A Datacentric Algorithm for Gamma-ray Radiation Anomaly Detection in Unknown Background Environments

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

James Michael Ghawaly

August 2020

I would like to dedicate this dissertation to my wife, Rebekah.

Acknowledgments

I would like to thank my family and friends for their support during my research and writing. I would also like to thank my colleagues at Oak Ridge National Laboratory in the Radiation Detection and Imaging group who supported me throughout my research by lending their knowledge and resources to my work. I would also like to thank my advisors and committee at UTK for their support.

Abstract

The detection of anomalous radioactive sources in environments characterized by a high level of variation in the background radiation is a challenging problem in nuclear security. A variety of natural and artificial sources contribute to background radiation dynamics including variations in the absolute and relative concentrations of naturally occurring radioisotopes in different materials, the wet-deposition of ^{222}Rn daughters during precipitation, and background suppression due to physical objects in the detector scene called "clutter." This dissertation presents a new datacentric algorithm for radiation anomaly detection in dynamic background environments. The algorithm is based on a custom deep neural autoencoder architecture called the Autoencoder Radiation Anomaly Detection (ARAD) model. An autoencoder is a type of neural network that compresses data at its input through a series of computational layers into a dimensionally-constrained representation called the latent space. It then uses another set of layers to decompress the latent space back into its original dimensionality. When trained on typical background radiation data, ARAD learns an efficient representation of the components comprising typical gamma ray background radiation spectra. If a gamma ray spectrum containing anomalous radiation signatures is presented to ARAD, it is unable to reconstruct the spectrum at its input which can be used to trigger a detection alarm. ARAD was trained and tested on both simulated and real world gamma ray radiation data. Results indicate that ARAD shows a high level of resilience to background dynamics while also displaying good detection performance on a variety of simulated and real world sources.

Table of Contents

List of Figures

Chapter 1

Introduction

The work in this dissertation is concerned with the development of a data-driven algorithm for the detection of illicit radioactive material in environments where it is challenging to do so using existing algorithms due to the presence of a dynamic background radiation. This chapter will begin by briefly describing the importance of radiation detection in terms of national security in the United States. The basics of gamma-ray radiation and its detection and analysis is then described to introduce the reader to the background topics necessary to fully understand the methods developed herein. A selection of some of the most influential existing gamma-ray radiation anomaly detection algorithms will then be described.

1.1 Problem Statement

The work in this dissertation is concerned with the development of a data-driven algorithm for the detection of illicit radioactive material in environments where it is challenging due to the presence of a dynamic background radiation. The algorithm is based on modern deep learning methods, particularly a custom autoencoder architecture for developing a data-driven compressed representation of what constitutes background gamma-ray radiation spectra collected by a detector. Using this representation of the background radiation constituents, we can analyze new radiation spectra to determine if they contain anomalous (non-background) components.

The main tasks/contributions performed by this work:

1. The design and development of a novel autoencoder neural network called the Autoencoder Radiation Anomaly Detection (ARAD) model that can extract a compressed representation of background radiation using a set of training data containing gamma-ray spectra that can either be collected or simulated.

2. Generation of radiation anomaly detection alarms by using the trained autoencoder to attempt reconstruction of new gamma-ray spectra and alarming based on the reconstruction error.

3. Demonstration of the ARAD model on a set of experimentally-validated Monte Carlo simulated data representing a mobile NaI(Tl) detector moving through an urban street loosely based on downtown Knoxville, TN [1, 2]. ARAD's resilience to NORM-induced false alarms is demonstrated. Also, the ability to tune the source and background activities in the model enabled the generation of source detection performance metrics for the ARAD model such as receiver operating characteristic and probability of detection curves. This was performed for five sources: highly enriched uranium, weapons grade plutonium, ^{131}I, ^{60}Co, and ^{99m}Tc.

4. Demonstration of the ARAD model on real-world gamma-ray detection data collected by a series of NaI(Tl) detectors deployed at the High Flux Isotope Reactor (HFIR) and Radiochemical Engineering Development Center (REDC) at Oak Ridge National Laboratory. ARAD's resilience to rain and clutter-induced false alarms and ability to detect a variety of real sources is demonstrated.

1.2 Motivation

After the terrorist events on September 11, 2001, the United States revamped its approach to national security, especially with regards to the potential transport of illicit nuclear or radiological material within the United States' borders. Part of the response to these events was the creation of the Department of Homeland Security (DHS), and within it the Domestic Nuclear Detection Office (DNDO) which is now jointly staffed with the Countering Weapons of Mass Destruction (CWMD) office. The official mission statement of the DNDO is to:

Figure 1.1: RAP team members performing a source search campaign [3].

> *Prevent nuclear terrorism by continuously improving capabilities to deter, detect, respond to, and attribute attacks, in coordination with domestic and international partners.*

With this statement, one of the major concerns of DNDO is to detect potential nuclear terrorism. As part of this effort, the Radiological Assistance Program that is managed by the Department of Energy (DOE) is often deployed to carry out radiation search operations during certain large public events [3]. This team uses a variety of detection methods to detect and locate potential illicit sources, including handheld detectors carried by personnel, ground vehicle mounted detection systems, and aerial mounted detection systems through the DOE Aerial Measurement System [3, 4]. Figure 1.1 shows an example of a few RAP team members carrying out a source detection campaign.

Many radiation detection systems such as some of the ones used by RAP have built in algorithms that alert the user to the presence of potentially non-background sources of radiation. As a result, it is of utmost importance that these detection algorithms can operate well in complex environments, especially urban environments, where the majority of search operations would likely take place. Section 1.7 will discuss some of the challenges in detecting radioactive sources in such environments.

1.3 Radioactivity

This section will give a brief overview of the basics of radioactivity. Various modes of nuclear decay will be discussed, with gamma ray decay being the focus of this dissertation.

1.3.1 Radioactive Decay

While there are many different radioactive decay products, the four primary ones of importance to radiation detection for nuclear security are alpha, beta, gamma, and neutron emission. This section will focus on gamma-ray decay, which is the modality of decay that is being detected in the work of this dissertation. For information on the other three modes of decay, the reader is referred to Knoll's textbook on the matter for an introduction to these topics [5].

Alpha and Beta Decay

When a nuclide has a N/Z ratio that causes it to become unstable, it decays by whatever mechanism will bring it closer to a more stable N/Z ratio. Some isotopes decay by emitting a bundle of 2 neutrons and 2 protons called an alpha particle, which is equivalent to a ^{4}He nucleus. This makes changes the the resulting nuclide into a new, lighter element. On the nuclide chart in Figure 1.2, an alpha decay shifts the nuclide 2 places down and 2 places to the left. Due to their large size, alpha particles do not travel very far in everyday materials (only a few centimeters in air). While they have some uses in nuclear forensics applications due to their emission energies being characteristic of the parent radionuclide, their limited range makes them difficult to use for radiation source search.

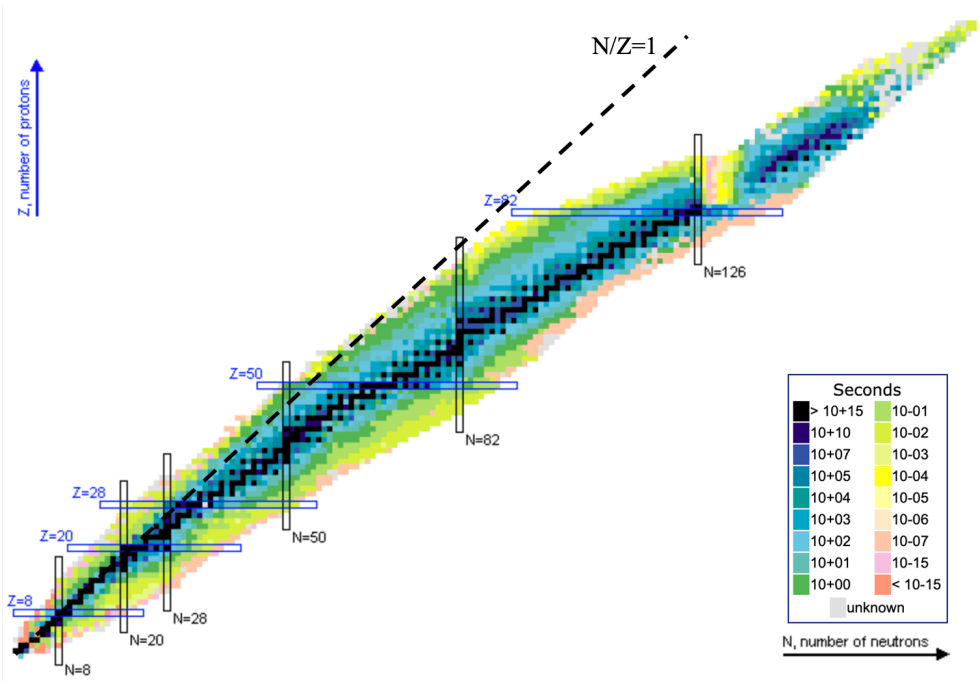


Figure 1.2: Chart of nuclides from the National Nuclear Data Center [6]. Colors represent the decay half life of each nuclide.

Another decay mechanism is beta decay. Beta decay is a process wherein a neutron changes into a proton, releasing a beta particle (electron) and an electron antineutrino, as shown in Equation 1.1. Beta decay changes the resultant nuclide into a new element, moving it up one and left one on the nuclide chart. This leads to a slight decrease in the N/Z ratio of the daughter nuclide. Beta particles can travel farther than alpha particles on average, generally less than a meter in air. Beta particles are emitted with energies that follow a distribution and can thus not effectively be used for rapid radioisotope identification. This, along with their limited range, makes them difficult for use in radiation source search.

$$n \longrightarrow p + e^{-} + \bar{\nu}_e \tag{1.1}$$

As opposed to beta decay where a neutron changes into a proton, it is also possible that a proton can change into a neutron, releasing a positron and electron neutrino in the process as shown in Equation 1.2. This process is called positron decay. Positron decay slightly

increases the N/Z ratio of the daughter nuclide, moving it down one and right one on the nuclide chart.

$$p \longrightarrow n + e^{+} + \nu_e \tag{1.2}$$

Gamma Decay

Often times, an alpha, beta, or other decay types will leave the daughter nucleus in an excited state. In order for the excited state to relax, the nucleus can release the energy in the form of photons called gamma rays. These gamma rays are generally between tens of keV and close to 10 MeV, corresponding to quantized shifts in the energy levels of the nucleus. Because of this quantization, gamma rays emitted by different radionuclides have energeis characteristic to that specific radionuclide. An example would be the decay of ^{60}Co as shown in Figure 1.3. 99.88% of the time, ^{60}Co beta decays into an excited state of ^{60}Ni and then decays down to the ground state by emitting a succession of gamma rays with energies of 1.1732 MeV and 1.3325 MeV. The other 0.12% of the time, the ^{60}Co beta decays down the the second excited state and then releases a gamma ray with an energy of 1.3325 MeV. Decay levels and radiation data for different radionuclides can be found on the National Nuclear Data Center (NNDC) site [6].

When compared to alpha and beta particles, gamma rays are extremely penetrative. This, along with the fact that gamma-rays are emitted with radionuclide-characteristic energies makes them ideal for radioactive source search applications.

1.4 Gamma Ray Interactions with Matter

This section will give a brief overview of the three primary mechanisms in which gamma rays interact with matter: the photoelectric effect, Compton scattering, and pair production. Understanding these concepts is important to understanding how gamma ray radiation is detected and how gamma rays emitted from radionuclides interact with the environment prior to hitting a detector.

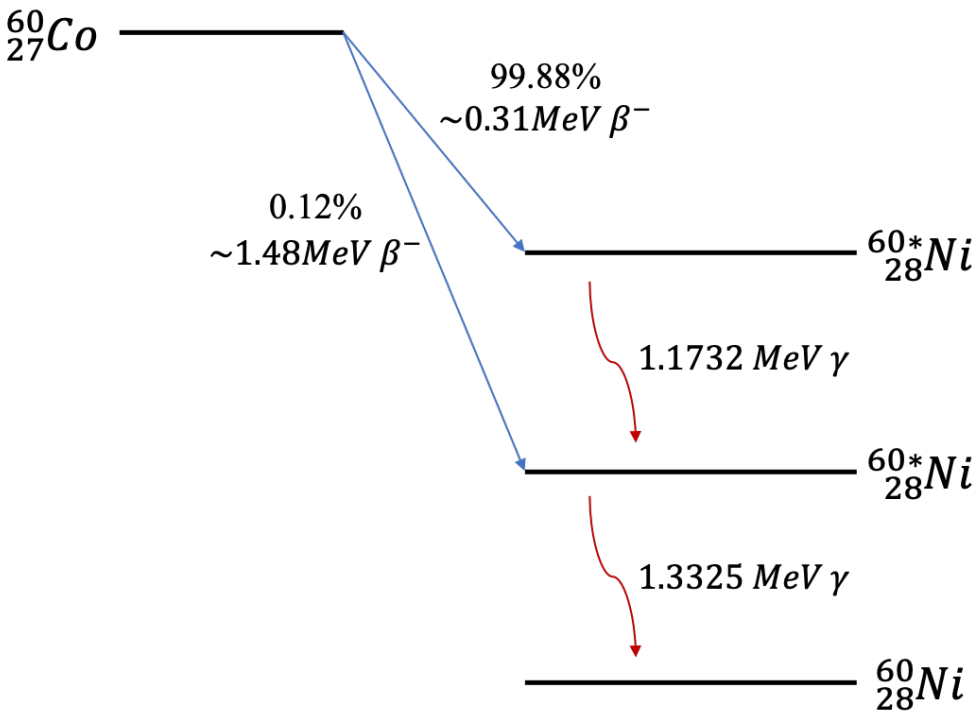


Figure 1.3: Decay of ^{60}Co into ^{60}Ni.

1.4.1 Photoelectric Effect

The photoelectric effect was one of the most important discoveries in physics in the early 19th century, partially leading to the development of quantum mechanics. The photoelectric effect is the process where a photon with some energy E_p collides with an atomic electron in the outer shell and is completely absorbed by said electron. The electron is then ejected with an energy E_e, which is given by Equation 1.3 where ϕ_0 is called the work function of the material [7]. The work function is the minimum amount of energy that the photon must have in order to eject the electron from the material.

$$E_e = E_p - \phi_0 \tag{1.3}$$

Under classical mechanics, electrons should be ejected by the material at a rate and with an energy dependent on the intensity of the incoming light. This is not the case however. In reality, the maximum kinetic energy of the ejected electrons is dependent on the energy of

the incoming photons and not their intensity. Instead, the intensity of the incoming photons is proportional to the intensity of the ejected electrons and not their energy, assuming the photon energy is greater than the work function of the material, .

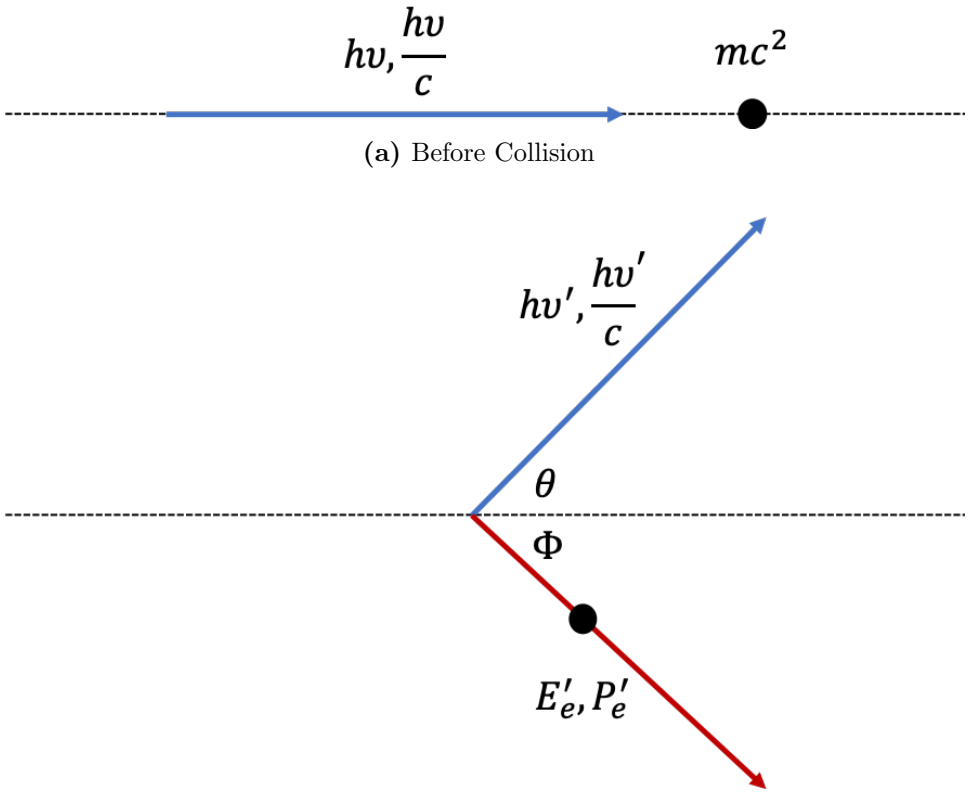
Photons can travel great distances in a material without interacting with it. The probability of interaction for a photon travelling through a material is largely dependent on the atomic number Z of the material and the energy E_p of the incoming photon. This is measured by the interaction cross section, which is proportional to Z and E as given by Equation 1.4 [7]. The key takeaway from this equation is that the probability of a photon undergoing the photoelectric effect in a material increases for high Z materials and lower energy photons.

$$\frac{Z^4}{E_p^3} \tag{1.4}$$

1.4.2 Compton Scattering

As opposed to photoelectric absorption, it is also possible for a photon to impact an electron and scatter off of it rather than being fully absorbed. As shown in Figure 1.4a, we will consider a photon with kinetic energy $E_p = h\nu$ where h is Planck's constant and ν is the frequency of the photon and momentum $\frac{h\nu}{c}$ where c is the speed of light [7]. In this figure, we consider the electron to be at rest with a total mass-energy equal to mc^2. After striking the electron as shown in Figure 1.4b, the photon scatters off of the electron at an angle θ with kinetic energy $h\nu^{'}$ and momentum $\frac{h\nu^{'}}{c}$ and the electron scatters at an angle Φ with kinetic energy $E_e^{'}$ and momentum $P_e^{'}$ [7].

Using conservation of kinetic energy and momentum, an expression for calculating the energy of the scattered photon can be formulated, as shown in Equation 1.5 [7]. Unlike for photoelectric absorption, the energy deposited in the material by a Compton scattering event is not discrete; the deposited energy is dependent on the angle of scattering. From Equation 1.5, we can see that the maximum energy deposition into the material (and thus the maximum energy loss in the photon) occurs when $\theta = 180\,\text{deg}$.



(a) Before Collision

(b) After Collision

Figure 1.4: Compton scattering process, figure is inspired by Figure 8.4 in Turner's textbook [7]

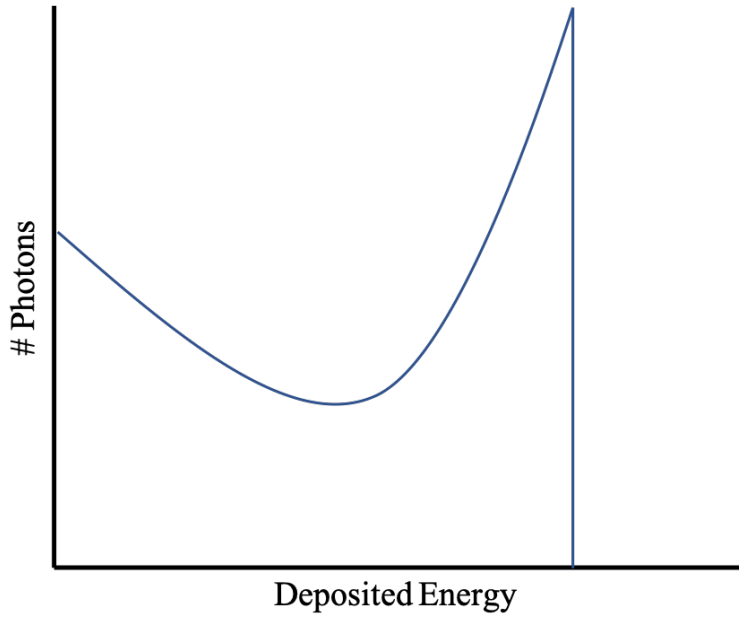


Figure 1.5: Approximate shape of the distribution of deposited energy for Compton scattering of a flux of monoenergetic photons, stemming from the Klein-Nishina formula. Figure inspired by Figure 8.5 in Turner's textbook [7].

$$h\nu^{'} = \frac{h\nu}{1 + \frac{h\nu}{mc^2}\,(1 - cos\theta)} \tag{1.5}$$

If one was to plot the distribution of energy deposition in a given material after bombarding said material with a flux of monoenergetic photons undergoing Compton scattering, the energy distribution would take the shape as seen in Figure 1.5, which stems from the Klein-Nishina formula [7]. The peak on the right of the distribution occurs near the maximum energy deposition for $\theta = 180\,\text{deg}$. This feature is called the Compton edge. The region between 0 and the Compton edge is called the Compton continuum. The probability of Compton scattering in the material is dependent on the electron density of the material [7].

1.4.3 Pair Production

The third important gamma ray interaction that we will consider is pair production. If a photon has an energy greater than or equal to twice the rest energy of an electron, $2m_ec^2 = 1.022$ MeV, and approaches near an atomic nucleus, there is a chance that the photon can be converted into an electron-positron pair. The energy deposited into the material from this interaction is very small, however, if the positron that was created meets an electron, there is a chance that it will annihilate with said electron and create a pair of 0.511 MeV photons that can then interact with the material via the photoelectric effect or Compton scattering. Pair Production becomes more and more likely as the photon energy increases beyond 1.022 MeV and is also proportional to Z^2 of the material.

Overall, for a given material, the photoelectric effect tends to dominate for lower energies (less than a few hundred keV), pair production tends to dominate at higher energies (over a few MeV), and Compton scattering tends to dominate in the energy region between a few hundred keV and a few MeV [5]. In NaI(Tl) for example, the photoelectric effect dominates below approximately 400 keV, Compton scattering dominates between about 400 kev and 7 MeV, and pair production dominates beyond 7 MeV [8]. These facts will be important when discussing gamma spectroscopy in Section 1.6.2.

1.5 Gamma Radiation Detection

A plethora of radiation detection technologies exist for the detection of gamma rays. This section will focus specifically on scintillation detectors, and particularly NaI(Tl) scintillation detectors, as this is the type of detector that is primarily targeted by the detection algorithm developed for this dissertation.

A scintillation detector is composed of a type of material that when radiation is absorbed, converts this radiation energy into a lower-wavelength light (visible or UV) such that it can be measured with typical optoelectronics equipment and methods. The two most common types of scintillation materials are solid inorganic and organic scintillators.

Organic scintillators are composed of polymerized aromatic hydrocarbon compounds. When ionizing radiation energy is absorbed by the organic molecules in the scintillator,

molecular electrons can be excited to higher energy levels and decay by releasing lower wavelength scintillation photons [5]. Organic scintillators are usually lightweight compared to inorganic scintillators and can be molded and extruded in many different shapes and sizes. Due to their low-Z composition however, Compton scattering is the primary photon interaction mechanism in the detector which makes them difficult to use for gamma spectroscopy applications.

Inorganic scintillators are made of crystals composed of inorganic compounds. When they absorb ionizing radiation, electrons in the crystal lattice that are occupying the stable valence band can become excited to the conductance band, in which the electron is free to drift throughout the crystal [5]. This creates an electron-hole pair in the crystal. The de-excitation of the electron from the conductance band to the valence band is too high of an energy such that the emitted scintillation photon is not in the visible spectrum [5]. In order to combat this, an impurity is usually added to the crystal to create excitation bands between the valence and conductance bands [5]. The electron holes migrate to these impurity sites in the crystal (called recombination sites). Free electrons in the conductance band can then relax at these recombination sites by emitting scintillation photons [5]. Because the excitation states at these impurity sites are at a lower energy than the conductance band (dependent on material), the scintillation photons can then be emitted at a lower, more easily detected energy [5]. Compared to organic scintillators, the high-Z composition of inorganic scintillators increases the probability of incoming gamma-rays to induce the photoelectric effect. As previously discussed, the photoelectric effect induces full energy deposition into the material. This improves the ability to perform gamma spectroscopy, which will be discussed in Section 1.6.2.

1.5.1 NaI(Tl) Radiation Detectors

Thallium activated sodium iodide, NaI(Tl), is one of the oldest and most common inorganic scintillators used in the radiation detection field. The scintillation photons have a maximum emission probability with a wavelength of roughly 415 nm [5]. NaI(Tl) also have a relatively high specific gravity of 3.67 and scintillation light yield of 38000 photons per MeV of energy absorbed [5]. Furthermore, they are relatively easy manufactured into many large shapes and

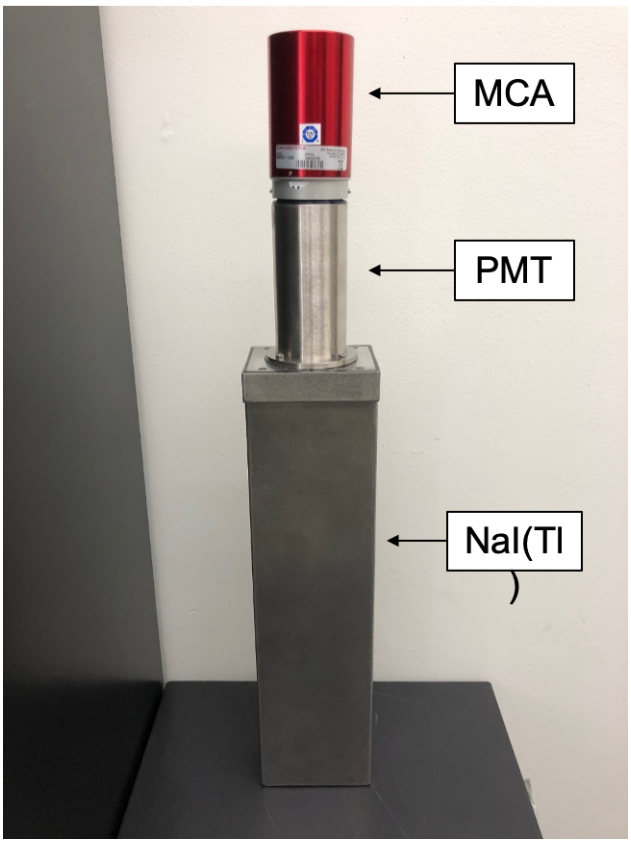


Figure 1.6: A 2"×4"×16" NaI(Tl) detector with major parts labeled.

sizes, a feature that man inorganic scintillators do not have [5]. Their ease in manufacturing, use of easily acquired materials, and many decades of research behind them makes NaI(Tl) detectors cheap compared to many other inorganic scintillators [5].

With regards to its disadvantages, NaI(Tl) is a highly hygroscopic and thus must be sealed into an air-tight container immediately after manufacturing. Furthermore, thallium is highly toxic to humans and other life, thus great care must be taken when using it to manufacture these crystals. Finally, compared to many other inorganic scintillators like $LaBr_3$(Ce), NaI(Tl) has a relatively low energy resolution (ability to discriminate between gamma rays of different energy).

Despite some of their disadvantages, NaI(Tl) are still widely used today. Their high light yield, detection efficiency, low cost, and ability to easily create large and difficult crystal geometries makes them very useful for a wide variety of nuclear security applications. Figure

1.6 shows an image of a 2"×4"×16" NaI(Tl) detector at Oak Ridge National Laboratory. This model of detector is the one used to collect and simulate the data described in Chapters 5 and 6.

Photon Measurement

In order to measure the scintillation photons that were released, the photons must be converted into an electric current. This generally follows a basic three-step process: conversion to electric current, current multiplication, and current to voltage conversion. Some considerations must be made when designing these three processes. One of these considerations is maintaining the original linearity of the energy/photon response throughout the photon to voltage conversion process. While the photon/energy response of NaI(Tl) is not perfectly proportional [9], the goal is to keep the linearity the same (or at least close) throughout the conversion process.

The first step is to convert the scintillation photons into electrons. This is normally done using a photocathode, which is a negatively charged wafer of material that converts photons to electrons via the photoelectric effect. This can be seen in Figure 1.7. The composition of the photocathode must be selected such that it is sensitive to the wavelength of the scintillation photon. For NaI(Tl), the material is often a bialkali [5]. Alternatively, wavelength shifters can be added that absorb the scintillation photons and re-emit them at the appropriate wavelength. Another important consideration when designing the photocathode is to maximize the number of scintillation photons that reach the photocathode. Usually, the detection crystal is surrounded by a specular or diffuse reflector to guide the photons towards the photocathode [5]. In some cases, especially when dealing with complex crystal geometries, light pipes that guide the photons towards the photocathode using total internal reflection are used [5]. Minimizing the loss of photons due to escape from the crystal is an important aspect of achieving maximal detection efficiency.

After conversion to an electric current, the current must be amplified at a typical factor of around 10^6 in order to be measured [5], thus, a device called a photomultiplier must be used. At the present time, there are two photomultiplier technologies that are often used. The first is the photomultiplier tube (PMT), which has been used for over a century. The other is the

smaller and more rugged silicon photomultiplier (SiPM), which has many practical benefits such as a smaller size, lower operating voltage, and higher durability. However, SiPMs are far more expensive, especially for larger volumes and thus are not used on the detectors used in this dissertation. For this reason, SiPMs will not be discussed to any extent in this dissertation.

PMTs generally consist of a glass vacuum tube containing a series of dynodes and an anode driven by a high voltage power supply and voltage divider circuit as shown in Figure 1.7. When electrons are emitted by the photocathode, they are accelerated to the next dynode by a high voltage potential, wherein they multiply via a process called secondary electron emission [5]. Each succesive dynode is held at a higher voltage potential by the voltage divider circuit, thus the electron acceleration/multiplication process continues until the electrons reach the anode. Standard PMT tubes can have current multiplication factors of over 10^7, depending on the dynode material [5].

Finally, the amplified current must be converted into a voltage. This is often done using an electric circuit called a transimpedance amplifier. A transimpedance, in its simplest form, is an operational amplifier (op-amp) with a current source between ground and the inverting terminal on the operational amplifier, with a resistor placed between the inverting and output terminals of the op-amp. The resistor is called the feedback resistor and can be used to set the gain value of the transimpedance amplifier. The transimpedance amplifier is often called the pre-amplifier in the radiation detection field.

1.5.2 Pulse Processing

Pulse Shaping

The output voltage pulse from the transimpedance amplifier generally takes the form of a tail pulse. The amplitude of the pulse is ideally proportional to the energy of the detected photon, thus, a circuit must be designed to extract the amplitude of this pulse and convert it into a digital value that can be processed. Unfortunately, the tail pulse is not ideal for digitization as it has a sharp peak that quickly decays, thus it is difficult to accurately sample the height of peak using discrete sampling. In order to combat this issue, a pulse shaping

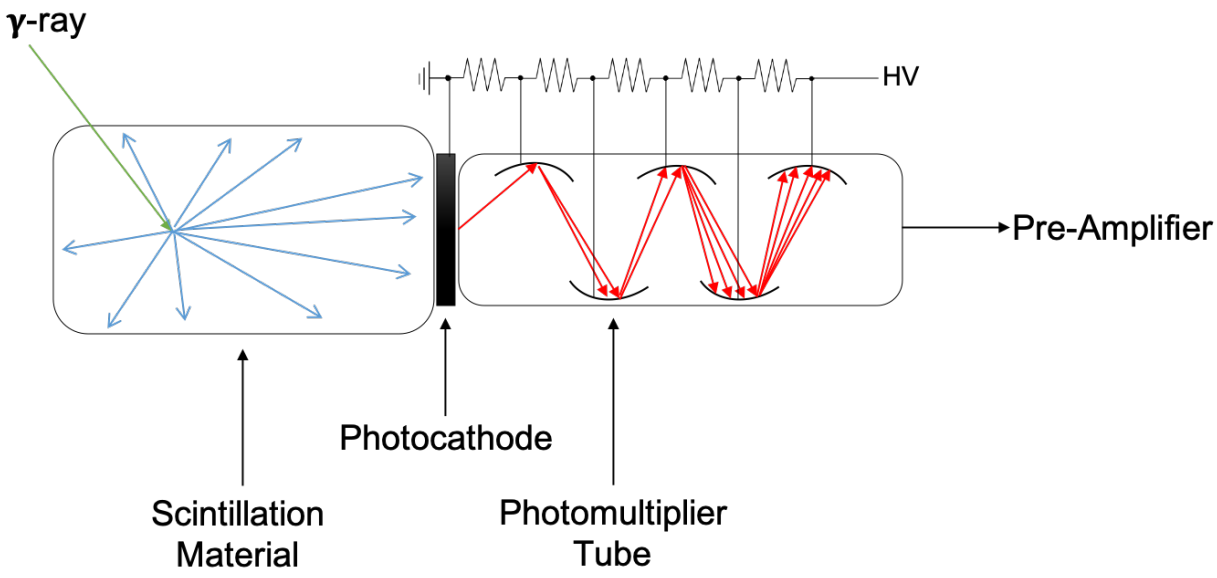


Figure 1.7: A simplified diagram of the process of converting a gamma ray into a current signal using a scintillator, photocathode, and photomultiplier tube.

circuit is added to the pulse processing chain that transforms the tail pulse into a Gaussian pulse so that the maximum peak height is maintained for a longer amount of time. This can be implemented using a simple RC low pass circuit. The output pulse can then be passed to another amplifier (such as an op-amp setup in a non-inverting amplifier configuration) to adjust the pulse height. Without pulse shaping, the measured peak voltage will not be accurate, thus leading to a lower energy resolution (lower ability to resolve the energy of the measured photon).

Digitization

After the pulse is amplified and shaped, the pulse heights can then be digitized using an analog to digital converter (ADC). The choice of the appropriate ADC is very important, particularly, the sampling rate, number of bits (determines the discretization of the ADC output), and the type of ADC technology used (e.g. direct conversion, sigma delta, and Wilkinson). The number of bits that the ADC has determines the discretization scheme of

the ADC. For example, an 8 bit ADC sampling voltages between 0 and 1000 V would be able to sample the voltage at a resolution of $\frac{1000\text{V} - 0\text{V}}{2^8\text{bits}} = 3.91$ V per bit.

In the radiation detection field, the ADC along with other electronics for data storage and pulse processing is often combined into a single package that connects to the PMT called a multichannel analyzer (MCA). An example of an MCA can be seen in Figure 1.6.

1.6 Data Analysis

There are many ways to process the data coming from the MCA. One method is to simply count the number of pulses output by the MCA over a period of time. The other is to histogram the output voltages in order to observed the energy spectrum of the detected photons. Both of these methods will be discussed in this section.

1.6.1 Counting

Many detection algorithms operate on a data format called the "gross count rate". The gross count rate G^* is simply a sum of the detection events (called counts) G, regardless of the energy of these counts, registered by the detector over a specified period of time called the integration time Δt, divided by said integration time. This is shown in Equation 1.6. Usually, the integration time takes the form of a sliding time window such that old data are discarded as the measurement continues.

$$G^* = \frac{G}{\Delta t} \tag{1.6}$$

Because radioactivity is a random process, it is important to consider the uncertainty in one's measurement of the gross count rate. When measuring the gross count rate for radiation source search applications, the primary source of uncertainty is usually the measurement of G and not Δt. When G is below about 20 counts, it can be modelled as a Poisson distribution [5]. Beyond 20 or 30 counts, G is well approximated as a Gaussian distribution [5]. As such, and when assuming no uncertainty in Δt, the uncertainty in the gross count rate is the standard deviation of the Gaussian distributed variable G given by Equation 1.7.

$$\sigma_{G^*} = \frac{\sqrt{G}}{\Delta t} \tag{1.7}$$

In a source search scenario, it is important to note that the gross count rate is *not* the same as the true activity of the source, A. First, we must consider that some photons that reach the detector may not register as a count. We call this the detector's intrinsic efficiency, ϵ_{int} and is given by Equation 1.8.

$$\epsilon_{\text{int}} = \frac{\text{\# of photons registered as counts}}{\text{\# of photons hitting detector}} \tag{1.8}$$

Likewise, not all photons emitted by the source will reach the detector. Consider an isotropic (4π) radiation source separated from a detector by a distance r. We can define a sphere with radius r centered at the position of the source. We can characterize the fraction of photons emitted by the source on the detector using a parameter called the solid angle, Ω. The solid angle is the projection of the detectors surface area A as seen by the source [5]. It is defined by Equation 1.9, where α is the angle between the normal to the detectors surface and the vector pointing from the detectors surface to the source [5]. Equation 1.9 is defined for a single face of the detector, thus it must be calculated for all faces of the detector that face the source. The solid angle for the 2"×4"×16" detector used in this dissertation is described in Reference [10]. With this in mind, we can define a term called the geometric efficiency ϵ_{geo} as given by Equation 1.10.

$$\Omega = \int_A \frac{\cos\alpha}{r^2} dA \tag{1.9}$$

$$\epsilon_{\text{geo}} = \frac{\text{\# of photons hitting detector}}{\text{\# of photons emitted by source}} = \frac{\Omega}{4\pi} \tag{1.10}$$

We must also consider any shielding between the detector and the source (including self attenuation) that is blocking radiation from reaching the detector. For the sake of simplicity, all shielding terms will combined into a single fraction S that represents the fraction of photons emitted by the source in the direction of the detector that do not get attenuated by shielding. By combining the concepts introduced in this section, we can relate

(approximately) the count rate in the detector G^* to the activity of the source A using Equation 1.11.

$$G^* \approx A\ \epsilon_{\text{int}}\ \epsilon_{\text{geo}}\ S \tag{1.11}$$

As can be seen, there are various ways to improve the uncertainty in the measured gross count rate. One way is to increase the geometric efficiency. There are two ways to do this; move the detector closer to the source or use a larger detector such that the solid angle is larger. The other way to improve uncertainty is to increase the integration time which allows more counts to be registered in the detector before estimating the count rate. Given, that the intrinsic efficiency is a function of the detector material, electronics, and energy of the incoming radiation quanta, for a given detector setup, this value is constant. For a radioactive source search scenario, the source position is unknown, thus integration time and detector size/shape are the main considerations for improving measurement uncertainty. It is important to select these parameters to achieve the correct balance between sensitivity and usability. For example, smaller detectors are easier to carry and conceal, however they may not be large enough to detect a source hidden in a distant building.

1.6.2 Gamma-ray Spectroscopy

Rather than integrating all of the detected counts into a single value, a histogram can be generated by integrating counts for pulse height bins. The resultant histogram is called a pulse height histogram, having B separate voltage bins each containing the number of counts registered in the detector within the voltage bounds. In order to analyze the energy spectrum of detected photons, i.e. perform gamma spectroscopy, the pulse height histogram (in units of Volts) must be converted into a photon energy histogram (units of keV, MeV, etc.). To do this, one performs a calibration by measuring pulse height histograms of a selection of radioisotope check sources with known gamma ray energies. If the energy response of the detector was linear, then only two photopeaks would be needed. However, as discussed earlier in this chapter, the energy response of these NaI(Tl) detectors is not linear, thus it is best to have a selection of photopeaks spanning the width of the energy spectrum that one

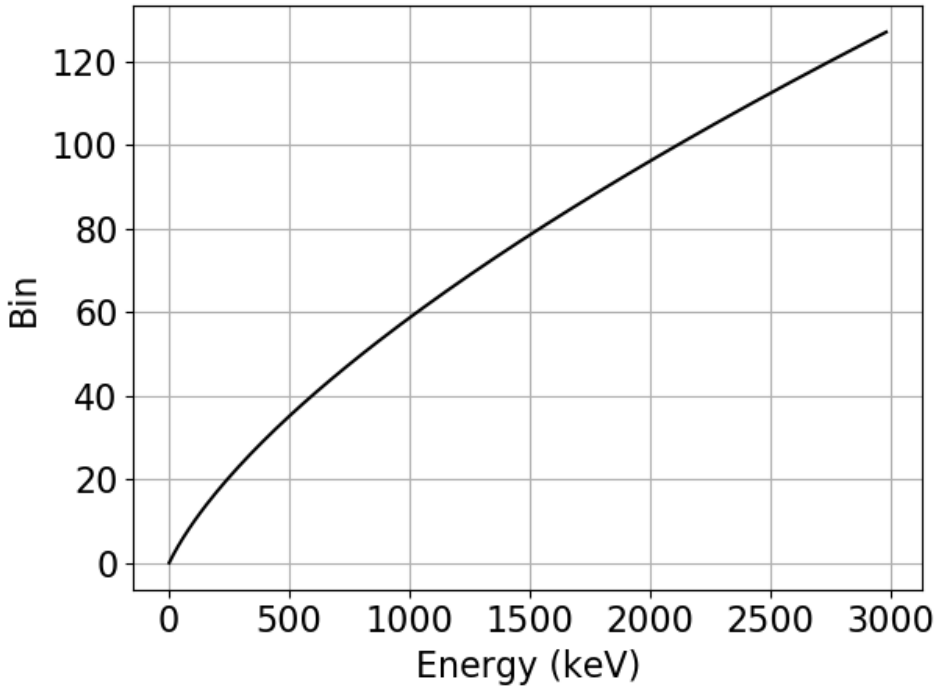


Figure 1.8: Example energy calibration for a 2"×4"×16" NaI(Tl) detector.

wishes to create. After collecting pulse height histograms of the calibration sources, a list of bin-energy pairs can be fit using a nonlinear function to convert the pulse height histogram into an energy spectrum. Many different fit functions are used by experts in the field and there does not appear to be a general consensus on what is the "best" calibration function, however a 2nd order polynomial is a popular choice [11]. Figure 1.8 shows an example energy calibration for a 2"×4"×16" NaI(Tl) detector for a 128 bin histogram.

Photon peaks in the energy spectrum for a perfect detector would take the form of a delta function, however, in reality the peak takes the form of a Gaussian distribution with a resolution defined as the full width at half maximum (FWHM) of the Gaussian peak divided by its mean value. There are many causes of this spread in detected photon energy, but one of the main factors is the statistical fluctuations in the number of photoelectrons generated at the photocathode [5]. Assuming the number of photoelectrons generated at the photocathode follows Poisson statistics, one would expect that the resolution of the detector

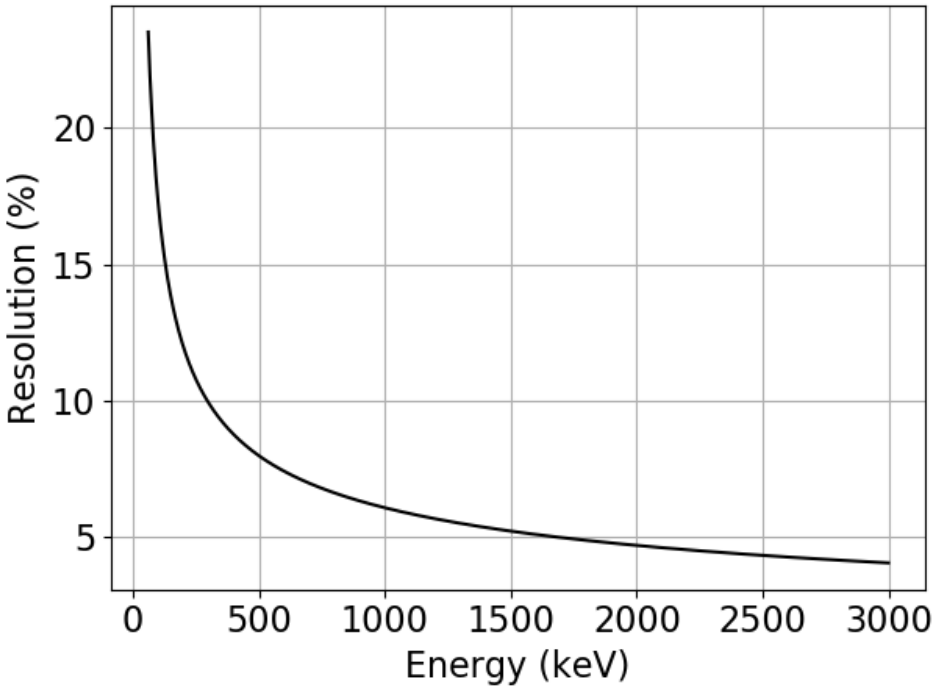


Figure 1.9: Example resolution calibration for a 2"×4"×16" NaI(Tl) detector.

would follow the inverse of the square root of the photon energy $1/\sqrt{E}$ [5] and as shown in Figure 1.9 this is indeed the case.

There are many cases throughout this dissertation in which we will refer to "features" in the gamma ray spectrum, thus it is important to discuss these features and their characteristics. The spectral features, as defined in this dissertation, arise from the gamma ray photon interactions in matter that were previously discussed in this chapter, that is: photoelectric absorption, Compton scattering, and pair production.

Photoelectric Effect

From the perspective of performing gamma spectroscopy, the photoelectric effect has some desirable properties. In particular, because the entire energy of the photon is deposited into the material, the resulting feature in the spectrum is a single Gaussian peak centered at the energy of the photon. This is one of the main reasons that NaI(Tl) is such a popular

scintillation material, because its high-Z value makes the photoelectric effect the dominant interaction mechanism for a larger gamma ray energy range with respect to many other materials.

Photoelectric absorption is often accompanied by the release of a characteristic X-ray from the absorber material [5]. For high-Z materials, the energy of this x-ray can be high enough to penetrate the scintillator wall and deposit its energy in the detector [5]. This is often a problem encountered when surrounding a detector in lead shielding, which has a characteristic photoelectric X-ray of 72 keV [12].

Compton Scattering

Compton scattering is a bit more problematic in terms of gamma spectroscopy due to the fact that the photon energy follows a distribution characterized by the photon/electron scattering angle. As previously discussed and shown in Figure 1.4, the Compton energy deposition distribution is maximum near a scattering angle of 180 deg. This is the point of maximum photon energy deposition in the detector via Compton scattering and the resultant feature in the spectrum is called the Compton edge. The region between no energy deposition in the detector and the Compton edge is called the Compton continuum and will also be present as a feature in the spectrum.

There is also a chance that the photon will undergo a Compton scattering interaction outside of the detector, with the scattered photon then interacting with the detector. This is called a backscatter and is often a significant part of the background gamma ray spectrum in the lower energy region (less than a few hundred keV). Because background radiation gamma rays are emitted with a broad range of energies (discussed in Section 1.7) and the energy of the scattered photons from Compton scattering also follows a wide distribution, the background backscatter hump (often called the background continuum) spans several hundreds of keV, often peaking around 200 keV [5]. This can be seen in Figure 1.12 in Section 1.7.

Pair Production

Finally, if the source photons are at least 1.022 MeV in energy, then the photon may undergo a pair production interaction along with a subsequent electron/positron annihilation which leads to the emission of a pair of 511 keV photons. There are five potential outcomes of this interaction in the detector and thus four different resultant features in the spectrum, as listed below:

1. The gamma ray of energy $h\nu$ undergoes pair production in the detector, depositing an energy of $h\nu - 2m_ec^2$ in the detector. Both annihilation photons escape the detector without interaction. The resultant feature in the spectrum is a Gaussian peak centered at $h\nu - 2m_ec^2$ called the double escape peak.

2. The gamma ray of energy $h\nu$ undergoes pair production in the detector, depositing an energy of $h\nu - 2m_ec^2$ in the detector. Only one of the annihilation photons escapes the detector without interaction, with the other depositing m_ec^2 of energy in the detector. Because the pair production and positron annihilation occur more or less in coincidence, the resultant feature in the spectrum is a Gaussian peak centered at $h\nu - m_ec^2$ called the single escape peak.

3. The gamma ray of energy $h\nu$ undergoes pair production in the detector, depositing an energy of $h\nu - 2m_ec^2$ in the detector. Both annihilation photons interact in the detector, each depositing m_ec^2 of energy. Because the pair production and positron annihilation occur more or less in coincidence, the resultant feature in the spectrum is a Gaussian peak centered at $h\nu$, the original energy of the gamma ray.

4. The gamma ray of energy $h\nu$ undergoes pair production outside of the detector. One of the annihilation photons interacts in the detector, depositing m_ec^2 of energy in the detector. The resultant feature is a Gaussian peak centered at m_cc^2.

5. The gamma ray of energy $h\nu$ undergoes pair production outside of the detector. Both of the annihilation photons interacts in the detector, depositing $2m_ec^2$ of energy in the detector. The resultant feature is a Gaussian peak centered at $2m_cc^2$. because the two annihilation photons are emitted at opposing directions, this tends to only

occur if the pair production occurs inside of a well detector cavity or other hollow detector geometry where it is possible for both annihilation photons to interact with the detector volume.

Pulse Pileup

Due to the finite sampling frequency of the detector and its associated electronics, there is a possibility that more than one photon interaction will be registered as a single interaction with energy equaling the sum of the original energies. This process is called peak pileup and often occurs for high count rates in the detector. This can occur regardless of the mode of photon interaction in the detector. Significant pulse pileup is not often encountered in the HFIR/REDC dataset described in Chapter 3, aside from a few particular radiological transfer events.

1.7 Radiation Background

One of the biggest challenges in radioactive source search for nuclear security is source/background discrimination. The radiation background in real world environments (that is, outside of the controlled conditions of a lab) is highly dynamic due to a variety of natural and man-made influences which will be discussed in this section. This problem is especially challenging in urban environments and even more so for mobile detection surveys for reasons that will soon be discussed. Understanding the sources of this dynamic radiation background is important to understanding the challenges in designing and deploying radiation detection algorithms. Three contributors to the dynamic radiation background are discussed herein: the presence of naturally occurring radioactive material (NORM), wet deposition of ^{222}Rn due to precipitation, and background suppression due to physical objects in the detection area called "clutter."

1.7.1 Naturally Occurring Radioactive Material

Both natural and artificial building materials contain highly varying levels of naturally occurring radioactive material (NORM), particularly ^{238}U, ^{232}Th, ^{40}K (often abbreviated

Table 1.1: KUT concentrations in a selection of common building materials, measured at the Fort Indiantown Gap Combined Arms Collective Training Facility [13]. Table and data directly adapted from Reference [13].

Material	Concentration (Bq/kg)		
	^{40}K	^{238}U	^{232}Th
Asphalt	97.52	24.34	3.96
Cinder Block 1	156.98	11.78	10.82
Cinder Block 2	200.05	14.07	7.90
Cinder Block 3	192.85	13.51	12.17
Cinder Block 4	318.62	15.75	13.25
Cinder Block 5	112.81	9.89	7.25
Concrete 1	236.27	17.93	10.33
Concrete 2	190.17	14.81	9.22
Concrete 3	231.26	18.21	10.20
Concrete 4	219.30	18.22	9.63
Concrete 5	241.28	17.64	10.45
Concrete 6	204.52	17.46	8.68
Gravel	51.00	23.02	3.58
Soil	412.63	25.88	37.75

as KUT), and their daughter products [13, 14, 15, 16, 17]. All three of these radioisotopes are primordial radionuclides with half lives on the order of 10^9. Both ^{238}U and ^{232}Th have large decay chains with many radioactive daughters, as shown in Figures 1.10 and 1.11, respectively. ^{40}K decays directly either to stable ^{40}Ca or ^{40}Ar.

Table 1.1 shows the concentrations of ^{238}U, ^{232}Th, ^{40}K in various common building materials as measured at the Fort Indiantown Gap Combined Arms Collective Training Facility (CACTF) by a team of researchers at ORNL [13]. The relative concentrations of these three radioisotopes is highly variable even among different samples of the same type of material. Some natural materials, especially igneous rocks such as granite, can contain higher levels of NORM when compared to other materials. KUT concentrations for various granites used as building materials can be seen in Reference [18].

Figure 1.12 shows a typical background gamma ray spectrum collected by a 2"×4"×16" NaI(Tl) detector. Most of the notable features in the spectrum are labeled. As can be seen, many of the photopeaks in the spectrum arise from the longer lived beta emitters

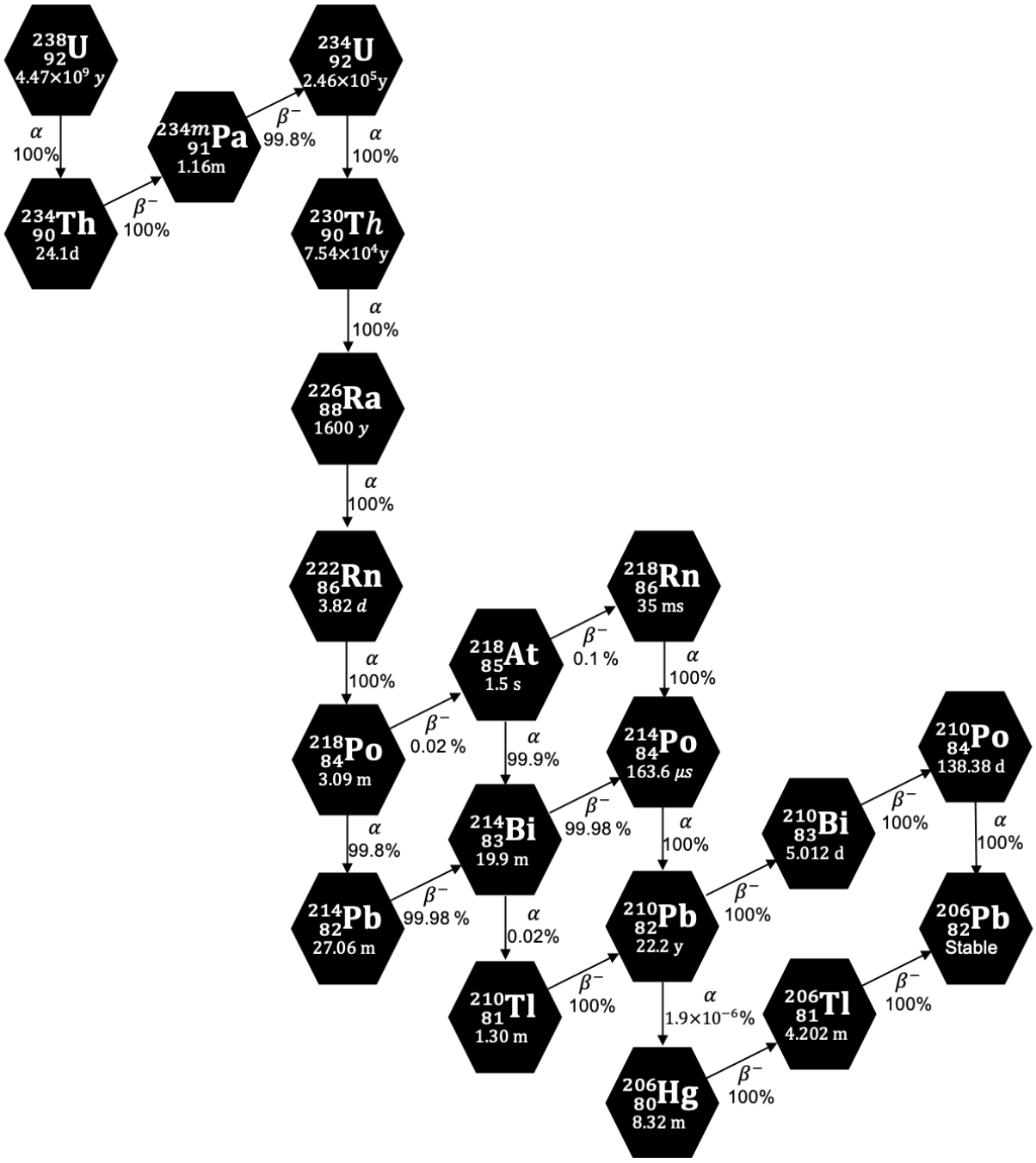


Figure 1.10: Decay chain for ^{238}U. Data from NNDC [6].

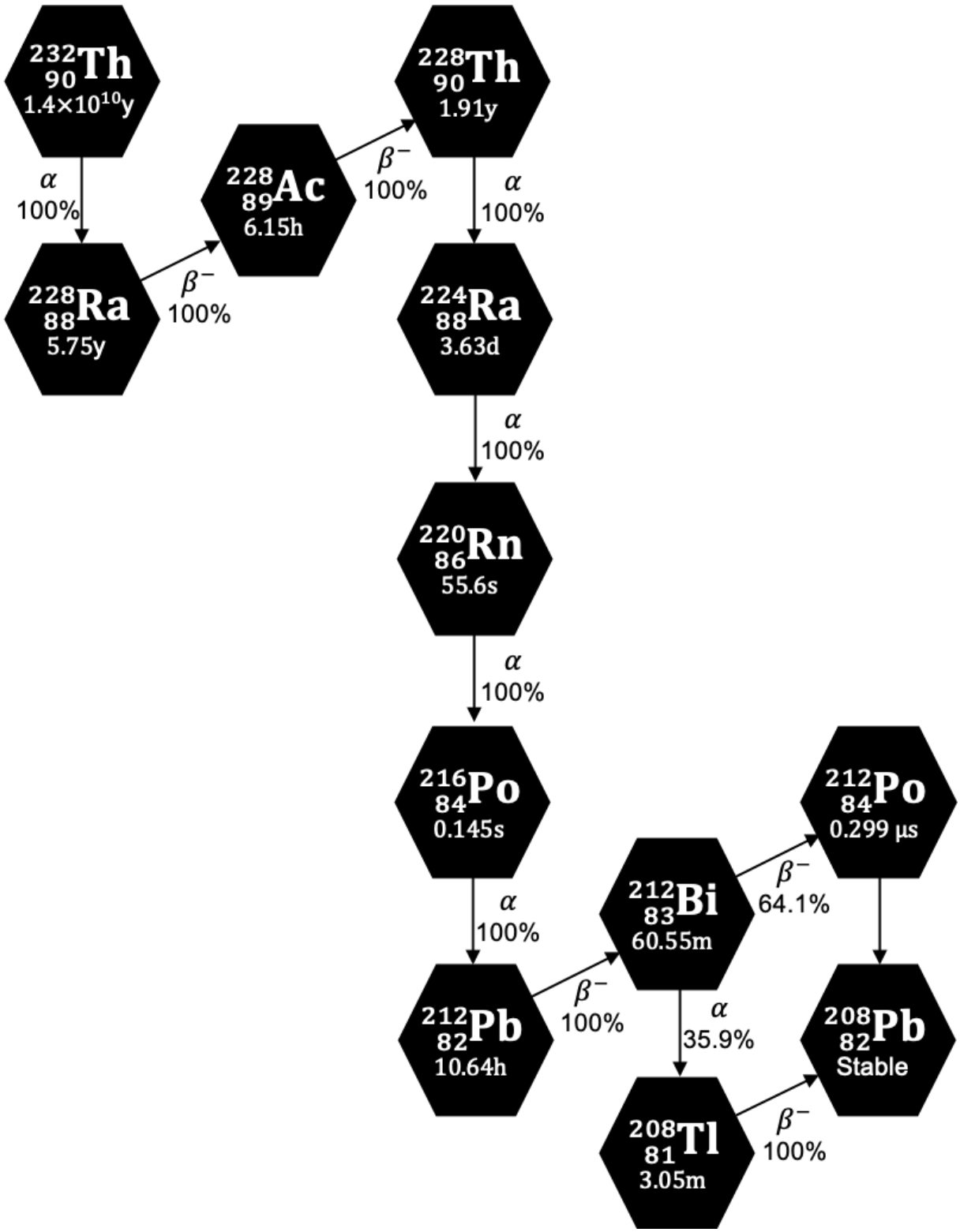


Figure 1.11: Decay chain for ^{232}Th. Data from NNDC [6].

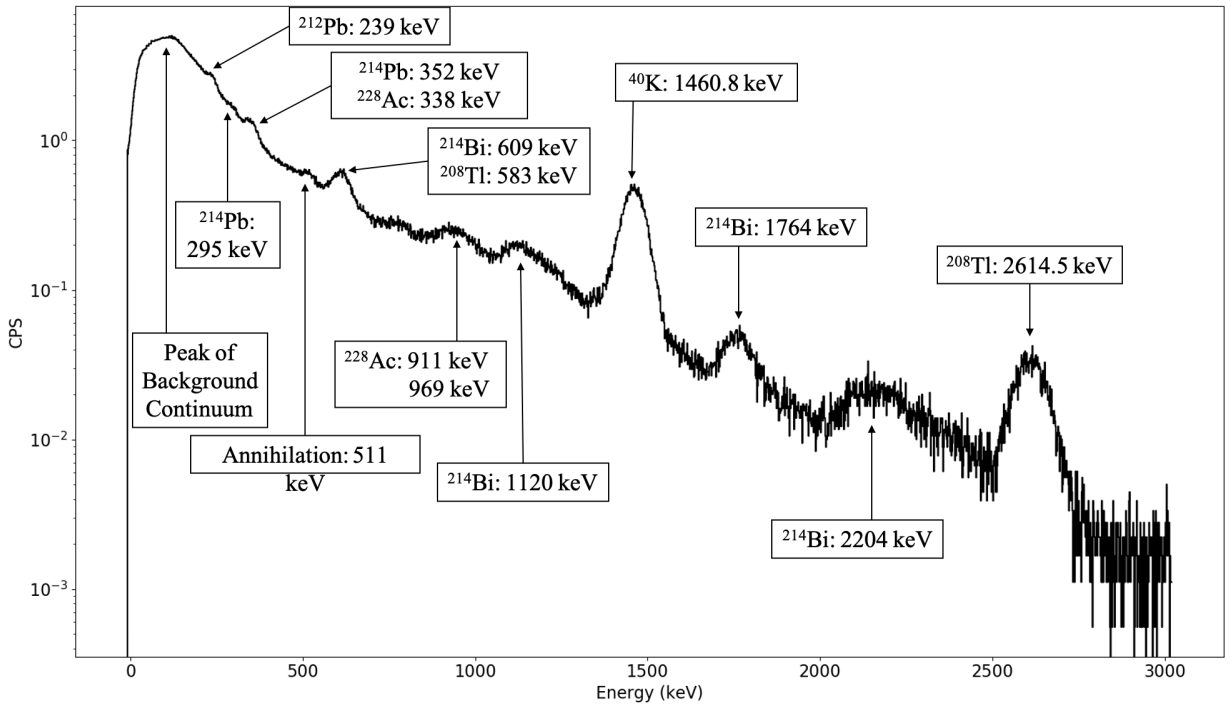


Figure 1.12: Typical background gamma ray spectrum collected by a 2"×4"×16" NaI(Tl) detector. Decay energy data from NNDC [6].

in the ^{238}U and ^{232}Th decay chains, particularly: ^{214}Pb (^{238}U series), ^{214}Bi (^{238}U series), ^{228}Ac (^{232}Th series), and ^{208}Tl (^{232}Th series). The 1460 keV photopeak from ^{40}K is also very prevalent in the spectrum. The relative and absolute peak areas of these features in the background spectrum vary between different materials according to the data in Table 1.1. This variability can be a challenge to radiation detection algorithms as the algorithm must be able to differentiate between this natural variation and real source signatures. Some examples of this will be demonstrated in a future section.

1.7.2 Precipitation

Another significant contributor to the dynamic background is the wet-deposition of ^{214}Pb and ^{214}Bi during precipitation [19, 20, 21]. As part of the ^{238}U decay chain, ^{222}Rn gas is present in the atmosphere. When clouds are forming in the atmosphere, ^{222}Rn can alpha decay to ^{218}Po and act as a nucleation sight for cloud droplets [21]. When precipitation from

the cloud occurs, ^{218}Po along with its two longer lived daughter products ^{214}Pb and ^{214}Bi are deposited on the ground [21]. Because ^{214}Pb and ^{214}Bi are both gamma emitters, as shown in the spectrum on Figure 1.12, the gross count rate as measured by detectors on the ground can increase significantly [19]. The initial increase in count rate decays exponentially after the precipitation subsides and remains measurable for a period of a few hours [19].

In order to demonstrate the effects that rain have on the gamma-ray spectrum, a gamma-ray spectrum was generated using a 2" ×4" ×16" NaI(Tl) placed outdoors during a rain event. The spectrum was integrated over a period of approximately 1.5 hours around the peak of the precipitation event. Also, a background spectrum of the same length was taken just prior to the start of the precipitation event and subtracted from the spectrum. The resultant spectrum is shown in Figure 1.13 with the relevant photopeaks labeled. As can be seen, there is a significant increase in the count rate for gamma-rays emitted by ^{214}Pb and ^{214}Bi, but not other isotopes. The peak increase in count rate for this particular event was approximately 41% (or 593 cps) higher than the mean background count rate, which is approximately 117 times higher than the standard deviation in the background count rate. Such an increase can cause false alarms in many types of radiation detection algorithms, especially those that do not consider spectral information in their detection decision.

1.7.3 Clutter

Another contributing factor to the dynamic background is the presence and movement of high density objects in the detector's field of view, which we will collectively call "clutter." It is natural to consider the fact that a large object such as a truck or other vehicle may inhibit a detection system's ability to detect a source simply due to the fact that said object acts as a shield that reduces the signal measured by the source. However, we must also consider the fact that such objects also suppress the background signal, the measurement and estimation of which is integral to many detection algorithms. Let's consider for example a detector moving throughout an urban street with buildings made of different materials. Imagine that as the detector moves along the street, a large granite building is present on the left, however, there is a large semi that blocks the detector's view (shields the background from the granite building). When the detector moves past the truck and the building comes

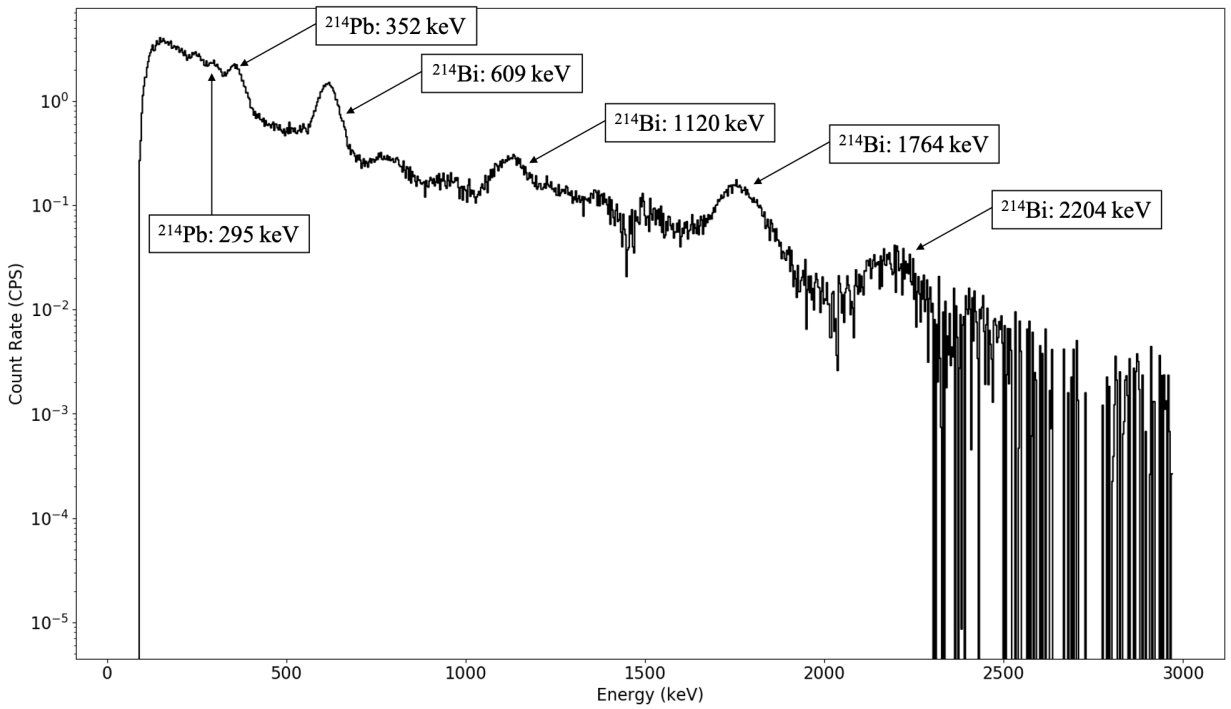


Figure 1.13: Background subtracted spectrum demonstrating the increase in count rate from ^{214}Pb and ^{214}Bi during precipitation. Integration time is 45 minutes. Decay energy data from NNDC [6].

into view, a spike in the detector's count rate will occur that can trigger a false alarm. This can also occur for a static detection scenario. Consider that the same detector was placed on the street with the semi truck parked between the detector and building. When the semi truck drives away, a spike in the count rate can occur. Considering the fact that many existing radiation detection algorithms (which are discussed in the next section) rely on an estimate of the background in order to make a decision, such clutter-induced background dynamics can lead to a false alarm.

The effects of clutter on static radiation portal monitoring systems have documented and studied for over a decade [22, 23]. Only recently however have researchers began to quantify the effects of different types of clutter in more complex environments such as urban city streets [24, 25]. As expected, the general trend is that background suppression due to

clutter increases as the clutter object gets closer to the detector and also increases for higher Z materials [24].

In addition to changing the count rate, clutter can also have drastic effects on the gamma ray spectrum. As was discussed in Section 1.3, photons at the energies we are considering most often interact with materials via three main processes: the photoelectric effect, Compton scattering, and pair production. Background attenuation in clutter objects from the photoelectric effect is the simplest to understand, with the background photon simply being absorbed by the clutter and not the detector. The result is a suppression of the full energy photopeak in the gamma ray spectrum, which reduces the signal to noise ratio (SNR) of that photopeak in the spectrum. The characteristic X-rays emitted by the clutter material after photoelectric absorption may also increase and interfere with signals from other radiosotopes.

Unfortunately, the effect of Compton scattering of background photons in the clutter is less straightforward. Rather than being fully absorbed by the clutter and reducing the full energy photopeak, the background photon can be backscattered by the clutter, resulting in both a reduction of the full energy photopeak and an increase in the background backscatter continuum in the spectrum. Further, the increase in the background backscatter continuum can overcome full energy photopeaks in the lower energy region, further reducing the ability to discriminate radioisotopes. This leads to a decrease in both the SNR for that particular photopeak and also for other photopeaks that are present in the region of backscatter continuum. Finally, higher energy photons that would normally not be detected because they have energies above the detector's upper level discriminator (set to 3 MeV in most of the data in this dissertation) may now backscatter and deposit energy into the detector at the higher energy regions, thus also reducing the SNR in the higher energy region.

The net effect of clutter on the measured gamma ray spectrum is a decrease in SNR which can be limited to just a single radioisotope in the case of the photoelectric effects or an entire energy region in the case of Compton scattering. Some demonstrations of how clutter affects gamma ray spectra will be discussed and demonstrated in further detail in Chapter 6. Overall, clutter must be heavily considered when designing a radiation detection

algorithm that utilizes either the gross count rate or the gamma ray spectrum, as it has significant effects on both types of data.

1.8 Prior Work

There are two primary types of source detection algorithms, those that operate on the gross count rate and those that operate on the gamma ray spectrum. Both methods have their advantages and disadvantages and are useful in different scenarios. The gross count rate can be used for observing the time variation of the total radiation level in a source search area. By monitoring the gross count rate while moving throughout the search scene and comparing it to the expected background count rate, one can in certain circumstances detect the presence of a source. As discussed in Section 1.7 however, variations in the detector count rate are often not indicative of the presence or absence of illicit radioactive sources as the radiation background is highly dynamic. Another potential problem with using gross count rate as a detection metric is that the lack of energy information makes it impossible to perform radioisotope identification/discrimination using the detector data alone without supporting contextual information. This being said, some detection algorithms based on the gross count rate are suitable in environments with a slowly changing radiation background count rate and certainly benefit from requiring lower integration times in order to achieve good counting statistics when compared to spectroscopic methods. This can be beneficial in scenarios where the window of opportunity to detect a source is very short, where methods that require longer integration times may fail due to integrating over too long a period of time.

Methods that utilize the full gamma ray spectrum naturally have much more information available in which to make detection decisions. By considering the entire energy spectrum, the algorithm can be designed to detect (and not detect) specific radioisotopes using gamma ray spectroscopy. This can make it much easier to filter out false alarms due to NORM. Further, a detection event from a spectrum-based algorithm inherently conveys more information to the end user that may be useful for the mission. Knowing the energy region (or regions) that triggered the detection event may help the end user to identify the potential source. While spectrum-based detection algorithms have many advantages over

gross count rate based algorithms, they are not without a few disadvantages. First, they are generally more complex to implement. Also, due to having to integrate counts across a histogram rather than a single value, higher integration times than are used for gross count rate algorithms are usually necessary in order to build up sufficient measurement statistics.

This section will briefly describe a selection of prior work on the development of automated radiation detection algorithms. Algorithms based on the gross count rate, the full gamma ray spectrum, and also region of interest count rates will be introduced.

1.8.1 K-Sigma

Introduced by Lloyd A. Currie in 1968, the k-sigma algorithm is arguably the simplest detection algorithm for radiation detection [26]. As its name entails, the k-sigma algorithm generates a detection event D at the point at which the gross count rate G increases above k times the standard deviation in background count rate σ_B plus the mean background count rate μ_B according to Equation 1.12. K-sigma works well in cases where the background, B can be accurately estimated using μ_B and σ_B. This is usually only the case in a controlled laboratory environment where B has little variation, but does not adapt well to dynamic background environments. As σ_B increases, the detection sensitivity with respect to the source activity S decreases. K-sigma can also be adapted to slow-moving variations in B by using a rolling time-window estimation of μ_B and σ_B. This helps to account for slow drifts in background count rate but may have the effect of washing out slow-moving sources as well.

$$D = \begin{cases} 0 & G < \mu_B + k\sigma_B \\ 1 & \text{otherwise} \end{cases} \tag{1.12}$$

1.8.2 SPRT

The sequential probability ratio test (SPRT) is simply a sequential form of the log likelihood ratio test used to compare two statistical models based on their log-likelihood ratios [27]. SPRT has been used for many radiation detection applications on a variety of different detection platforms with a good level of success [28]. In the case of radiation detection, the null hypothesis H_0 is that the detected count rate λ can be attributed to background

terms alone: i.e. that λ is equal to the nominal background count rate λ_0. The alternative hypothesis H_1 is that λ has a source term present: i.e. that λ is equal to the nominal background plus source count rate λ_1. For a sequence of observed count rates where x_i is the detector count at the ith interval with an integration time of ΔT, the log-likelihood ratio Z_i is given by Equation 1.13 [28].

$$Z_i = \log\left(\frac{\lambda_1}{\lambda_0}\right) x_i - \Delta T(\lambda_1 - \lambda_0) \tag{1.13}$$

The test runs for each count sequentially such that for n observations, $Z_n = \sum_{i=1}^{n} Z_i$. The decision is made by comparing Z_i to two thresholds a and b [27, 28]. a and b can be calculated using Equation 1.14 by setting target false positive and false negative probabilities, α_0 and β_0 respectively [27, 28]. The null hypothesis (background only) is accepted when $Z_n <= b$ and the alternative hypothesis (source) is accepted when $Z_n >= a$. While $a > Z_n > b$, the test continues until a decision can be made

$$a = \log\left(\frac{1-\beta_0}{\alpha_0}\right) \quad , \quad b = \log\left(\frac{\beta_0}{1-\alpha_0}\right) \tag{1.14}$$

1.8.3 ROI Countrate

One method of handling dynamic background count rates and increasing specificity of the algorithm is to consider only specific portions of the detector's energy spectrum. This method is called the region of interest (ROI) method. In general, all of the same algorithms applied to gross count rate data can also be applied to ROI count rate methods. The question becomes how to accurately estimate the countrate within a given ROI which is usually centered around a gamma-ray photopeak of interest, e.g. 662 keV for ^{137}Cs. The background count rate underneath the photopeak must be estimated, one common method being to use two energy windows surrounding either side of the photopeak to estimate the background continuum [29]. Given that the background continuum is roughly linear with respect to energy, this method yields decent results [29]. The ROI method is not without its problems however. Detector gain drift due to temperature and other environmental factors can cause photopeaks to drift partially or completely outside of the ROI energy windows. Further,

this method assumes that the energy windows used to estimate the background continuum do not overlap with other photopeaks from the source. This places a hard limit on which photopeaks can be used for the algorithm as one must select peaks that are separated enough to prevent overlap and detectors with an energy resolution capable of discriminating peaks of interest. Further, Compton scattering of the photons from clutter or other shielding sources can prevent the development of photopeaks within specified ROIs.

1.8.4 PCA and Matched Filter

Principal components analysis (PCA) is a procedure that performs the eigendecomposition of the covariance matrix $\mathrm{X}^T\mathrm{X}$ where X is a set of training data containing M measurements of size N. In this case, N is the number of bins in each gamma-ray energy spectrum and M is the number of spectra in the training set where each spectrum is considered to be background. PCA performs an orthogonal translation on X such that the variance of X in the direction of the first component is maximized. Each succeeding component must be orthogonal to the preceding component and the number of components is set by the user. Once the principal components are extracted from the training data, they can be used to reconstruct a given test spectrum S_i to produce S_i' by projecting S_i onto the principal components and back again. The norm of the residual between S_i to produce S_i' can then be used to establish whether or not an anomalous component is present in the spectrum.

One potential problem of PCA for some systems is that the covariance matrix $\mathrm{X}^T\mathrm{X}$ is size M×M, thus as the number of measurements in the training set increases, the size of $\mathrm{X}^T\mathrm{X}$ grows exponentially (as well as the computational complexity). Nevertheless, PCA has been successfully applied to the detection problem for portal monitor systems [30].

A slightly different alternative to PCA is called the matched filter. The matched filter assumes that the user has adequate knowledge of the threat source spectra in the form of a source template library [31]. The matched filter performs standard PCA to extract the non-background components of the test spectrum then calculates the inner product between the reconstruction and the source templates in the source template library.

1.8.5 Non-negative Matrix Factorization

Non-negative matrix factorization (NMF) is a linear algebra technique in which a non-negative matrix V of size $M \times N$ is factorized into two smaller non-negative matrices W (size $M \times r$) and H (size $r \times N$), such that $W\,H = V$. There is generally no single solution to the problem, so numerical techniques are used to approximately solve the problem, $W\,H \approx V$. The size of the smallest dimension of matrices W and H, r, is set by the user and must be less than $\min(M, N)$. This r value can be considered analogous to the number of components in PCA as described in 1.8.4. The non-negativity property of NMF is considered to be more physical compared to PCA when used for radiation anomaly detection [32].

The solution to $W\,H \approx V$ can be found my minimizing the Kullback-Leibler Divergence (KLD) between V and $W\,H$. The Kullback-Leibler Divergence $D_{\mathrm{KL}}(P\,||\,Q)$ measures the relative information entropy between two probability distributions $P(x)$ and $Q(x)$ where both distributions share the same probability space χ, as defined in Equation 4.6 [33]. Solving the problem using the KLD has been shown to be identical to the probabilistic latent semantic analysis clustering technique [34].

$$D_{\mathrm{KL}}(P\,||\,Q) = \sum_{x \in \chi} P(x) \log \left[\frac{P(x)}{Q(x)} \right] \tag{1.15}$$

After performing the minimization, each column i in matrix W is a single basis vector, denoted w_i. When V contains background-only gamma-ray spectra, the basis vectors can be visualized and patterns representing recognizable background components can often be seen [32]. Examples of such components include ^{40}K, ^{222}Rn, and components resembling typical background Compton continua. When a new gamma-ray spectrum S_n is collected, the basis vectors can then be linearly weighted and added together to create a reconstructed spectrum S_R using on the basis vectors, $S_R = \sum_{i=0}^{r} w_i\,p_i$, where p_i is the parameter to fit for basis vector w_i. This can be done using a variety of optimization routines.

After performing the optimization procedure, a reconstruction error function can be used to evaluate how well the basis vectors were able to reconstruct the input spectrum. If the reconstruction error is above a certain threshold, the input spectrum is considered to contain non-background components, and vice versa. The ideal selection for the reconstruction error

function would be noise-invariant (insensitive to changes in statistical noise from dynamic gross count rate) and sensitive to non-background spectral features (photoelectric peaks, Compton edges, etc.). As will be discussed in Chapter 4, the KLD is a decent choice for a spectral reconstruction error function.

1.8.6 NSCRAD

The nuisance-rejection spectral comparison ratio anomaly detection (NSCRAD) algorithm is an ROI-based technique developed by a team of researchers at Pacific Northwest National Laboratory (PNNL). The algorithm utilizes a set of measured background spectra labelled the benign source population (BSP) dataset to define the ROI count rate characteristics of background spectra. Consider two spectra both belonging to the BSP each with N-bins, with one representing a reference spectrum and the other at time k, $\bar{B}_0$ and B_k respectively. Under the assumption that the shape of background spectra remains relatively stable and that the two separate background spectra can be related by a single scaling factor, then Equation 1.16 can be used to obtain the counts in bin 1 from the counts in bin j for either spectrum [35].

$$B_{1_k} = \left(\frac{\bar{B}_{1_0}}{B_{j_0}}\right) B_{j_k} \tag{1.16}$$

If the spectra collected at time k contains components not belonging to the BSP, then the spectra cannot be related by a single scale factor and will incur an error, α_{1j_k}, which is defined as the spectral composition ratio (SCR) [35]. This is defined in Equation 1.17 where C_{j_k} and C_{1_k} are the measured counts in bin j and 1 of the non-BSP spectrum, respectively [35].

$$\alpha_{1j_k} = C_{1_k} - \left(\frac{\bar{B}_{1_0}}{B_{j_0}}\right) C_{j_k} \tag{1.17}$$

The authors of NSCRAD settled on 12 separate ROIs. These ROIs are not defined manually, rather, they are calculated using a multivariate optimization technique in order to maximize sensitivity to a set of target threat sources [35]. Later devleopments of NSCRAD adopted the use of simulated annealing for performing this optimization [36]. In order to

run the algorithm in a real-time sequence of spectral data, a Kalman filter is used to predict the appropriate reference spectrum $\bar{B}_0$ to compare the spectrum in question [35]. This step is necessary in order to account for systematic variations in the count rate arising from a variety of sources. The SCR can then be calculated and thresholded to produce an alarm.

1.8.7 WAVRAD

The wavelet-assisted variance reduction anomaly detection (WAVRAD) algorithm was developed by a team of researchers at the Remote Sensing Laboratory. A mathematical formulation of the algorithm is not published, however the conceptual background has been vaguely described [37]. The algorithm works in real time by sequentially performing a continous wavelet transform on the spectra in order to perform variance reduction on the spectra [37]. The merit function for the wavelet transform is not defined. The algorithm then computes an expectation based on a rolling series of previously collected data and compares this expectation to the actual measured spectra in order to establish an anomaly alarm [37]. This method has the advantage that it is resistant to false alarms arising from *slowly* changing background and detector parameters (such as gain drift) [37].

1.8.8 Autoencoders

This dissertation presents the first known use of an autoencoder neural network for radiation anomaly detection, however there are a few instances in which they have been applied to various other applications in the radiation detection field. For an in-depth overview of autoencoders and deep learning in general, refer to Chapter 2.

Bellinger et. al. developed a denoising autoencoder for generating synthetic threat source gamma ray spectra [38]. This autoencoder was created for the purpose of upsampling the threat source class in training datasets used to train machine learning-bsased algorithms for real time monitoring of gamma ray spectra [38]. Due to class imbalance between the background and threat source data in many of these datasets, this autoencoder enabled the author's to generate realistic synthetic threat source spectra that balanced the datasets [38].

Another team investigated the use of an autoencoder combined with a standard neural network for calculating dose rate correction factors in handheld spectroscopic radiation detectors [39]. The team used an autoencoder trained on a dataset containing gamma ray spectra from different sources to perform dimensionality reduction of acquired spectra [39]. The dimensionally-reduced representation of the spectrum created by the autoencoder is then used by a standard neural network to predict dose rate correction factors in the handheld device [39].

Finally, another group used autoencoders to reconstruct the Compton edges in noisy, Gaussian energy broadened (GEB) gamma ray spectra from plastic scintillators [40]. The autoencoder was trained using a synthetic Monte Carlo dataset containing pairs of GEB and non-GEB gamma ray spectra for a variety of sources [40]. The final model showed good reconstruction performance for noisy real-world spectra despite being trained on synthetic data [40].

Chapter 2

Deep Learning

Deep learning is a branch of machine learning that utilizes multilayer neural networks that learn to perform tasks based on feature learning methods. Their application is far reaching and extremely broad, with deep neural networks that are able to perform tasks all the way from object detection and identification to autonomous vehicle control and climate forecasting. This technology has become a staple part of everyday life, being used for tasks that traditional computing methods tend to struggle such as facial recognition, natural language processing, autonomous vehicle navigation, and modelling/forecasting of complex dynamic systems (such as climate). This chapter will give a brief introduction to deep neural networks with special emphasis on a specific type of deep neural network called an autoencoder, which in the past decade has proven to be one of the most powerful methods for performing anomaly detection in highly complex data sets. This is a very broad technology and thus this chapter does not intend to be a all-encompassing introduction to the material, rather, it intends to give the reader enough information in order to understand the algorithm described in the next chapter.

2.1 Artificial Neural Networks

Artificial neural networks are a type of connectionist computing system very loosely inspired by the workings of the human brain. The fundamental building block of a neural network is the artificial neuron (sometimes called a unit), which is shown in Figure 2.1. The artificial

neuron integrates signals from other neurons (x_1, x_2, x_3, ...) at its input, with each of these inputs being multiplicatively scaled by their associated connection weights (w_1, w_2, w_3, ...) to produce a weighted sum. A scalar variable called the bias, b, is also added to the weighted sum, the result of which we will denote as z. A nonlinear function called the activation function, $f(z)$, is then applied to z. The output of this activation function is then the output of the neuron and is sent to other neuron's inputs to which it is connected. The output of a single neuron, which we will denote as a, is given by Equation 2.1 where N is the number of inputs to the neuron. Common choices of the activation function are the logistic function (sigmoid), hyperbolic tangent (tanh), and rectified linear unit (ReLu). The activation function must be nonlinear in order for the neuron/network to learn nonlinear decision boundaries between the input data and desired output response. This will become more clear throughout this section. The activation function must also be differentiable in order to be used with the backpropagation training algorithm that we will be discussed shortly.

$$a = f(z) = f\left(\sum_{i=1}^{N} x_i\, w_i + b\right) \tag{2.1}$$

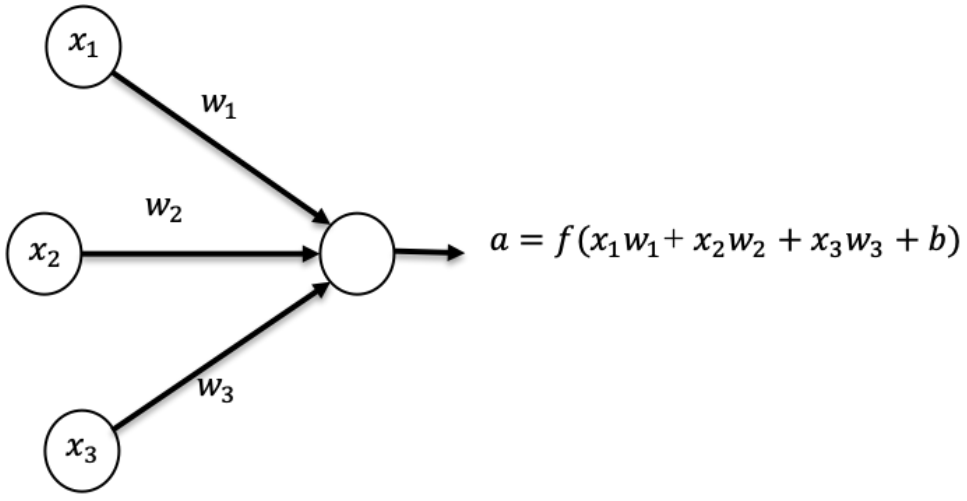


Figure 2.1: Single neuron diagram with three inputs.

Neurons are organized into separate layers which can then be stacked into a multilayer structure as shown in the example in Figure 2.2. The first layer is called the input layer and is where input data is injected into the network. The final layer is called the output layer and as its name states, is the layer usually responsible for producing the final output of the network. Layers in between the input and final layer are called hidden layers. Networks with many hidden layers are called deep neural networks. It has been mathematically proven that neural networks with at least one hidden layer are universal function approximaters, i.e. they can approximate any continuous function given a sufficient number of neurons in the hidden layer and choice of an appropriate activation function [41]. This being said, the number of neurons needed in a network with a single hidden layer can be extremely (prohibitively) large and the universal approximation theorem does not guarantee that such a network is actually trainable. Instead, it was more recently shown that the universal approximation theorem can be extended to deep neural network with bounded hidden layer widths [42, 43].

Let's consider the network in Figure 2.2. The goal of the network is to predict a tomato crop's yield using three input variables: the tomato seed count, the soil depth, and the soil acidity. The network has a single input layer with three input neurons that takes these three input values, applies the activation functions to them, and sends them to the two neurons in the single hidden layer across the weighted connections (sometimes called synapses). The neurons in the hidden layer then each calculate the weighted sum, apply an activation function, and propagate the outputs across the weighted connections to the output layer. The output layer consists of a single neurons whose output value is the output of the network.

The goal of training the network is to adjust the weights and biases of the network in order to minimize the loss between actual neural network output, Y^*, and the target network output, Y, for a set of training data containing target input/output pairs, (X, Y). In this context, the term loss refers to a measure of how consequential an error between Y^* and Y is to the network's performance and thus a function of this error, $L(Y - Y^*)$. In this particular example, the training data may consist of a set of data collected from last year's tomato crops that contain last year's tomato seed count, soil depth, and soil acidity. The goal is to adjust the weights and biases of the network to minimize the loss on the training data, $L(Y - Y^*)$.

There are many choices of the loss function (also called the cost function), however all must be differentiable. Some common choices for regression problems are the mean squared error (MSE) and the logarithm of the hyperbolic cosine (logcosh). For classification problems where the target outputs of the network are binary values, a common choice is the cross entropy loss function.

2.1.1 Gradient Descent via Backpropagation

In order to tune the weights and biases in the network, a process called training, an algorithm called backpropagation is used [44, 45]. At its core, backpropagation works by taking the partial derivative (gradient) of the network loss $L(Y - Y^*)$ with respect to each and every weight and bias in the network, which we will collectively denote as θ as shown in Equation 2.2. The weights and biases are then moved in the opposite direction of each of their gradients, scaled by a hyperparameter learning rate ν, as shown in Equation 2.3. In essence, this process is calculating the contribution of each weight and bias in the network to the

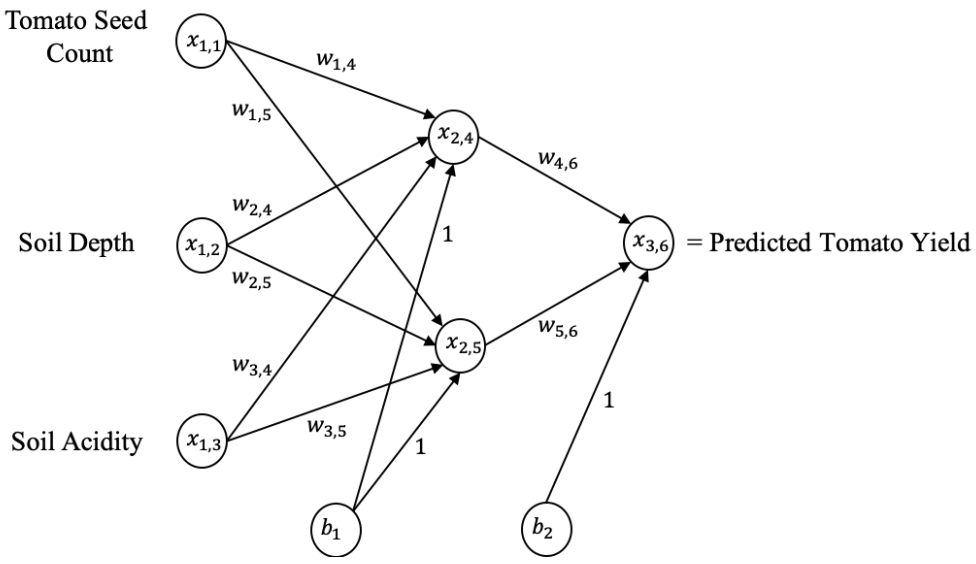


Figure 2.2: Simple neural network example with a single input layer, hidden layer, and output layer.

network's loss and we seek to modify these values in order to minimize their calculated contribution to the network loss.

$$\frac{\partial L\left(Y - Y^*\right)}{\partial \theta}, \text{ for each } \theta = (w_1, w_2, w_3, ..., b_1, b_2, b_3, ...) \tag{2.2}$$

$$\theta_{i+1} = \theta_i - \nu \frac{\partial L\left(Y - Y^*\right)}{\partial \theta}, \text{ for each } \theta_i = (w_1, w_2, w_3, ..., b_1, b_2, b_3, ...) \tag{2.3}$$

In order to perform Equations 2.2 and 2.3 for an arbitrary weight or bias in the network, we must use the chain rule to backpropagate errors from the output layer to earlier layers in the network. We can use the network in Figure 2.2 to better demonstrate this concept. For example, assume that we have a training example $(X, Y) = ([x_{\text{seed}}, x_{\text{depth}}, x_{\text{acidity}}], y_{\text{yield}})$. Now assume that during training we forward propagate (run) these inputs into the network and it makes a prediction, Y^*. We now want to estimate the contribution that each weight and bias made to the loss, $L(y_{\text{yield}} - Y^*)$, but for the sake of this example, let us just consider the weight $w_{1,4}$ only. The contribution that $w_{1,4}$ made to the loss is given by the partial derivative of the loss with respect to the weight:

$$\frac{\partial L(y_{\text{yield}} - Y^*)}{\partial w_{1,4}}$$

For the sake of readability, we will simplify $\partial L(y_{\text{yield}} - Y^*)$ as ∂L. We can now use the chain rule to break this expression down to:

$$\frac{\partial L}{\partial w_{1,4}} = \frac{\partial L}{\partial y^*} \frac{\partial y^*}{\partial w_{1,4}}$$

We know that:

$$y^* = x_{3,6} = f(z_{3,6}) = f\left(\sum_{i=N_1+1}^{N_1+N_2} x_{2,i}\, w_{2,i} + b_2\right) = f\left(x_{2,4} w_{4,6} + x_{2,5} w_{5,6} + b_2\right)$$

where N_1 and N_2 are the number of neurons in layer 1 and 2 respectively. We can now compute the partial derivative of the rightmost expression using the chain rule to get:

$$\frac{\partial L}{\partial w_{1,4}} = \frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial w_{1,4}} = \frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial z_{3,6}}\frac{\partial z_{3,6}}{\partial w_{1,4}} = \frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial z_{3,6}}\frac{\partial \left(x_{2,4}w_{4,6} + x_{2,5}w_{5,6} + b_2\right)}{\partial w_{1,4}}$$

Because b_2 and $x_{2,5}w_{5,6}$ are constant with respect to $w_{1,4}$, their partial derivatives are both 0, leading to:

$$\frac{\partial L}{\partial w_{1,4}} = \frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial z_{3,6}}\frac{\partial \left(x_{2,4}w_{4,6}\right)}{\partial w_{1,4}} = w_{4,6}\frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial z_{3,6}}\frac{\partial x_{2,4}}{\partial w_{1,4}}$$

Given:

$$x_{2,4} = f(z_{2,4}) = f\left(\sum_{i=1}^{N_1} x_{1,i}\, w_{1,i} + b_1\right) = f\left(x_{1,1}w_{1,4} + x_{1,2}w_{2,4} + x_{1,3}w_{3,4} + b_1\right)$$

we get:

$$\frac{\partial L}{\partial w_{1,4}} = w_{4,6}\frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial z_{3,6}}\frac{\partial x_{2,4}}{\partial z_{2,4}}\frac{\partial z_{2,4}}{\partial w_{1,4}} = w_{4,6}\frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial z_{3,6}}\frac{\partial x_{2,4}}{\partial z_{2,4}}\frac{\partial \left(x_{1,1}w_{1,4} + x_{1,2}w_{2,4} + x_{1,3}w_{3,4} + b_1\right)}{\partial w_{1,4}}$$

Because b_1, $x_{1,2}w_{2,4}$, and $x_{1,3}w_{3,4}$ are all constant with respect to $w_{1,4}$, their partial derivatives are all 0, leading to:

$$\frac{\partial L}{\partial w_{1,4}} = w_{4,6}\frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial z_{3,6}}\frac{\partial x_{2,4}}{\partial z_{2,4}}\frac{\partial \left(x_{1,1}w_{1,4}\right)}{\partial w_{1,4}} = x_{1,1}w_{4,6}\frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial z_{3,6}}\frac{\partial x_{2,4}}{\partial z_{2,4}}$$

The quantities $\frac{\partial L}{\partial y^*}$, $\frac{\partial y^*}{\partial z_{3,6}}$, and $\frac{\partial x_{2,4}}{\partial z_{2,4}}$ can now be easily calculated and $x_{1,1}$ and $w_{4,6}$ are both known. The weight can now be adjusted based on Equation 2.3 as:

$$w_{1,4} = w_{1,4} - \nu x_{1,1}w_{4,6}\frac{\partial L}{\partial y^*}\frac{\partial y^*}{\partial z_{3,6}}\frac{\partial x_{2,4}}{\partial z_{2,4}}$$

This entire process can then be repeated for all of the other weights and biases in the network. At this point, backpropagation has completed for one single training pair in the training dataset.

2.1.2 Training Process

In essence, backpropagation provides for a way to calculate the gradient of the network's training error with respect to each of the weights and biases in the network. With this knowledge, we can now investigate the actual implementation of gradient descent via backpropagation for training neural networks on a set of data. There are three different methods that will be discussed, these are: stochastic gradient descent, batch gradient descent, and mini batch gradient descent.

1. **Batch Gradient Descent**: Batch gradient descent (sometimes called vanilla gradient descent) calculates the loss gradient for the entire training dataset and then performs a single weight update based on this gradient [46].

2. **Stochastic Gradient Descent**: Stochastic gradient descent (SGD) is a form of backpropagation where the loss gradient calculations and consequent weight updates are performed for each example in the training set separately and iteratively [47, 46].

3. **Mini Batch Gradient Descent**: Mini batch gradient descent calculates the loss gradient for a small batch of training examples (batch size is generally between 8 and 64) and then performs a weight update based on the gradient.

Regardless of which of these three methods is used, each methods is generally carried out over the entire dataset multiple times. Each full run through the entire dataset is called an epoch. The network is generally trained for either a pre-defined number of epochs or is run for as many epochs as is required to allow the network's performance to stabilize into its final state.

The choice of which gradient descent optimization method to use depends on many factors, including but not limited to the computational resources available (both RAM/VRAM and CPU/GPU performance), the nature of the problem, and the size of the dataset.

The batch gradient descent algorithm is computationally intensive from both a raw processing speed and memory usage standpoint because each weight update requires the loss gradient to be calculated for the entire dataset (entire dataset must be loaded into RAM). This may or may not be a limitation that the user needs to worry about for their

particular dataset or computational resources. From a training standpoint, the trajectory of the gradient over time during training tends to remain smooth when compared to SGD, making it easier to complete the training process. For this same reason however, batch gradient descent may also be more prone to getting stuck in non-optimal local minima for non-convex problems than for SGD, as will be discussed shortly.

Unlike batch gradient descent, in SGD, the gradient trajectory over time tends to be noisy and can cause the network's performance to rapidly oscillate during training. This corresponds to a potential advantage of SGD over batch gradient descent, as the oscillations in the gradient can allow the network to more easily escape non-optimal local minima and jump into a better solution space on the gradient surface [46]. On the other hand, these oscillations can make it difficult to know when to stop training the network as large increases in network error are often followed by an overall net decrease in the error. From a computational standpoint, SGD requires less RAM/VRAM than batch gradient descent, especially for large datasets, and also allows the network to be trained in an online manner [46].

Mini batch gradient descent lies in the middle ground between batch and stochastic gradient descent. As such, from a computational standpoint, mini batch gradient descent has the advantage over batch gradient descent as it must only load a few training examples into memory at a time rather than the entire dataset. Likewise from a training standpoint, if the batch size is optimized appropriately, mini batch gradient descent can maintain the smoothness of the gradient trajectory as in batch gradient descent while also maintaining the ability to more easily escape certain non-optimal local minima like in SGD. For these reasons, mini batch gradient descent is the training process used for the work in this dissertation, as is described in the next chapter.

There are many different forms of these three basic optimization methods that have been developed over the past few decades. Many of these tackle the problem of non-optimal local minima, training speed, and computational efficiency. As will be discussed in the next chapter, the optimization method used for the work in this dissertation is the adaptive learning rate optimization with nesterov momentum (NAdam) algorithm [48], which is built on top of SGD or mini batch gradient descent. The exact details of this algorithm will not be

discussed in this section, however the reader is referred to the original paper on Adam [49] and the addition of Nesterov momentum into the Adam algorithm [48] for more information.

2.1.3 Learning Rate

While there are many hyperparameters in training neural networks that must be carefully selected, the learning rate is arguably one of the most important. The learning rate defines how fast the weights change for any single training event. Larger learning rates will cause the network to descend the gradient faster, however they may cause the weight change to overshoot the gradient and thus oscillate *around* the minimum rather than descend into it. Likewise, too small of a learning rate can cause the network to get stuck in non-optimal local minima without ever reaching the global minimum.

In practice, it is often good to have a variable learning rate that starts at a higher value and decays over time to a smaller learning rate. This is called learning rate annealing and allows the network to more quickly converge to the general location of an optimal local minimum in the earlier stages of training and then more slowly converge into it later on in the training process. The net effect of this is a network that will generally learn more quickly and settle on a more accurate final state than if using a static learning rate [50]. There are many different annealing techniques that can be used for the learning rate, with some good suggestions outlined in Reference [50]. Some of these methods are fixed learning schedules, that is, the learning rate will change over time regardless of the trajectory of the network performance. Other methods are adaptive such that the learning rate depends on some internal state of the network or training process. In the Adam algorithm for example, the initial learning rate is adjusted for each parameter based on the local gradient moments for that parameter [49]. It is not uncommon to use a standard learning rate schedule (static or adaptive) alongside a gradient-based learning rate modifier such as Adam or Nadam during training.

2.1.4 Overfitting Prevention

When training a neural network, we want the network to learn a good internal representation of the relationship between the input and output training data such that the network is able to make useful decisions about new input data. That is, we want the network to *generalize* well to the problem space. One can imagine however, that given the (potentially) large number of free parameters in the network, the network may simply memorize specific training examples rather than learn an actual useful representation of the problem space. This potential problem is called overfitting and is a major concern for neural networks with a large number of neurons and/or layers that must be addressed. The first step in understanding how to combat overfitting is to learn how to detect it. During training, one can track the loss of the network on the training set and also on a separate dataset (that is not used for training) called the validation set. One sign of overfitting is that the training loss continues to improve while the validation accuracy stops improving or begins to degrade. This is demonstrated in Figure 2.3. The point at which the validation loss stagnates or begins to worsen while the training loss continues to improve is the turning point between the network continuing to learn and starting to overfit the training data. Fortunately, many methods have been developed to tackle overfitting; a selection of some of the most prevalent ones are briefly described below:

1. **Regularization**: One of the methods to reduce overfitting in these networks is to penalize weights in the network from having large values, which reduces their ability to fit specific data points (such as noise) in the training set [51]. Such methods are called regularization techniques and are often implemented by adding an extra regularization term to the training loss function. The two most commonly used regularization terms are the L1 and L2 regularization. In L1 regularization, the new loss function becomes $L\left(Y - Y^*\right) = L_0\left(Y - Y^*\right) + \frac{\lambda}{N_t}\sum_i^{N_w}|w_i|$ where L_0 is the original loss function, N_w is the number of weights in the network, N_t is the number of training samples in the mini batch, and λ is a regularization hyperparameter [51]. Likewise, L2 regularization adds a slightly different term to the original loss function: $L\left(Y - Y^*\right) = L_0\left(Y - Y^*\right) + \frac{\lambda}{2N_t}\sum_i^{N_w} w_i^2$. The difference between L1 and L2 regularization is a subtle one. L2

tends to discourage weight sparsity more so than L1 regularization [51]. The choice of which to use depends on the application.

2. **Dropout**: Dropout has become one of the most popular methods to prevent overfitting due to its simplicity, interpretability, and effectiveness. Dropout is simply the process of randomly disabling (dropping out) neurons in the network during each step of training [52]. This has shown to encourage the network to develop multiple independent sub-networks that collectively contribute to the final network decision [52].

3. **Early Stopping**: Early stopping is arguably the easiest overfitting prevention method to implement. Early stopping, as its name entails, simply means to track the training and validation loss and stop training at the point that the network begins to overfit. This is a simple but effective method to prevent overfitting and is often used in conjunction with other methods such as regularization or dropout. Some research has been down on optimizing the point at which to stop training, the reader is referred to Reference [53] for more information on this.

4. **Weight Constraints**: As previously discussed, weight regularization terms such as the L1 and L2 terms can be used to encourage weights in the network to be small. In some cases, however, such a penalty may not be aggressive enough and we used a different, but similar technique to enforce weight limitations called weight constraints. Rather than adding a term to the loss function, weight constraints actually scale the weights on each neuron in each layer according to a pre-defined method during training. Some methods are the unit-norm, max normalization, and min-max normalization which scales the weights on each neuron to have a unit norm, stay below a pre-defined maximum value, or stay within a pre-defined minimum and maximum value, respectively. Weight constraints have proven to be useful in several published works [54, 52].

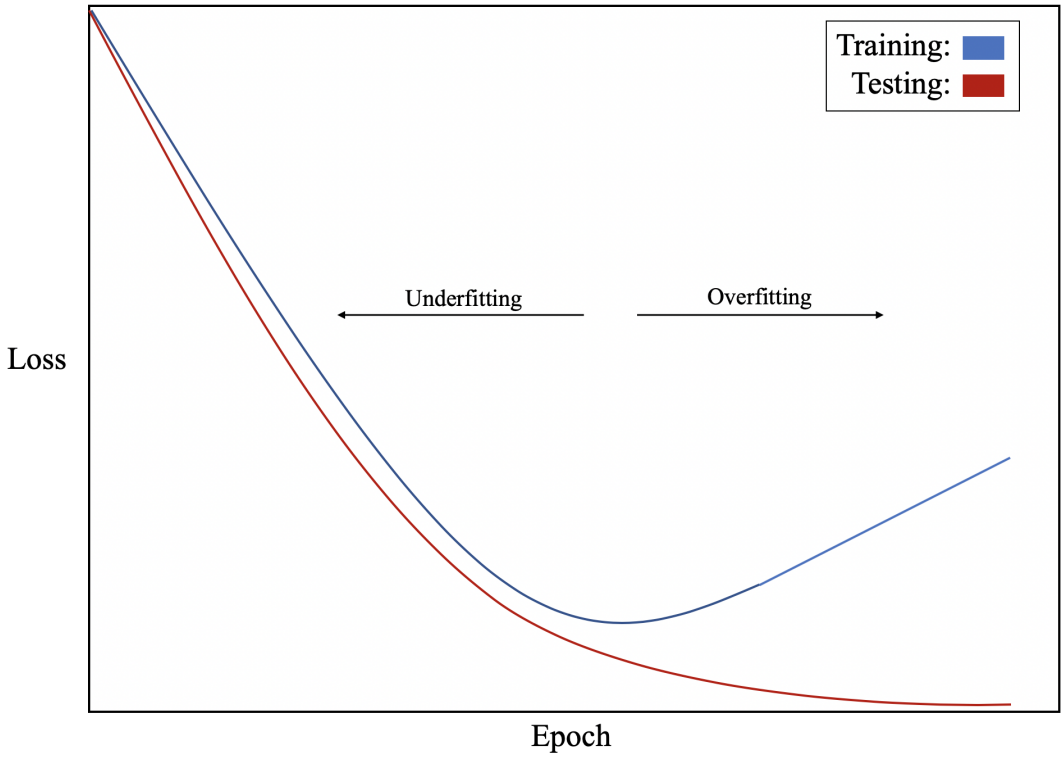


Figure 2.3: Training and validation loss example to demonstrate overfitting.

2.2 Convolutional Neural Networks

2.2.1 Convolutional Layers

Up until this point, all of the networks we have discussed used a type of neuron connectivity scheme called dense connectivity: that is, every neuron in layer i is connected to every neuron in layer $i+1$. For dense connectivity, the number of weights between layer i containing N neurons and layer $i+1$ containing M neurons is $N \times M$. One can imagine, especially considering the fact that many modern deep neural networks contain dozens of layers and thousands of neurons, that the number of free parameters could become extremely large. Most modern networks, however, are predominantly composed of a different connectivity scheme called convolutional connectivity. Networks that use this connection scheme are called convolutional neural networks (CNN) and are an integral part of the deep learning field. Unlike dense networks, neurons in a convolutional layer are grouped into separate groups

of neurons filters. Each neuron in a single filter is connected to a local region in the input space and a single weight matrix is shared between each neuron in each filter. The portion of the input space that each neuron is connected to is called the neuron's receptive field. The size of the receptive field is called the kernel size and the spacing between neighboring receptive fields is called the stride. These concepts are best understood visually as shown in Figure 2.4, which shows an example convolutional layer with connections to an input layer representing an image.

The shared weights among the receptive fields for neurons in a single filter introduces an important concept called spatial invariance. With each neuron in a filter being sensitive to the same feature, the output of this filter is analogous to performing a convolution of the weight matrix with the input. The introduction of convolution to the neural network field stems from many disciplines. Image convolution is used in the traditional computer vision field for manipulating images in many different ways: smoothing, edge detection, taking derivatives, etc. Thus, the inclusion of convolutional layers is partly inspired by computer vision practices. Interestingly, the visual cortex of the human brain uses locally connected layers, which are similar in concept to convolutional layers minus the weight sharing. CNNs became the dominant connectivity scheme in neural network design sometime in the first decade of the 21st century. A series of publications by Dr. Yann LeCun, especially his LeNet-5 CNN architecture [55], were arguably the catalyst to the rise of CNN network designs. One of LeCun's papers demonstrated that CNNs performed better at handwritten digit classification than the other state of the art methods at that time [**?**].

Besides the introduction of spatial feature invariance, convolutional layers also drastically reduce the number of free parameters in the model. The number of unique weight values in a convolutional layer is equal to $N_f \times N_k$ where N_f is the number of filters in the layer and k is the kernel size. In the example network shown in Figure 2.4, let's assume that the input layer is size 28×28 pixels, which is the size of the handwritten digit images in the MNIST dataset [56]. Now let's assume that the kernel size of the receptive fields is $8 \times 8 = 64$ and the vertical and horizontal stride are both 5. In this case, the number of weights between the input and convolutional layer is $3 \times 64 = 192$ because each of the 3 filters has 64 weights. If this was not a convolutional layer and was instead a dense layer, then the number of weights

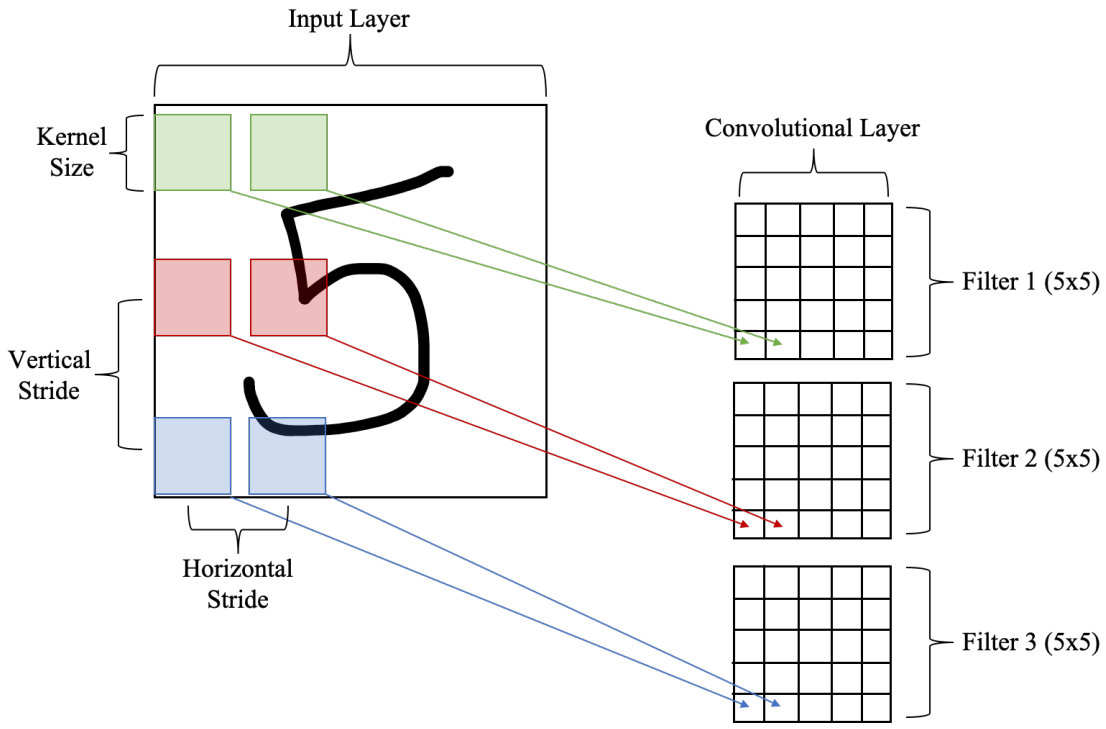


Figure 2.4: Example of a convolutional layer with important parts labeled. Each colored square represents a neuron's receptive field (not all are shown) and each receptive field of the same color uses the same weight matrix.

would be $(28 \times 28) \times (3 \times 5 \times 5) = 58800$ weights, which is clearly a significant increase compared to the convolutional case.

2.2.2 How CNNs Process Information

Convolutional layers tend to learn to identify spatially invariant salient features present in their inputs. When stacking multiple convolutional layers to create a CNN, each consecutive convolutional layer tends to build up information from the previous layers in a bottom-up feature representation. Consider a CNN designed for image recognition as an example. The first layer(s) often learn to extract elementary features from the input data such as oriented edges of different color. Future layers then build up these elementary edge features into more and more complex shapes. The end result is a network that has learned to extract complex information from the dataset based on combinations of elementary features; this is called

bottom-up learning and is also a large part of the way that the human visual cortex works [57]. Reference [58] exhaustively demonstrates these concepts in a CNN trained for image recognition and also gives a thorough overview of how one can visualize CNN activations and use them to interpret their model [58]. The main takeaway from this subsection is that while CNNs are functionally a black box, the ways in which they operate can be understood and interpreted at a high level.

2.2.3 Classifiers

There are many different types of neural network architectures designed for different purposes. One of the most common problem spaces in which neural networks are applied are classification problems. Neural networks designed for classification often consist of a series of convolutional layers of different types followed by a series of dense layers that ultimately lead to a group of neurons where each neuron represents a particular class. Some classic examples of classification networks are ResNet [59], AlexNet [60], VGGNet [61], and Inception [62, 63, 64].Because the network designed in this dissertation is not a classifier, we will no longer discuss them here, however I believe that understanding how they work and reviewing the papers on the network models just mentioned is important for understanding some of the motivations behind the design of ARAD.

2.3 Autoencoders

Autoencoders are a type of unsupervised neural network architecture consisting of two main parts, the encoder and the decoder. The encoder is a series of layers which can be convolutional, dense, locally connected, etc. whose purpose is to take high dimensionality data as its input and compress it into a low dimensionality representation called the latent space. The decoder is then a series of layers that takes this low dimensionality latent space and expands it into the original dimensionality of the data. Unlike a supervised classifier, the autoencoder is trained to reproduce the input data as its output. In doing so, the hope is that the network will be able to learn to extract a salient encoding of the input data at the latent space, similar in concept to the principal components in PCA or the basis vector in

NMF. Figure 2.5 shows an example of a typical autoencoder network designed to reconstruct images of handwritten digits at its output. While the overall goals of PCA and autoencoders are often the same, the ways in which they work are fundamentally different. Autoencoders, like other neural networks, use feature/representation learning methods to come up with an efficient data encoding. On the other hand, PCA is fundamentally a statistical method, using eigendecomposition of the covariance matrix of the training data to extract an efficient data encoding. As such, PCA is limited to extracting only linear mappings between the input data and the data encoding, whereas autoencoders can extract complex nonlinear mappings. The important of this nonlinear property depends on the nature of the data itself: if the data lies on a nonlinear manifold, then nonlinear dimensionality reduction techniques would be more appropriate. With this in mind, it is often very difficult to determine the linear/nonlinear nature of a specific dataset, especially if that dataset is very complex. Finally, autoencoders may be less computationally intensive from a memory standpoint than PCA because PCA requires the entire dataset to be loaded into memory at once in order to perform SVD whereas the autoencoder can be trained on small batches (minibatch gradient descent) or even single data points (stochastic gradient descent) over a period of time. This also makes autoencoders more appropriate for online training purposes, especially on a system deployed at the edge where computational resources may be limited.

2.3.1 Anomaly Detection with Autoencoders

Autoencoders have found use in many different problem spaces. They have been used for image compression [65], image denoising [66, 67], semantic hashing of textual documents [68], and of course dimensionality reduction [69, 70].

With regards to this dissertation, we are especially interested in the use of autoencoders for anomaly detection. If an autoencoder is trained on data representing the typical behavior of a system, then the internal representation that it learns from that data will be those representing typical system behavior. As such, if an atypical data sample is shown to the network post-training, it will be unable to reconstruct the input data at its output, leading to a spike in the reconstruction error between the input and the output. This spike can be used to trigger an anomaly detection event. Detecting this spike generally entails designing an

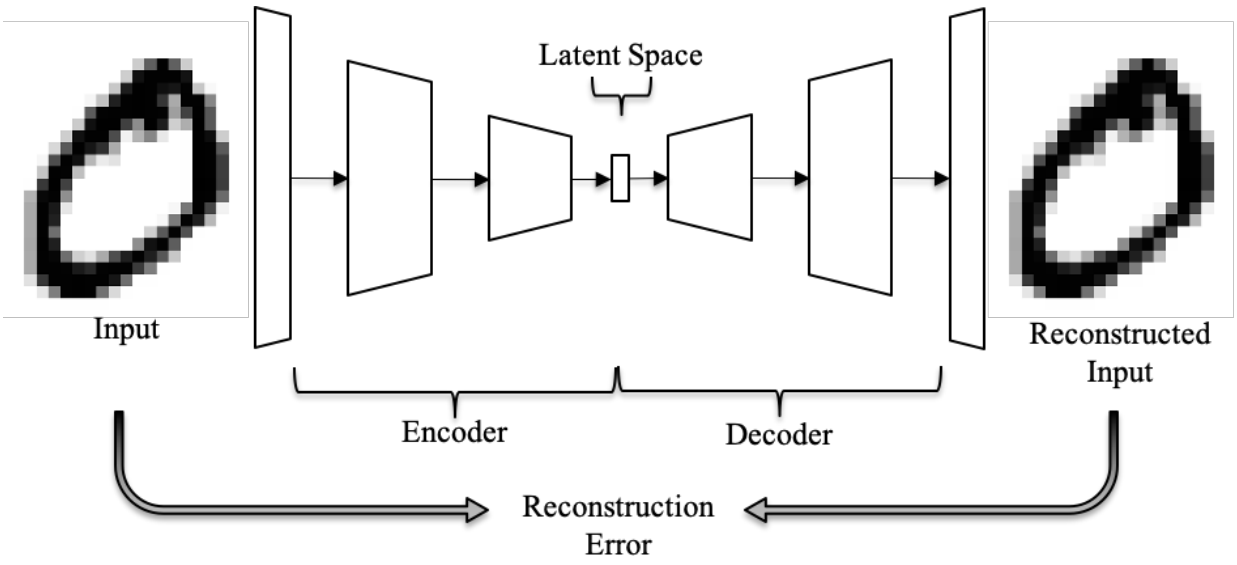


Figure 2.5: Example diagram of the typical layout of an autoencoder neural network.

appropriate reconstruction metric that determines how well the model is able to reproduce the input data at its output. One can then establish a threshold value in which to trigger an anomaly detection alarm.

Autoencoders have widely been used for anomaly detection, including for financial fraud detection [71], detecting anomalies in spacecraft telemetry data [72], detection of atypical behavior in high performance computing nodes [73], and detection of abnormal events in video streams [74, 75]. In the next chapter, we will discuss the autoencoder model that I have designed and developed for performing anomaly detection on gamma ray spectra collected by NaI(Tl) detectors. Chapters 5 and 6 will then discuss the peformance of this model on simulated and real world data, respectively.

Chapter 3

Description of Datasets

This chapter will describe the datasets used to train and evaluate the ARAD algorithm. Section 3.1 will give an overview of a set of simulated data representing a mobile urban source search. As will be described, these simulated data allow for the unique ability to assess algorithm performance in a well controlled environment, which allows for the ability to accurately quantify algorithm performance. Section 3.2 will then describe the data collected by a suite of radiation detectors situated outside of the High Flux Isotope Reactor and Radiochemical Engineering Development Center at Oak Ridge National Laboratory. These data make it possible to evaluate the ARAD algorithm on a multitude of real world sources and background dynamics contributors.

3.1 Simulated Urban Source Search Dataset

Urban environments are one of the most challenging in which to perform a source search campaign, owing largely to the many background dynamics contributors present in such an environment (see Chapter 1). In order to encourage the development of new detection, identification, and localization algorithms that are suitable for urban search scenarios, a consortium of national laboratories, including ORNL, developed a Monte Carlo model of an urban street with realistic background and injected sources [2, 1]. This dataset was used for a public data competition hosted on the Topcoder website [1]. The data has also been made

available to the public through ORNL and is an invaluable resource for others developing detection and identification algorithms [76].

3.1.1 The Model

The model was developed using the SCALE/MAVRIC Monte Carlo code and a suite of custom Python scripts [77]. A full overview of this is outlined in an article that is under review for publishing [2], however this section intends to provide the reader with enough information to understand the data. The model is composed of an urban city street, loosely modeled on the layout of Gay Street in downtown Knoxville, TN [2]. The street contains s series of buildings grouped together in blocks, surrounding a center street with four individual lanes [2]. Aside from buildings, grassy areas resembling small parks, asphalt parking lots, and parking garages are also modeled [2]. Buildings are each composed of a single construction material being either granite, concrete, or brick [2]. The KUT concentrations in the buildings are based on real world experimental data in order to provide realistic NORM signals to the detector [2]. The buildings are hollow rectangular prisms with a 6" thick wall [2]. The buildings blocks are each modeled in a separate simulation [2]. The reason for this is so that different building blocks can be laid out on the street in different orientations without having to run the Monte Carlo simulation again. Figure 3.1 shows a mock up of what one of these configurations may look like.

There are five injected sources that are modeled at 15 different locations in the model [2]. These five sources are highly enriched uranium (HEU), weapons grade plutonium (WGPu), ^{60}Co, ^{99m}Tc, ^{131}I, and a combined HEU/^{99m}Tc source [2] [2]. ^{60}Co, ^{99m}Tc, and ^{131}I are modelled at each location with a source strength of 1 μCi [2]. The detector response for each of these sources (unshielded) is shown in Figure 3.2. The HEU and WGPu sources are each modeled as a single IAEA significant quantity [2]. When using the model to create actual dataset runs, a Python script is used to linearly scale the strength of the source [2]. Shielded sources are also included in the model, however these are not used for the dataset used to train and evaluate ARAD.

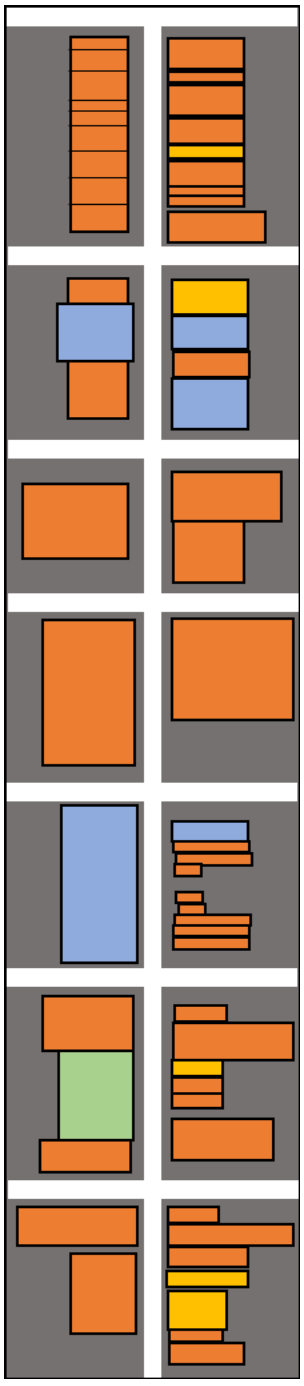


Figure 3.1: Example street model. Individual blocks (separated by roads in the image) can be placed in different orders to create different street configurations. Dark grey is concrete, granite is yellow, blue is asphalt, soil is green, and brick is orange.

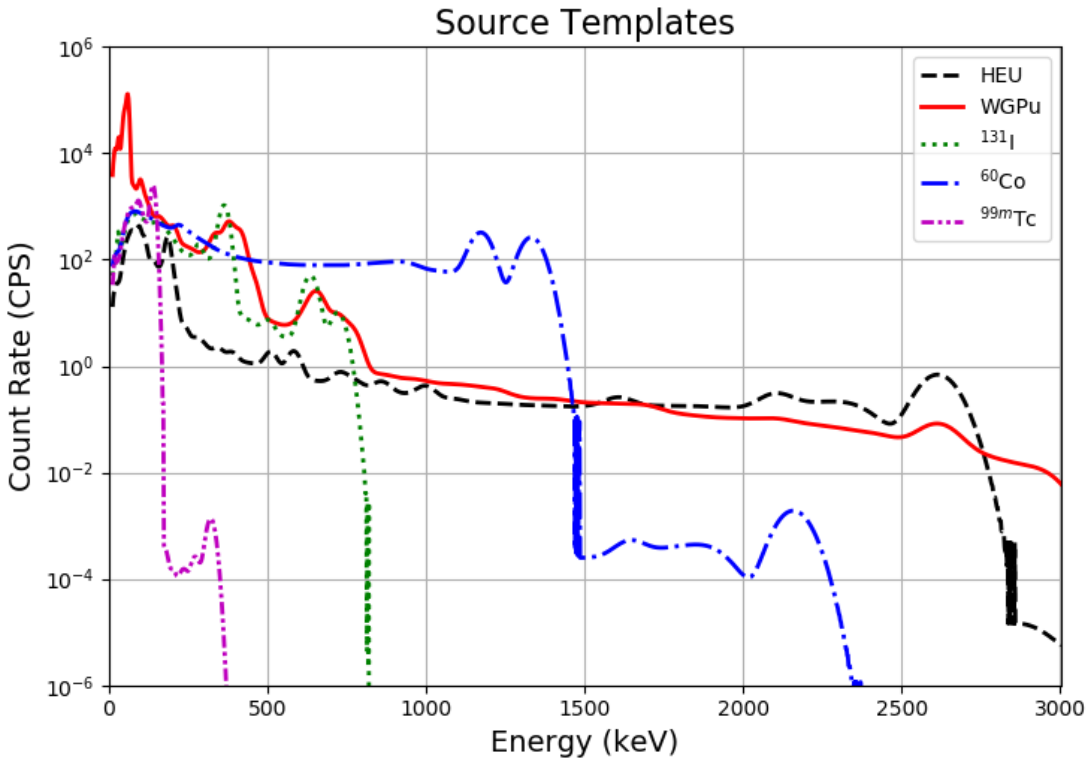


Figure 3.2: Detector response for the five sources in the simulation data [2].

3.1.2 The Topcoder Dataset

The dataset used to train this model is a subset of that used for the Topcoder competition. The Topcoder dataset consists of 25540 runs, each representing list mode data collected by a simulated 2"×4"×16" NaI(Tl) detector moving through the simulated street [1, 2]. For each run, the detector speed was held at a constant value between 1 and 13.4 m/s and the detector stayed in the same lane of travel. The KUT concentrations of the building materials change from run to run, but do not vary by more than 80% from typical real-world values [2]. Of these 25540 runs, 9700 are designated as training data, with the rest making up the testing set. Overall, there is approximately 350 hours of just background data and 550 hours of source data available [2].

3.1.3 Custom Performance Evaluation Dataset

In order to test the ARAD model's performance for urban source search, a custom dataset was created using a modified form of the Python scipt used to generate the list mode data from the Monte Carlo model for the Topcoder competition. A total of five sources were selected: ^{131}I , ^{60}Co, ^{99m}Tc, HEU, and WGPu. For each of these sources, separate runs were made with varying source strengths ranging from a source to background ratio (SBR) of 0.05 to 0.6. It was selected to define the source strength in terms of SBR rather than absolute units of Ci as the ability to detect a source is not just dependent on the strength of the source, but also the strength of the background. Note that the Monte Carlo model was only run a single time with source strengths of 1 μCi for single radioisotopes and 1 significant quantity for the SNM sources. In order to generate the list mode data, the Python script scales the sources counts S by a linear scaling factor c. The scaling factor c required to achieve the designated SBR is calculated using Equation 3.1, where S and B are the source and background counts integrated over a period of 24 seconds surrounding the point of closest approach between the detector and source. For this testing set, the speed of the detector is held at a constant value of 1 m/s for all runs, correlating to a distance of 24 m for surrounding the source for the calculate of c.

$$SBR = \frac{c\,S}{B} \tag{3.1}$$

For each of the 5 sources, 11 incremental SBR values between 0.05 to 0.6 were used to test ARAD on source ranging from very low to high. For each of these different source strengths, 24 different KUT background models were used to vary the background. This leads to a total of 264 separate runs (list mode files) for each of the 5 sources. As an overview, the testing dataset has the following features:

- 264 list mode files for each source, with 24 different background models for each of the 11 different SBR values ranging from 0.05 to 0.6.
- Detector path, lane, and speed of 1 m/s were held constant across all runs.

- Source location stayed the same for each run. The source location was selected to be one with minimal shielding and direct line of sight between the detector and source. This dataset is intended to test the algorithm detection performance as a function of SBR in an environment with realistic levels of NORM variation. The purpose is not to evaluate performance with respect to shielding.

- The building block configuration stayed the same for all runs.

Figure 3.3 shows the gross count rate for one of these runs without a source present. Note the extremely large variation in the gross count rate as the detector moves through the street. The large peak in the count rate near 150 seconds is the location of one of the granite buildings, which as described in Chapter 1, granite has a higher concentrations of KUT than many other building materials. The k-sigma and SPRT algorithms do not perform remotely well on this data as they are not able to establish a reliable background estimate because the count rate changes very quickly.

The metric used to evaluate the algorithm's performance is the receiver operating characteristic (ROC) which plots the algorithm's true positive rate (TPR) as a function of the false positive rate (FPR). This is done by running the algorithm on each run with a range of detection threshold values and tallying the alarm rate and alarm times of the algorithm. A true positive is tallied when the detector produces an alarm within 40 seconds of closest approach to the source. This 40 second window is highlighted in Figure 3.3. Note that there are no spikes in the background gross count rate within this 40 second wind.

Figure 3.4 shows the count rate profile for a selection of runs with SBR values ranging from 0.05 to 0.3 for ^{131}I , ^{60}Co, and ^{99m}Tc. Likewise, Figure 3.5 shows the count rate profiles for the same SBR values for the two SNM sources.

3.2 HFIR/REDC Dataset

3.2.1 HFIR/REDC

For the past several years, the team at ORNL, including myself, has been developing a set of sensor suites positioned outside of the High Flux Isotope Reactor (HFIR) and Radiochemical

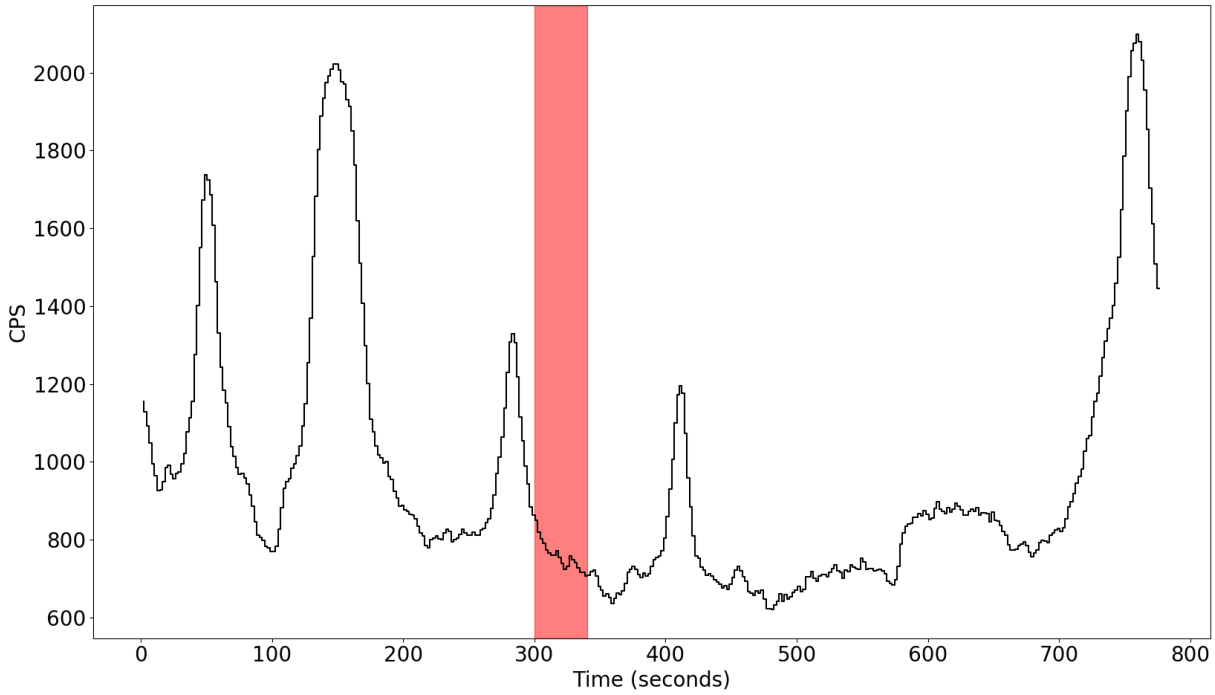
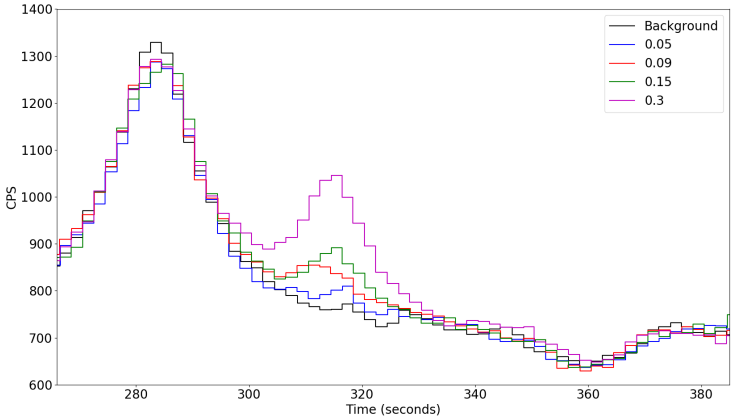


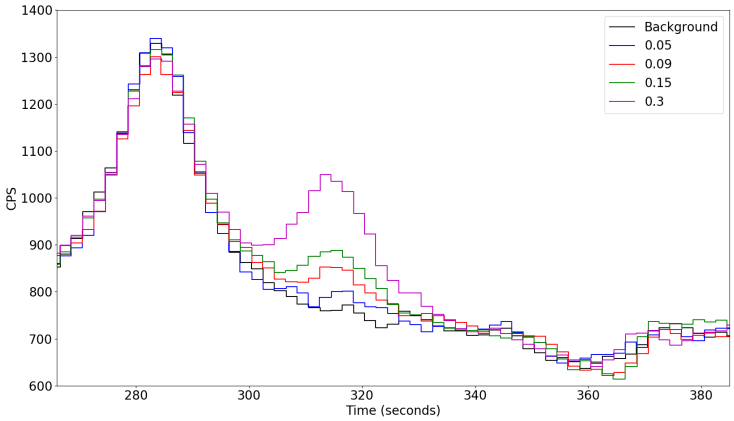
Figure 3.3: Background only gross count rate for one of the runs in the perform evaluation testing dataset. The red highlighted region shows the 40 second window surrounding the point of closest approach between the detector and source. Note that no source is present in this particular figure.

Engineering Development Center (REDC) located at ORNL. There are six sensor suites, called nodes, located at various locations surrounding the two facilities. Each of these nodes contains a single 2"×4"×16" NaI(Tl) detector, a 3" cylindrical NaI(Tl) detector, one to two cameras, a single Velodyne VLP-16 LIDAR sensor, a weather station, environmental sensors, and a network connected single board computer for capturing the sensor data and relaying it to a central data collection server. The node located at that point of entry to the HFIR/REDC facility is called the supernode. The position of the sensor nodes at the HFIR/REDC facility are shown in Figure 3.6.

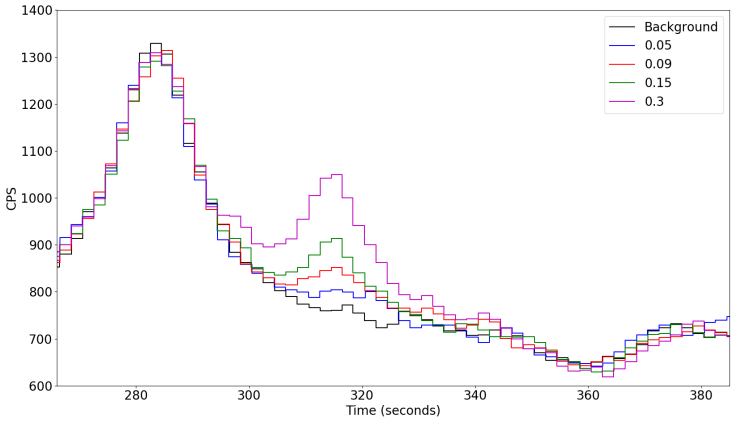
These nodes have been collecting data persistently for several years, with many tens of terabytes of data now available. All of the data points, including the radiation data, is sampled and stored at an interval of 100 ms. This data contains many real world radiation events that are part of the everyday operations of the HFIR/REDC facilities.



(a) ^{131}I

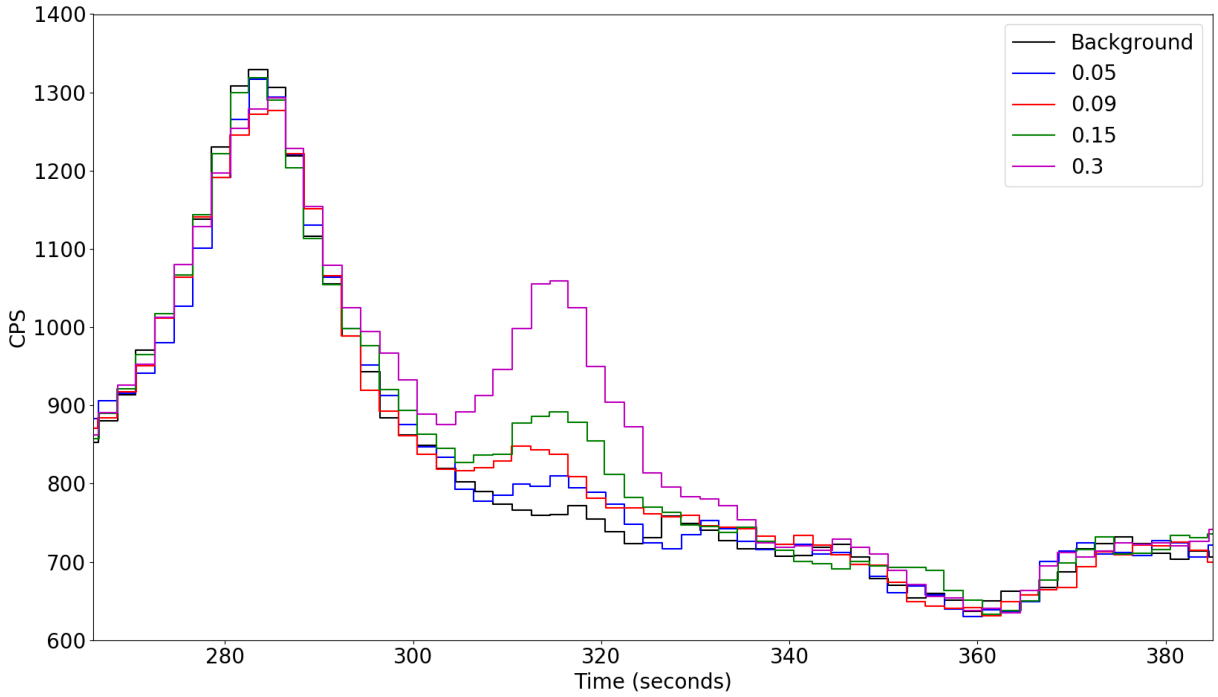


(b) ^{60}Co

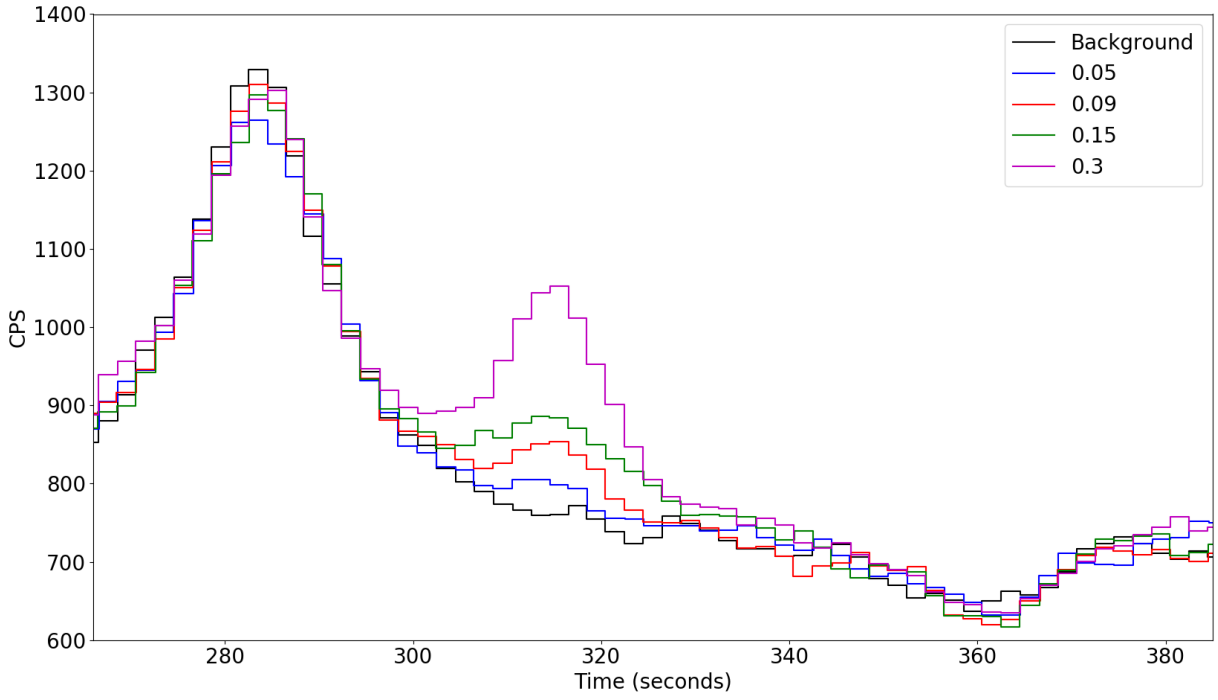


(c) ^{99m}Tc

Figure 3.4: Count rate profile for a range of SBR values for ^{131}I (top), ^{60}Co (middle), and ^{99m}Tc (bottom).



(a) ^{131}I



(b) ^{60}Co

Figure 3.5: Count rate profile for a range of SBR values for HEU (top) and WGPu (bottom).

Figure 3.6: Layout of the sensor nodes at the HFIR/REDC facility.

In particular, various types of radioactive material such as irradiation targets, spent fuel, and medical/industrial sources are transferred between the two facilities. Beyond this, the data is incredibly well documented. The majority of the radioactive material transfers are documented from ground truth. Having cameras and LIDAR also enables us to correlate radiation events with events that occur in the scene around the detector. Further, the weather station can be used to verify and quantify the presence of precipitation and correlate this with changes in the measured radiation signal. Finally, the detectors are in climate-controlled boxes with gain-stabilization algorithms running in order to reduce the effects of possible temperature-induced gain shift in the detector (this is also corrected for in the training set). The takeaway is that this data is incredibly useful for testing radiation detection algorithms on real world radiation events including those coming from both natural and artificial sources.

Chapter 4

The Autoencoder Radiation Anomaly Detection Algorithm

4.1 Purpose

The autoencoder radiation anomaly detection (ARAD) model is a convolutional autoencoder designed to learn a compressed, efficient representation of the features comprising typical background gamma ray spectra in dynamic background environments. The goal of the network is to be able to detect anomalous sources present in gamma ray spectra collected in the field by evaluating the network's spectral reconstruction performance on this spectra. This chapter will outline the process of designing the model.

4.2 Spectrum Pre-processing

Before discussing the design of the model, it is important to first describe the format of the input data and the pre-processing stages applied to the data prior to analysis by the ARAD network. As a reminder to the context of the problem at hand, we are considering the situation where a NaI(Tl) detector (or other low-moderate resolution detector) is performing a radioactive source search in an environment characterized by a high level of variation in the background radiation environment. Examples of such a scenario are a mobile search in an urban environment and a detector positioned at a busy point of ingress.

4.2.1 Binning Size

In such an environment where the detectable signature of the illicit source may be short lived, it is ideal to sample spectra and perform analysis with a small integration time. As described in Chapter 1 however, the uncertainty in the count rate (per energy bin or gross counts) is equal to the square root of the counts, thus the longer the integration time, the smaller the uncertainty in the data. With this in mind, the integration time must be selected to strike the correct balance between uncertainty in the raw data and minimizing the risk of missing short-lived source signals. In practice, sampling times on the order of seconds to tens of seconds appear to be good choices, at least for the data described in Chapters 5 and 6.

Aside from the integration time consideration, there are other ways to increase uncertainty in the raw data than increasing the integration time. As discussed in Chapter 1, one of the advantages of gross count rate algorithms over spectrum-based algorithms is the fact that because all of the counts are integrated into a single measurement, the integration time can be much shorter than when they are binned into 1028 or 2048 separate energy bins (these are standard binning sizes). We must consider however that NaI(Tl) detectors do not have great resolution; for the detectors used in this study, the FWHM at 662 keV is normally about 46 keV. With an LLD and ULD set to 0 and 3 MeV, respectively, the average bin width of the histogram for 1028/2048 bins would be about 2.9/1.5 keV. With this in mind, we can safely use far less bins in our histogram without sacrificing much information in the spectrum.

The real-world dataset described in Chapter 6 contains spectra binned with 1000 energy bins. In order to reduce this value, the spectra must be rebinned from the original calibration, $E_{\text{cal},0}$ into a new calibration E_{cal}. The main challenge in rebinning is determining how to handle the case where the counts in one of the original energy bins must be split into multiple bins in the new histogram. In this work, I take a probabilistic approach, where we assume that counts in each bin in the original histogram are distributed uniformly between the energies of the bin edges. Considering the large FWHM with respect to the 3 keV bin width, this is a reasonable assumption.

The rebinning algorithm, written in Python 3, is shown in Listing 4.2.1. For each bin in the original spectrum, each count energy is sampled from the bin uniformly between the left and right bin edge energies and then binned into the new histogram according to E_{cal}. Conceptually, the algorithm can be interpreted as sampling the original spectrum into individual energy events and then rebinning them back into a new spectrum. Figure X shows an example spectrum binned into 64, 128, 256, 512, and 1024 bins. Using 128 bins was shown to produce good results at integration times down to 5 seconds for the data used in this dissertation. Using less bins tends to wash out certain features in the spectra which impacted algorithm performance. More than 256 bins required longer integration times to get reasonable statistics above 400 keV, which resulted in missing alarms that were very short in duration. 128 bins is thus used for the ARAD algorithm.

Listing 4.1: Spectrum rebinning algorithm

```
1  import numpy as np
2  def rebin(histogram: np.ndarray, obe: np.ndarray, nbe: np.ndarray):
3      sampled_energies = []
4
5      for index, bin_count in np.ndenumerate(histogram):
6          for i in range(bin_count):
7              sampled_energies.append(np.random.uniform(obe[index],
8                                                        obe[index + 1]))
9
10     n_spec, _ = np.histogram(sampled_energies, nbe)
11
12     return n_spec
```

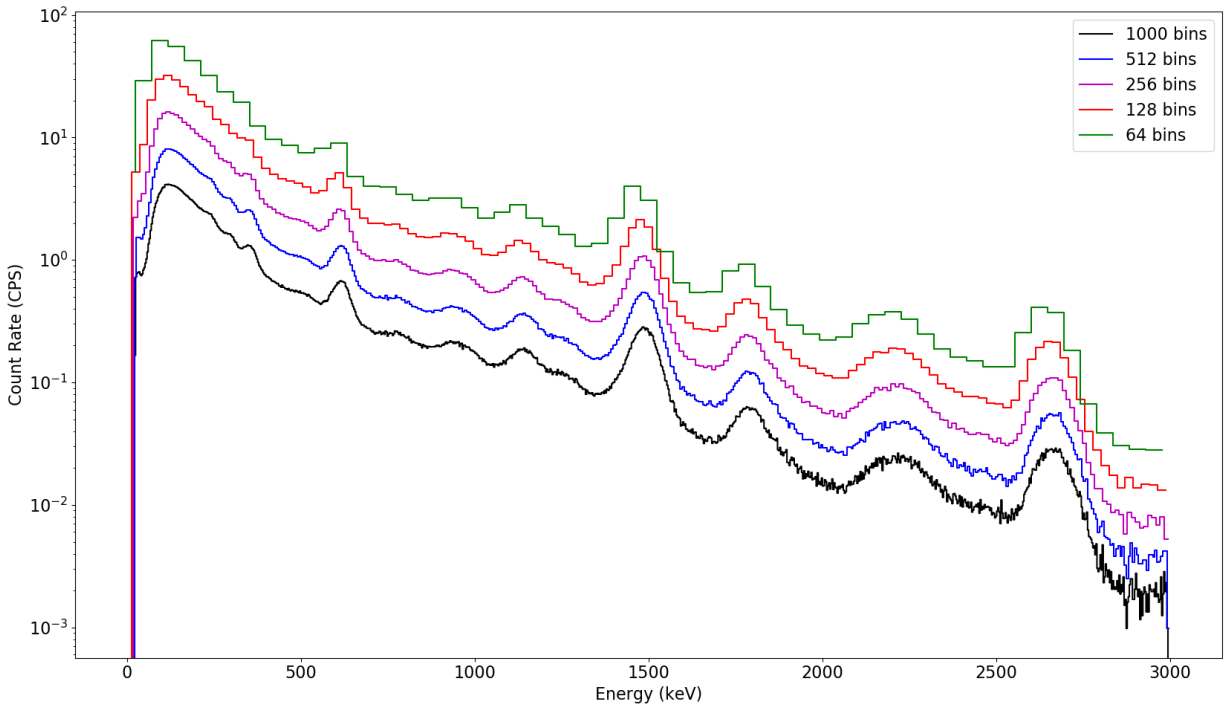


Figure 4.1: Rebinning a 1000 bin gamma spectrum into 64, 128, 256, and 512 bins using the algorithm in Listing 4.2.1.

4.2.2 Resolution Normalization

As will be discussed in the next section, the first layer of the ARAD model is convolutional. In order to select an appropriate kernel size, it is best to have an understanding of the size of the features in which the network may learn to capture. In this case, the most obvious features in the spectra are the photopeaks, which vary in width according to the energy resolution nonlinearity (see Section 1.6.2 for more information on the source of this nonlinearity). In order to allow a single convolutional kernel size to capture all of the photopeaks in the spectrum, the new calibration prior to rebinning the spectrum is calculated such that the resolution is constant across the entire spectrum. To do this, the resolution nonlinearity of the detectors are modelled according to Equation 4.1, where $\sigma(E)$ is the standard deviation of the photopeak at energy E, abd a, b, and c are fit parameters. Equation 4.1 is fitted using data collected from ^{241}Am, ^{166}Ho ^{137}Cs, ^{152}Eu, ^{60}Co, and ^{208}Tl sources. The resolution calibration for the 2"×4"×16" NaI(Tl) detector on Node 1 of the HFIR/REDC dataset (described in

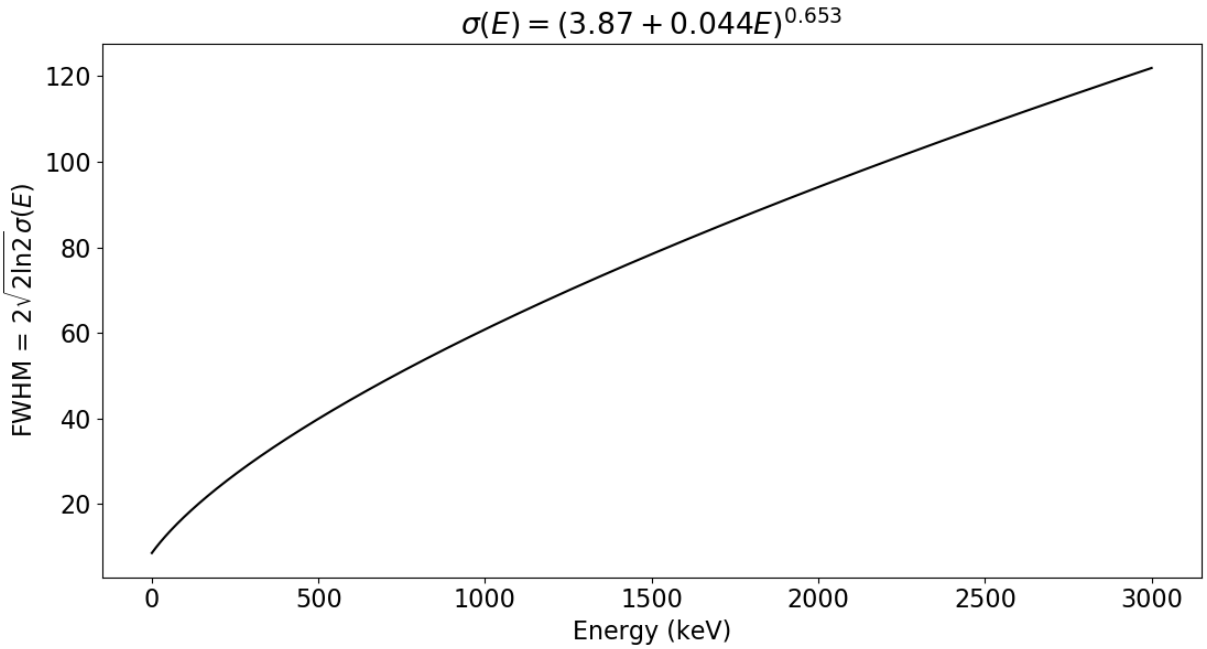


Figure 4.2: Resolution of the 2"×4"×16" NaI(Tl) detector on Node 1 of the HFIR/REDC dataset.

Chapter 6) is shown in Figure 4.2. As can be seen, the resolution approximately follows the square root of the energy as predicted by the uncertainty induced in the photocathode photon to electron conversion process. By making the exponential term, c, a fittable parameters however, we allow the resolution function model to better capture other contributing factors to the resolution nonlinearity that may not follow the square root of the energy.

$$\sigma(E) = (a + b \times E)^c \tag{4.1}$$

Performing the resolution normalization step did not result in significant algorithm performance, likely due to the fact that the branched architecture (described in the next section) allows for the variability in resolution. This being said, it should *theoretically* make it easier for the network to develop "photopeak" detection-related filters, as these filters would no longer have to be scale-invariant. Also, considering that both energy and resolution calibration can be performed using the same data, there is no reason not to include

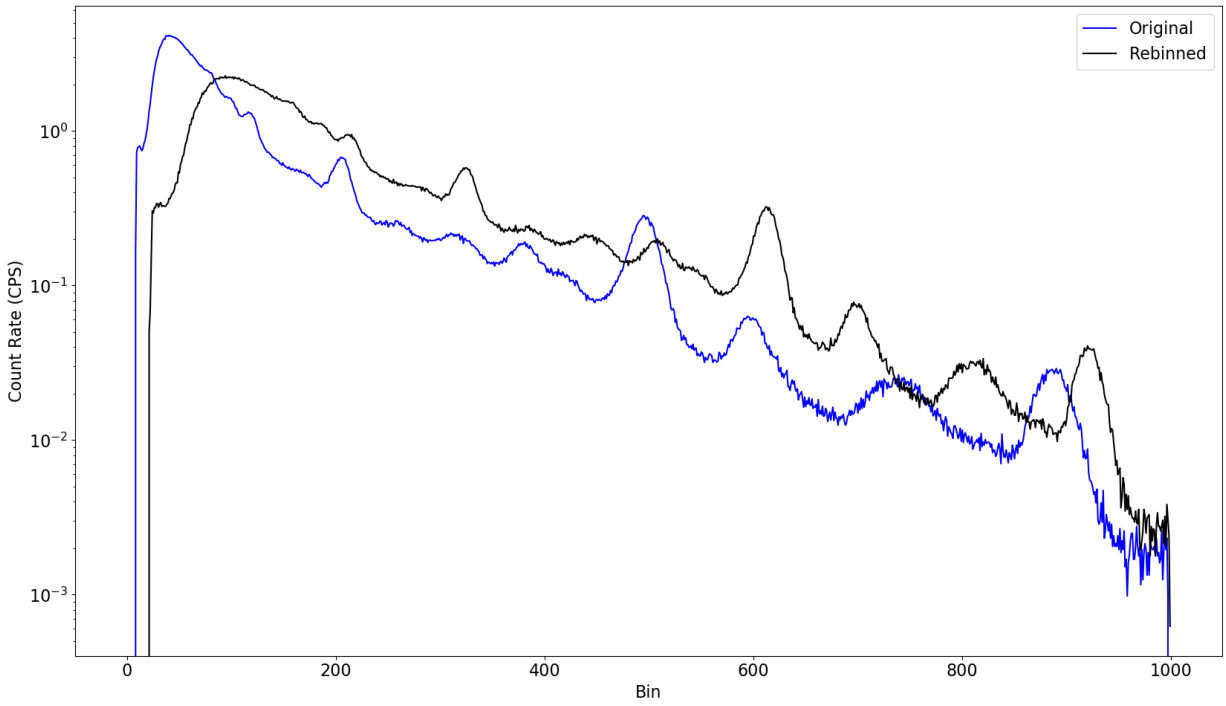


Figure 4.3: Resolution normalized and rebinned background spectrum.

this step in the process pipeline. A sample background spectrum that has been rebinned and resolution-calibrated is shown in Figure 4.2.

4.2.3 Spectrum Normalization

A standard practice in preparing datasets for machine learning problems, especially neural networks, is to normalize the input data in order to constrain the range of values that the network must learn. The method used here is min-max normalization, according to Equation 4.2. This process is done both for the training and inference mode.

$$S^* = \frac{S - \min(S)}{\max(S) - \min(S)} \tag{4.2}$$

4.3 Model Architecture and Design

4.3.1 Initial Architecture and Design Process

The ARAD model underwent many iterations prior to establishing the final design. The initial working design is shown in Figure 4.4, with layer labels outlined in Table 4.1. The encoder part of the model consisted of a series of convolutional and max pooling layers that reduced the original 256 bin spectra into a latent space of only 3 components. These 3 components were then fed into a series of dense layers that expanded the dimensionality back to the original 256 bins. All neurons in the network used the sigmoid activation function, which displayed the most stable training trajectory when compared to relu and tanh. Each of the convolutional and dense layers (excluding the output layer) were followed by a dropout operation with a dropout probability of 80%. Values between 20% and 80% had similar final performance, but 80% was selected to minimize co-adaptation in the network. Finally, each of the dropout operations was followed by a batch normalization step to stabilize the training process. Instead of using a densely connected decoder, a set of deconvolutional layers symmetric to the encoder was also tested but made the training process unstable and did not result in any improvement in network performance.

The model was trained using mini batch gradient descent via Adam with a batch size of 32 and the mean squared error (MSE) as the loss function as shown in Equation 4.3 where N_b is the number of bins in the spectrum. Training occurred for 25 epochs at a static learning rate of 1×10^{-5}. A total of 8 separate ARAD models were trained; 4 networks were trained on the simulation data with training spectra integrated at 5, 10, 30, and 60 seconds, respectively. Likewise, 4 different network were trained on the HFIR/REDC data with training spectra integrated at 5, 10, 30, and 60 seconds.

$$MSE = \frac{\sum_{i=1}^{N_b} (Y_b - Y_b^*)^2}{N_b}. \tag{4.3}$$

After the training process, a metric must be designed to evaluate the spectral reconstruction error during the testing phase. The natural option would be to use the same loss function as was used for the training step, which in this case is the MSE. It was found that while the

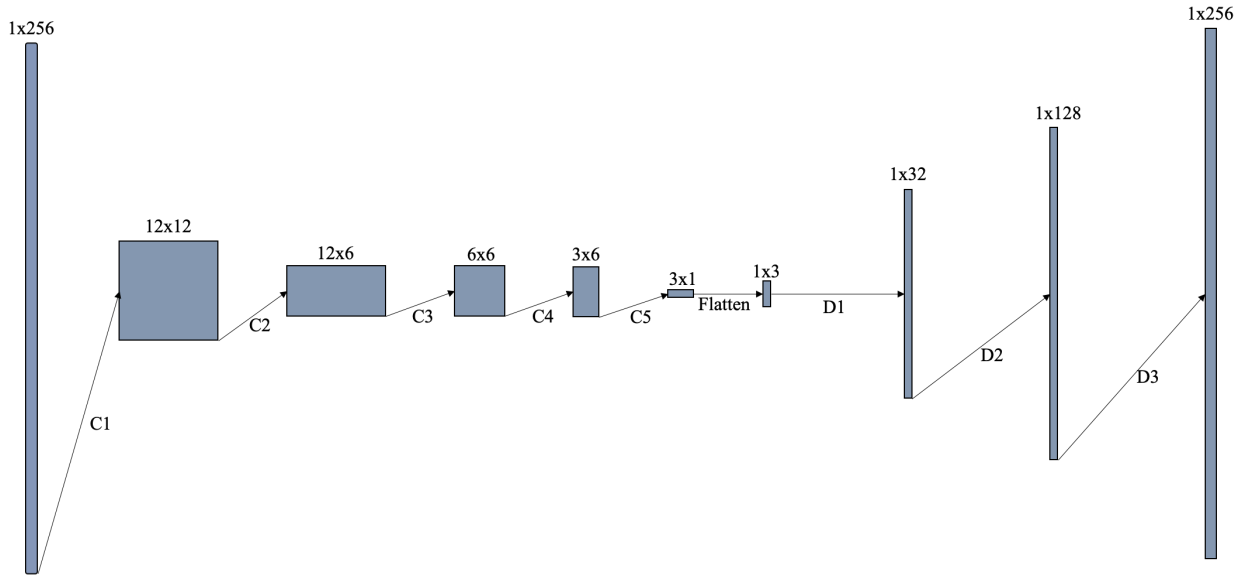


Figure 4.4: Initial design of the ARAD model. Layer labels are shown in Table 4.1

MSE was appropriate for use as a loss function, it did not work so well in testing because its magnitude is exponentially proportional to mean-centered noise in the spectrum (such as that arising from counting statistics). The result is that the MSE increases exponentially as a function of gross count rate. Instead of the MSE, the squared mean error (SME) was used as the reconstruction metric, which does not display as much proportionality to the count rate because the errors for each bin are summed prior to squaring which has the effect of partially cancelling out the effect of mean-centered statistical noise.

$$SME = \left(\frac{\sum_{i=1}^{N_b} (Y_b - Y_b^*)}{N_b} \right)^2 , \tag{4.4}$$

The first convolutional layer was somewhat arbitrarily selected to use 12 filters with a kernel size of 12. The main motivation was to select a kernel size that is large enough to capture the different feature types present in the spectrum. The next few convolutional and max-pooling layers in the encoder continue to reduce the dimensionality down to 3 components. The intuition for using 3 components was that there are three primary

Table 4.1: Operations performed at each layer in the initial ARAD model. The location that each label references in the model can be seen in Figure 4.4

ARAD Initial Layer Operations	
Label	Operation
C1	Convolution: 12 × 1 kernel, 21 stride,12 filters
C2	Batch Normalization Dropout Max Pooling: 2 × 1 kernel, 2 stride
C3	Convolution: 6 × 1 kernel, 1 stride, 6 filters
C4	Batch Normalization Dropout Max Pooling: 2 × 1 kernel, 2 stride
C5	Convolution: 1 × 1 kernel, 1 stride, 3 filters
D1	Batch Normalization Dropout Dense connection: 32 units
D2	Batch Normalization Dropout Dense connection: 128 units
D3	Batch Normalization Dropout Dense connection: 256 units

radioisotopes in background NORM: K, U, and Th. Despite this intuition, the 3 components in the latent space did not learn to represent these components. The functional mapping between the input and the latent space and between the latent space and the output is far more abstract than that, and in many cases the relationship between the latent space components and the input spectra is not interpretable. Unlike the basis vectors in PCA and NMF, the neurons in an autoencoder's latent space are not the only parameters of the model. Rather, every neuron in every layer of the model can represent an important piece of information contributing to the spectral reconstruction. With this in mind, the latent space cannot be interpreted in the same way as that of the basis vectors in PCA and NMF, they simply represent the location in the model with the highest level of compression.

Before designing the final model, a variety of optimizations were made on the initial one, which largely inspired the final design. First, the number of filters in the first convolutional layer was increased three-fold. Likewise, the number of filters in the consecutive layers of the encoder were all increased and another convolutional layer was added. With regards to the kernel size of the first convolutional layer, it was found that a relatively large range of kernel sizes (about 50 keV to 150 keV) produced similar results on both datasets.

The depth of the decoder was varied. There was no significant increase in performance when increasing the depth beyond the 3 dense layers used in the initial model. Likewise, reducing the depth by one layer did not change performance, but reduced the number of parameters in the model which is advantageous from a computational perspective.

One of the problems with the initial model was that the latent space size was determined by the convolutional layer kernel sizes and strides, thus it was not an adjustable parameter of the model. This made it difficult to test how the latent space size affects performance, which is one of the motivations for the final network design that will be discussed in the next section.

4.3.2 Final Architecture

When selecting the kernel size of the initial convolutional layer in a neural network, it can be useful to consider the size of the features that the layer might need to detect. In gamma ray spectra, the main features important to spectroscopy are full energy photopeaks. Full energy photopeaks present as Gaussian shaped peaks with a standard deviation given by Equation 4.1. The FWHM of these peaks varies between 40 keV at 500 keV and 120 keV at 3 MeV. For a 128 bin spectrum with equal sized bins between 0 and 3 MeV, this corresponds to 2 or 3 bins at 500 keV and 5 or 6 bins at 3 MeV. Because of this spread, it is not clear what the best choice for kernel size would be. Also, different detectors can have better energy resolution than others (resolution tends to be better for smaller detectors than for larger detectors), which adds an additional consideration for kernel size selection. Compton edges and stacked photopeaks are also important features that typically are larger in size than a typical photopeak.

Rather than using just one kernel size to try and capture all of these features into a single convolutional layer, I designed the encoder stage of the network to contain multiple convolutional branches with different kernel sizes that each observe the spectrum at different energy scales. This can be seen on the final model diagram in Figure 4.5 with operation labels listed in Table 4.2. The theory behind this branched structure is that each branch may be better suited to a particular type of feature than just a single one. In testing this theory, it was found that multiple branches was indeed slightly beneficial to network training convergence and did not significantly impact the computational speed of the model (convolutional layers are computationally cheap). The reduction in final validation set loss peaks at around 3 branches, with more not leading to any significant improvement in the validation loss. The total improvement when going from 1 to 3 branches is a reduction in final validation loss of 15% for the network trained on Node 11 data. As such, the final network has three convolutional branches in the encoder with first-layer kernel sizes of 3, 5, and 10 respectively. Larger kernel sizes were also tested, but using more than 10 did not make any measurable difference in training convergence. A single branch with a kernel size of 10 works pretty well on its own, but given the minimal difference in computational speed between the network with 3 branches and a single branch, the slight gains in performance were worthwhile.

Each convolutional branch ends with a single dense layer that converges down to 5 neurons per branch. Prior to having this dense layer, the output of the convolutional layers were simply concatenated together into a single row. The dense layers were placed here in order to allow for the kernel sizes and strides to be modified in the convolutional layers without affecting the number of neurons at the output of each branch. The outputs of each dense layer are then concatenated together to create a single feature map of size 15.

As previously discussed, one of the main issues with the original model was the inability to change the size of the latent space without changing the kernel sizes and strides of the convolutional layers. This problem was solved in the final model by placing a dense layer between the branch convergence layer (size 15) and the latent space. With this structure, the latent space can be anywhere between 1 and 15 neurons wide.

While it was already tested when designing the initial model, replacing the dense decoder with a convolutional decoder was also tested on the final model. Because a true "deconvolutional" layer does not exist, this is typically implemented by a series of upsampling layers followed by a convolutional layer that learns to interpret the upsampled activations. Unfortunately, the use of a convolutional decoder made the network significantly more difficult to train and in the few times it was successful, it did not reach the performance of the dense model. Thus, the dense decoder did not change much from the initial design except that the number of layers were cut from 3 to 2 since the network was designed from 128 bin spectra rather than 256 bin spectra.

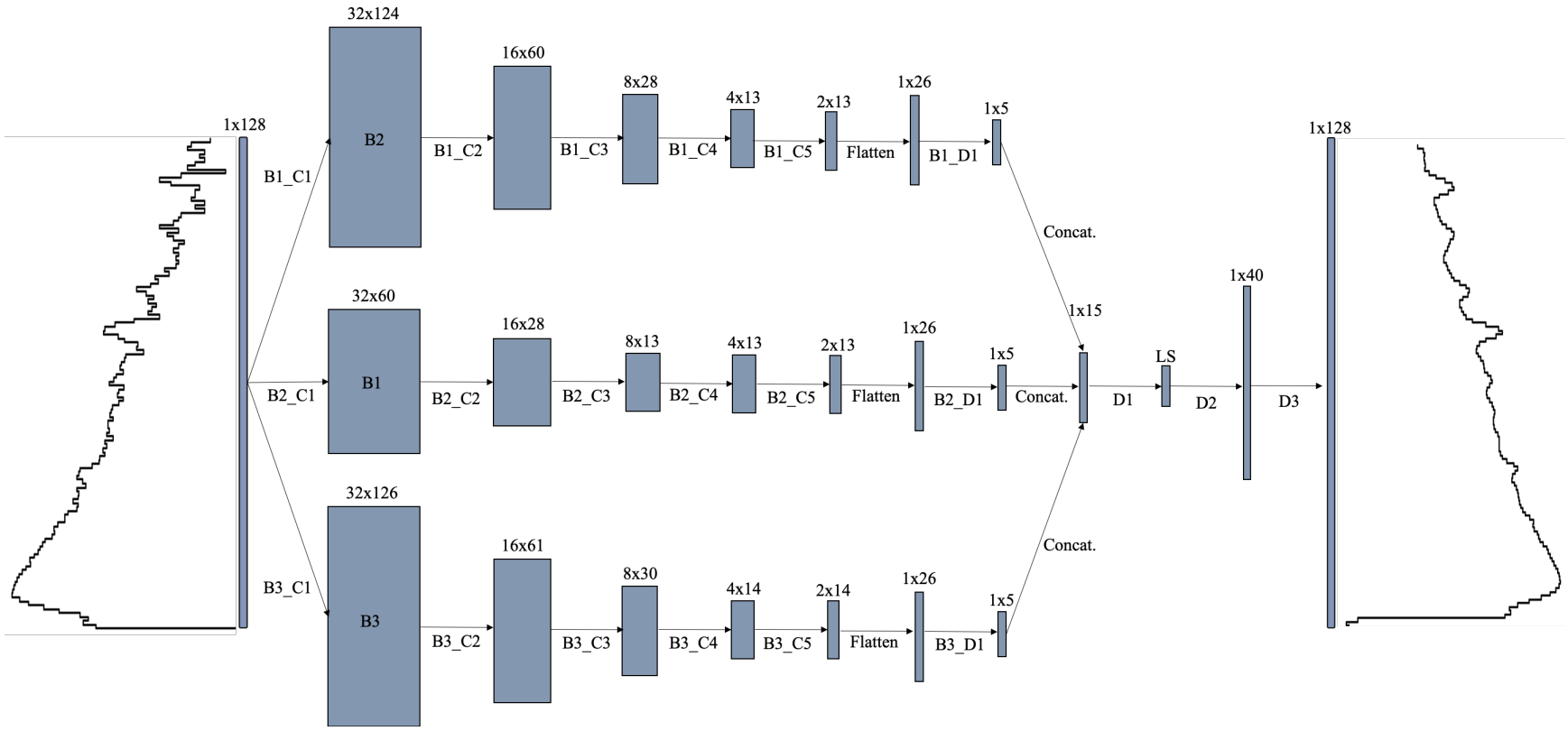


Figure 4.5: Final design of the ARAD model. Layer labels are shown in Table 4.2. The spectra on each end of the autoencoder are actual input and output background spectra from Node 11 in the HFIR/REDC dataset.

4.3.3 Latent Space Size

The latent space is the point of highest compression in the network, thus its size can be an important factor in network performance and generalization. The latent space size should be large enough to capture the dynamics present in the background data, but not so large that the network is able to start capturing random variations in the data that do not represent true physical features in the data. Like most aspects of network design, determining the point at which this occurs is mostly qualitative. If the spectrum reconstructions on the training dataset appear to fit the input spectra so well as to capture the statistical variations of the true signal, then there are too many neurons in the latent space. This can be thought of as a form of overfitting. The output spectrum of the network should contain well-formed Gaussian photopeaks, even when noise is present in the input spectrum. This indicates that the network has learned the true underlying signal in the data. Figure 4.6 shows an example of this.

The process of determining whether the latent space is too small is very similar. If the reconstructed spectrum is the same for any and all input spectra, then the latent space is too small and the network has learned to produce a single output that approximately fits the majority of the input data. This is similar to the problem described in Section 4.4.2, however these two problems are different in that the problem described in in Section 4.4.2 occurs regardless of the size of the latent space.

It was found that a latent space size of between 4 and 8 works well for most cases. For the final network, a latent space size of 5 is used and produces very good results for both the simulated and real world data. Nevertheless, the architecture is designed such that the latent space can be easily changed if the need ever arises.

Finally, the latent space can be encouraged to learn a sparse representation of the input data by including a regularizer in the loss function that penalizes the latent space when it deviates from a user-defined level of activation sparsity [78]. The regularizer used in this model is the Kullback-Leibler divergence (KLD) with a sparsity parameter of 0.001 (lower means more sparsity) and a regularization parameter value of 0.05 (higher means more weight

Table 4.2: Operations performed at each layer in the final ARAD model. The location that each label references in the model can be seen in Figure 4.5.

ARAD Final Layer Operations	
Label	Operation
B1_C1	Convolution: kernel=5, stride=1, filters=32 → Batch Norm. → Dropout
B1_C2	Convolution: kernel=6, stride=2, filters=16 → Batch Norm. → Dropout
B1_C3	Convolution: kernel=6, stride=2, filters=8 → Batch Norm. → Dropout
B1_C4	Convolution: kernel=4, stride=2, filters=4 → Batch Norm. → Dropout
B1_C5	Convolution: kernel=1, stride=1, filters=2 → Batch Norm. → Dropout
B1_D1	Dense: 5 units → Batch Norm.
B2_C1	Convolution: kernel=10, stride=2, filters=32 → Batch Norm. → Dropout
B2_C2	Convolution: kernel=6, stride=2, filters=16 → Batch Norm. → Dropout
B2_C3	Convolution: kernel=4, stride=2, filters=8 → Batch Norm. → Dropout
B2_C4	Convolution: kernel=1, stride=1, filters=4 → Batch Norm. → Dropout
B2_C5	Convolution: kernel=1, stride=1, filters=2 → Batch Norm. → Dropout
B2_D1	Dense: 5 units → Batch Norm.
B3_C1	Convolution: kernel=3, stride=1, filters=32 → Batch Norm. → Dropout
B3_C2	Convolution: kernel=6, stride=2, filters=16 → Batch Norm. → Dropout
B3_C3	Convolution: kernel=3, stride=2, filters=8 → Batch Norm. → Dropout
B3_C4	Convolution: kernel=4, stride=2, filters=4 → Batch Norm. → Dropout
B3_C5	Convolution: kernel=1, stride=1, filters=2 → Batch Norm. → Dropout
B3_D1	Dense: 5 units → Batch Norm.
D1	Dense: Latent Space Size → Batch Norm.
D2	Dense: 40 units → Dropout
D3	Dense: 128 units

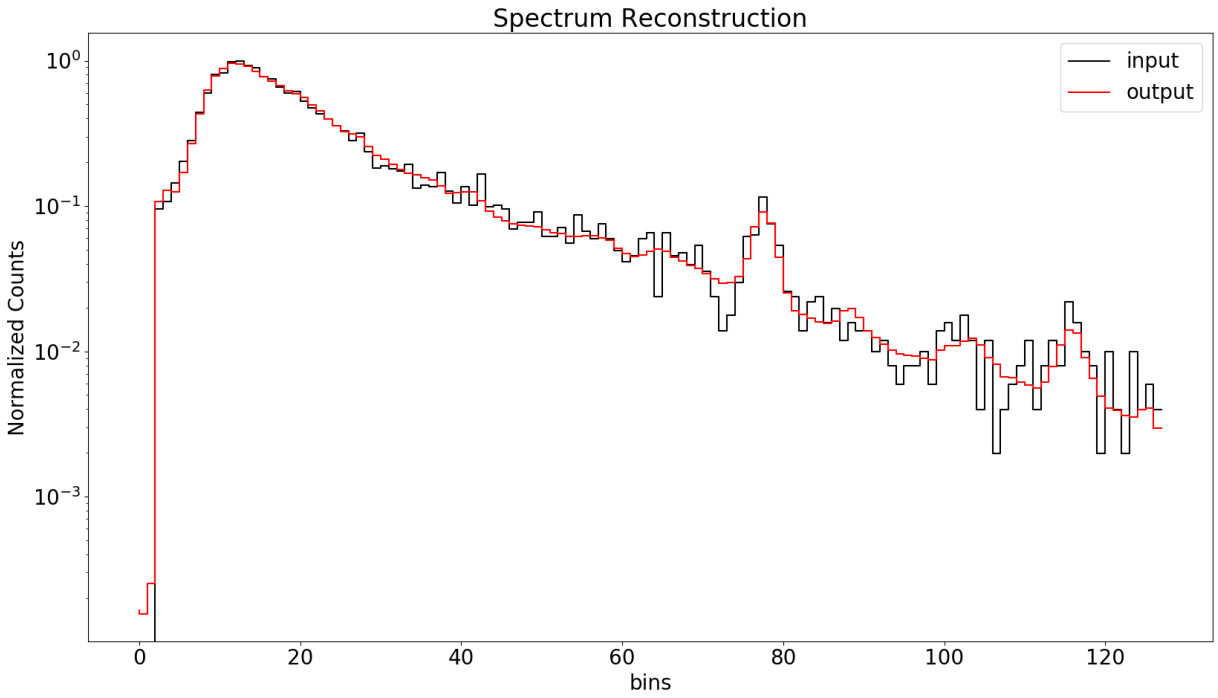


Figure 4.6: The reconstructed spectrum should contain well-formed Gaussian photopeaks (as seen here) and should not fit to the noise in the input spectrum. The integration time for this spectrum is 5 seconds.

is given to enforcing the sparsity). The theory behind this is discussed in detail in Reference [78].

4.3.4 Loss Function Selection

When analyzing gamma ray spectra, the spectra are often viewed on a logarithmic y axis because it makes it easier to visualize the higher energy photopeaks. This is because the background continuum is prevalent in the lower energy region, giving that region a typical countrate around two orders of magnitude higher than the high energy regions. Not surprisingly, better network convergence was found when using loss functions that operate logarithmically on the error between the input and output. Such examples are the logcosh and mean squared logarthmic error. Of these two, logcosh demonstrated the most stable

training. Loss functions such as the mean squared error and mean absolute error do not work as well, especially in the higher energy regions where the count rate is very low.

4.3.5 Activation Function Selection

Of the *many* different hyperparameters that can be used to tune the model, the activation function was one of the most significant contributors to the performance of the model. Many activation functions were tested, including: sigmoid (logistic function), hyperbolic tangent (tanh), rectified linear unit (relu), exponential linear unit (elu), leaky relu, and softplus. Of course, each activation function was tested with different learning rates and optimizers in order to develop a good understanding of each functions true performance. One of the most popular and widely used activation functions in the past few years has been the relu and its derivatives (such as leaky relu). Despite its popularity, especially on image processing tasks, I was unable to train a useful model using the relu functions. The leaky relu works slightly better, however neither were able to converge to a stable solution when training. The first few working network designs used either tanh or sigmoid which both resulted in stable training and good network performance. The final network however uses the softplus function which maintains the training stability of the sigmoid and tanh functions but tends to converge faster. Softplus is also a relatively modern activation function [79]. It has roughly the same shape as the relu activation function, but rather than having the discontinuity at 0, the transition from $y = 0$ to a linear form occurs gradually and smoothly. The softplus is given be Equation 4.5 and is plotted in Figure 4.7.

$$y(x) = \ln\left(1 + \mathrm{e}^{x}\right) \tag{4.5}$$

4.4 Training

4.4.1 Optimizer Selection

A total of 6 different optimizers were tested for the training process: Adam/Nadam, standard SGD, SGD+momentum, SGD+Nesterov momentum, Adadelta, Adagrad, and RMSProp.

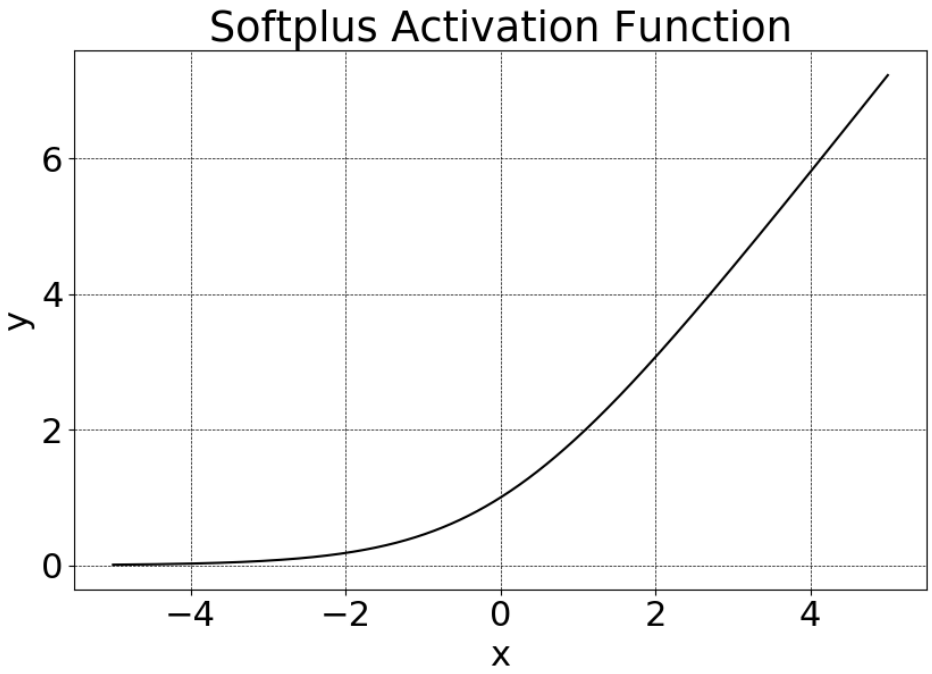


Figure 4.7: Softplus activation function [79].

Further, a range of values for the various hyperparameters for each of these optimizers were tested. The best results were had for Adam/Nadam both in terms of final accuracy and training time. SGD+momentum was able to achieve similar accuracy to Adam/Nadam, however the training time required to reach that level of accuracy was several hundred epochs longer even when using a scheduled learning rate. With respect to Adam versus Nadam, both optimizers achieved the same or very similar results in performance and training time, thus Nadam was selected as the optimizer for training the final model.

4.4.2 Local Minima

By far the greatest challenge in training this network is preventing the network from learning to simply produce the mean of the training data. Because the variance between training samples is so low, the mean of the training data is actually not that bad of a solution. From analyzing the networks that do fall into this local minimum, it was found the output of the

network is nearly identical for all inputs, even zero input. This indicates that the network was learning to tune the neuron biases in order to output the mean without actually processing the input data. Applying a bias regularizer (L1 and L2 norm were both tested) lead to other problems with training the network. Regularization of the weights enhanced the problem by further encouraging the network to cancel out the weights and rely solely on the bias.

It is important to note that this particular local minimum does not just occur for the final network architecture, in fact, it occurs on every architecture that was tested. A solution was found however. Based on the idea that the weights were shrinking and the network was learning to rely on the biases, a min-max norm constraint was added to the weights, such that after every weight update, the weights coming into each neuron were scaled in order to have a norm no less than a user-defined minimum and no more than a user-defined maximum. The constrains the network's ability to ignore the weights and reliably enables the network to escape the local minimum.

4.4.3 Overfitting Prevention

Because the network is very difficult to train as it is, overfitting is not a major concern for this model. This being said, steps were taken to prevent the potential for overfitting. First, a dropout of 10% was used. Dropout not only helps to prevent overfitting but also encourages a more robust representation of the data by allowing the network to act as an ensemble of multiple, separate networks [52]. Early stopping is also enabled during the training process, which monitors the training and validation loss and stops the training when overfitting is detected [53]. While it used primarily to stabilize the training process, batch normalization also plays a role in reducing the chance of overfitting [80]. Finally, the weight constraints used for preventing the network from falling into the mean local minimum may also limit the ability of the network to overfit, however this has not been verified.

4.4.4 Training Data Selection and Formatting

Chapter 3 introduced the datasets use for training and evaluating the ARAD model in detail. This section intends to describe how this data was sampled and formatted for creating

the training and validation sets for the network. Further, one of the top questions when considering a datacentric algorithm such as ARAD is the amount of data required to train the network. This is important from both a computational and an operational standpoint. Likewise, it is important to consider what sort of background dynamics that these data must contain: i.e. what proportion of the data must contain typical background, cluttered background, precipitation, etc. From investigation, it turns out that the answer to these questions is not so clear and is dependent on the background dynamics at the location of interest, however, some insights were learned and are presented here.

HFIR/REDC Dataset

Two separate models are trained for the HFIR/REDC dataset, one for the Node 01 and one for Node 11. The reason that these two nodes were chosen is because they represent both extremes in the dataset. Node 01 has a relatively stable background with very little clutter present. On the other hand, Node 11, being right next to the REDC loading dock has a very high level of clutter from the loading crane and large transport vehicles. Both nodes experience the typical NORM and precipitation-induced background variations. The clutter-induced variations in the NODE 11 spectra are far more complex than that arising from NORM and precipitation. Rain induces photopeaks from ^{214}Bi and ^{214}Pb and NORM results in variations in the KUT photopeaks. On the other hand, the spectral changes resulting from clutter depend on the energy of the photons being Compton scattered and also the shielding configuration of the clutter object.

The amount of data required to train the model depends on the complexity of the background. For cases where there is a lot of clutter, more data is needed than for cases without a high level of clutter. This being said, from a machine learning standpoint it is best to have equal amounts of data for each scenario, that is, equal amounts of precipitation background, cluttered background, and NORM background. If one of the classes is over-represented in the data, the model may become biased towards that particular class [81]. With this in mind, the size of the training dataset is determined by the most complex class of background dynamics present in the data, which in the case of NODE 11, is clutter. As such,

the process for determining the amount of data needed for each class is actually determined by the amount of data needed for the clutter class.

The entire dataset for Node 11 contains spectra generated from 16 hours each of NORM, precipitation, and cluttered background, for a total of 48 hours. Unlike was done for the initial model design, the spectra are integrated with integration times of 5, 10, 30, 60, and 120 seconds and then combined into one larger dataset such that the actual amount of data in the dataset is $5 \times 48 = 240$ hours. The dataset is also split into a training and validation set, such that the training set contains 120 hours of spectra generated from 24 hours of actual data. Using more data did not affect the model's performance, however reducing the amount of data to only 12 hours of actual data does negatively impact performance.

The data generation procedure for Node 01 was the same as that of Node 11, however only 38 hours of actual data were used to generate the spectra, roughly 19 hours each for precipitation and NORM background. This results in a training set containing roughly 95 hours of spectra generated from 19 hours of actual data.

It is important to note that while only a few days of actual radiation data is required to train these two nodes, this does not mean that this amount of data can be collected in a single period of this time. The training data for Node 01 are extracted from different days between October of 2018 and February of 2019. Likewise, the training data for Node 11 are extracted from different days between August of 2018 and January of 2019.

It is important to mention that there is no true upper limit to the amount of data that can be used for training. If more data is available than is required, there is no reason, other than for computational considerations, why more data should not be used. With this being said, it is important to ensure that the amounts of data from the different classes of background dynamics are equally (roughly) represented, as it is well known the unbalanced datasets can lead to biased models [81]. Also, while neural network in general are relatively robust to mislabeled data [82], especially when using the logcosh loss function, it is important to ensure that the training data only contain background spectra because the inclusion of source data may desensitize the network to that source.

Simulation Dataset

The simulated dataset does not contain any clutter or precipitation-induced background variations, however it does have a high level of variation in the NORM KUT components. Good results were obtained when using 12 hours of actual radiation data to generate the dataset. Like was done for the HFIR/REDC data, the spectra were integrated with integration times of 5, 10, 30, 60, and 120 seconds, resulting in a total of 60 hours of spectra. These data are split into training and validation test sets, resulting in a total of 30 hours of spectral data in the training set.

4.4.5 Training Results

Figure 4.8 shows the training and validation loss for the simulated data. As can be seen, early stopping was engaged at roughly 145 epochs. Figures 4.9 and 4.10 shows the loss for Nodes 01 and 11, respectively. Early stopping was engaged at roughly 130 for Node 01 and 100 for Node 11. The number of epochs required to train is largely dependent on the amount of data which explains why Node 11 required fewer epochs than Node 01 which required fewer than the simulation data.

4.4.6 Training Parameters

The training parameters are stored in a JSON config file, making it easy to test different parameter values for training. The config file (minus unused parameters) for training the final ARAD model is shown in Listing A in the Appendix.

4.5 Spectrum Reconstruction Metric

In order to detect an anomaly, there must be a way to quantify how well the network is able to reconstruct the input spectrum. This is called the spectrum reconstruction metric. Choosing an appropriate metric is actually quite difficult. The ideal spectrum reconstruction metric would have the following characteristics:

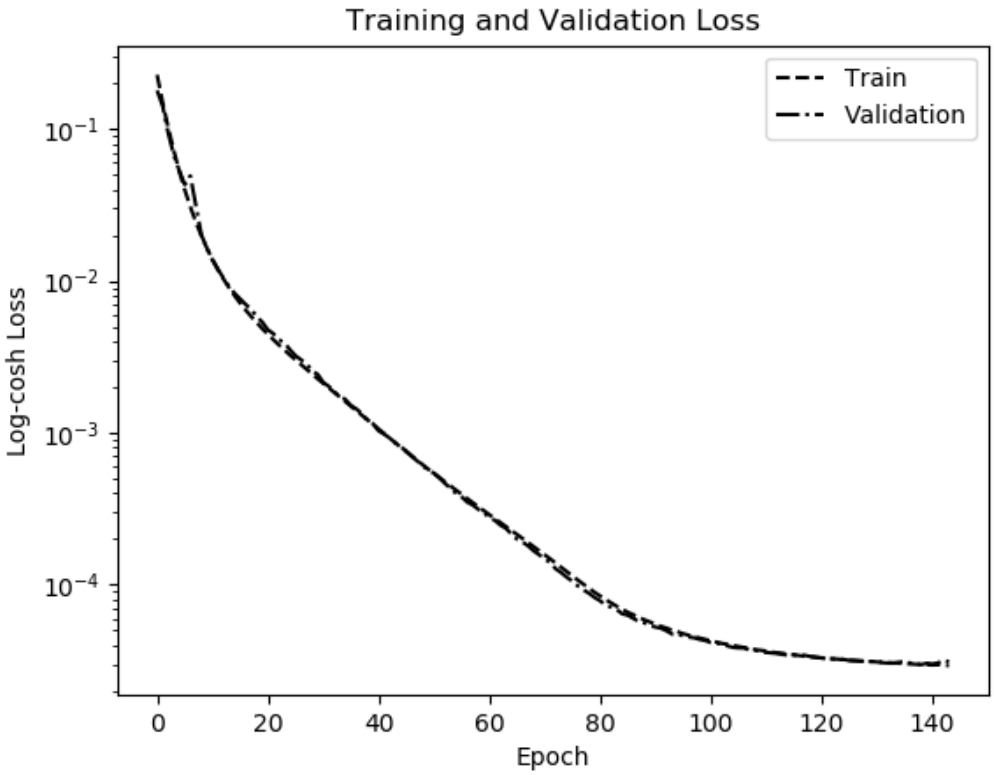


Figure 4.8: Training and validation loss when training ARAD on the simulated dataset.

- The metric should be independent from the gross count rate. As discussed in Chapter 1, changes in the gross count rate are not always indicative that a source is present.
- The metric should be sensitive to differences between the features present in the input spectrum and the output spectrum regardless of the signal to noise ratio. As discussed in Section 4.3.3, the reconstructed spectrum does not (and should not) have the information capacity to reproduce the statistical noise in the input spectrum, thus the reproduced spectrum will always have a higher SNR than the input spectrum. The ideal metric would have a magnitude *dependent* on differences in the underlying signal and *independent* of the difference in the statistical noise between the input and output spectrum. If this is not the case, then the metric would indicate worse reconstruction

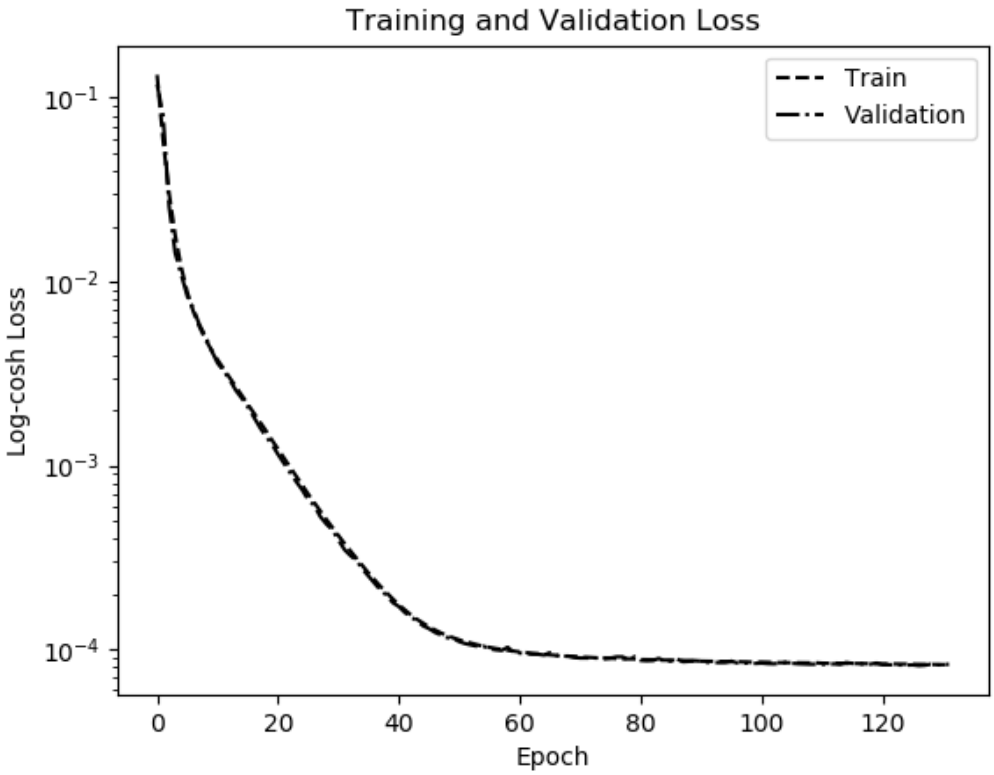


Figure 4.9: Training and validation loss when training ARAD on Node 01 data from the HFIR/REDC dataset.

performance on spectra with a lower gross count rate than on spectra with a higher gross count rate and less uncertainty.

Many different metrics were tested in order to best meet the preceding criteria. The squared mean error (SME, used in the initial design), logarithm of the hyperbolic cosine (logcosh), reduced chi squared statistic, Jensen-Shannon distance [83], and the Kullback-Leibler divergence (KLD) [33] were all tested. These metrics were compared by running the trained network on a run from the simulation dataset with a highly dynamic background count rate and a weak source. The highly dynamic background count rate induces large changes in the SNR of the observed signal. The spectrum integration time is held constant at 5 seconds. As stated, the ideal metric would have a large magnitude increase at the position of the source (changed in feature space) and should otherwise not change much as

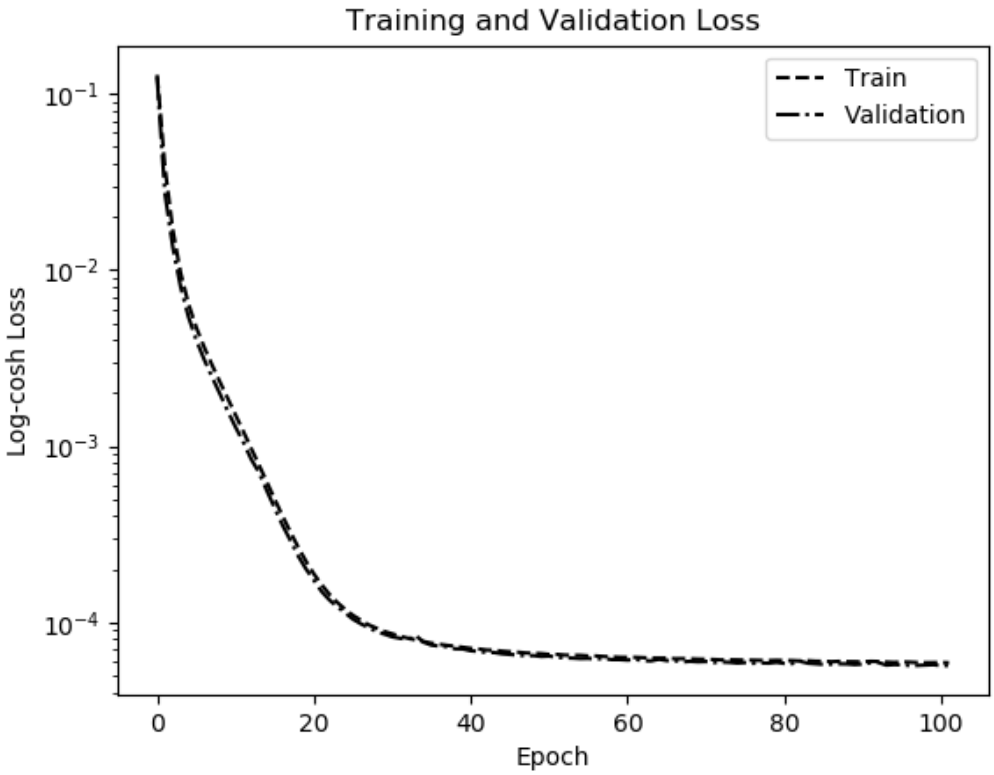


Figure 4.10: Training and validation loss when training ARAD on Node 11 data from the HFIR/REDC dataset.

a function of SNR. An example of this comparison test can be observed in Figure 4.11; of course, it was also performed for many more runs and also on the HFIR/REDC dataset.

From Figure 4.11, it can be seen that the Jensen-Shannon distance is clearly not appropriate for this application. Likewise, the SME appears to have missed the source and in general is quite noisy and spurious. The logcosh, reduced chi squared, and KLD all follow very close to one another, with the logcosh and KLD being nearly identical. The reduced chi squared is slightly more sensitive than logcosh and KLD to fluctuations in the background statistics. Due to its foundation in information theory as a measure of the difference between two probability distributions, the KLD was selected as the metric to use. The KLD reconstruction metric is given by Equation 4.6, where $Q(x)$ is the input spectrum and $P(x)$ is the output spectrum, both defined on the same probability space χ. From a

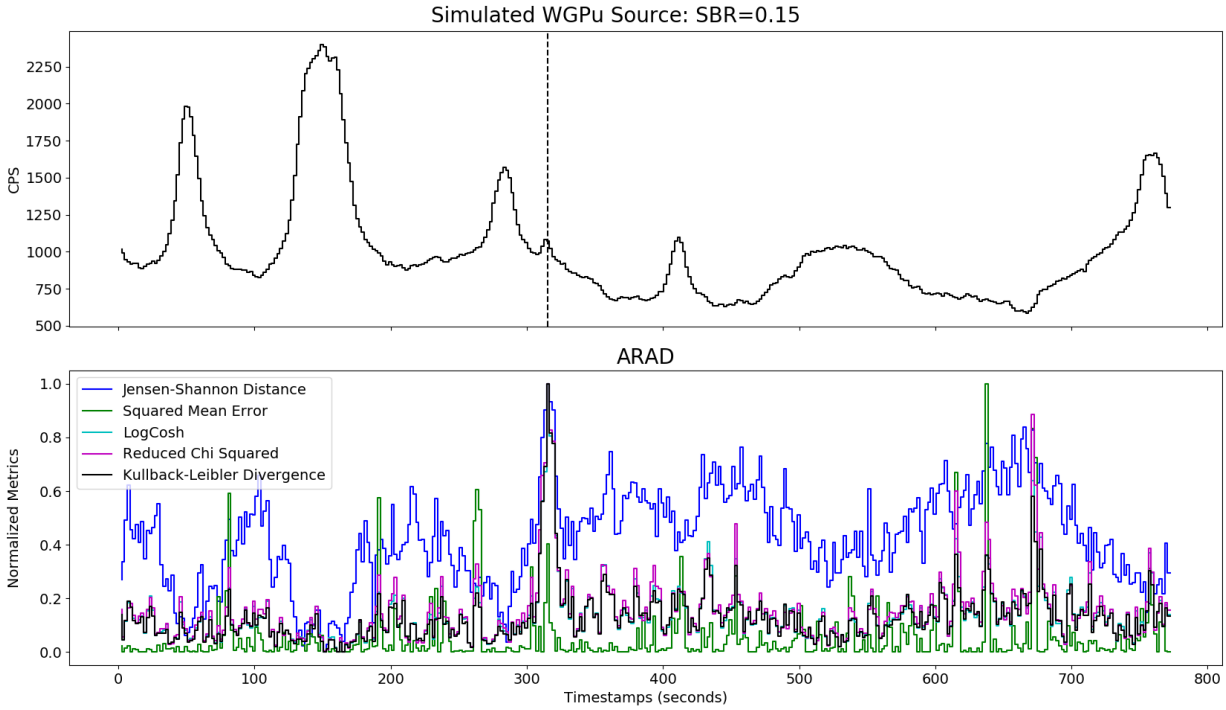


Figure 4.11: Comparison between various spectrum reconstruction metrics when evaluated on a run from the simulated dataset. Logcosh and KLD are both nearly identical, thus they are difficult to distinguish in this figure.

statistical standpoint, it can be viewed as the expectation of the log-likelihood ratio between the input and output spectra. a KLD of zero indicates that the two spectra are identical and its value increases as the two distributions diverge from one another.

$$\mathrm{D_{KL}} = \sum_{x \in \chi} P(x) \log \left(\frac{P(x)}{Q(x)} \right) \tag{4.6}$$

4.6 Detection Threshold Optimization

An anomaly alarm is triggered when the KLD metric increases above a user-defined threshold. The threshold can be selected by running the algorithm on the training and validation data with a range of threshold values and scoring the false alarm rate as a function of detection threshold. To set the detection threshold, one can select the desired false alarm rate and use the corresponding threshold.

Choosing the threshold based on the false alarm rate is a decent method to use when only background data is available. This is the method used for establishing the detection thresholds on the HFIR/REDC detectors nodes and has shown to work well in practice. If data is available that also contains source data with varying *known* source strengths and types, then a receiver operating characteristic (ROC) can be generated which plots the algorithm's true alarm rate against its false alarm rate for a range of detection thresholds. The threshold can then be found by considering both the true and false alarm rate simultaneously. This is the method used for the simulation dataset. If source data is not present for a particular dataset, one can potentially use a radiation transport model to "inject" simulated source data into the real-world background data. In practice however, this is not always possible or practical from an operations standpoint.

Chapter 5

ARAD Performance in a Simulated Source Search

As discussed in Chapter 3, the Topcoder dataset presents a unique opportunity to develop and evaluate detection algorithms in a well-controlled and realistic urban testbed. Having the data generation scripts also allowed for the creation of a custom dataset from the Monte Carlo model for generating algorithm detection performance curves. Because Chapter 3 already introduced the dataset and model, this chapter will focus on the results of applying the ARAD algorithm to the data.

5.1 Example Alarm Spectra

This section presents a set of alarm spectra for each of the sources in the simulated dataset. These figures demonstrate ARAD's ability to detect anomalous sources, even those with very weak spectral signals, in a highly dynamic background environment. The alarm threshold was set such that the false positive rate was 1 alarm per 8 hours. All of the spectra shown in this section were integrated with an integration time of 5 seconds. It may be useful to go back and look at the source detector response in Figure 3.2 in Chapter 3 before reading through this section.

5.1.1 Highly Enriched Uranium (HEU)

The gamma ray energies emitted by the HEU source in this model are relatively broad, with energies spanning across nearly the entire 3 MeV spectrum. There are a few photopeaks of interest to note here: there is a peak situated around 186 keV which is a stacked photopeak consisting of gamma rays of energy 185.7 keV from ^{235}U and 186.2 keV from ^{226}Ra. The detector response peaks at about 100 keV, which are actually X-ray emissions from uranium and plutonium. Also notable is the 1001 keV peak that stems from ^{234m}Pa, which is a part of the ^{238}U decay series. Finally, there is a rather large peak at 2614 keV from ^{208}Tl, part of the ^{232}Th decay series.

Figure 5.1 shows an alarm spectrum for HEU with a source to background ratio of 0.15, that is, the count rate in the detector from HEU at the point of closest approach to the source is 15% of the background count rate. From just observing the spectrum, it can be seen that there are no distinguishable photopeaks in the spectrum from the HEU source. Instead, there is just a slight increase in the amplitude of the spectrum between 100 and 200 keV. In its reconstruction, the network was unable to account for this lower energy increase, leading to an alarm. One way to visualize the cause of the alarm is to plot the Kullback-Leibler divergence as a function of energy (that is, calculate the KLD per energy bin without summing the result). In this figure, it can be seen that the location in the spectrum that contributed most to the alarm was in this lower energy region, as expected.

5.1.2 Weapons Grade Plutonium (WgPu)

As seen in Figure 3.2, the WGPu gamma rays are concentrated in the lower energy region of the spectrum. Of particular interest is the prevalent 60 keV gamma ray from ^{241}Am. When ^{239}Pu absorbs two neutrons it becomes ^{241}Pu which then beta decays into ^{241}Am. Also of note are the stacked photopeaks stemming from the 375 keV, 414 keV, and a plethora of gamma rays in the 330 keV region that come from ^{239}Pu. Finally, the large hump between about 600 and 800 keV are the result of dozens of different gamma ray energies from ^{239}Pu, ^{240}Pu, and ^{241}Am. A more in-depth overview of the gamma ray signatures from plutonium is available in a report by Los Alamos National Laboratory [84].

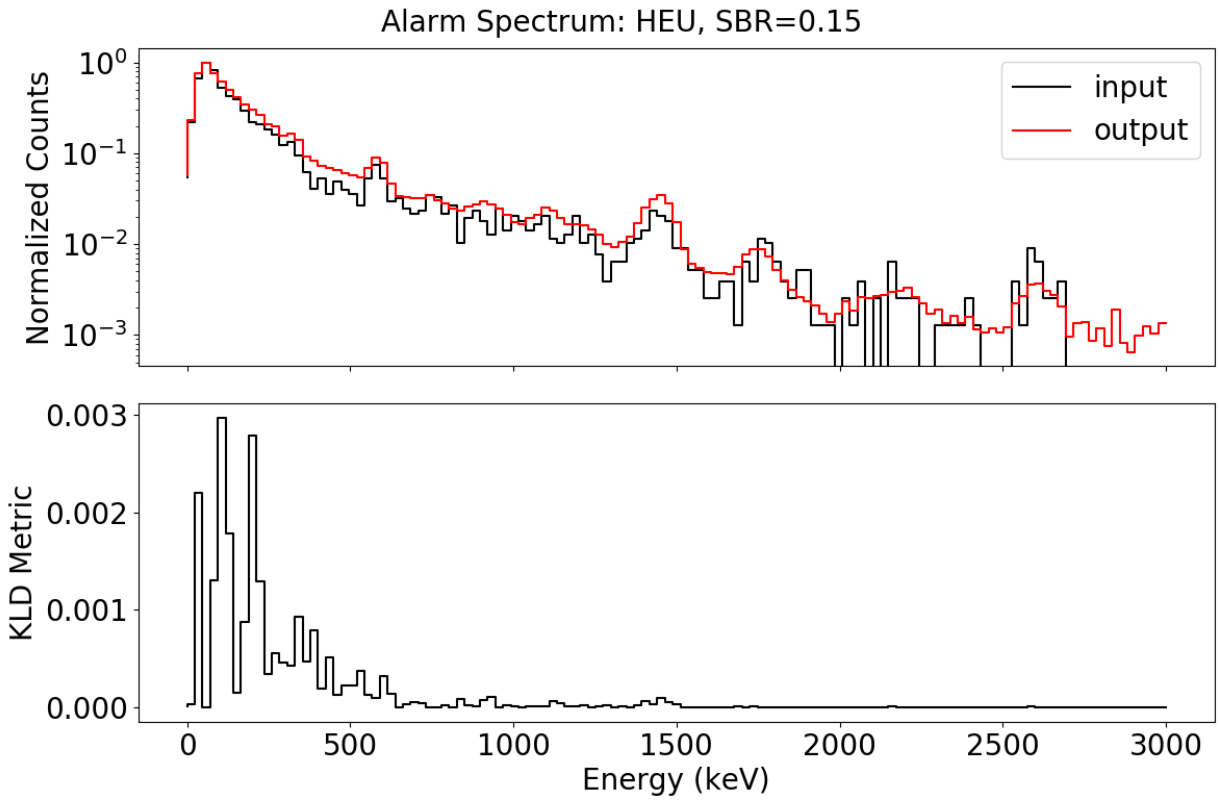


Figure 5.1: Alarm spectrum with reconstruction for HEU with an SBR of 0.15. Also shown is the KLD metric as a function of energy.

Figure 5.2 shows an example alarm spectrum for a WGPu source with an SBR of 0.07. As expected, the biggest contributor to the reconstruction error stems from the very intense 60 keV peak from ^{241}Am. The peaks in the 300 keV region and the 600 keV region also played a role, however their effects were minuscule compared to the 60 keV peak.

5.1.3 ^{131}I

^{131}I is a very short lived medical isotope with a half life of just slightly over 8 days. About 82% of emitted gamma rays have an energy of 364.5 keV, with the rest mostly having an energy of 637 keV. ^{131}I also has two characteristic X-rays at 29.5 and 29.8 keV and another at 80.2 keV. These three peak region tend to line up with those from WGPu as can be seen in Figure 3.2.

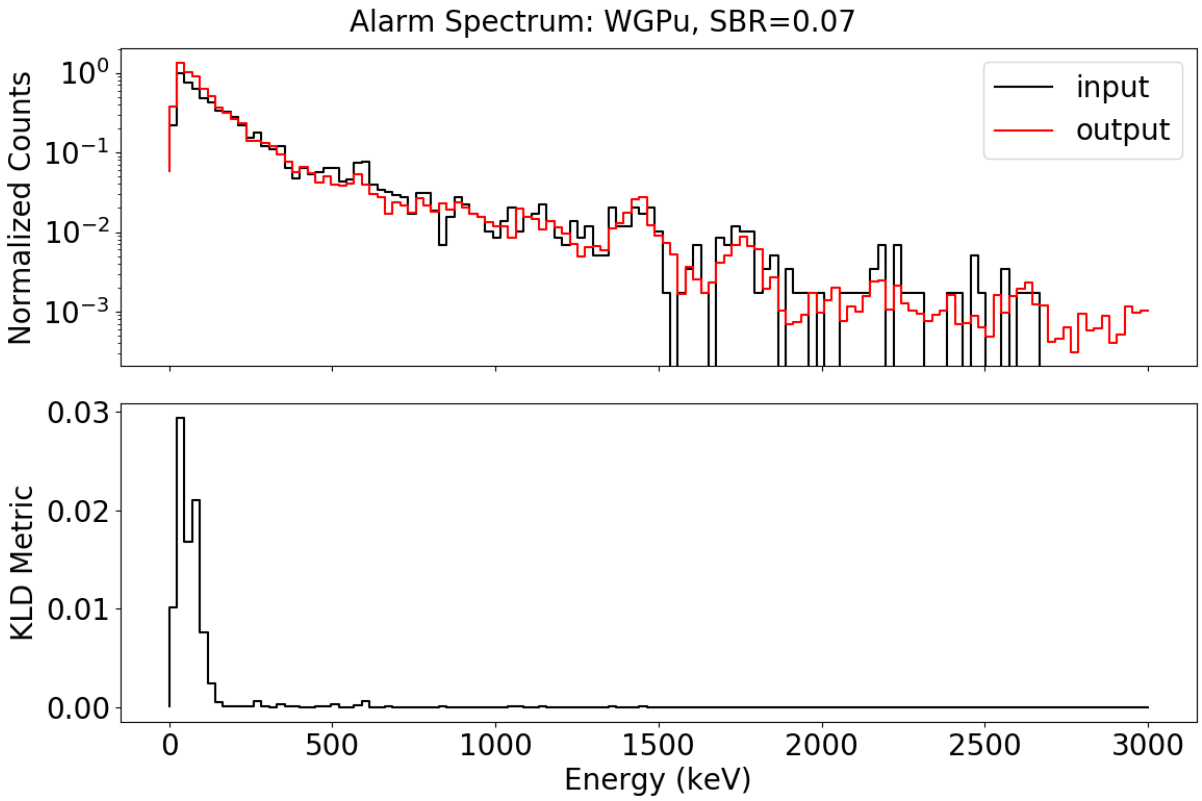


Figure 5.2: Alarm spectrum with reconstruction for WGPu with an SBR of 0.07. Also shown is the KLD metric as a function of energy.

Figure 5.3 shows an alarm spectra for ^{131}I with an SBR of 0.1. As can be seen, the biggest contributor to the reconstruction error came from the 364.5 keV gamma ray. There is also a notable peak in the KLD around 637 keV and also from the characteristic X-rays.

5.1.4 ^{60}Co

^{60}Co is a common industrial isotope with two very prevalent gamma rays at 1173 and 1332 keV. 99.85% of the time, ^{60}Co will emit both of these gamma rays in near coincidence [6]. Otherwise, only the 1332 keV gamma is emitted. At these energies, Compton scattering plays a significant role in the photon interactions in the detector as can be seen in Figure 3.2, which makes it a bit more difficult to detect than some of the lower energy sources (this is discussed in the next section).

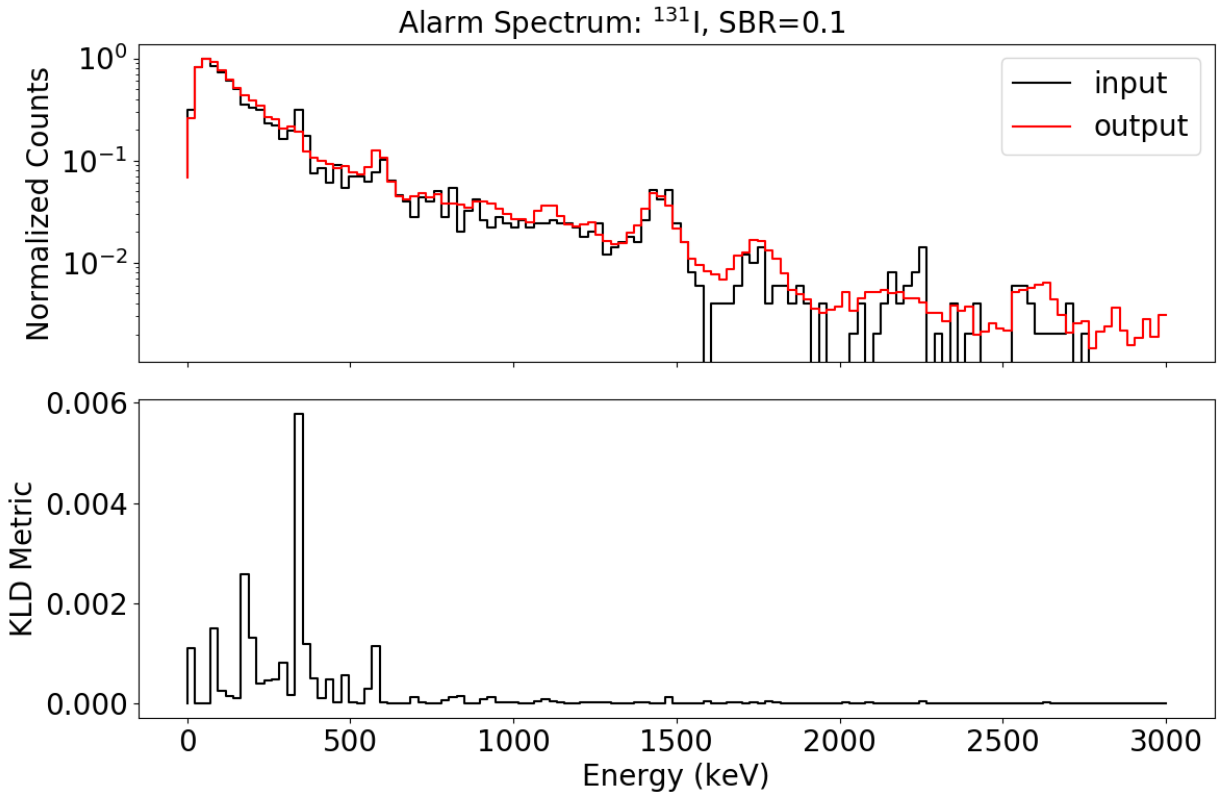


Figure 5.3: Alarm spectrum with reconstruction for ^{131}I with an SBR of 0.1. Also shown is the KLD metric as a function of energy.

Figure 5.4 shows the alarm spectrum for a ^{60}Co source with an SBR of 0.2. On the KLD metric it can be seen that the two gamma ray photopeaks play a big role in the reconstruction, however, the biggest contributor is the lower energy Compton scattering.

5.1.5 ^{99m}Tc

^{99m}Tc is a very common medical isotope that emits a a single 140 keV gamma ray alongside several characteristic X-rays around 20 keV. Figure 5.5 shows an alarm spectrum for a ^{99m}Tc source with an SBR of 0.07. The low energy of the gamma rays emitted by ^{99m}Tc cause the normally rounded shape of the background continuum peak to take on a much narrower peak that the network is unable to reproduce, leading to an alarm.

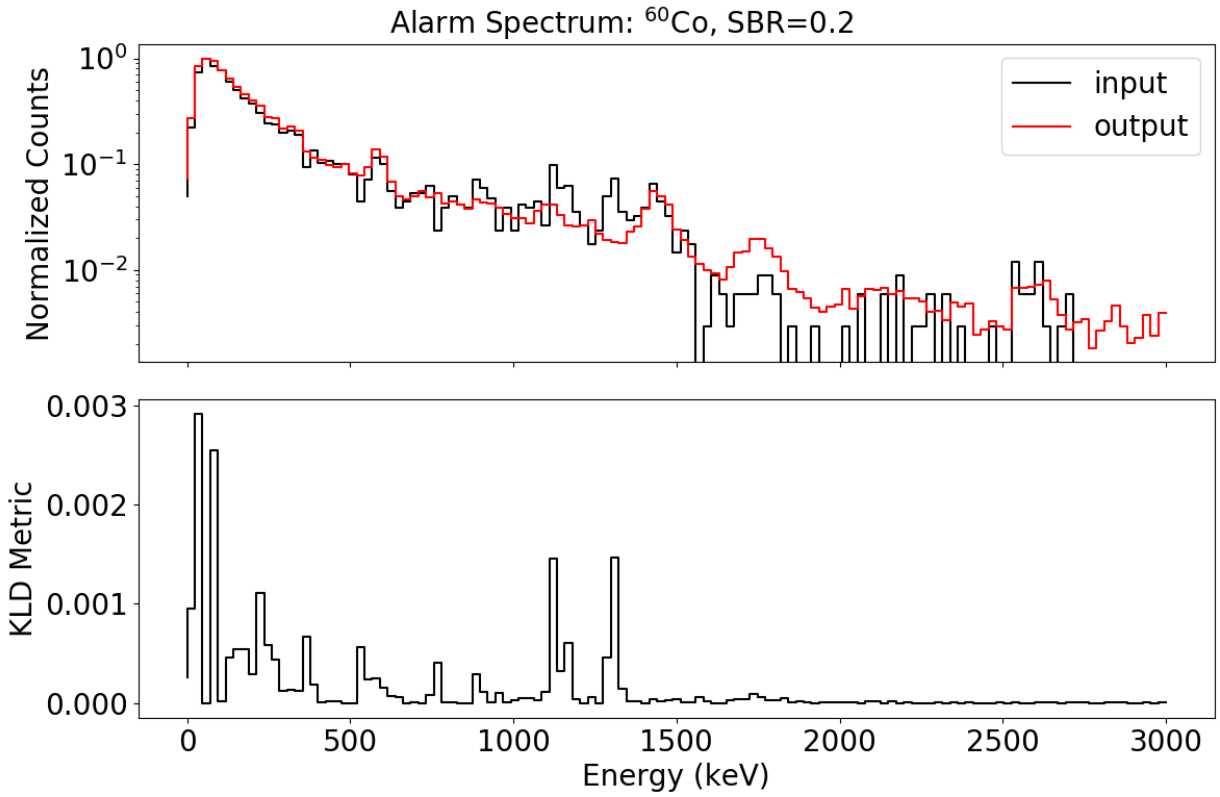


Figure 5.4: Alarm spectrum with reconstruction for ^{60}Co with an SBR of 0.1. Also shown is the KLD metric as a function of energy.

5.2 Performance Evaluation

This section shows the individual ROC curves generated for each of the five sources in the simulated dataset. A single ROC curve was also generated which shows ARAD's performance for all 5 sources. Whereas the ROC curves for each particular source demonstrate ARAD's performance for specific source, the single conglomerate ROC curve is the one most important in an operational setting where the source of interest may not be known. The conglomerate ROC curve is the one used to generate the probability of detection curves that are also shown in this section.

The source to background ratio required to achieve a 100% TPR along with a 0% FPR is one of the measures that will be used to compare ARAD's performance on different isotopes.

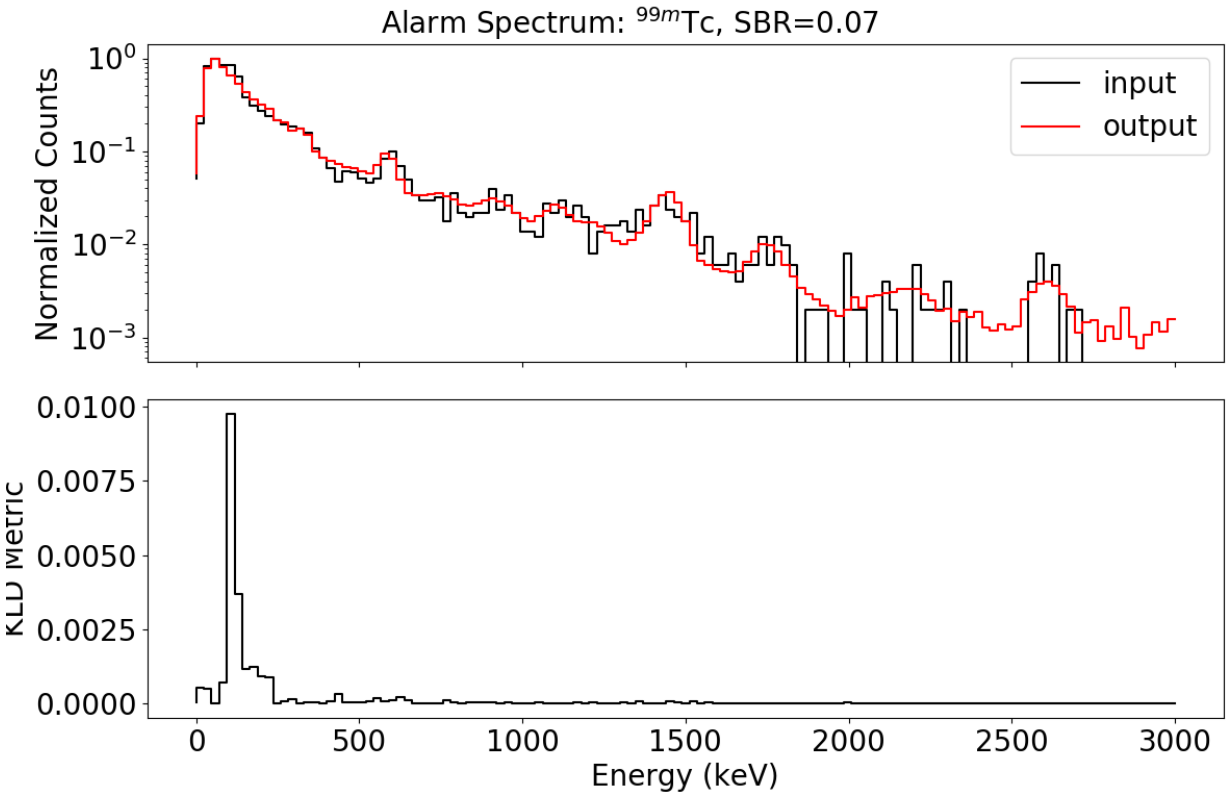


Figure 5.5: Alarm spectrum with reconstruction for ^{99m}Tc with an SBR of 0.07. Also shown is the KLD metric as a function of energy.

This measure will be defined as $\text{SBR}_{\text{min,s}}$ where s is the source ID. This section only presents the results. Chapter 7 contains a discussion on these results.

5.2.1 Comparison to NMF

As discussed in Chapter 1, non-negative matrix factorization (NMF) is another recent unsupervised algorithm that has been developed for performing radiation anomaly detection. An NMF algorithm was developed and trained using the same training data and evaluation procedures as that used for ARAD. A quantity of 5 basis vectors were used, just as 5 units in the latent space were used for ARAD. Spectrum reconstruction error was likewise measured using the Kullback-Leibler divergence. Each of the ARAD ROC and PD curves displayed in this section will also have an accompanying ROC/PD curve for NMF. The SBR values on

each figure may not be the same for all sources, as they are selected to best show the full range of performance for each algorithm/source.

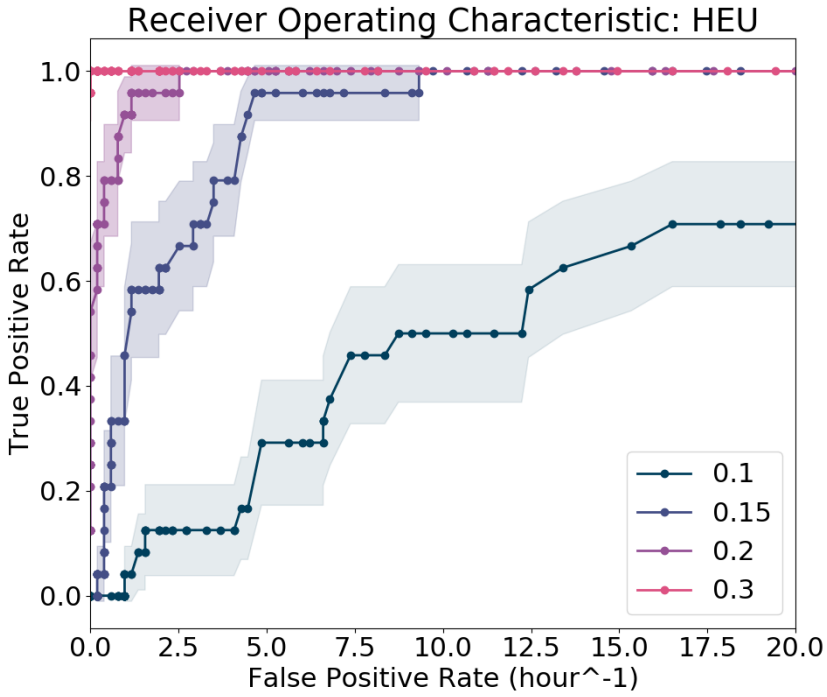
It is important to emphasize that the purpose of this work was not to compete with other algorithms. This being said, the results for the NMF algorithm present the opportunity to compare ARAD to another modern radiation anomaly detection algorithm. Overall, NMF performs very well on the simulated data, showing performance similar to that of ARAD. ARAD performed significantly better than NMF for WGPu, able to achieve a 100% TPR and 0% FPR for WGPu at a SBR of 0.07, whereas this value was 0.25 for NMF. Also, ARAD performed slightly better than NMF for ^{99m}Tc, with a 100% TPR and 0% FPR at an SBR of 0.15 whereas NMF required an SBR of 0.2. ARAD also had a higher TPR than NMF for lower SBR values. Both algorithms achieved a 100% TPR and 0% FPR for ^{60}Co at an SBR of 0.3, however, NMF had a higher TPR than ARAD at lower SBR values. NMF performs slightly better than ARAD on HEU, with a 100% TPR and 0% FPR at an SBR of 0.25 for NMF and 0.3 for ARAD. ARAD performed slighly better than NMF on ^{131}I. When evaluated on all 5 sources, NMF achieves a 100% TPR and 0% FPR at an SBR of 0.3, whereas ARAD achieves this at an SBR of 0.4. Based on the PD curve generated for all 5 sources, ARAD appears to outperform NMF for weaker sources (between 0.05 and 0.15 SBR) and NMF slightly outperforms ARAD for the stronger SBR sources (0.15 to 0.4).

5.2.2 Highly Enriched Uranium (HEU)

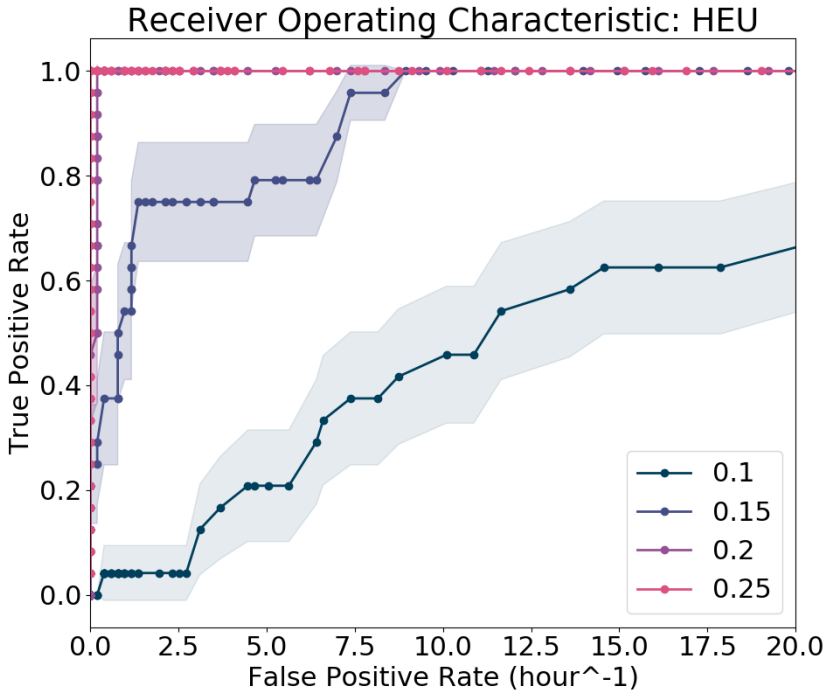
Of the five sources in the simulation dataset, HEU was the one that has the highest $\text{SBR}_{\text{min,s}}$ (tied with ^{60}Co). As can be seen on the ROC curve displayed in Figure 5.6, $\text{SBR}_{\text{min,HEU}} \approx 0.3$, that is, with the HEU countrate being 30% of the background count rate, ARAD can detect HEU with a 100% detection rate and 0% false alarm rate.

5.2.3 Weapons Grade Plutonium (WGPu)

WGPu was the source that ARAD was most easily able to detect. Based on the ROC curve shown in Figure 5.7, $\text{SBR}_{\text{min,WGPu}} \approx 0.07$. That is, if the count rate in the detector from WGPu is at least 7% of the background count rate, ARAD can detect WGPu with a 100%



(a) ARAD



(b) NMF

Figure 5.6: ARAD (top) and NMF (bottom) ROC curves for HEU at 4 different source to background strength ratios. Error bars indicate 90% confidence interval assuming a Binomial distribution.

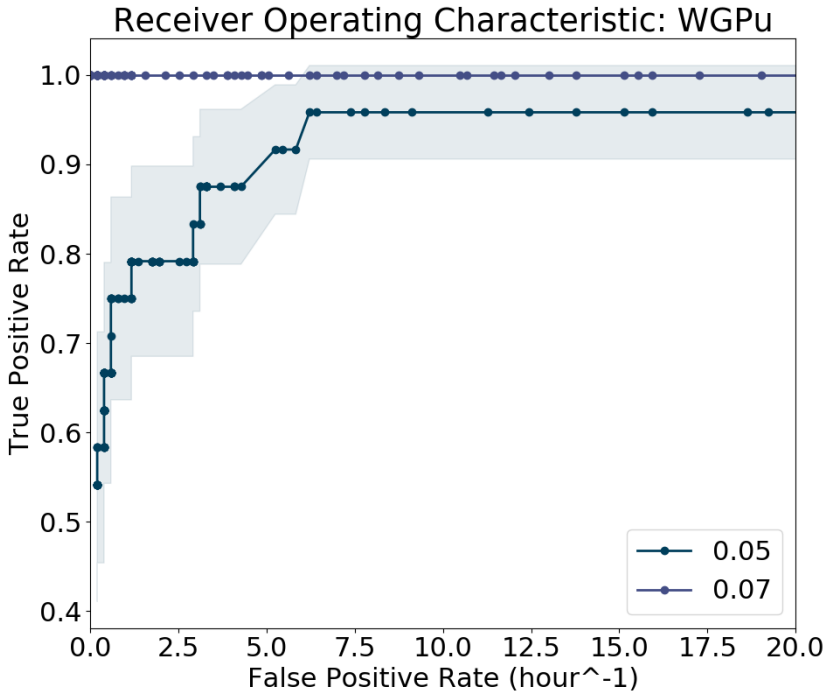
probability of detection and 0% false alarm rate. Likewise, ARAD is able to detect WGPu with a count rate 5% of background with an 80% detection probability and false alarm rate of about 1 per hour.

5.2.4 ^{131}I, ^{60}Co, and ^{99m}Tc

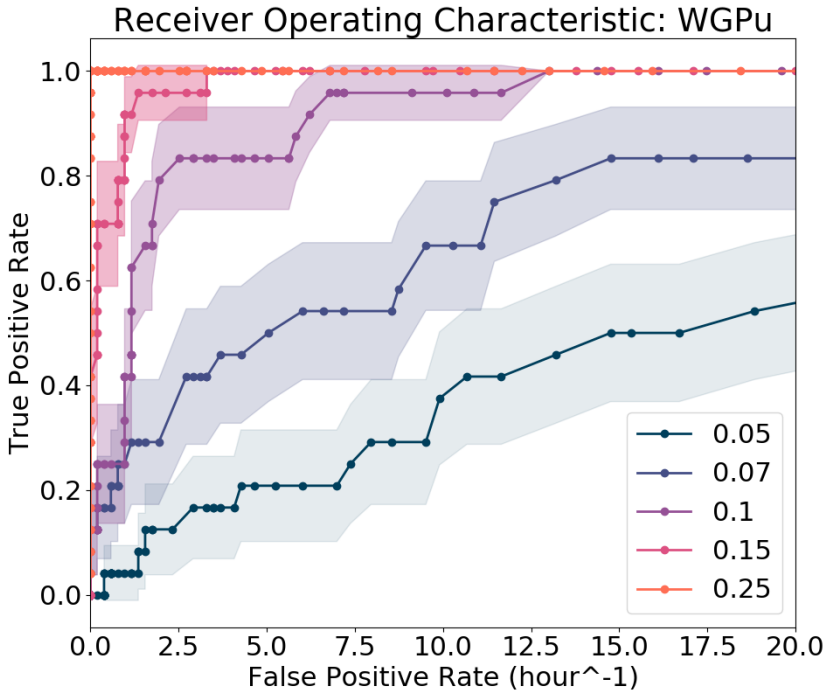
Based on the ROC curve in Figure 5.8, $\text{SBR}_{\text{min,I-131}} \approx 0.2$. ^{60}Co, whose ROC curve is shown in 5.9 has the same $\text{SBR}_{\text{min,s}}$ as that of HEU, however, it actually performs worse than HEU at lower SBR values. On the other hand, ^{99m}Tc, with $\text{SBR}_{\text{min,Tc-99m}} \approx 0.15$ comes in second place with regards to detectability.

5.2.5 All 5 Sources

Figure 5.11 shows the ROC curve generated when running ARAD on all 5 of the sources. In an operational setting, the source of interest may not be know, thus this conglomerate ROC curve can be used to establish a reasonable detection threshold when the source characteristics are unknown. $\text{SBR}_{\text{min,ALL}} \approx 0.4$. In reality however, a certain acceptable false alarm rate can usually be established. Using these false alarm rates, once can plot the probability of detection (PD) as a function of SBR. The PD curves for false alarm rates of 1 per hour and 1 per 8 hours for all 5 sources are presented in Figure 5.12.

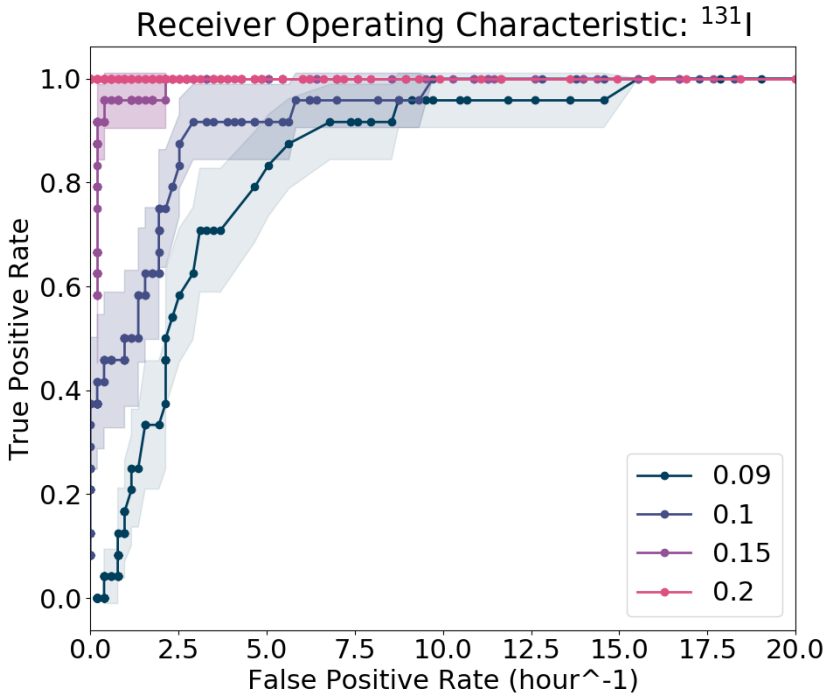


(a) ARAD

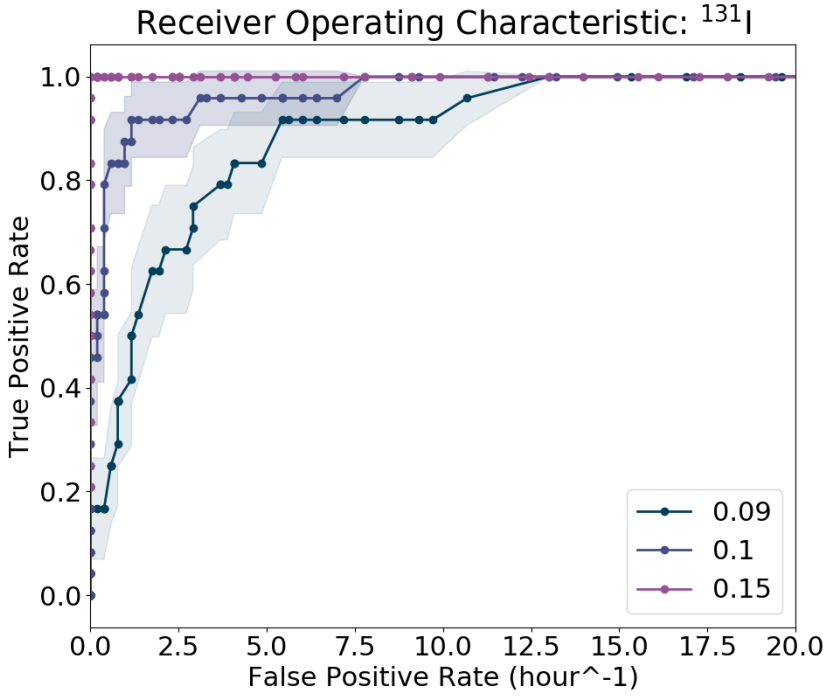


(b) NMF

Figure 5.7: ARAD (top) and NMF (bottom) curves for WGPu at 2/5 different source to background strength ratios. Error bars indicate 90% confidence interval assuming a Binomial distribution.

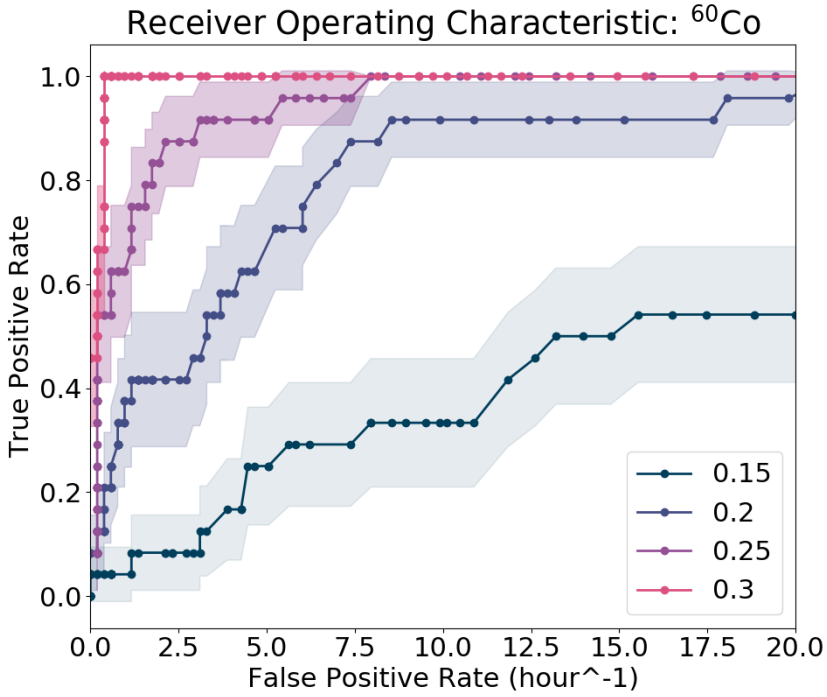


(a) ARAD

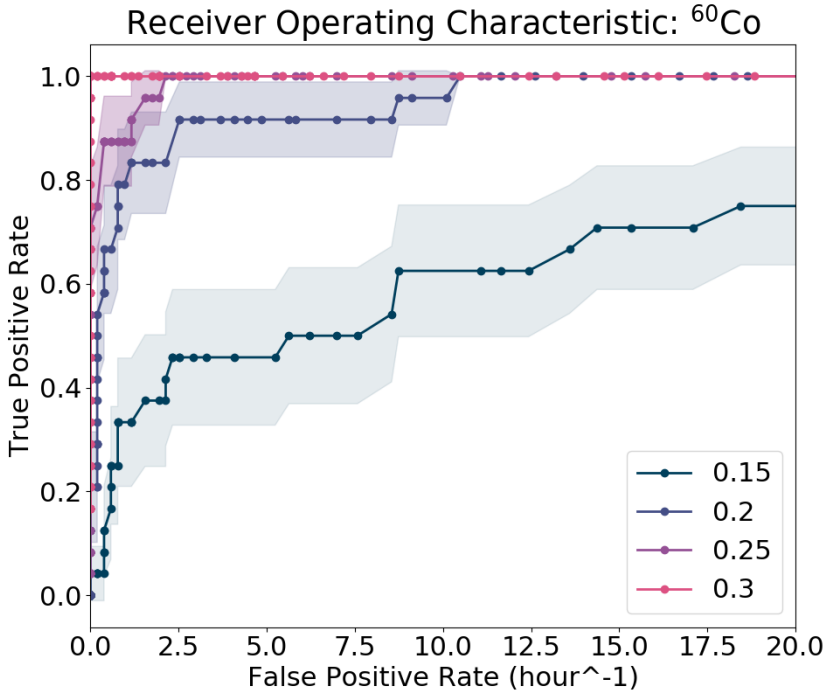


(b) NMF

Figure 5.8: ARAD (top) and NMF (bottom) curves for ^{131}I at 4/3 different source to background strength ratios. Error bars indicate 90% confidence interval assuming a Binomial distribution.

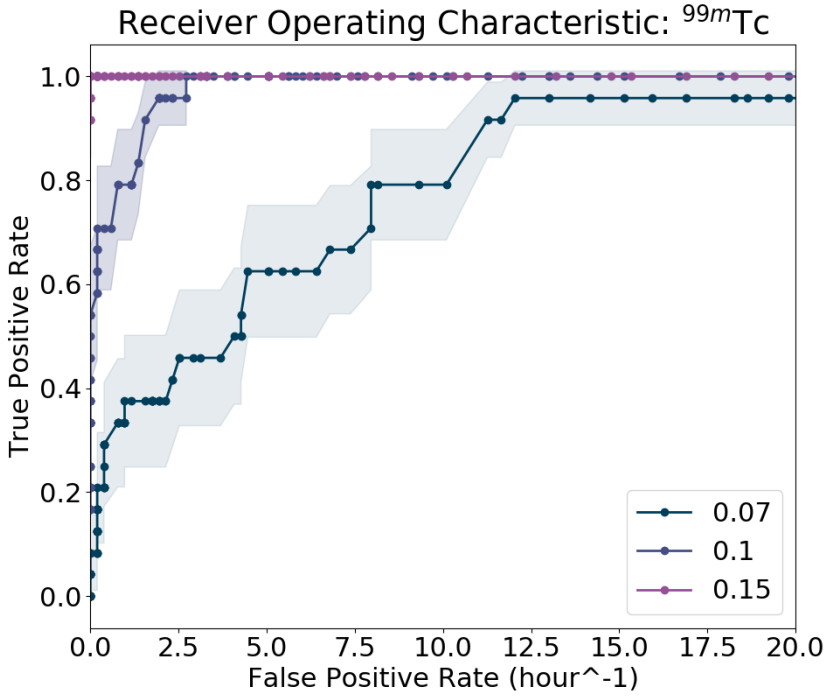


(a) ARAD

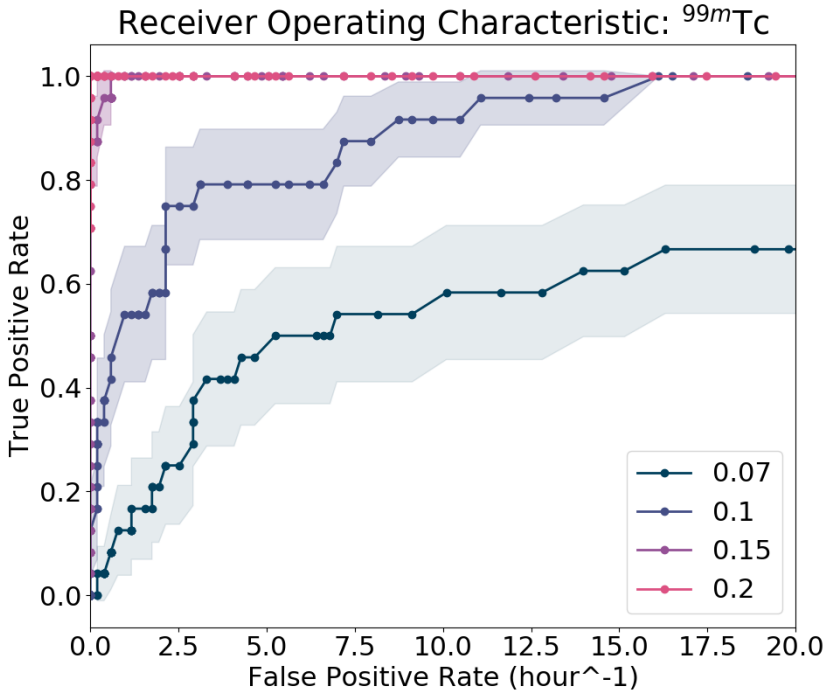


(b) NMF

Figure 5.9: ARAD (top) and NMF (bottom) curves for ^{60}Co at 4 different source to background strength ratios. Error bars indicate 90% confidence interval assuming a Binomial distribution.

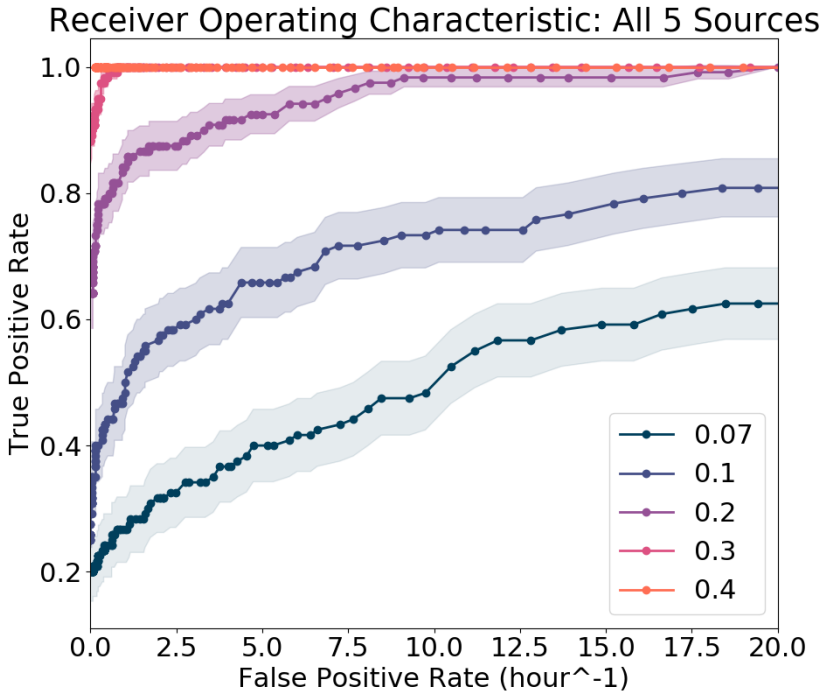


(a) ARAD

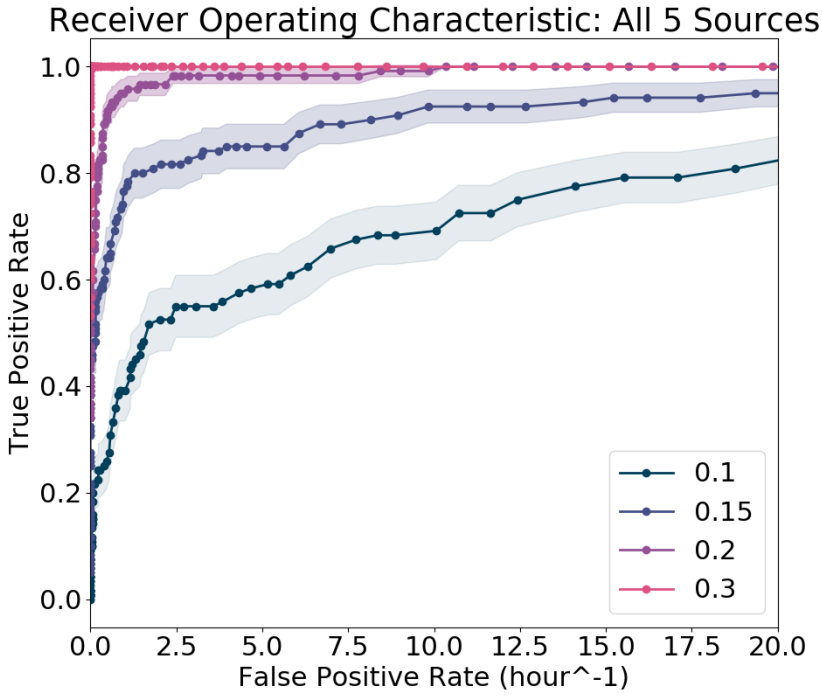


(b) NMF

Figure 5.10: ARAD (top) and NMF (bottom) curves for ^{99m}Tc at 3 different source to background strength ratios. Error bars indicate 90% confidence interval assuming a Binomial distribution.

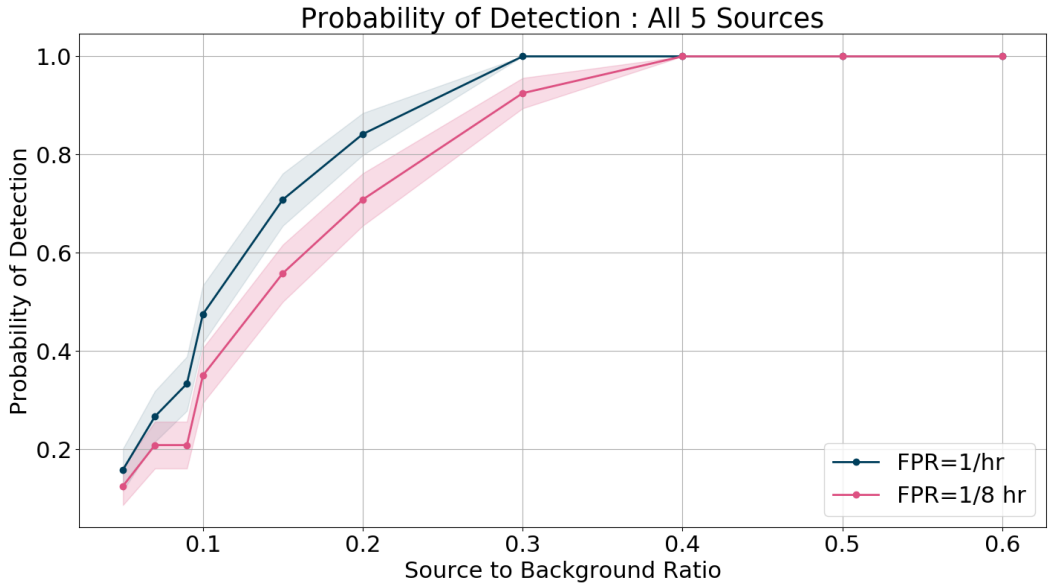


(a) ARAD

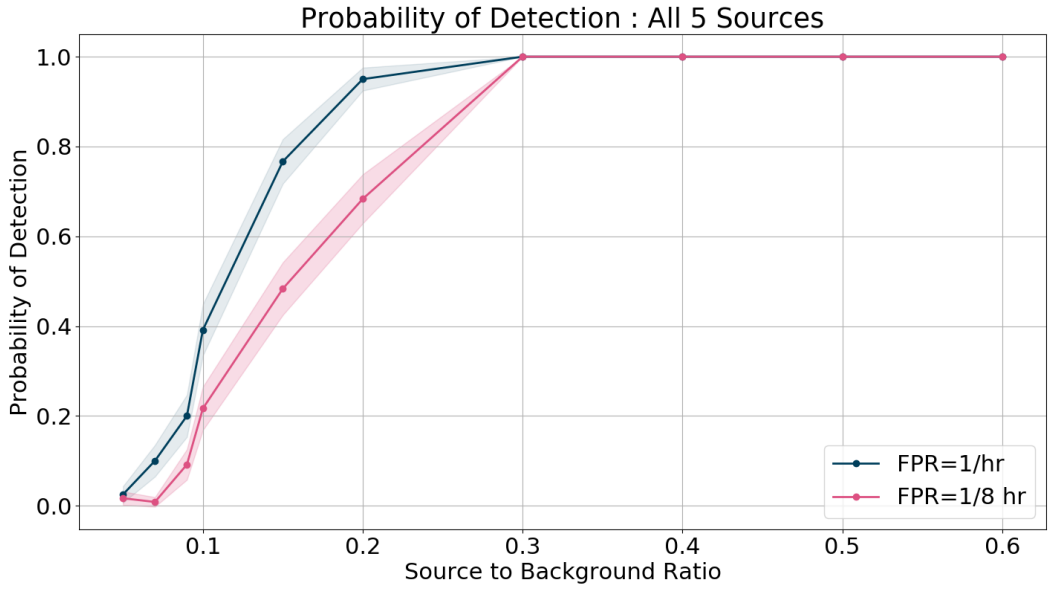


(b) NMF

Figure 5.11: ARAD (top) and NMF (bottom) curves for all 5 sources at 5/4 different source to background strength ratios. Error bars indicate 90% confidence interval assuming a Binomial distribution.



(a) ARAD



(b) NMF

Figure 5.12: ARAD (top) and NMF (bottom) probability of detection curve for all five sources for a false positive rate of 1 per hour and 1 per 8 hours. Error bars indicate 90% confidence interval assuming a Binomial distribution.

Chapter 6

ARAD Performance in a Real World Environment

6.1 Background Dynamics

Both Nodes 01 and 11 are affected by precipitation-induced background dynamics from wet-deposition of ^{222}Rn. Being located right next to the REDC loading dock and a waste cask storage location, Node 11 frequently experiences clutter-induced background dynamics. This section will show ARAD's resilience to these types of events. Also shown is the response of the k-sigma, SPRT, and NMF algorithms for these events.

6.1.1 Node 11 Clutter

Figures 6.1 and 6.2 show the gross count rate and algorithm response for two particular clutter events that occurred at Node 11. Both events are the result of a vehicle moving between the detector and a set of radioactive waste casks located near the detector. For both cases, false alarms are raised by both the k-sigma and SPRT algorithms. Neither NMF nor ARAD produce an alarm, however. The clutter events are verified using video footage as discussed in Chapter 3.

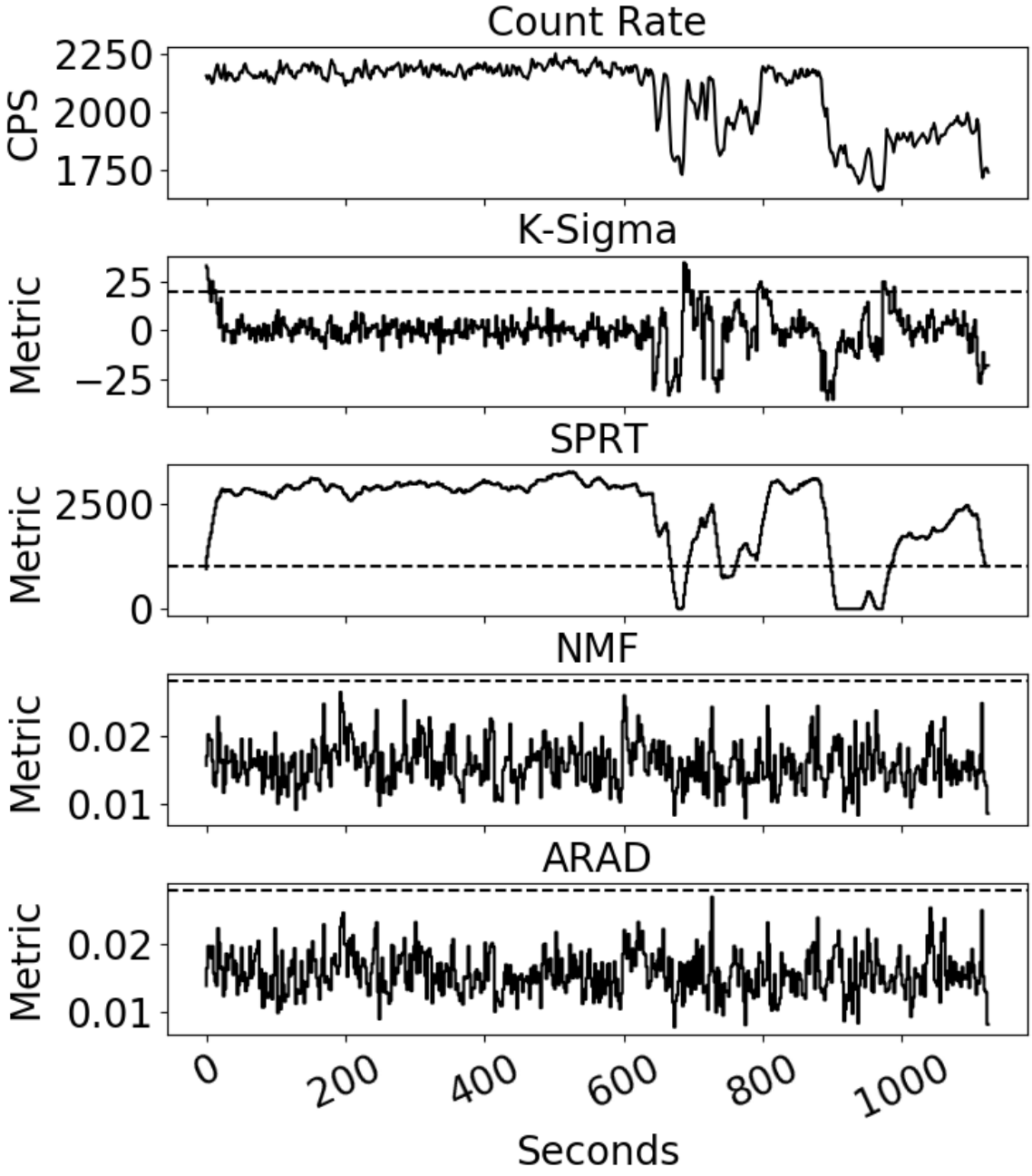


Figure 6.1: Gross count rate and algorithm response for a clutter-induced background variation at Node 11. Horizontal lines indicate detection thresholds for each algorithm.

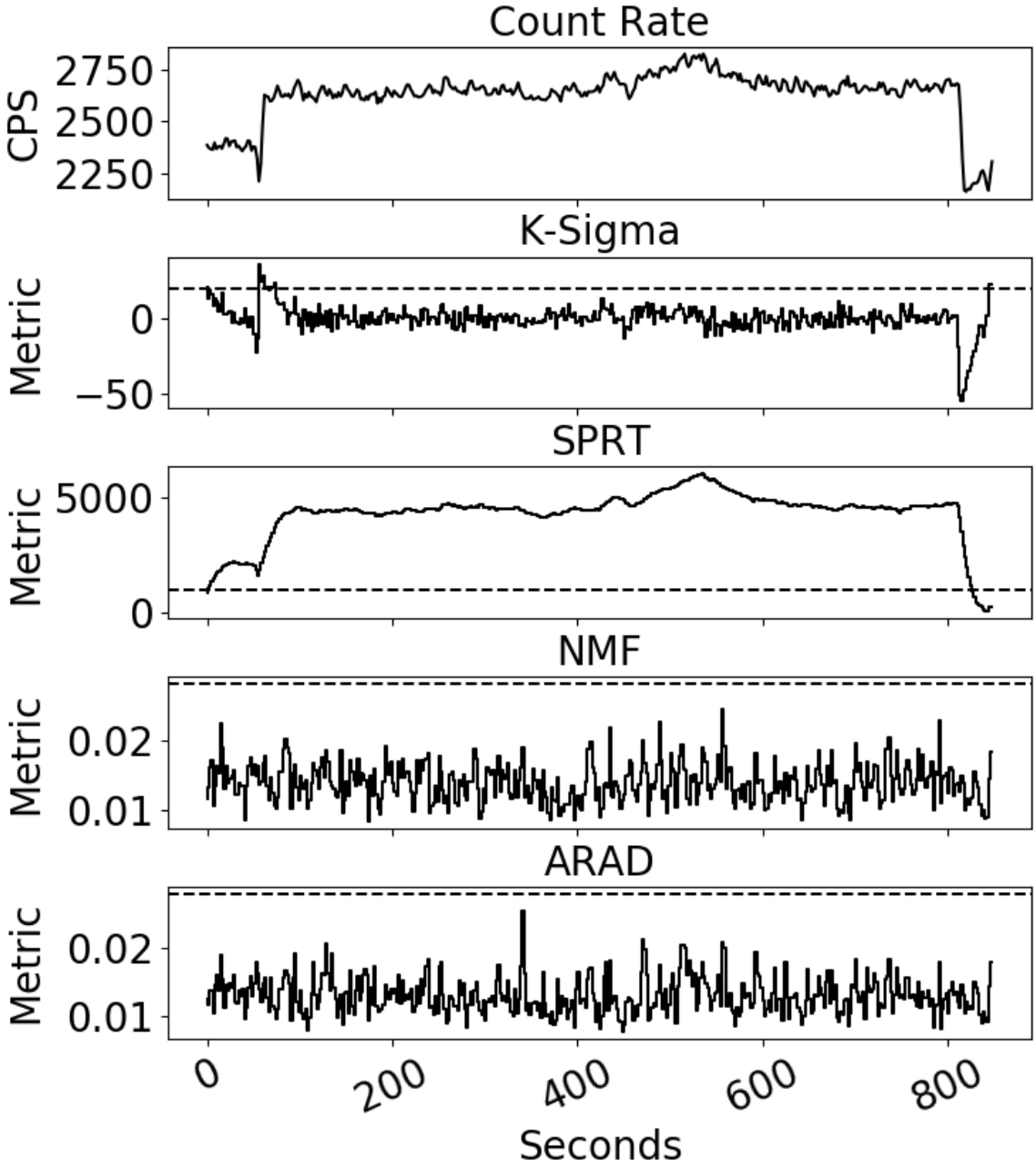


Figure 6.2: Gross count rate and algorithm response for a clutter-induced background variation at Node 11. Horizontal lines indicate detection thresholds for each algorithm.

6.1.2 Node 01 and 11 ^{222}Rn Wet-Deposition

Figure 6.3 shows the gross count rate and algorithm response for a rain-induced background fluctuation at Node 01. As stated in Chapter 3, the rain events are verified with weather data and video footage. As can be seen in the figure, the SPRT algorithm generated an alarm for the precipitation event. The increase in gross count rate occurred over a long enough period of time that it did not cause an alarm in the rolling k-sigma algorithm. ARAD did not alarm for the event as it was able to construct the ^{214}Bi and^{214}Pb peaks in the spectrum as shown in Figure 6.4. Figure 6.5 shows a rain event that occurred at Node 11, leading to a false alarm in SPRT and no alarm in k-sigma, NMF, and ARAD.

6.2 Source Detection Events

As a nuclear research reactor and radiochemistry processing center, the HFIR/REDC complex has many source events on which to test radiation detection algorithms. Many of these sources are transfer events where radioactive material is moved between both facilities. Other events such as the ^{41}Ar effluents occur as a result of operations occurring at these facilities. This section will show alarm spectra for a selection of interesting source events.

6.2.1 ^{41}Ar Effluent

The short lived gaseous radioisotope ^{41}Ar is sometimes released as effluent from HFIR. It is generated through neutron absorption in ^{40}Ar, which is naturally present in the atmosphere in small quantities. It has a very short half life of only 1.8 hours and decays into stable ^{41}K via beta minus decay. After this beta decay, a gamma ray of energy 1293.6 keV is emitted with 99.16% probability [6]. Because it is diluted in the air by the time it reaches the detectors, the signal is usually quite weak. Figure 6.6 shows the gross count rate profile and KLD metric for one of these events. The 1293.6 keV gamma ray is clearly visible in the alarm spectrum for this event, which is presented in Figure 6.7.

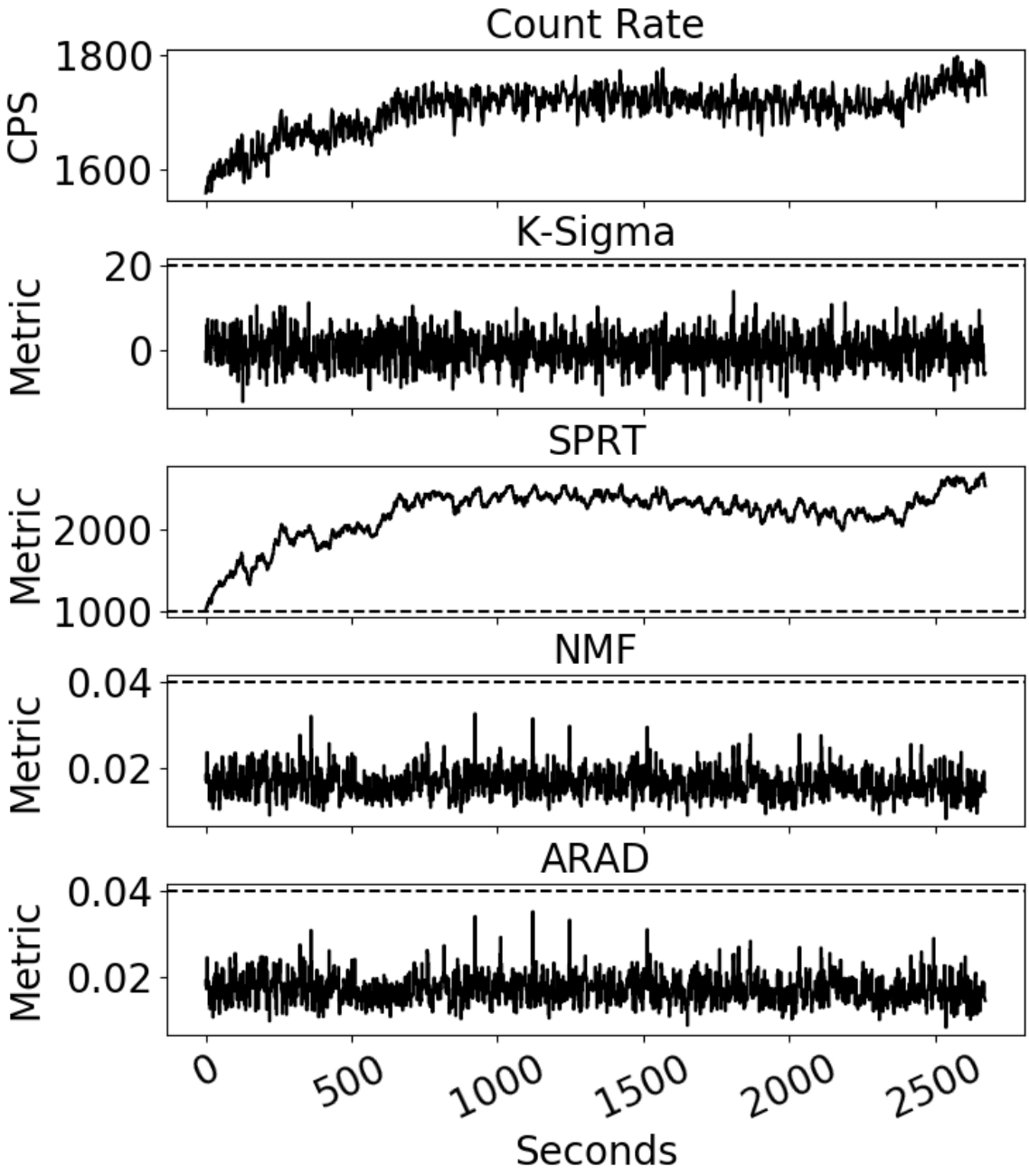


Figure 6.3: Gross count rate and algorithm response for a rain-induced background variation from wet-deposition of ^{222}Rn daughters at Node 01. Horizontal lines indicate detection thresholds for each algorithm.

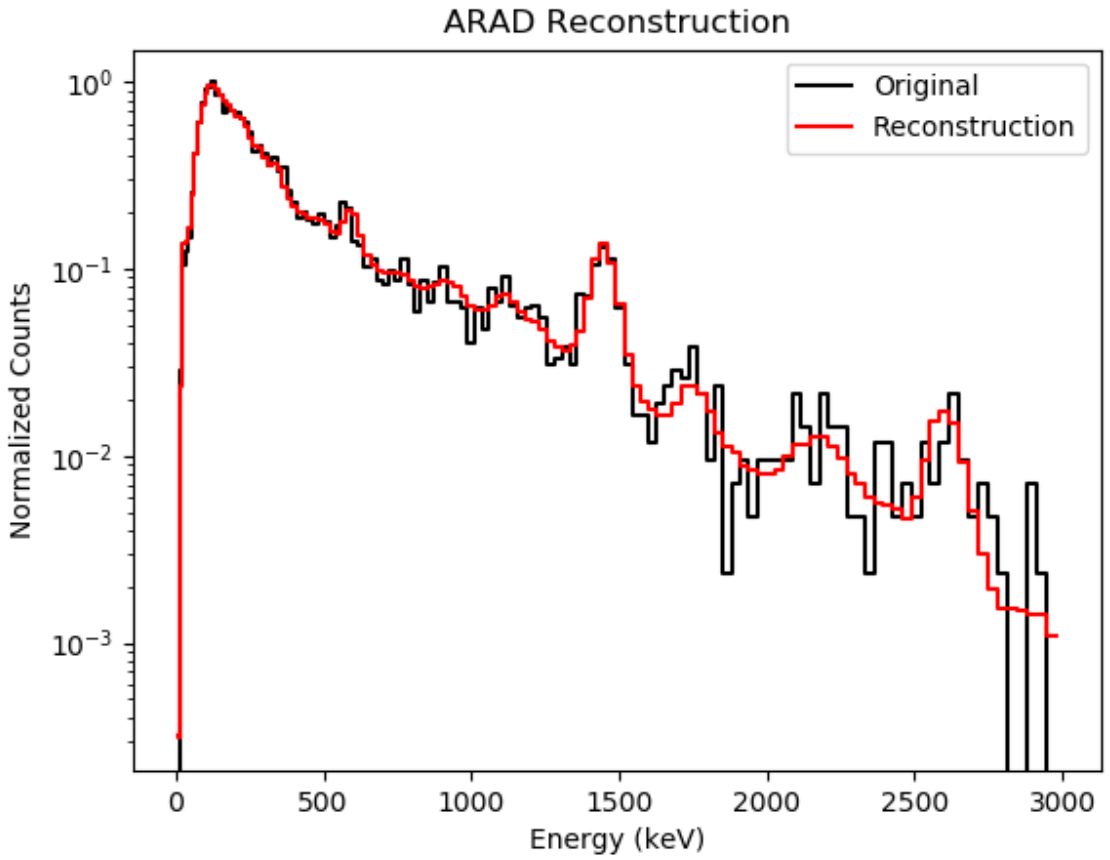


Figure 6.4: Reconstruction of a gamma ray spectrum by ARAD for the precipitation event shown in Figure 6.5. Node 01.

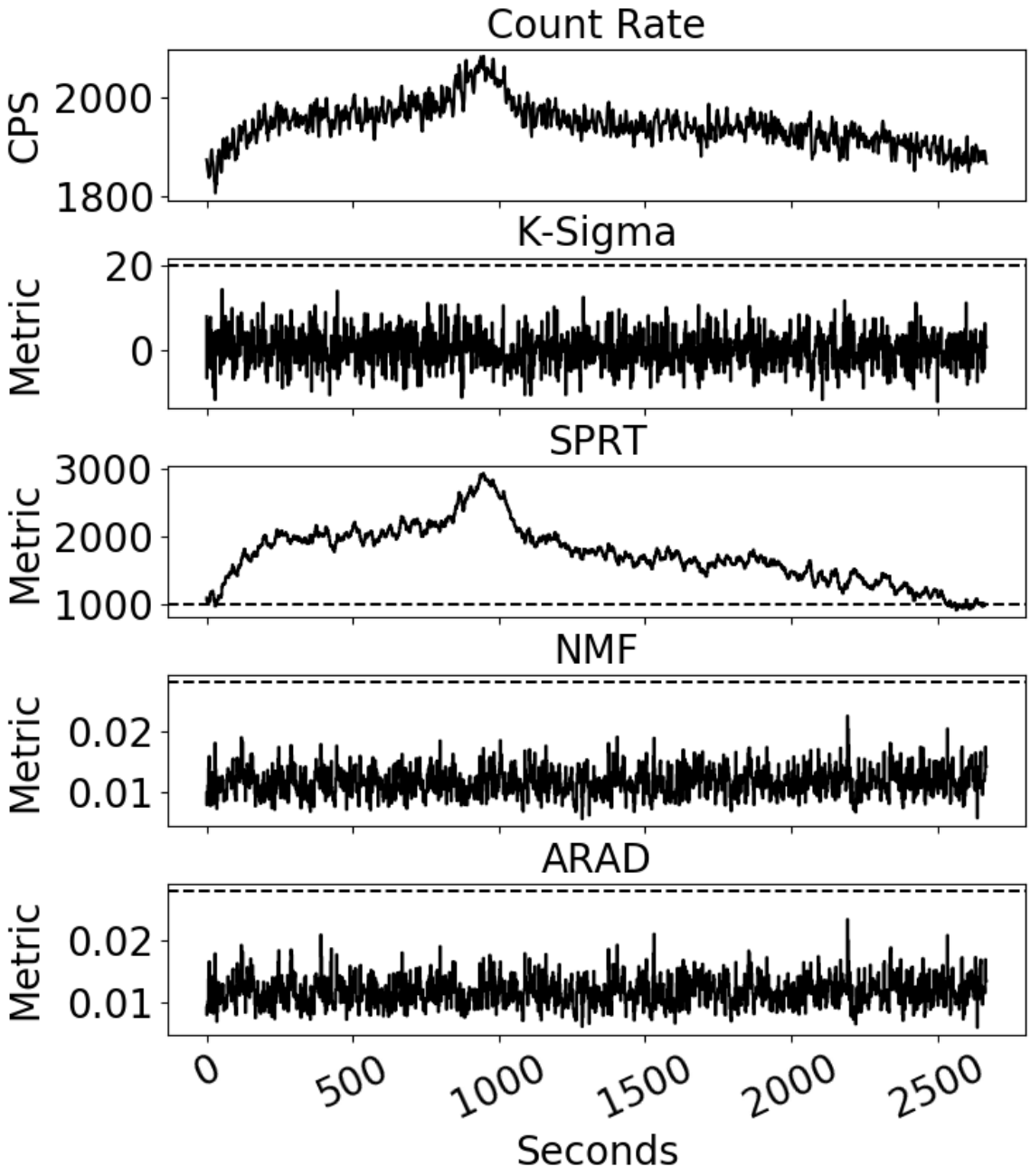


Figure 6.5: Gross count rate and algorithm response for a rain-induced background variation from wet-deposition of ^{222}Rn daughters at Node 11. Horizontal lines indicate detection thresholds for each algorithm.

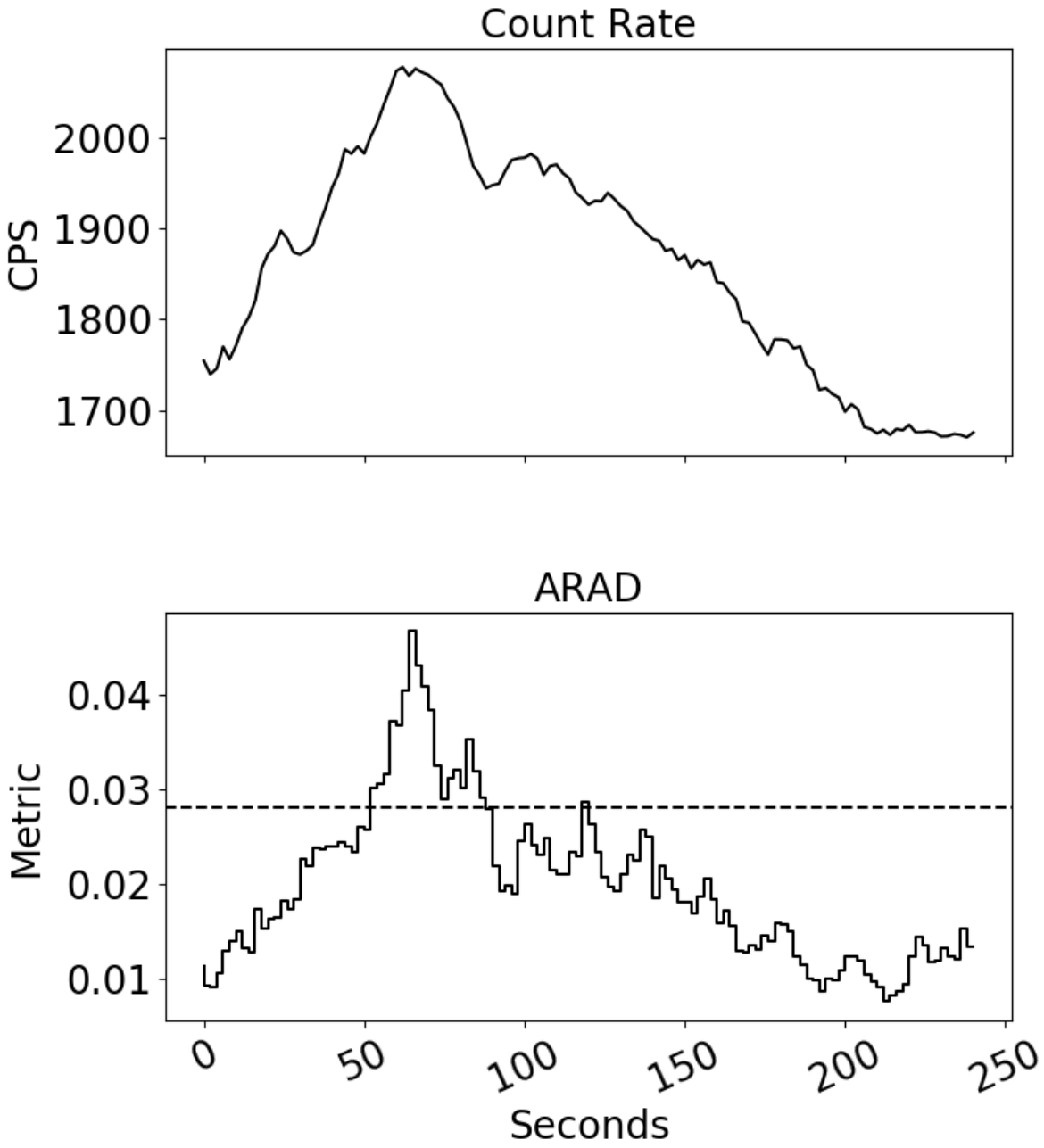


Figure 6.6: Gross count rate and ARAD KLD metric for an ^{41}Ar effluent event. Node 11. Horizontal dotted line indicates detection threshold.

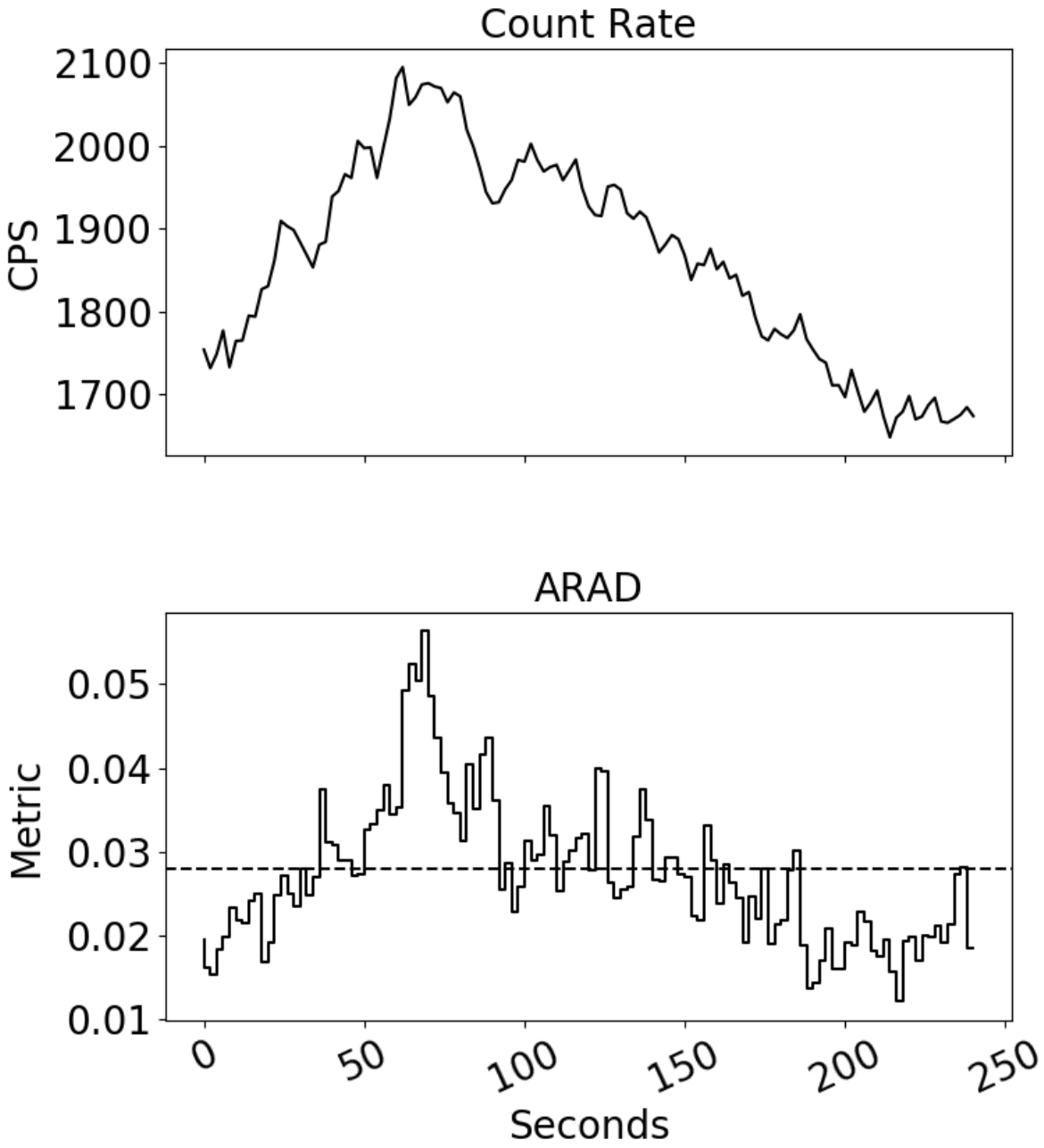


Figure 6.7: Alarm spectrum with ARAD reconstruction for the ^{41}Ar effluent event in Figure 6.6. Node 11.

6.2.2 ^{247}Np Cermet Targets

Part of ^{238}Pu production process that takes place at HFIR/REDC is the neutron bombardment of ^{237}Np targets. These targets come in many different material forms, one of which is a ceramic-metal (cermet). The gross count rate and ARAD KLD metric for a particular cermet ^{237}Np target transfer event is shown in Figure 6.8. There are many gamma rays emitted by the radioisotopes in the ^{237}Np decay chain, however some notable ones are the 311.9 keV gamma from ^{233}Pa and the 440.45 keV gamma from ^{213}Bi [6]. The 311.9 keV gamma can be clearly seen in the alarm spectrum shown in Figure 6.9.

6.2.3 ^{225}Ac Shipment

^{225}Ac is a medical radioisotope used for targeted alpha therapy. Until recently, ORNL was the only place in the world in which ^{225}Ac was produced [85]. It has a very short half life of only 10 days. In alpha decays to ^{221}Fr, after which it follows a decay chain to eventually lead to stable ^{205}Tl. Notable gamma rays emitted from ^{225}Ac and its daughters are the 218 keV gamma from ^{221}Fr and the 440.45 keV gamma from ^{213}Bi [6]. Figure 6.10 shows the gross count rate and ARAD KLD metric for an ^{225}Ac shipment. The 218 and 440.45 keV gamma rays can both be seen in the alarm spectrum presented in Figure 6.11.

6.2.4 Waste Transfer

Various forms of radioactive waste are transferred around the HFIR/REDC facility. Figure 6.12 shows the gross count rate and ARAD KLD metric for a particular waste transfer event. The waste is shielded, thus the majority of the signal comes from high energy photons that were downscattered prior to reaching the detector. Because of the high energy of the gammas, a 511 keV peak from electron/positron annihilation is often present in the spectrum. Also there is often a peak around 2.2 MeV arising from the $^{1}H(n,\gamma)^{2}H$ reaction between neutrons emitted from the waste and hydrogenous materials surrounding the waste and detector. Both the 511 keV and 2.2 MeV gamma peaks can be seen in the alarm spectra presented in Figure 6.13.

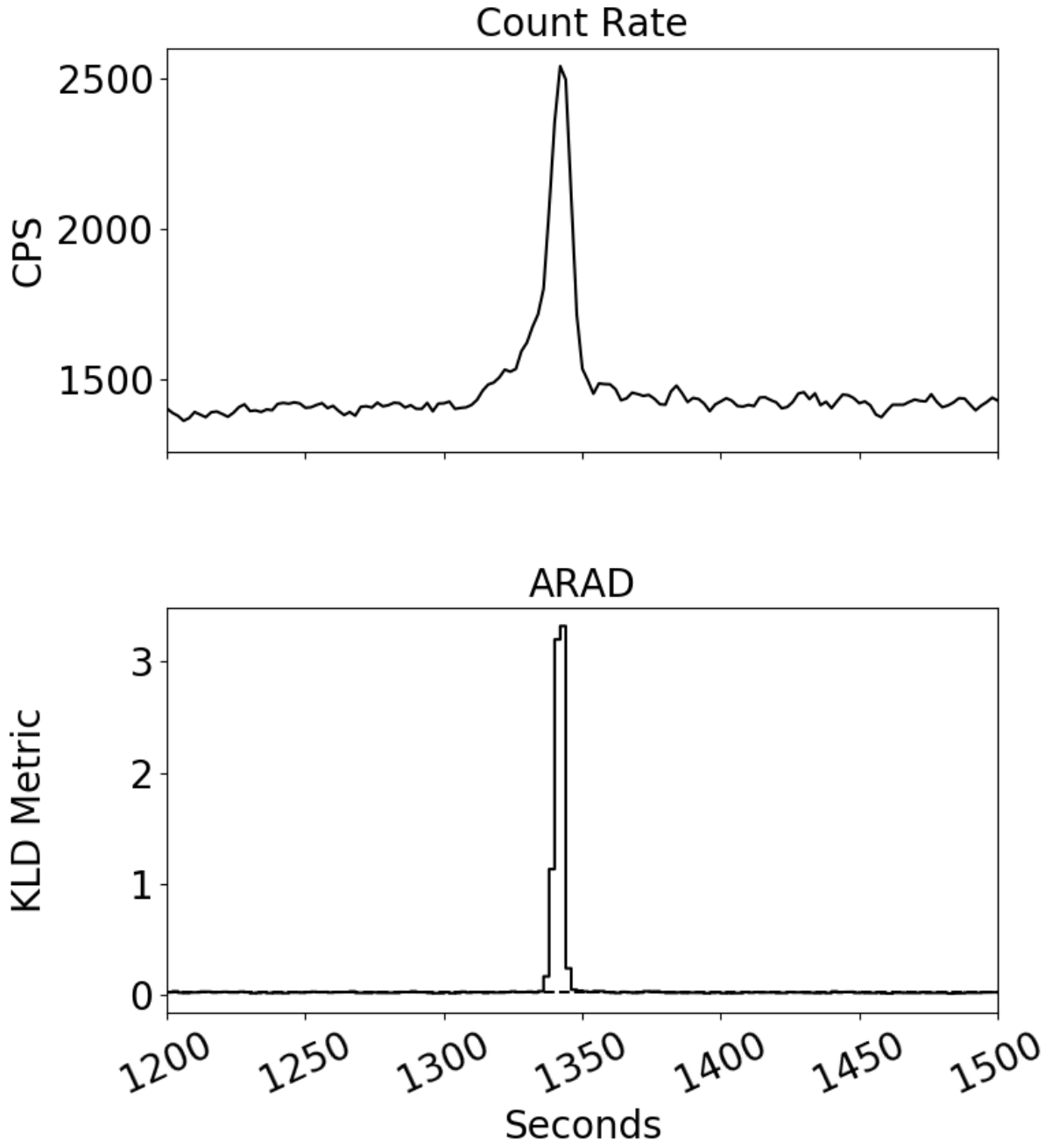


Figure 6.8: Gross count rate and ARAD KLD metric for a cermet ^{237}Np transfer event. Node 11. Horizontal dotted line indicates detection threshold.

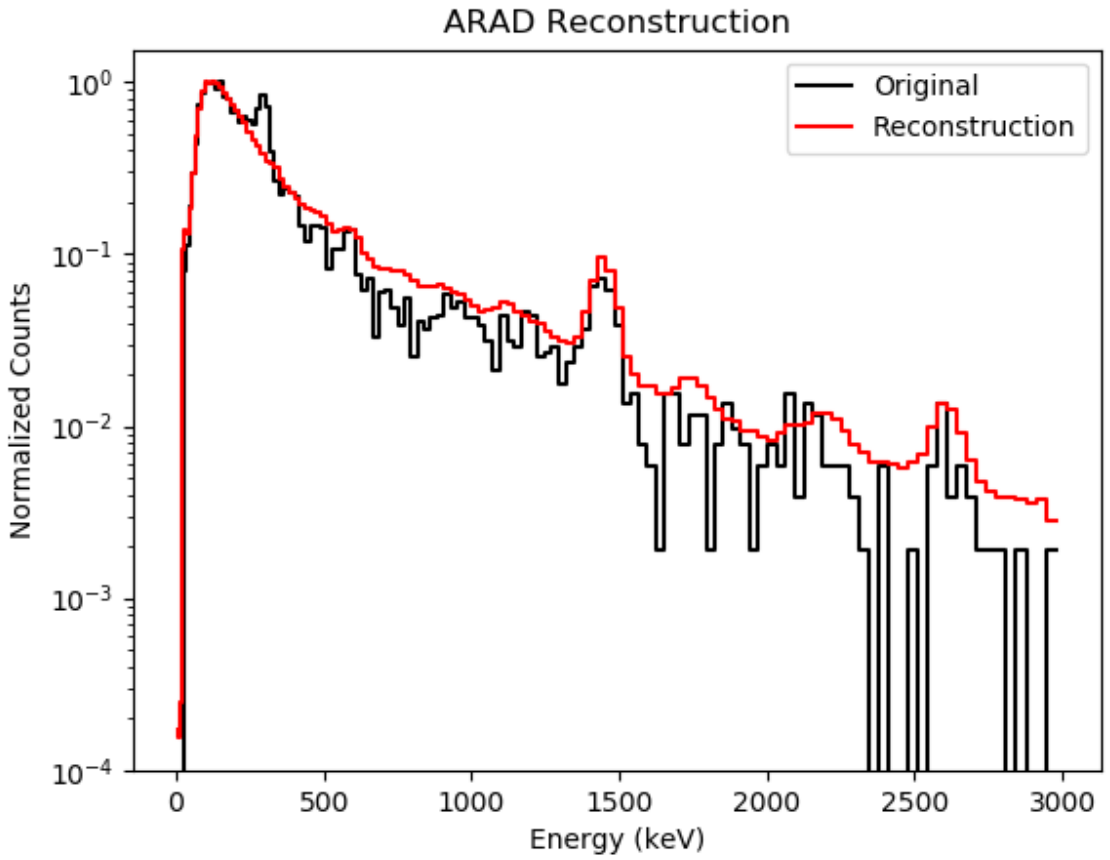


Figure 6.9: Alarm spectrum with ARAD reconstruction for the cermet ^{237}Np transfer event in Figure 6.8. Node 11.

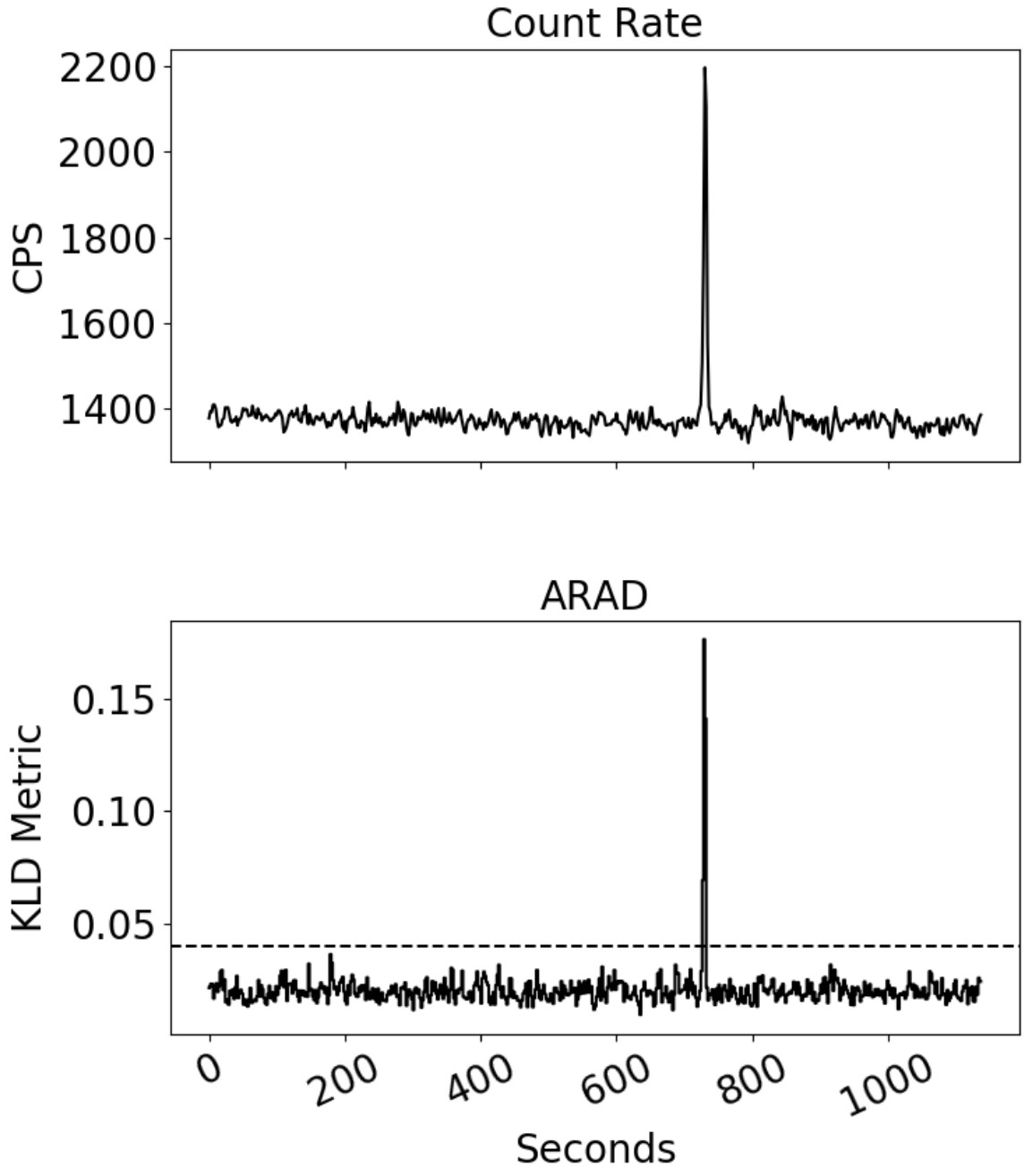


Figure 6.10: Gross count rate and ARAD KLD metric for an ^{225}Ac shipment event. Node 01. Horizontal dotted line indicates detection threshold.

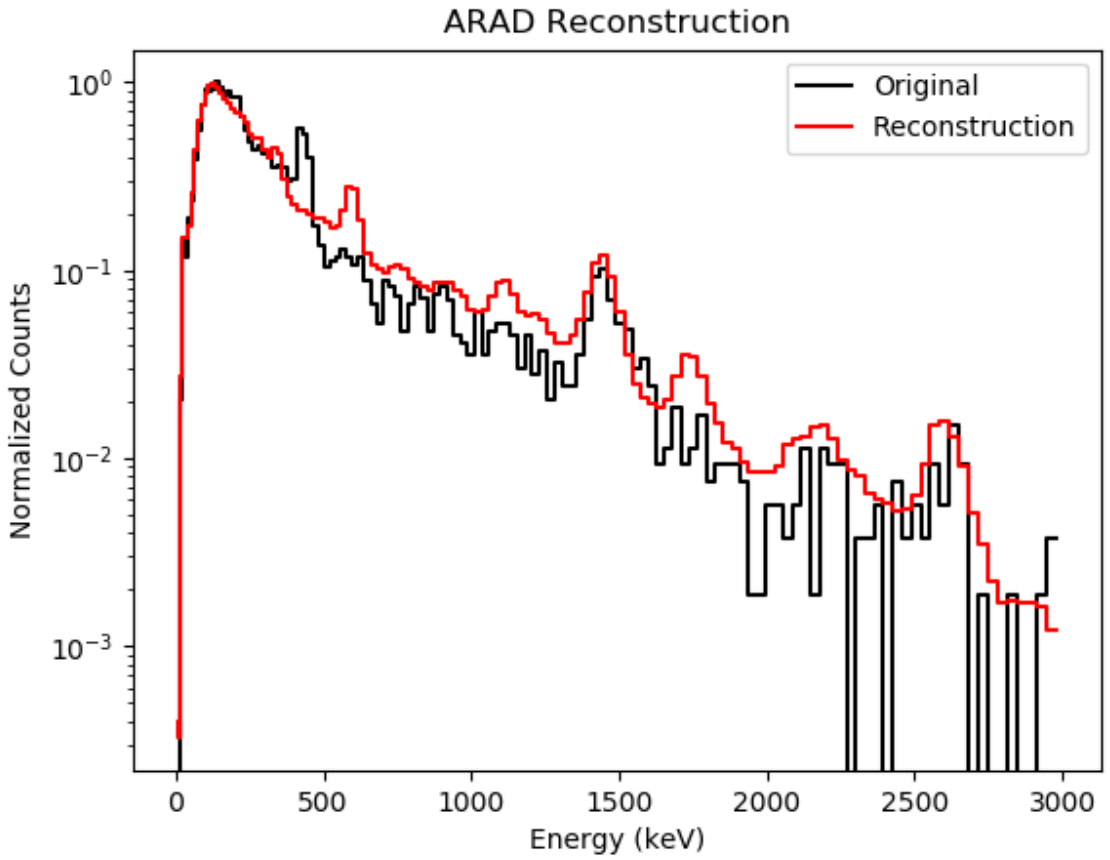


Figure 6.11: Alarm spectrum with ARAD reconstruction for the ^{225}Ac shipment event in Figure 6.10. Node 01.

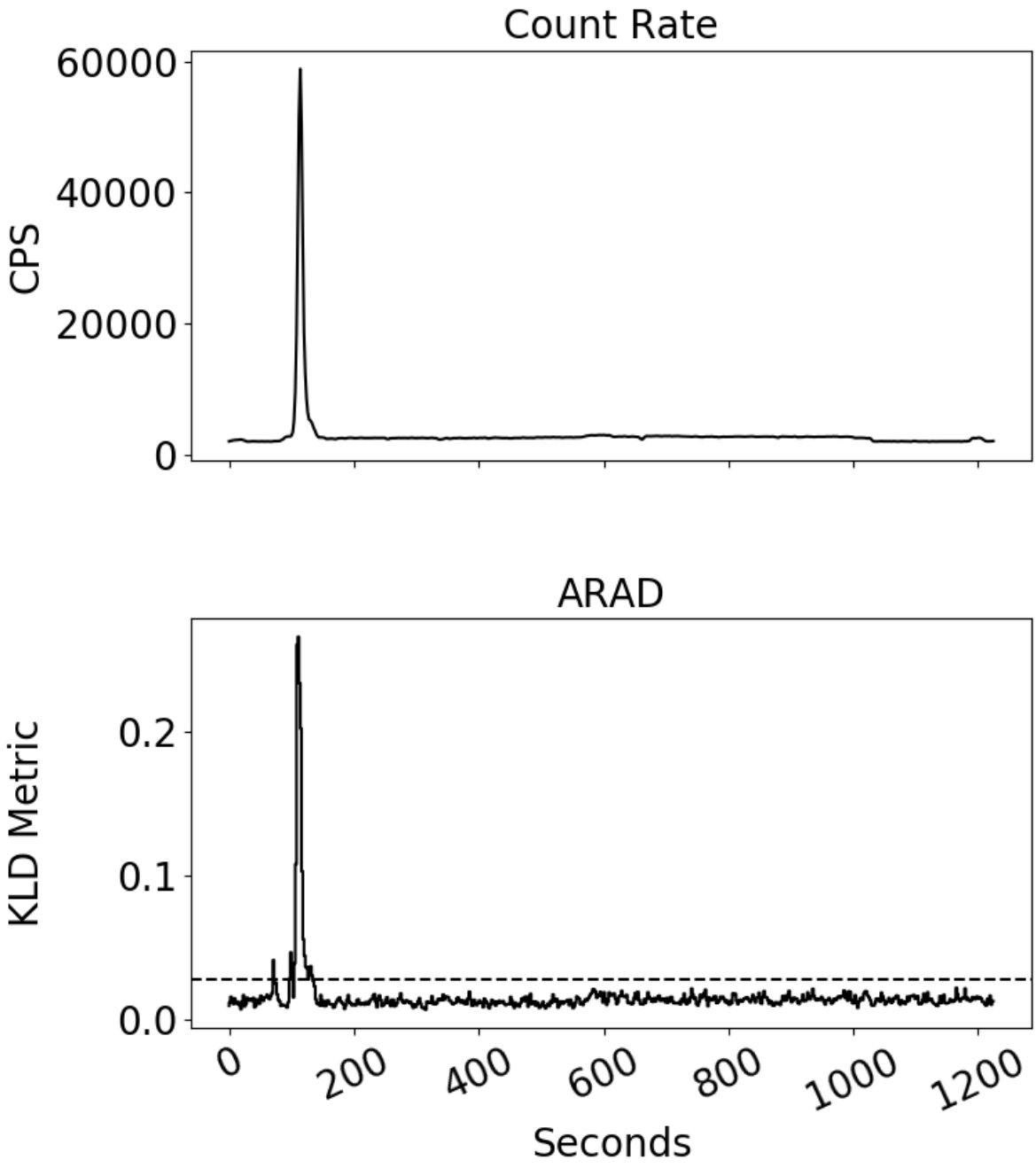


Figure 6.12: Gross count rate and ARAD KLD metric for a radioactive waste transfer event. Node 11. Horizontal dotted line indicates detection threshold.

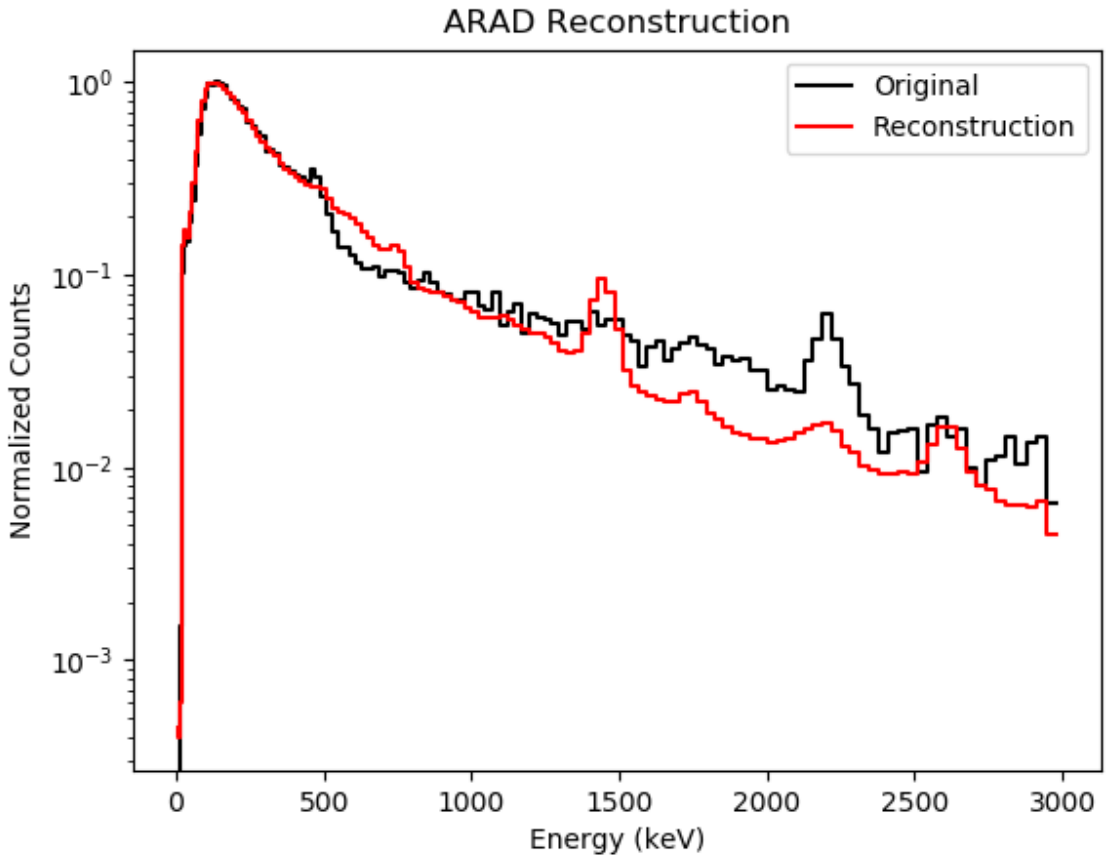


Figure 6.13: Alarm spectrum with ARAD reconstruction for the radioactive waste transfer event in Figure 6.12. Node 11.

6.2.5 ^{137}Cs Transfer

^{137}Cs is a prevalent fission product that decays via beta minus decay to ^{137m}Ba, which then releases a prominent gamma ray with an energy of 661.7 keV [6]. Figure 6.14 show the gross count rate and ARAD KLD metric for a ^{137}Cs transfer event near Node 11. The 661.7 keV peak can be seen in the alarm spectrum in Figure 6.15.

6.2.6 ^{60}Co Transfer

Like ^{136}Cs, ^{60}Co is a prominent fission product in nuclear reactors. It can also be created by neutron bombardment of ^{59}Co targets. Figure 6.16 shows the gross count rate and ARAD KLD metric for a particular ^{60}Co material transfer event. This particular event only lasted a total of 10 seconds, emphasizing the importance of using small integration times for generating the spectra. The two prominent gamma rays from ^{60}Co have energies of 1173 and 1332 keV, both of which can be seen in the alarm spectrum shown in Figure 6.17.

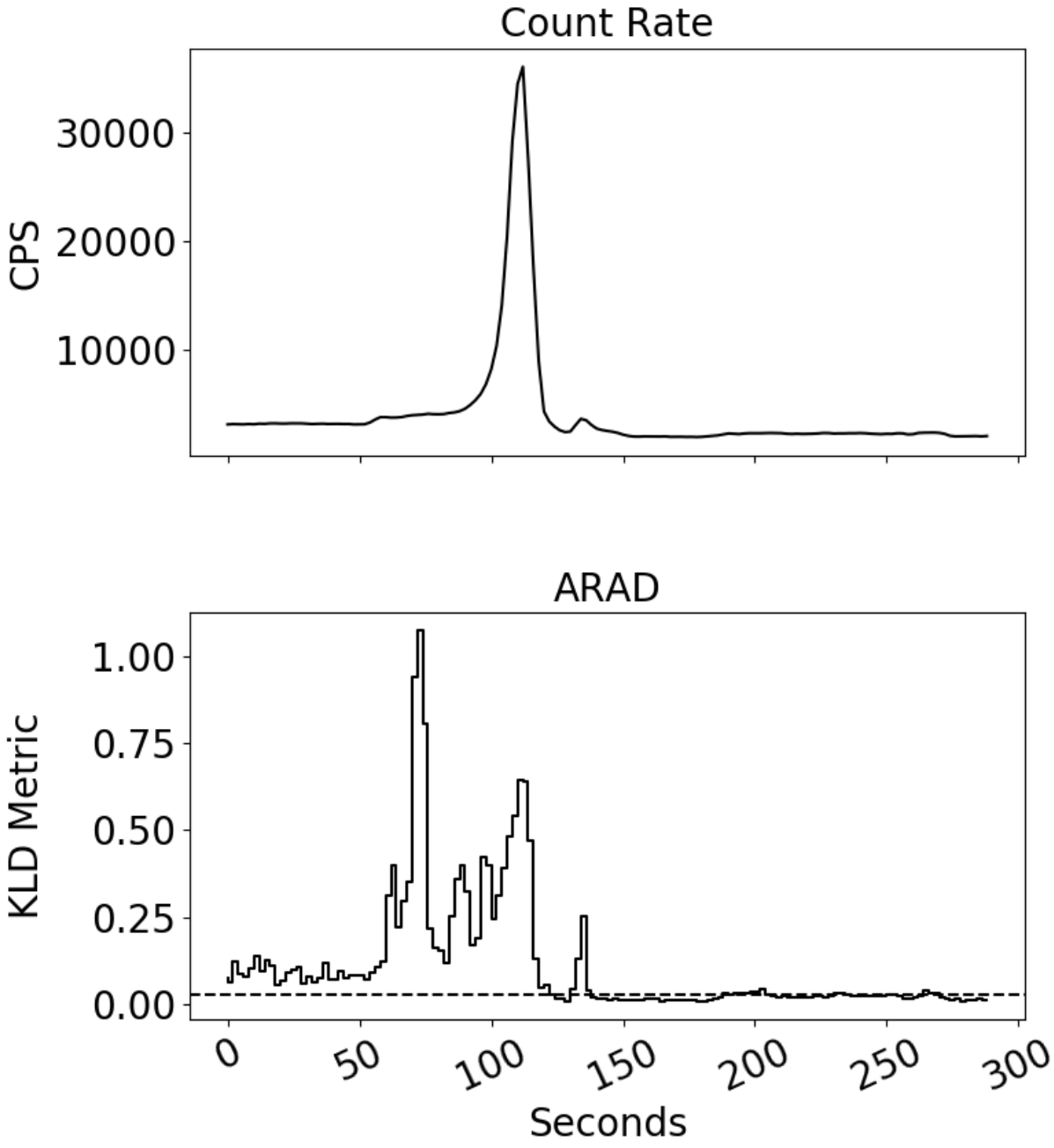


Figure 6.14: Gross count rate and ARAD KLD metric for an ^{137}Cs transfer event. Node 11. Horizontal dotted line indicates detection threshold.

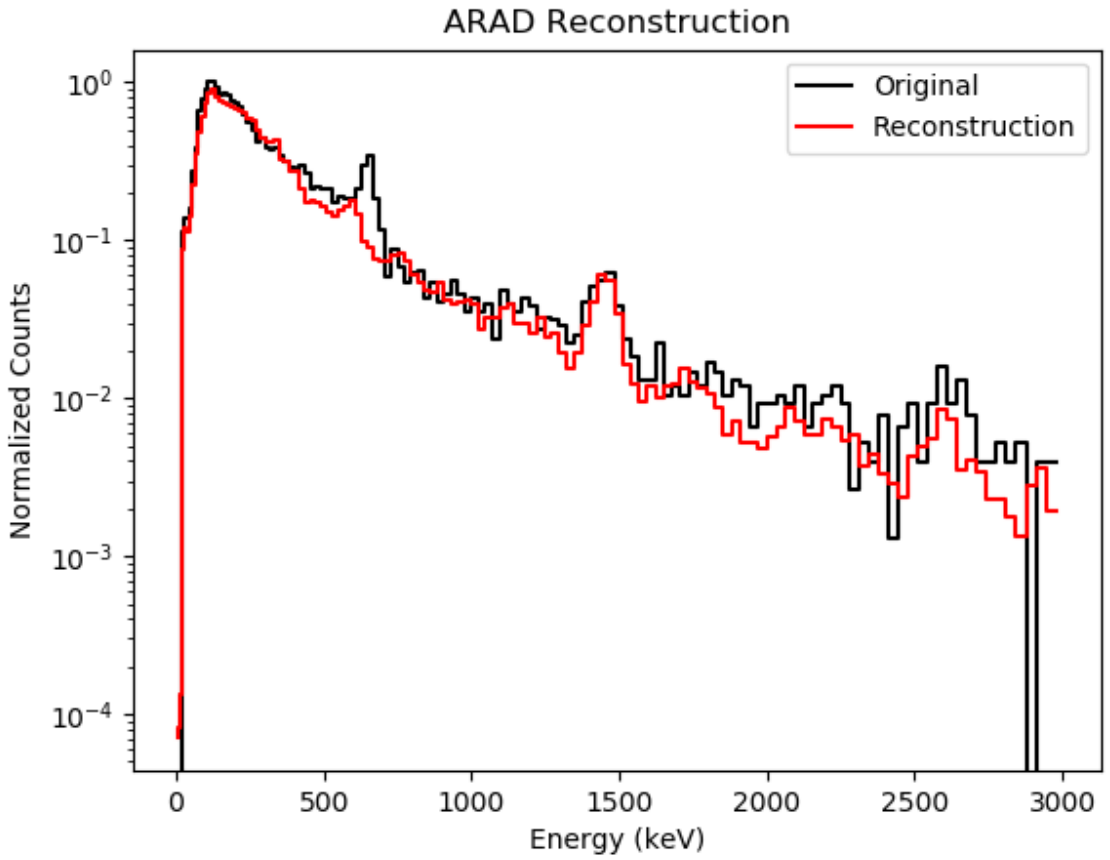


Figure 6.15: Alarm spectrum with ARAD reconstruction for the ^{137}Cs shipment event in Figure 6.14. Node 11.

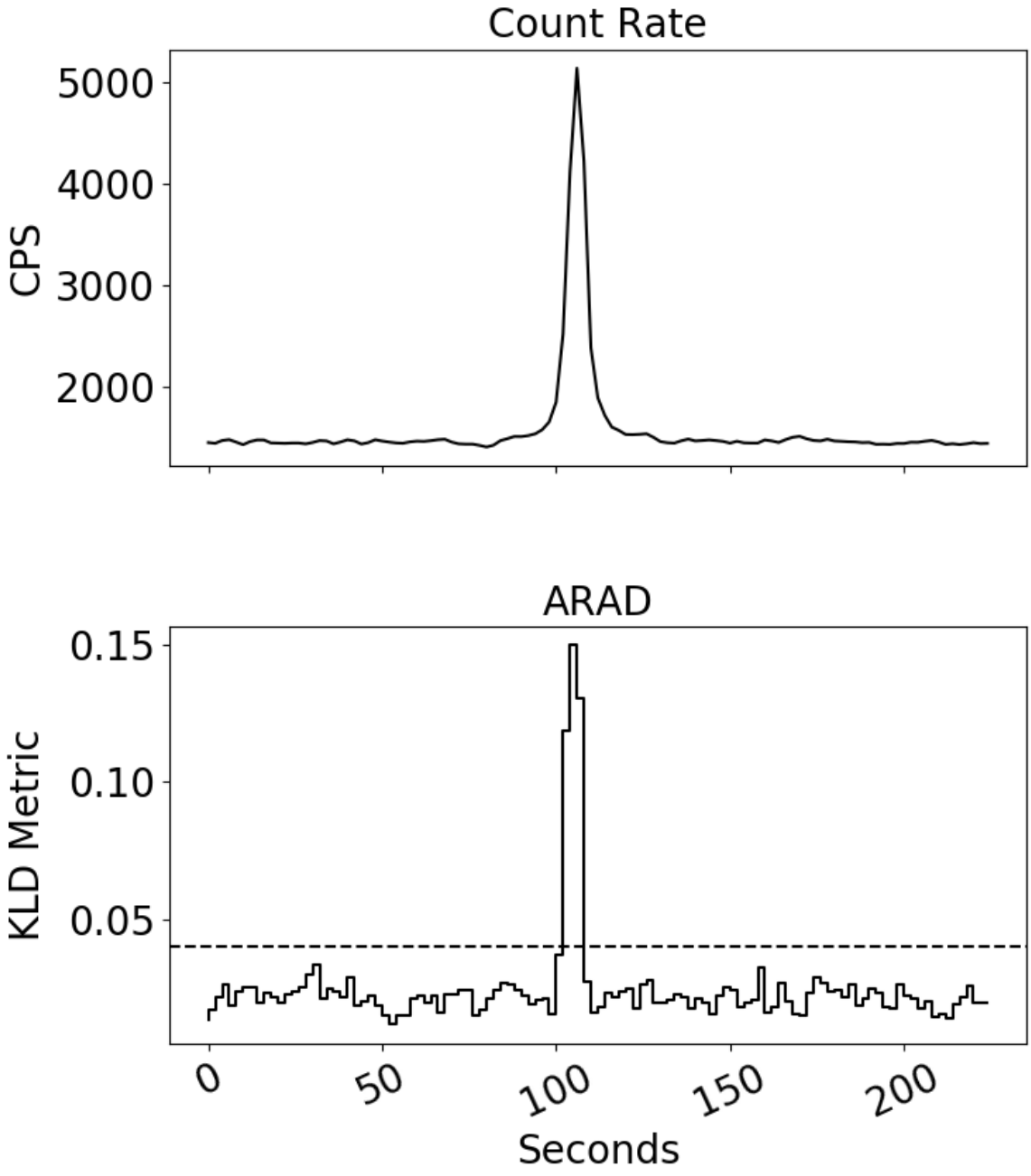


Figure 6.16: Gross count rate and ARAD KLD metric for an ^{60}Co transfer event. Node 01. Horizontal dotted line indicates detection threshold.

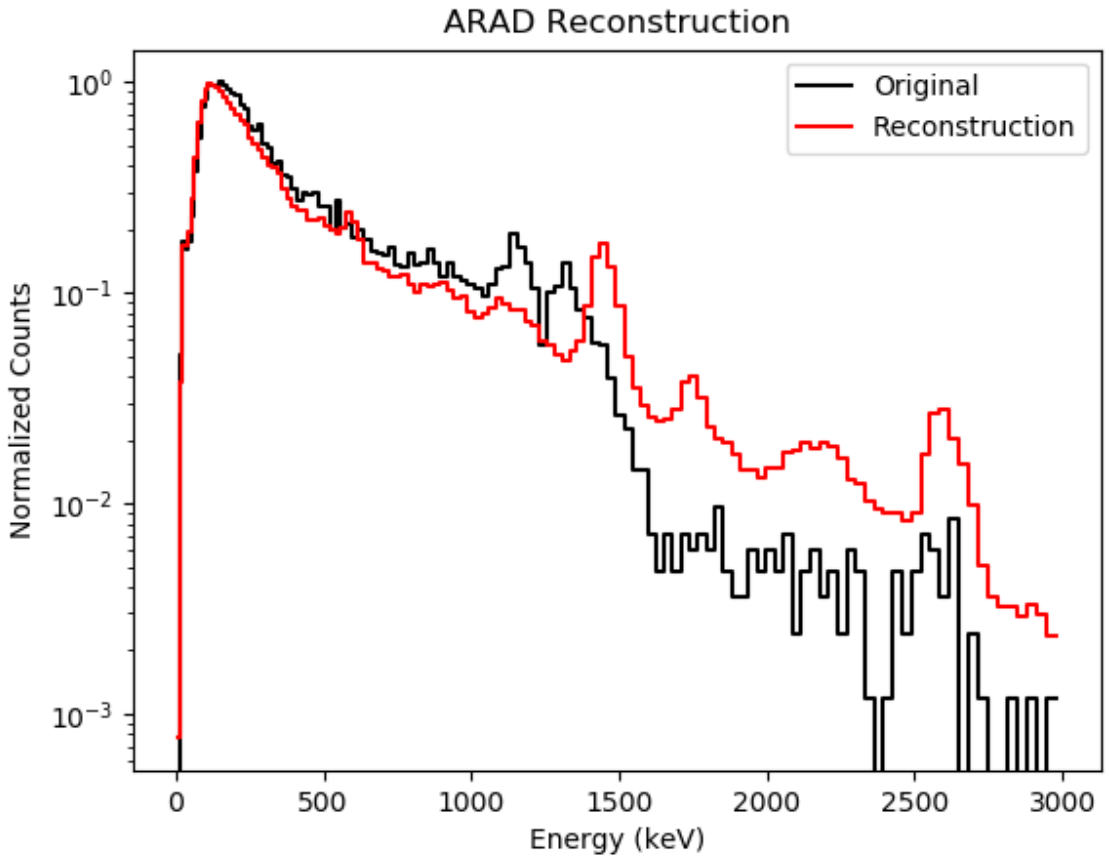


Figure 6.17: Alarm spectrum with ARAD reconstruction for the ^{60}Co transfer event in Figure 6.16. Node 01.

Chapter 7

Discussion and Conclusions

7.1 Discussion

Chapter 4 already contains discussion on the ARAD model design and architectural considerations. Also, Chapter 3 contains plenty of discussion on the datasets. This section will focus specifically on discussing the results presented in Chapters 5 and 6.

7.1.1 Simulated Data

The lower performance on HEU compared to the other 4 sources is actually not due to any fundamental flaw in the algorithm, it actually has to do with the nature of the source itself. Because the gamma ray energy distribution is so broad, a higher source activity (with respect to background) is required to make any significant feature change in the spectrum. Consider HEU versus ^{131}I as an example; if we assume that the sources each have a gross count rate of 1000 CPS and the spectrum integration time is 1 second, then for ^{131}I we would have roughly 1000 counts in the 140 keV region in the spectrum (ignoring Compton scattering), whereas for HEU, those 1000 counts would be spread amongst the dozens of different gamma rays in the energy distribution. In short, the algorithm's performance is not so much a function of the activity of the source as it is a function of how easy it is for the source to leave a distinguishable change in the energy spectrum. This is in contrast to gross count rate based algorithms where the ability to detect a source is determined by the activity of that source

relative to background, regardless of whether or not any change would be induced in the spectrum.

The type of interaction that the photon makes in the detector also plays a role in the detectability of the source (using ARAD). As was discussed in Chapter 1, photoelectric absorption of photons in the detector is the most useful interaction from a gamma spectroscopy standpoint because it results in full energy deposition of the photon and a well-formed Gaussian peak. Compton scattering on the other hand has a wide distribution of possible energies that the scattered photon can acquire, making it more difficult to perform spectroscopy. This is the same case for ARAD. For two monoenergetic gamma ray sources, ARAD is able to detect the source with the lower energy gamma ray at a lower minimum activity than that of the source with higher energy photons. This is demonstrated in the simulation data when comparing ARAD's performance on ^{99m}Tc and ^{60}Co. Most of the ^{99m}Tc gamma rays are emitted with an energy of 140 keV. Likewise, most of the ^{60}Co gamma rays are emitted at either 1332 or 1173 keV. Because the 140 keV photons from ^{99m}Tc have such a low energy, the interaction in the detector is primarily the photoelectric effect. On the other hand, the gammas for ^{60}Co interact primarily via Compton scattering, which can also be seen in the detector response function in Figure 3.2. This means that for any given source activity, a significant fraction of the photons will be downscattered for ^{60}Co, which distributes the counts among many energy bins, reducing the change in the spectrum.

7.1.2 HFIR/REDC Data

ARAD's robustness to clutter and rain-induced background dynamics was demonstrated. Clutter in particular is a real challenge for many radiation detection algorithms because of the speed in which the gross count rate can change, which lead to false alarms in both the SPRT and k-sigma algorithms for both clutter events that were presented. On the other hand, precipitation-induced increases in the gross count rate occur quite slowly such that algorithms that use a short time window to estimate the background count rate, such as this implementation of k-sigma, may be able to avoid a false alarm. One must also consider however that a true source event may not always occur quickly, such that one may run the risk of missing a slow moving source if using a detection algorithm with a short

rolling background estimation window. In fact, this is the reason why both the k-sigma and SPRT algorithms are used side by side for the HFIR/REDC detectors; the k-sigma algorithm is able to catch fast moving and relatively strong sources whereas SPRT is more sensitive to weak sources that may ramp up over a longer period of time. Unlike many other algorithms, including k-sigma and SPRT, ARAD does not need to continuously estimate the current background count rate over time. ARAD's entire concept of what constitutes typical background is contained within its internal representation that was learned from the training data. Because rain events were included in the training data, as was described in Chapter 3, ARAD was able to reconstruct the precipitation-induced features in the spectra and did not generate an alarm.

ARAD's ability to detect a variety of real world radiation anomaly events was demonstrated. These events included radioactive waste transfer, ^{41}Ar effluent gasses, medical isotope shipments, ^{237}Np target transfers used for ^{238}Pu production, and industrial source transfers.

7.2 Conclusions

The goal of this project was to investigate the use of deep learning for developing a data-driven radiation detection algorithm that is robust to dynamic background environments. With that goal, this project was very successful. The algorithm performed well from both a source detection and a background dynamics resistance standpoint, the combination of which is notoriously difficult to achieve in radiation detection algorithm development. ARAD also performed well on the real-world data. The resistance of ARAD to clutter and rain-induced background dynamics was demonstrated. The algorithm's ability to detect a variety of real world sources, from relatively weak to very strong was also shown.

7.2.1 Summary of Results

The following list outlines some of the major achievements of this project.

- A custom autoencoder neural network architecture was designed and developed for radiation anomaly detection. This project is the **first known use** of neural autoencoders for this application space.

- The ARAD algorithm's high resilience to clutter and precipitation-induced background dynamics was demonstrated on real world data collected in a complex radiation background environment. The algorithm's ability to detect real world sources in this challenging environments was also demonstrated.

- The algorithm's resilience to NORM-induced background changes was also demonstrated on a set of simulated mobile detector data in an urban environment. The algorithm's source detection performance was quantified using a custom version of this dataset, showing similar performance to NMF on this particular dataset.

- The algorithm can process a gamma ray spectrum in inference mode in approximately 1 ms on a standard consumer grade laptop CPU, enabling its use on edge computing devices.

7.3 Future Work

There are many opportunities for future work on this algorithm. Some of these potential opportunities are listed here:

- The model can and should be tested on other datasets in different types of background and source environments.

- Modify the ARAD model to use a generative adversarial architecture, which may result in a better internal representations of the background components and may be useful for generation of synthetic data.

- Evaluate the algorithm's performance on other detectors such as $LaBr_3$(Ce) and HPGe. While the algorithm was tested on PVT data and worked, an in-depth study on this may be useful, especially for deployment on radiation portal monitors.

- A long-term study can be performed to evaluate the algorithm's effectiveness in a real world operational setting.

Bibliography

[1] Topcoder. Detecting radiological threats in urban areas. https://www.topcoder.com/challenges/30085346, 2019. Accessed: 2020-02-25. 2, 57, 60

[2] Andrew D Nicholson, Douglas P Peplow, Michael J Willis, and E Archer, Daniel. Generation of synthetic data for a radiation detection algorithm competition. *IEEE Transactinos on Nuclear Science*, Submitted. xi, 2, 57, 58, 60

[3] National Nuclear Security Administration, United States Department of Energy. Nuclear incident response. https://www.energy.gov/nnsa/nuclear-incident-response, 2017. x, 3

[4] Radiological Assistance Program. Rap60 detection identification analysis response. web, 2018. 3

[5] Glenn F Knoll. *Radiation detection and measurement*. John Wiley & Sons, 2010. 4, 11, 12, 13, 14, 15, 17, 18, 20, 21, 22

[6] RR Kinsey, CL Dunford, JK Tuli, and TW Burrows. The NUDAT/PCNUDAT program for nuclear data. Technical report, Brookhaven National Lab., Upton, NY (United States), 1996. x, 5, 6, 26, 27, 28, 30, 98, 114, 120, 127

[7] James E. Turner. *Atoms, Radiation, and Radiation Protection, 3rd, Completely Revised and Enl.* John Wiley & Sons, 2007. x, 7, 8, 9, 10

[8] Robley Dunglison Evans and RD Evans. The atomic nucleus. 1955. 11

[9] John E Jaffe, David V Jordan, and Anthony J Peurrung. Energy nonlinearity in radiation detection materials: Causes and consequences. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 570(1):72–83, 2007. 14

[10] J Cook. Solid angle subtended by two rectangles. *Nuclear Instruments and Methods*, 178(2-3):561–564, 1980. 18

[11] R Casanovas, JJ Morant, and M Salvadó. Energy and resolution calibration of nai (tl) and labr3 (ce) scintillators and validation of an egs5 monte carlo user code for efficiency calculations. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 675:78–83, 2012. 20

[12] Rachel A Powsner, Edward R Powsner, and Rachel A Powsner. Essential nuclear medicine physics. 2006. 22

[13] D.E. Archer, M.S. Bandstra, D.S. Bolme, S.L. Cleveland, J.C. Curtis, G.G. Davidson, I. Garishvili, J.O. Johnson, A.K. Mikkilineni, M.S.L. McLean, A.D. Nicholson, D.E. Peplow, A.A. Plionis, M.J. Poska, B.J. Quiter, W.R. Ray, A.J. Row, I.R. Stewart, M.W. Swinney, and M.J. Willis. Mid-Project Report for the Modeling Urban Scenarios and Experiments (MUSE) Project. Technical Report ORNL/TM-2018/964, Oak Ridge National Laboratory, August 2018. 25

[14] Matthew W. Swinney, Douglas E. Peplow, B. W. Patton, Andrew D. Nicholson, Dan E. Archer, and Michael J. Willis. A methodology for determining the concentration of naturally occurring radioactive materials in an urban environment. *Nuclear Technology*, 203:325–335, 2018. 25

[15] J. G. Ingersoll. A survey of radionuclide contents and radon emanation rates in building materials used in the us. *Health Physics*, 45:363–368, 1983. 25

[16] S. Croft and I. G. Hutchinson. The measurement of u, th and k concentrations in building materials. *Applied Radiation and Isotopes*, 51:483–492, 1999. 25

[17] Z. Szab, P. Vlgyesi, H. . Nagy, C. Szab, Z. Kis, and O. Csorba. Radioactivity of natural and artificial building materials a comparative study. *Journal of Environmental Radioactivity*, 118:64–74, 2013. 25

[18] WJ Llope. Activity concentrations and dose rates from decorative granite countertops. *Journal of environmental radioactivity*, 102(6):620–629, 2011. 25

[19] RJ Livesay, Christopher S Blessinger, Tyler F Guzzardo, and Paul A Hausladen. Rain-induced increase in background radiation detected by radiation portal monitors. *Journal of environmental radioactivity*, 137:137–141, 2014. 28, 29

[20] J.-F. Mercier, B.L. Tracy, R. d'Amours, F. Chagnon, I. Hoffman, E.P. Korpach, S. Johnson, and R.K. Ungar. Increased environmental gamma-ray dose rate during precipitation: a strong correlation with contributing air mass. *Journal of Environmental Radioactivity*, 100(7):527 – 533, 2009. 28

[21] Naoto Fujinami. Observational study of the scavenging of radon daughters by precipitation from the atmosphere. *Environment International*, 22:181 – 185, 1996. The Natural Radiation Environment VI. 28, 29

[22] Tom Burr and Kary Myers. Effects of background suppression of gamma counts on signal estimation. *Applied Radiation and Isotopes*, 67(9):1729–1737, 2009. 30

[23] Charles A Lo Presti, Dennis R Weier, Richard T Kouzes, and John E Schweppe. Baseline suppression of vehicle portal monitor gamma count profiles: A characterization study. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 562(1):281–297, 2006. 30

[24] Michael J Willis, Ian R Stewart, Andrew D Nicholson, Daniel E Archer, and William R Ray. Systematic study of variation in radiation data in cluttered environments. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 954, 2018. 30, 31

[25] Ian R Stewart, Andrew D Nicholson, Daniel E Archer, Michael J Willis, Mathew W Swinney, Irakli Garishvili, and William R Ray. Understanding and quantifying the systematic effects of clutter within a radiation detection scene. *Journal of Radioanalytical and Nuclear Chemistry*, 318(1):727–737, 2018. 30

[26] L. A. Currie. Limits for qualitative detection and quantitative determination. *Application to Radiochemistry*, 40:586–593, 1968. 33

[27] A. Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16:117–186, 1945. 33, 34

[28] K. D. Jarman, L. E. Smith, D. K. Carlson, and D. N. Anderson. Sequential probability ratio test for long-term radiation monitoring. *2003 IEEE Nuclear Science Symposium*, 2:1458–1462, 2003. 33, 34

[29] T. J. Aucott, M. S. Bandstra, V. Negut, J. C. Curtis, D. H. Chivers, and K. Vetter. Effects of background on gamma-ray detection for mobile spectroscopy and imaging systems. *IEEE Transactions on Nuclear Science*, 61:985–991, 2014. 34

[30] David Boardman, Mark Reinhard, and Alison Flynn. Principal component analysis of gamma-ray spectra for radiation portal monitors. *IEEE Transactions on Nuclear Science*, 59(1):154–160, 2012. 35

[31] E. Lei. Robust detection of radiation threat. *IEEE Nuclear Science Symposium and Medical Imaging Conference*, 2017. 35

[32] K J Bilton, Joshi, M. S T H., Bandstra, Curtis J C, B J Quiter, R J Cooper, and K Vetter. Non-negative matrix factorization of gamma-ray spectra for background modeling, detection, and source identification. *IEEE Transactinos on Nuclear Science*, 66(5), 2019. 36

[33] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951. 36, 91

[34] Chris Ding, Tao Li, and Wei Peng. On the equivalence between non-negative matrix factorization and probabilistic latent semantic indexing. *Computational Statistics & Data Analysis*, 52(8):3913–3927, 2008. 36

[35] D. M. Pfund, R. C. Runkle, K. K. Anderson, and K. D. Jarman. Examination of count-starved gamma spectra using the method of spectral comparison ratios. *IEEE Transactions on Nuclear Science*, 54:1232–1238, 2007. 37, 38

[36] D. M. Pfund, K. K. Anderson, K. D. Detwiler, McDonald B. S. Jarman, K. D., B. D. Milbrath, M. J. Myjak, N. C. Paradis, S. M. Robinson, and M. L. Woodring. Improvements in the method of radiation anomaly detection by spectral comparison ratios. *Applied Radiation and Isotopes*, 110:174–182, 2016. 37

[37] S. Mukhopadhyay, R. Maurer, R. Wolff, and S. Guss, P. a d Mitchell. Radiation anomaly detection algorithms for field-acquired gamma energy spectra. *Hard X-Ray, Gamma-Ray, and Neutron Detector Physics XVII*, 9593, 2015. 38

[38] Colin Bellinger, Nathalie Japkowicz, and Christopher Drummond. Synthetic oversampling for advanced radioactive threat detection. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 948–953. IEEE, 2015. 38

[39] Marcus J Neuer, Nikolai Teofilov, Wolf Schykowski, Christian Henke, and Elmar Jacobs. Machine learning algorithms for improving the dose rate measurement in handheld homeland security instrumentation. In *2019 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*, pages 1–3. IEEE, 2019. 39

[40] Byoungil Jeon, Youhan Lee, Myungkook Moon, Jongyul Kim, and Gyuseong Cho. Reconstruction of compton edges in plastic gamma spectra using deep autoencoder. *Sensors*, 20(10):2895, 2020. 39

[41] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991. 42

[42] Boris Hanin. Universal function approximation by deep neural nets with bounded width and relu activations. *Mathematics*, 7(10):992, 2019. 42

[43] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: A view from the width. In *Advances in neural information processing systems*, pages 6231–6239, 2017. 42

[44] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012. 43

[45] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986. 43

[46] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. 46, 47

[47] Jack Kiefer, Jacob Wolfowitz, et al. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952. 46

[48] Timothy Dozat. Incorporating nesterov momentum into adam. 2016. 47, 48

[49] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 48

[50] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012. 48

[51] Michael A Nielsen. *Neural networks and deep learning*, volume 2018. Determination press San Francisco, CA, USA:, 2015. 49, 50

[52] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. 50, 86

[53] Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 1998. 50, 86

[54] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. 50

[55] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 52

[56] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010. 52

[57] Randall C. O'Reilly, Yuko Munakata, Michael J. Frank, Thomas E. Hazy, and Contributors. *Computational Cognitive Neuroscience*. Wiki Book, 1st Edition, URL: http://ccnbook.colorado.edu, 2012. 54

[58] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014. 54

[59] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 54

[60] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 54

[61] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 54

[62] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 54

[63] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 54

[64] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017. 54

[65] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*, 2017. 55

[66] Kyunghyun Cho. Simple sparsification improves sparse denoising autoencoders in denoising highly corrupted images. In *International Conference on Machine Learning*, pages 432–440, 2013. 55

[67] Lovedeep Gondara. Medical image denoising using convolutional denoising autoencoders. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, pages 241–246. IEEE, 2016. 55

[68] Ruslan Salakhutdinov and Geoffrey Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009. 55

[69] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. 55

[70] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015. 55

[71] E. L. Paula, M. Ladeira, R. N. Carvalho, and T. Marzago. Deep learning anomaly detection as support fraud investigation in brazilian exports and anti-money laundering. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 954–960, 2016. 56

[72] Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, pages 4–11, 2014. 56

[73] Andrea Borghesi, Andrea Bartolini, Michele Lombardi, Michela Milano, and Luca Benini. Anomaly detection using autoencoders in high performance computing systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9428–9433, 2019. 56

[74] Yong Shean Chong and Yong Haur Tay. Abnormal event detection in videos using spatiotemporal autoencoder. In *International Symposium on Neural Networks*, pages 189–196. Springer, 2017. 56

[75] Mohammad Sabokrou, Mahmood Fathy, and Mojtaba Hoseini. Video anomaly detection and localisation based on the sparsity and reconstruction error of auto-encoder. *Electronics Letters*, 52(13):1122–1124, 2016. 56

[76] Andrew Nicholson, Douglas Peplow, Christine Anderson-Cook, Christopher Greulich, James Ghawaly Jr., Kary Myers, Daniel Archer, Michael Willis, and Brian Quiter. Detecting radiological threats in urban areas. https://doi.ccs.ornl.gov/ui/doi/74. Accessed: 2020-02-25. 58

[77] Douglas E. Peplow. Monte carlo shielding analysis capabilities with mavric. *Nuclear Technology*, 174(2), 1 2011. 58

[78] Andrew Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19, 2011. 81, 83

[79] Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. In *Advances in neural information processing systems*, pages 472–478, 2001. xii, 84, 85

[80] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 86

[81] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):27, 2019. 87, 88

[82] David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*, 2017. 88

[83] Bent Fuglede and Flemming Topsoe. Jensen-shannon divergence and hilbert space embedding. In *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, page 31. IEEE, 2004. 91

[84] Thomas E Sampson. Plutonium isotopic composition by gamma-ray spectroscopy: A review. Technical report, Los Alamos National Lab., NM (USA), 1986. 96

[85] US Department of Energy. How scientists discovered a new way to produce actinium-225, a rare medical radioisotope, June 2018. 120

Appendices

A ARAD Training Parameters

Listing 1: Training Config File

```
{
  "num_inputs": 128,
  "latent_space": 5,
  "batch_size": 64,
  "learning_rate": 0.0001,
  "learning_rate_decay": 0,
  "epochs": 160,
  "loss": "logcosh",
  "kernel_regularizer": null,
  "kernel_regularization_parameter": 0.01,
  "bias_regularizer": null,
  "bias_regularization_parameter": 0.1,
  "activity_regularizer": null,
  "activity_regularization_parameter": 0.01,
  "latent_regularizer": "kld",
  "latent_regularization_parameter": 0.05,
  "latent_sparsity_parameter": 0.001,
  "kernel_constraint": "min_max_norm",
  "min_norm": 1,
  "max_norm": 2,
  "dropout": 0,
  "hidden_activation": "softplus",
  "output_activation": "softplus",
  "early_stopping_enabled": true,
  "early_stopping_wait_epochs": 5,
  "early_stopping_min_delta": 1.0e-7,
  "output_bias": true,
  "lr_decay_enabled": true,
  "lr_decay_factor": 0.1,
  "lr_decay_min_delta": 2.0e-8,
  "lr_decay_min_lr": 1.0e-8,
  "track_cosine_similarity": false,
  "optimizer": "nadam",
```

```
34    "sgd_momentum": 0,
35    "sgd_nesterov": true,
36    "lrs": false
37  }
```

Vita

James Ghawaly Jr. was born and raised in Southeast Louisiana. While in high school, James developed an interest in nuclear physics and decided to pursue a Bachelor's degree at the University of Tennessee, Knoxville. After completing his Bachelor's degree, James continued to study at UTK to pursue a Master's degree in Computer Engineering and a Ph.D. in Nuclear Engineering. James is passionate about the application of modern machine learning techniques to the analysis of radiation detection data. After graduation, James plans to begin a career in data science and machine learning for radiation detection, identification, and localization as they apply to national nuclear security.