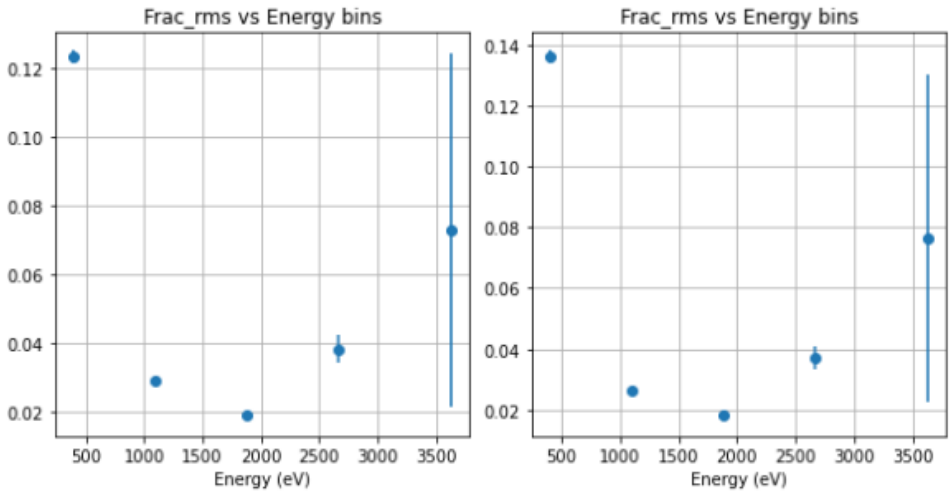

Figure 4.24: CNN trained on high, low, and no noise (left) vs CNN trained on 8 PSDs (right) tested on "high" noise. Although the reconstruction is slightly weaker when more PSDs are included in the training, it is still a strong energy predictor.

Chapter 5

Conclusions

In this thesis I have explained and described the integration of the Signal Distribution Unit for SuperCDMS detectors into the current data acquisition system. Additionally, I have outlined my work in attempting to use neural networks for simulated event reconstruction.

5.1 Findings on neural network performance for reconstruction

Applying a convolutional neural network to simulated physics events is an effective energy reconstruction technique. Even an exceedingly simple CNN can reconstruct energies accurately if the data contains no noise. However, if noise is added to the dataset, the neural network must be trained on a dataset with the same noise model, or the reconstruction breaks.

A neural network trained solely on the noisy data can achieve an acceptable reconstruction. However, in my studies I found that providing the dataset's noise PSD as an input improved the quality of the reconstruction, especially for low-energy events where noise fluctuations were proportionally higher compared to the relatively small amplitude of the data peaks.

Since SuperCDMS requires an analysis tool that can handle changing noise levels, and I found these neural network to be quite sensitive to noise, I aimed to provide the network with a tool that would allow it to compensate for noise changes by learning features from noise PSDs. My hope was that the neural network would achieve flexibility in its training and be able to handle a PSD it had never seen before. This was successful for a test where the noise PSDs were simply rescaled, but the shape and features were the same. This CNN was trained on high-level, low-level, and no noise, and it was able to correctly interpret data created from a mid-level noise PSD. Thus this simple CNN became capable of dealing with changes in the noise level.

However, once the noise PSD changed drastically and noise was generated from a different shape of PSD, the neural network became hopeless at

predicting the energy, with extremely poor resolution. Unfortunately, the neural network was not able to successfully reconstruct energies for data drawn from a PSD dissimilar to those that the CNN was trained with.

Nevertheless, after generating a more complex dataset from 8 different PSDs and training a neural network on it, it was able to handle a testing sample made from the same PSDs quite well. So as long as the training sample is varied, the neural network can achieve some flexibility and even perform well on new data as long as the PSD is similar to the ones used in training.

Therefore, while this simple approach shows promise as a denoising technique, it seems that with unexpected noise changes, the neural network must be retrained entirely. As this is not efficient, it is unlikely that it would be practical to apply these CNNs in a SuperCDMS analysis. In the next section, I describes possible ways in which this method could be improved.

5.2 Future directions

The ultimate goal of my neural network studies was to produce a neural network that could effectively learn how to denoise based off of a PSD that is provided separately for each event, allowing one NN to handle data sets in which different events have different noise PSDs. In order to do so, the training would require potentially hundreds of different PSDs, and a more complex neural network architecture than the one-layer forked CNN I've used in this thesis. I attempted to create a more complex neural network architecture with four convolutional layers on the input data, four layers on the PSD, and then adding the two layers as before, before including an additional convolutional layer, in the hopes that this added complexity would allow the NN more flexibility and robustness in testing. The guiding principle behind this neural network is the ability to not only detect more features (and smaller features), but to find correlations between the features, thus improving the accuracy of the energy predictions. By performing a convolution after the two neural network branches are added, in principle the neural network should find ways to account for the noise from the PSD convolution.

After creating this neural network, I then tested it on the 3-PSD dataset from the previous analysis, to determine its efficacy. This neural network was more precise, but less accurate, and the neural network was underpredicting the energy on average.

Attempting to train the complex network on the 8-PSD dataset, however,

did not fare well. Training iterations often resulted in a NN that predicted nonsense energy distributions; only in a few instances did it actually produce something reasonable.

Due to time constraints I was not able to improve the training of this NN to give better results, but this remains a promising direction for future developments. The fact that the neural network is essentially non-functional is not an uncommon problem—since NNs frequently contain a large number of hyperparameters, there are simply combinations of hyperparameters which will not produce good results. In order to avoid changing hyperparameters by hand, the machine learning Python package scikit-learn [38] contains a function named GridSearchCV which searches over specified parameters for an estimator. GridSearchCV does not function, however, for neural networks with multiple inputs; using it for this complex-architecture network is, then, impossible without modifying the GridSearchCV source code.

It is still possible that a more complex neural network architecture would be able to more effectively learn from a noise PSD and perform noise reduction that way. In order to implement such a network, given the optimization issues, it is necessary to implement routines which search through hyperparameters in order to find ideal configurations. While this is computationally more intensive, it is not difficult to implement. Even if the difficulties in training this more complex network architecture are overcome, however, it is not known that it would be able to handle the wide variety of different PSDs that might be encountered in real life, and if it would robustly and correctly handle PSDs dissimilar to those present in its training set. This is a subject for future study.

To truly prove this method is viable for event reconstruction for SuperCDMS WIMP searches, it would be necessary to test it on a set of real events with known true energies. Having a source of real events with known energies and positions would be incredibly powerful for the SuperCDMS collaboration in general. For this analysis, instead of creating simulated events from a noise PSD, we would have natural events from which we could construct a noise PSD, then determine if including the noise PSD in the training aids in the quality of the reconstruction.

Bibliography

[1] Albert Einstein. The Foundation of the General Theory of Relativity. *Annalen Phys.*, 49(7):769–822, 1916.

[2] Edwin Hubble and Milton L. Humason. The Velocity-Distance Relation among Extra-Galactic Nebulae. *The Astrophysical Journal*, 74:43, July 1931.

[3] F. Zwicky. On the Masses of Nebulae and of Clusters of Nebulae. *The Astrophysical Journal*, 86:217, October 1937.

[4] K. C. Freeman. On the Disks of Spiral and S0 Galaxies. *ApJ*, 160:811, June 1970.

[5] V. C. Rubin, Jr. Ford, W. K., and N. Thonnard. Rotational properties of 21 SC galaxies with a large range of luminosities and radii, from NGC 4605 (R=4kpc) to UGC 2885 (R=122kpc). *The Astrophysical Journal*, 238:471–487, June 1980.

[6] Susan Kassin, Roelof de Jong, and Benjamin Weiner. Dark and baryonic matter in bright spiral galaxies: Ii. radial distributions for 34 galaxies. *The Astrophysical Journal*, 643, 02 2006.

[7] Douglas Clowe, Maruša Bradač, Anthony H. Gonzalez, Maxim Markevitch, Scott W. Randall, Christine Jones, and Dennis Zaritsky. A direct empirical proof of the existence of dark matter. *The Astrophysical Journal*, 648(2):L109–L113, Aug 2006.

[8] S. Burles. Deuterium and big bang nucleosynthesis. *Nuclear Physics A*, 663-664:861c – 864c, 2000.

[9] M. R. Nolta, J. Dunkley, R. S. Hill, G. Hinshaw, E. Komatsu, D. Larson, L. Page, D. N. Spergel, C. L. Bennett, B. Gold, and et al. Five-year wilkinson microwave anisotropy probe observations: Angular power spectra. *The Astrophysical Journal Supplement Series*, 180(2):296–305, Feb 2009.

[10] Planck Collaboration, Ade, P. A. R., Aghanim, N., Armitage-Caplan, C., Arnaud, M., Ashdown, M., Atrio-Barandela, F., Aumont, J., Baccigalupi, C., Banday, A. J., Barreiro, R. B., Bartlett, J. G., Battaner, E., Benabed, K., Benoît, A., Benoit-Lévy, A., Bernard, J.-P., Bersanelli, M., Bielewicz, P., Bobin, J., Bock, J. J., Bonaldi, A., Bond, J. R., Borrill, J., Bouchet, F. R., Bridges, M., Bucher, M., Burigana, C., Butler, R. C., Calabrese, E., Cappellini, B., Cardoso, J.-F., Catalano, A., Challinor, A., Chamballu, A., Chary, R.-R., Chen, X., Chiang, H. C., Chiang, L.-Y, Christensen, P. R., Church, S., Clements, D. L., Colombi, S., Colombo, L. P. L., Couchot, F., Coulais, A., Crill, B. P., Curto, A., Cuttaia, F., Danese, L., Davies, R. D., Davis, R. J., de Bernardis, P., de Rosa, A., de Zotti, G., Delabrouille, J., Delouis, J.-M., Désert, F.-X., Dickinson, C., Diego, J. M., Dolag, K., Dole, H., Donzelli, S., Doré, O., Douspis, M., Dunkley, J., Dupac, X., Efstathiou, G., Elsner, F., Enßlin, T. A., Eriksen, H. K., Finelli, F., Forni, O., Frailis, M., Fraisse, A. A., Franceschi, E., Gaier, T. C., Galeotta, S., Galli, S., Ganga, K., Giard, M., Giardino, G., Giraud-Héraud, Y., Gjerløw, E., González-Nuevo, J., Górski, K. M., Gratton, S., Gregorio, A., Gruppuso, A., Gudmundsson, J. E., Haissinski, J., Hamann, J., Hansen, F. K., Hanson, D., Harrison, D., Henrot-Versillé, S., Hernández-Monteagudo, C., Herranz, D., Hildebrandt, S. R., Hivon, E., Hobson, M., Holmes, W. A., Hornstrup, A., Hou, Z., Hovest, W., Huffenberger, K. M., Jaffe, A. H., Jaffe, T. R., Jewell, J., Jones, W. C., Juvela, M., Keihänen, E., Keskitalo, R., Kisner, T. S., Kneissl, R., Knoche, J., Knox, L., Kunz, M., Kurki-Suonio, H., Lagache, G., Lähteenmäki, A., Lamarre, J.-M., Lasenby, A., Lattanzi, M., Laureijs, R. J., Lawrence, C. R., Leach, S., Leahy, J. P., Leonardi, R., León-Tavares, J., Lesgourgues, J., Lewis, A., Liguori, M., Lilje, P. B., Linden-Vørnle, M., López-Caniego, M., Lubin, P. M., Macías-Pérez, J. F., Maffei, B., Maino, D., Mandolesi, N., Maris, M., Marshall, D. J., Martin, P. G., Martínez-González, E., Masi, S., Massardi, M., Matarrese, S., Matthai, F., Mazzotta, P., Meinhold, P. R., Melchiorri, A., Melin, J.-B., Mendes, L., Menegoni, E., Mennella, A., Migliaccio, M., Millea, M., Mitra, S., Miville-Deschênes, M.-A., Moneti, A., Montier, L., Morgante, G., Mortlock, D., Moss, A., Munshi, D., Murphy, J. A., Naselsky, P., Nati, F., Natoli, P., Netterfield, C. B., Nørgaard-Nielsen, H. U., Noviello, F., Novikov, D., Novikov, I., O´Dwyer, I. J., Osborne, S., Oxborrow, C. A., Paci, F., Pagano, L., Pajot, F., Paladini, R., Paoletti, D., Partridge, B., Pasian, F., Patanchon, G., Pearson, D., Pearson, T. J., Peiris, H. V., Perdereau, O., Perotto, L., Perrotta, F., Pettorino, V., Piacentini, F., Piat, M., Pier-

paoli, E., Pietrobon, D., Plaszczynski, S., Platania, P., Pointecouteau, E., Polenta, G., Ponthieu, N., Popa, L., Poutanen, T., Pratt, G. W., Prézeau, G., Prunet, S., Puget, J.-L., Rachen, J. P., Reach, W. T., Rebolo, R., Reinecke, M., Remazeilles, M., Renault, C., Ricciardi, S., Riller, T., Ristorcelli, I., Rocha, G., Rosset, C., Roudier, G., Rowan-Robinson, M., Rubiño-Martín, J. A., Rusholme, B., Sandri, M., Santos, D., Savelainen, M., Savini, G., Scott, D., Seiffert, M. D., Shellard, E. P. S., Spencer, L. D., Starck, J.-L., Stolyarov, V., Stompor, R., Sudiwala, R., Sunyaev, R., Sureau, F., Sutton, D., Suur-Uski, A.-S., Sygnet, J.-F., Tauber, J. A., Tavagnacco, D., Terenzi, L., Toffolatti, L., Tomasi, M., Tristram, M., Tucci, M., Tuovinen, J., Türler, M., Umana, G., Valenziano, L., Valiviita, J., Van Tent, B., Vielva, P., Villa, F., Vittorio, N., Wade, L. A., Wandelt, B. D., Wehus, I. K., White, M., White, S. D. M., Wilkinson, A., Yvon, D., Zacchei, A., and Zonca, A. Planck 2013 results. xvi. cosmological parameters. *A&A*, 571:A16, 2014.

[11] G. F. Smoot, C. L. Bennett, A. Kogut, E. L. Wright, J. Aymon, N. W. Boggess, E. S. Cheng, G. de Amici, S. Gulkis, M. G. Hauser, G. Hinshaw, P. D. Jackson, M. Janssen, E. Kaita, T. Kelsall, P. Keegstra, C. Lineweaver, K. Loewenstein, P. Lubin, J. Mather, S. S. Meyer, S. H. Moseley, T. Murdock, L. Rokke, R. F. Silverberg, L. Tenorio, R. Weiss, and D. T. Wilkinson. Structure in the COBE Differential Microwave Radiometer First-Year Maps. *The Astrophysical Journal Letters*, 396:L1, September 1992.

[12] D. Cirigliano, H. J. de Vega, and N. G. Sanchez. Clarifying inflation models: The precise inflationary potential from effective field theory and the wmap data. *Physical Review D*, 71(10), May 2005.

[13] Planck Collaboration, Ade, P. A. R., Aghanim, N., Arnaud, M., Ashdown, M., Aumont, J., Baccigalupi, C., Banday, A. J., Barreiro, R. B., Bartlett, J. G., Bartolo, N., Battaner, E., Battye, R., Benabed, K., Benoît, A., Benoit-Lévy, A., Bernard, J.-P., Bersanelli, M., Bielewicz, P., Bock, J. J., Bonaldi, A., Bonavera, L., Bond, J. R., Borrill, J., Bouchet, F. R., Boulanger, F., Bucher, M., Burigana, C., Butler, R. C., Calabrese, E., Cardoso, J.-F., Catalano, A., Challinor, A., Chamballu, A., Chary, R.-R., Chiang, H. C., Chluba, J., Christensen, P. R., Church, S., Clements, D. L., Colombi, S., Colombo, L. P. L., Combet, C., Coulais, A., Crill, B. P., Curto, A., Cuttaia, F., Danese, L., Davies, R. D., Davis, R. J., de Bernardis, P., de Rosa, A., de Zotti, G., Delabrouille, J., Désert, F.-X., Di Valentino, E., Dickinson, C.,

Diego, J. M., Dolag, K., Dole, H., Donzelli, S., Doré, O., Douspis, M., Ducout, A., Dunkley, J., Dupac, X., Efstathiou, G., Elsner, F., Enßlin, T. A., Eriksen, H. K., Farhang, M., Fergusson, J., Finelli, F., Forni, O., Frailis, M., Fraisse, A. A., Franceschi, E., Frejsel, A., Galeotta, S., Galli, S., Ganga, K., Gauthier, C., Gerbino, M., Ghosh, T., Giard, M., Giraud-Héraud, Y., Giusarma, E., Gjerløw, E., González-Nuevo, J., Górski, K. M., Gratton, S., Gregorio, A., Gruppuso, A., Gudmundsson, J. E., Hamann, J., Hansen, F. K., Hanson, D., Harrison, D. L., Helou, G., Henrot-Versillé, S., Hernández-Monteagudo, C., Herranz, D., Hildebrandt, S. R., Hivon, E., Hobson, M., Holmes, W. A., Hornstrup, A., Hovest, W., Huang, Z., Huffenberger, K. M., Hurier, G., Jaffe, A. H., Jaffe, T. R., Jones, W. C., Juvela, M., Keihänen, E., Keskitalo, R., Kisner, T. S., Kneissl, R., Knoche, J., Knox, L., Kunz, M., Kurki-Suonio, H., Lagache, G., Lähteenmäki, A., Lamarre, J.-M., Lasenby, A., Lattanzi, M., Lawrence, C. R., Leahy, J. P., Leonardi, R., Lesgourgues, J., Levrier, F., Lewis, A., Liguori, M., Lilje, P. B., Linden-Vørnle, M., López-Caniego, M., Lubin, P. M., Macías-Pérez, J. F., Maggio, G., Maino, D., Mandolesi, N., Mangilli, A., Marchini, A., Maris, M., Martin, P. G., Martinelli, M., Martínez-González, E., Masi, S., Matarrese, S., McGehee, P., Meinhold, P. R., Melchiorri, A., Melin, J.-B., Mendes, L., Mennella, A., Migliaccio, M., Millea, M., Mitra, S., Miville-Deschênes, M.-A., Moneti, A., Montier, L., Morgante, G., Mortlock, D., Moss, A., Munshi, D., Murphy, J. A., Naselsky, P., Nati, F., Natoli, P., Netterfield, C. B., Nørgaard-Nielsen, H. U., Noviello, F., Novikov, D., Novikov, I., Oxborrow, C. A., Paci, F., Pagano, L., Pajot, F., Paladini, R., Paoletti, D., Partridge, B., Pasian, F., Patanchon, G., Pearson, T. J., Perdereau, O., Perotto, L., Perrotta, F., Pettorino, V., Piacentini, F., Piat, M., Pierpaoli, E., Pietrobon, D., Plaszczynski, S., Pointecouteau, E., Polenta, G., Popa, L., Pratt, G. W., Prézeau, G., Prunet, S., Puget, J.-L., Rachen, J. P., Reach, W. T., Rebolo, R., Reinecke, M., Remazeilles, M., Renault, C., Renzi, A., Ristorcelli, I., Rocha, G., Rosset, C., Rossetti, M., Roudier, G., Rouillé d´Orfeuil, B., Rowan-Robinson, M., Rubiño-Martín, J. A., Rusholme, B., Said, N., Salvatelli, V., Salvati, L., Sandri, M., Santos, D., Savelainen, M., Savini, G., Scott, D., Seiffert, M. D., Serra, P., Shellard, E. P. S., Spencer, L. D., Spinelli, M., Stolyarov, V., Stompor, R., Sudiwala, R., Sunyaev, R., Sutton, D., Suur-Uski, A.-S., Sygnet, J.-F., Tauber, J. A., Terenzi, L., Toffolatti, L., Tomasi, M., Tristram, M., Trombetti, T., Tucci, M., Tuovinen, J., Türler, M., Umana, G., Valenziano, L., Valiviita, J., Van Tent, F., Vielva, P., Villa, F., Wade, L. A., Wandelt, B. D., Wehus,

I. K., White, M., White, S. D. M., Wilkinson, A., Yvon, D., Zacchei, A., and Zonca, A. Planck 2015 results - xiii. cosmological parameters. *A&A*, 594:A13, 2016.

[14] Yonit Hochberg, Tongyan Lin, and Kathryn M. Zurek. Absorption of light dark matter in semiconductors. *Physical Review D*, 95(2), Jan 2017.

[15] Jeffrey Filippini. *A Search for WIMP Dark Matter Using the First Five-Tower Run of the Cryogenic Dark Matter Search.* PhD thesis, University of California, Berkeley, 2008.

[16] Charlie Conroy, Risa H. Wechsler, and Andrey V. Kravtsov. Modeling luminosity-dependent galaxy clustering through cosmic time. *The Astrophysical Journal*, 647(1):201–214, Aug 2006.

[17] Marc Kamionkowski. Wimp and axion dark matter. `arXiv:hep-ph/9710467`, 1997.

[18] Roberto D. Peccei. *Axions.* Springer Berlin Heidelberg, 2008.

[19] R. D. Peccei and Helen R. Quinn. Constraints imposed by CP conservation in the presence of pseudoparticles. *Phys. Rev. D*, 16:1791–1797, Sep 1977.

[20] Georg Raffelt. Axions in astrophysics and cosmology. `arXiv:hep-ph/9502358`, 1995.

[21] David J. E. Marsh. Axions and alps: a very short introduction. `arXiv:1712.03018[hep-ph]`, 2017.

[22] T. Aralis, T. Aramaki, I. J. Arnquist, E. Azadbakht, W. Baker, S. Banik, D. Barker, C. Bathurst, D. A. Bauer, L. V. S. Bezerra, and et al. Constraints on dark photons and axionlike particles from the supercdms soudan experiment. *Physical Review D*, 101(5), Mar 2020.

[23] Bob Holdom. Two u(1)'s and ϵ charge shifts. *Physics Letters B*, 166(2):196 – 198, 1986.

[24] R. Foot and S. Vagnozzi. Dissipative hidden sector dark matter. *Physical Review D*, 91(2), Jan 2015.

[25] William Page. *Searching for Low-Mass Dark Matter with SuperCDMS Soudan Detectors.* PhD thesis, University of British Columbia, 2 2019.

[26] R. Agnese, A. J. Anderson, T. Aralis, T. Aramaki, I. J. Arnquist, W. Baker, D. Balakishiyeva, D. Barker, R. Basu Thakur, D. A. Bauer, and et al. Low-mass dark matter search with cdmslite. *Physical Review D*, 97(2), Jan 2018.

[27] Andrea Giuliani. *Dark Matter Direct and Indirect Detection*, pages 295–328. Springer Netherlands, Dordrecht, 2011.

[28] R. Agnese, T. Aralis, T. Aramaki, I. J. Arnquist, E. Azadbakht, W. Baker, S. Banik, D. Barker, D. A. Bauer, T. Binder, and et al. First dark matter constraints from a supercdms single-charge sensitive detector. *Physical Review Letters*, 121(5), Aug 2018.

[29] William Page. Data acquisition for supercdms snolab. Master's thesis, University of British Columbia, 7 2016.

[30] The MidasDAQ Manual. `https://cdms-gw.triumf.ca/daq_doc_demo/`.

[31] The MIDAS Data Acquisition System. `https://midas.triumf.ca/MidasWiki/index.php/Main_Page`.

[32] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, Jan 2015.

[33] Glosser.ca. File:colored neural network.svg. `https://commons.wikimedia.org/wiki/File:Colored_neural_network.svg`, Feb 2013.

[34] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[35] Jason Brownlee. How to avoid overfitting in deep learning neural networks. *Machine Learning Mastery*, Aug 2019.

[36] Matthew Stewart. Simple introduction to convolutional neural networks. `https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac`, Feb 2019.

[37] Winchell, J., Morales, J., and Toback, D. DAQSim package. SuperCDMS code repository (not publicly available). `http://titus.stanford.edu:8080/git/summary/?r=Simulations/DAQSim.git`, 2019.

[38] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.