Gamma Spectroscopy Data Augmentation for Self-Supervised Machine Learning Applications to Nuclear Nonproliferation on Measured Data with Limited Ground-Truth

by

Jordan R. Stomps

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Nuclear Engineering and Engineering Physics)

at the

UNIVERSITY OF WISCONSIN–MADISON

2023

Date of final oral examination: 12/07/2023

For Sara

Acknowledgments

Thank you first to my advisor, Paul Wilson. Now—after several years steeped in the world of national security—I can appreciate how important nuclear nonproliferation research is to the world and the country. I discovered a vocation in my graduate studies, and it is thanks to Paul's guidance which has formed me as a scientist. I could not have asked for a more patient and approachable advisor, and I commend you for the community you have fostered. Thank you also to all the amazing researchers in CNERG that shared meetings, offices, and coffees with me.

To the National Nuclear Security Administration and the Consortium for Enabling Technology and Innovation (ETI), I owe a great deal of thanks. This work is supported by the Department of Energy / National Nuclear Security Administration under Award Number(s) DE-NA0003921. If the goal of university consortia is for them to act as a professional pipeline to the DOE research complex, then they are a success. Much of the collaboration for this work was the result of ETI facilitated interactions. I am proud to be a member of this network and a small part of the amazing research undertaken by ETI. I cherish the friendships I have made with my fellow consortium graduate students and look forward to seeing familiar faces during conference season. Thank you to Rob Nowak and Danica Fliss for teaching me (formally and informally) about machine learning and building my intuition for its application.

I should certainly acknowledge the impact of Oak Ridge National Laboratory on this research and my graduate experience. Thank you to Ken Dayman for being the best mentor and showing me how rewarding research in data science for nuclear nonproliferation can be. You poured way more time and effort into guiding me than I deserve. I am particularly grateful for your help in articulating why this research matters and how it aligns with our national security objectives. Thank you also to the close network of friends and colleagues I have worked with at ORNL, including Birdy Phathanapirom, Jason Hite, and the entire DSEN team. I am also thankful for the help of the MINOS collaboration at Oak Ridge National Laboratory. Thank you to Jared Johnson for your patience and diligence in responding to my torrent of emails. Thank you to Brian Quiter for your guidance on data processing. Thank you to Daniel Archer, Michael Willis, James Ghawaly, and Andrew Nicholson for your role in collecting and curating the data I used in this work. Without this data my results would be a lot less interesting.

Finally, thank you to my support system. Thank you to my family for being role models in diligent hard work. The burden falls heaviest on Sara, who has suffered through all the highs and lows of graduate school and beyond alongside me. You are my inspiration without which none of this would be possible.

Contents

List of Tables

List of Figures

Nomenclature

AEA Atomic Energy Act of 1954

AI Artificial Intelligence

ANN Artificial Neural Network

BEADS baseline estimation and denoising using sparsity

BP backpropagation

CNN Convolutional Neural Network

DL Deep Learning

DOE Department of Energy

EAAT Exponential Averaging Adversarial Training

ERM Empirical Risk Minimization

FWHM Full-Width Half-Maximum

HFIR High-Flux Isotope Reactor

IAEA International Atomic Energy Agency

KUT Potassium-Uranium-Thorium

L-BFGS Limited-memory Broyden-Fletcher-Goldfarb-Shanno

LEU Low Enriched Uranium

MINOS Multi-Informatics for Nuclear Operating Scenarios

ML Machine Learning

MLP Multilayer Perceptron

MSE Mean Square Error

MT Mean Teacher

NaI sodium iodide

NLP Natural Language Processing

NNSA	National Nuclear Security Administration
NPP	nuclear power plant
NPT	Treaty on the Non-Proliferation of Nuclear Weapons
OOD	out-of-distribution
ORNL	Oak Ridge National Laboratory
PCA	Principal Component Analysis
PDF	Probability Distribution Function
RBF	Radial Basis Function
REDC	Radiochemical Engineering Development Center
ROC	Receiver Operating Characteristic
ROI	Region of Interest
S^{3}VM	Semi-Supervised Support Vector Machine
SGD	Stochastic Gradient Descent
SimCLR	Simple Framework for Contrastive Learning of Visual Representations
SME	Subject-Matter Expert
SNM	special nuclear material
SPRT	Sequential Probability Ratio Test
SSL	Self-Supervised Learning
SSML	Semi-Supervised Machine Learning
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TSVM	Transductive Support Vector Machine
VAT	Virtual Adversarial Training

Abstract

The timely detection of special nuclear material (SNM) transfers is an important monitoring objective in nuclear nonproliferation. Labeling sufficient volumes of radiation data for successful supervised machine learning can be too costly when manual analysis is employed. Therefore, this work is developing a machine learning model built on semi-supervised learning to utilize both labeled and unlabeled data and therefore alleviate the cost of labeling.

As a preliminary experiment, radiation measurements collected with sodium iodide (NaI) detectors from the Multi-Informatics for Nuclear Operating Scenarios (MINOS) testbed at Oak Ridge National Laboratory (ORNL) are used. Anomalous measurements are identified using a method of statistical hypothesis testing. After background estimation, an energy dependent spectroscopic analysis is used to characterize an anomaly based on its radiation signatures in a noisy labeling heuristic. These noisily labeled spectra are used in training and testing classification models that estimate a binary label: SNM transfer or other anomalous measurement. Supervised logistic regression—trained only on limited labeled data—serves as a baseline to compare three semi-supervised machine learning models all trained on the same limited labeled data and a larger volume of unlabeled data: co-training, Label Propagation, and a Convolutional Neural Network (CNN). In each case, the semi-supervised models outperform logistic regression, suggesting unlabeled data can be valuable when training and demonstrating performative value in semi-supervised nonproliferation implementations.

This work uses a self-supervised contrastive learning framework to efficiently extract information from unlabeled data. A contrastive model learns patterns by perturbing data instances using a set of label-invariant data augmentations, meaning augmented samples preserve labeling information present in an original measurement. A set of transformations are designed for gamma spectra, tailored for specific principles of radiation detection. MINOS measurements are augmented, and an encoder is contrastively trained to produce meaningful high-dimensional representations of spectra. A supervised classifier then uses these encoded representations to assign a label estimating whether a given transfer spectrum was of tracked nuclear material or not. Even a simple linear model built on these representations and trained on limited labeled data can achieve a balanced accuracy score of 80.30%.

Several tools are employed for evaluating the efficacy of augmentations, representations, and classification models. Principal Component Analysis (PCA) is used to demonstrate that representations provide a richer feature space for detecting nuclear material transfers by embedding distributional information from unlabeled data. Integrated Gradients connect a classifier's decision boundary to spectral features, suggesting the framework learns relevant patterns in spectra that can be used for detecting transfers. When labeled data are scarce, this work suggests that training a supervised classifier should be prioritized over semi-supervised (compared to self-supervised) contrastive learning an encoder to maximize detection accuracy. Hyperparameter optimization was conducted, finding a locally optimum maximum cross-validated balanced accuracy score. Overall, a methodology has been established for using semi-supervision to accurately classify SNM transfers without the prohibitive cost of labeling.

Chapter 1

Introduction

Largely because of the history of nuclear weapons and safety concerns during the Cold War, the Treaty on the Non-Proliferation of Nuclear Weapons (NPT) was signed on July 1, 1968. At the time, there existed five nations possessing nuclear weapons: the United States, Russia, the United Kingdom, France, and China. The treaty was an attempt to limit, rather than increase, the world's supply of nuclear weapons. Signatories without nuclear weapons agreed never to acquire them and nuclear weapons states agreed to "the cessation of the nuclear arms race and to undertake effective measures in the direction of nuclear disarmament [1]." This is similar in spirit to President Eisenhower's speech to the United Nations on December 8, 1953, in which is articulated the societal benefits of civilian nuclear power while also acknowledging the dangers of nuclear proliferation in the twentieth century. Such distinctions are further accentuated in light of increased concern for the affects of climate change and how nuclear power as a clean energy source can play a role in the global electricity portfolio.

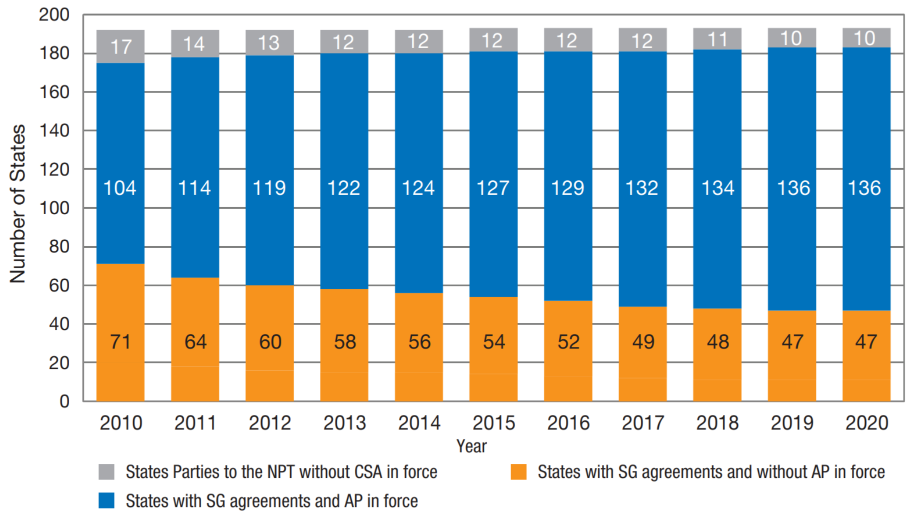
Globally, the International Atomic Energy Agency (IAEA) is responsible for monitoring and verifying that nations states with nuclear fuel cycles comply with the NPT and accurately report their nuclear activity as expressed by the IAEA's mandate to "seek to accelerate and enlarge the contribution of atomic energy to peace, health and prosperity throughout the world [2]." In 2000, the National Nuclear Security Administration (NNSA) was formed under the Department of Energy (DOE) to promote national security within the United States. This is covered by the four pillars of their mission: maintaining the nuclear stockpile, counterterrorism and counterproliferation, powering the nuclear navy, and nuclear nonproliferation. That is, "[the] NNSA works to prevent nuclear weapon proliferation and reduce the threat of nuclear and radiological terrorism around the world. The agency endeavors to prevent the development of nuclear weapons and the spread of materials or knowledge needed to create them [3]."

1.1 Motivation

Through a set of tools and technologies, organizations such as the IAEA monitor nuclear activity and ensure that nation states or individuals are abiding by laws, regulations, and international agreements. The Atomic Energy Act of 1954 (AEA) defines SNM as "plutonium, uranium enriched in the isotope 233 or in the isotope 235, and any other material which the Commission...determines to be special nuclear material [4]." The illicit use of SNM and other radioactive materials therefore necessitates timely detection and characterization—important steps toward ensuring nuclear nonproliferation.

As nuclear technology expands, the international community will have an increased burden on verification with limited resources [5, 6]. Methods that detect and track nuclear material are valuable for monitoring material inflow and outflow throughout the nuclear fuel cycle. Of course, these tools can be employed by IAEA inspectors for use in treaty verification, but other nuclear stakeholders can benefit from material tracking technology. Those responsible for nuclear facilities can use material tracking to ensure operational integrity. Site security personnel could also tracking as an additional layer monitoring access controls.

This can be a difficult objective to achieve without prior knowledge of the event itself. Traditional radiation detection and manual analysis can be unreliable, highly variable, time consuming, and require subject-matter expertise. Confounding variables and complex physical behavior can occlude radiation signatures, precipating careful and costly evaluation for every individual measurement. Machine Learning (ML)—designed for analytical analysis of complex observed systems—could aid in radiation detection and/or characterization, thereby enabling rapid development and deployment of monitoring and tracking systems.



Year	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
States Parties to the NPT without CSA in force	17	14	13	12	12	12	12	12	11	10	10
States with SG agreements and AP in force	104	114	119	122	124	127	129	132	134	136	136
States with SG agreements and without AP in force	71	64	60	58	56	54	52	49	48	47	47

Figure 1.1: From the IAEA's 2020 Annual Report, this chart tracks the evolution of state safeguards agreements with the IAEA over time. Note that the number of states with Additional Protocols (blue) enforced constitutes the majority of safeguards agreements and is increasing over time. As more nation states express interest in nuclear fuel cycles, the IAEA will be expected to conduct more treaty verifications despite no real growth in its annual budget [6]. (Image Source: [5])

However, many established ML implementations require large volumes of data and/or significant computing costs for pattern recognition. Abundant data and computing resources may not apply in nuclear nonproliferation scenarios where an increased burden on verification with limited resources [5] necessitates resource efficiency. The primary motivation of this work is to focus on tools that can be employed to distinguish SNM transfers from other environmental effects. This involves answering the following: (1) is a nuclear material transfer occurring, (2a) what kind of material is it, and (2b) how much material is present? Achieving both objectives without prior knowledge unique to the transfer event is essential, provided that the implementation alleviates the high computing and domain costs typical for traditional ML methods. In other words, if data can feasibly be collected, but is costly to label, this work highlights alternative ML techniques that reduce labeling requirements and leverage unlabeled data.

Without precise contextual information, signatures from a material transfer may be difficult to resolve with rudimentary detection algorithms or can be prone to misidentification. Thus, the cost of labeling training data needed for appropriately generalizable models can be as prohibitive as manually evaluating measurement samples with an Subject-Matter Expert (SME). Methods employing Semi-Supervised Machine Learning (SSML) attempt to address this by incorporating unlabeled data to achieve a performant model with the limited labeled data expected in constrained resource scenarios. As shown herein, accuracies competitive with supervised methods are observed when the volume of unlabeled data is much larger than the volume of labeled data, making semi-supervised machine learning worthwhile when labeled data are costly to produce, rare, or limited in volume. This means that monitoring institutions can still utilize information gathered even if it has not been extensively labeled, thus establishing a methodology enabling radiation monitoring in data-rich, label-poor environments.

1.2 Methodology

First, envision a generalized workflow for detecting SNM transfers without the use of advanced computational techniques. A radiation detector would be used to take gamma spectroscopic measurements of a transportation vehicle. This measurement would result in a spectrum of gamma radiation. A realistic measurement would contain energy-dependent counts associated with background radiation from environmental factors. This traditionally includes radiation from ^{40}K, ^{232}Th, and ^{238}U, or the Potassium-Uranium-Thorium (KUT) decay chain, all of which are radioactive isotopes with natural abundance on Earth.

If an SNM transfer occurs, the measurement will include signatures associated with the transfer superimposed upon the counts associated with background. The specific energy-dependent signature from the transfer will depend upon the type of material carried. In this scenario, it would be the responsibility of the monitoring individual to analyze such a spectrum and recognize off-normal, or anomalous measurements that would be associated with an SNM transfer. Of course, more than just SNM transfers can appear anomalous in gamma measurements, including other environmental behaviors. For example, rain can "washout" radioactive ^{222}Rn which has a half-life of approximately 3.8 days and is also a result of the decay of ^{238}U. These radon isotopes would have to be distinguished from an event of interest like an SNM transfer.

1.2.1 The Role of Machine Learning

This process naturally requires an SME with training to analyze each measurement and make a confident evaluation for the presence of controlled material flows. This can be costly for the domain expert as the frequency or volume of spectroscopic measurements increases. One way to alleviate this is to build machine learning models that can identify SNM transfers faster than a manual analysis. Unsupervised machine learning algorithms involve running a model on large amounts of data that groups based on similar properties. No prior labeling of data samples is required. Because of the lack of classification context, these methods are typically used in pre-task investigations of data patterns and properties.

For direct labeling or classification, supervised machine learning algorithms can be deployed. The design of these models incorporates physical or statistical properties of the subject matter to make a classification. Supervised machine learning requires a dataset containing instances of measurements (x_i) and its associated label (y_i, e.g. background, SNM, ^{222}Rn). These are used to train models that can make prediction, $\hat{y}_i$ after being trained to reduce the error, or loss, between the prediction and label for the labeled samples used in training. This loss optimization could take the form of mean squared error for example.

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{1.1}$$

Supervised machine learning algorithms require substantial amounts of labeled data to train accurate models. The cost of labeling this much training data for appropriately generalizable models can be as prohibitive as manually evaluating measurement samples above. Semi-supervised machine learning methods attempt to address this by incorporating unlabeled data to achieve a performant model with limited labeled data as expected in constrained resource scenarios. As shown herein, semi-supervised machine learning can outperform supervised methods when unlabeled data is plentiful but labeled data is costly to produce or rare and therefore limited in volume. This means that inspectors can still utilize information gathered even if it has not been extensively labeled, reducing the burden of cost on monitoring institutions.

1.3 Goals

This work evaluates how semi-supervised machine learning can utilize information gained from both unlabeled and labeled data in pattern recognition algorithms for application in nuclear nonproliferation. In this application space, the goal is to optimally leverage large volumes of radiation data with limited ground-truth data. The efficacy of SSML models will be demonstrated on real-world data containing measurements of shielded radiological material transfers collected at ORNL. Of particular interest is the SSML subset entitled Self-Supervised Learning (SSL) which, guided by a modest amount of labeled data, iteratively learns patterns in unlabeled data instances in a self-guided manner. One SSL approach utilizes physically viable data augmentations to generate a larger training corpus. These data augmentations allow an appropriate model (e.g. CNN) to learn representations of data samples that maximize classification agreement. That is, if a data sample (x_i) is transformed ($\tilde{x_i}$) in a way designed to maintain its labeling information (y_i), both samples given to a model should result in the same output prediction. The result of these perturbations is a more robust model that can harness both forms of training data in a semi-supervised context. This follows from a brief survey of existing semi-supervised machine learning techniques and their relevance to nuclear nonproliferation and spectroscopic data.

The rest of this document starts with background material. This includes an overview of the nuclear fuel cycle, SNM signatures, and gamma spectroscopy. Then, a review of machine learning concepts is offered with an emphasis on the emerging semi-supervised paradigm.

This is followed by an overview of traditional self-supervised algorithms used for image classification. SSML applications to nuclear data, other than in nuclear nonproliferation, will be briefly examined.

Next, preliminary work to demonstrate the value of semi-supervised models to nuclear nonproliferation is presented. The full workflow includes data preprocessing and a hypothesis testing algorithm to identify anomalous spectral measurements in temporally varying gamma radiation measurements. Anomalous measurements are noisily labeled with a labeling heuristic intended to employ domain aware data analysis without direct access to ground-truth information. Training, testing, and unlabeled subsets are formed for binary classification (SNM or other anomaly). Three semi-supervised models (co-training, Label Propagation, and a semi-supervised CNN) are compared to a traditional supervised model (logistic regression). In each case, the SSML model outperforms the supervised model in classification accuracy. This demonstrates the intrinsic value of SSML models in nuclear nonproliferation and costly labeling regimes.

The doctoral research detailed here involves an investigation of self-supervised machine learning and data augmentations. Since data transformations for images are maturely developed, this work attempts to present a set of valid transformations with equal applicability to spectral nuclear data. Several initial non-exhaustive examples are presented, and their physical applicability is discussed. These perturbations will be used in training two types of self-supervised models based on the SimCLR framework developed by Google [7]. First, the value of data augmentation is quantified in regimes where limited positive samples (e.g. SNM transfer spectra) are available. Second, these data augmentations are used in a broader SSML paradigm where labeled and unlabeled data is used. In this case, positive and negative samples are produced from transformations and used to create robust representations of the data distribution and allow a ML model to classify more accurately SNM transfers by fine-tuning the previous SSL implementation.

Chapter 2

Background

This section reviews the necessary background information for machine learning in nuclear nonproliferation. First, radiation monitoring is described. The flow of nuclear material and the physics behind its detection is detailed in Section 2.1. Then, a primer on machine learning is provided in Section 2.2. This includes a review of the basics for linear regression followed by some brief mathematical concepts regarding the construction of neural networks.

Note the SSML survey in Section 2.3. This section will summarize the body of research on SSML at present, including notable assumptions about efficacy and example algorithms. Contrastive learning will be introduced in Chapter 4 for use later in this research. This is followed by a literature review of existing SSML implementations for nuclear engineering in Section 2.4. For a detailed review of data analytic tools used in this work, refer to Appendix I.

2.1 Existing Radiation Monitoring for Nonproliferation

A schematic of the nuclear fuel cycle can be seen in Figure 2.1. Uranium used in energy production is mined and milled from the Earth's crust, enriched to levels appropriate for electricity production (Low Enriched Uranium (LEU)), fabricated into fuel, used in a nuclear reactor, and then sent to interim or long-term storage. Each box in Figure 2.1 is typically a separate facility, sometimes residing in entirely different nation states or institutions.

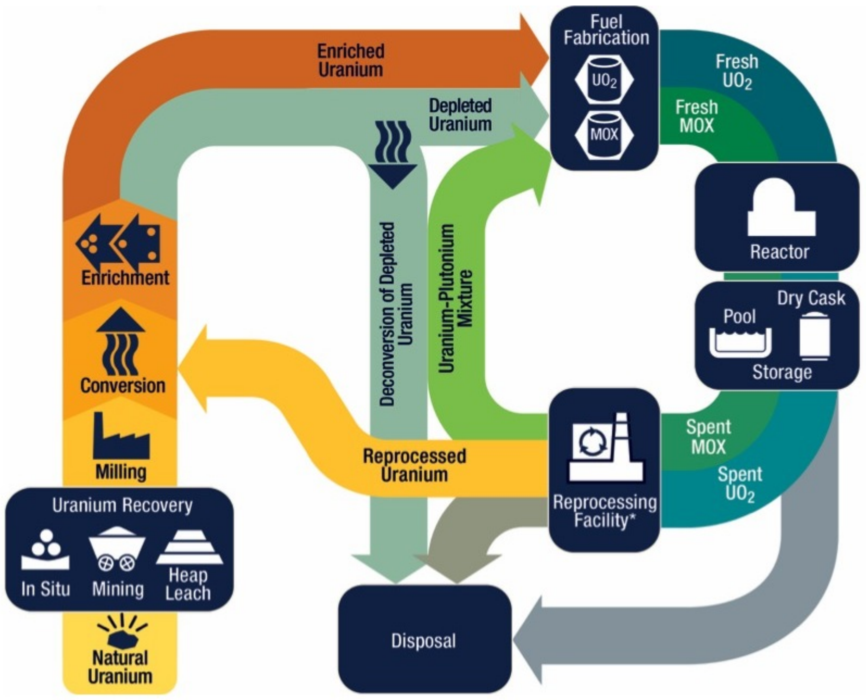


Figure 2.1: An overview of the nuclear fuel cycle from mining to permanent storage. Note that most countries do not have a closed nuclear fuel cycle where spent nuclear fuel is reprocessed. Instead, countries like the United States have an open fuel cycle where spent nuclear fuel is sent to interim or permanent storage. (Image source: [8])

2.1.1 Nuclear Material Transfers

A flow of materials likely includes transportation of nuclear materials between facilities, conducted by semi-truck or similar when ground transportation is necessary. However, as nuclear technology expands, the international community, particularly the IAEA, will have an increased burden on verification with limited resources [5]. Methods that detect and track nuclear material are valuable for monitoring flows throughout the nuclear fuel cycle. Advancing this capability would enhance the IAEA's ability to autonomously verify material movement and therefore be more resource efficient.

One method of detection would be radiation monitoring. An example gamma radiation spectrum can be seen in Figure 2.2. Some photopeaks visible in the spectrum are the result of persistent background radiation that naturally occurs around Earth. This includes signatures related to the KUT decay chain (^{40}K and ^{208}Tl, for example). Other signatures present

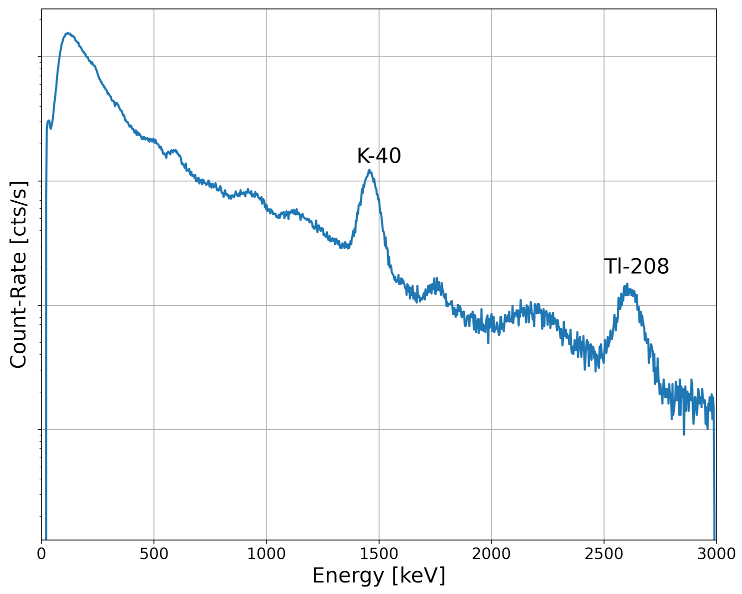


Figure 2.2: Here is an example radiation spectrum taken from MINOS. Note several spectral features, including photopeaks associated with background radiation (labeled).

in the spectrum must be measured and identified. Figure 2.3 is a spectrum taken during a nuclear material transfer. To accentuate the portion of the spectrum that is associated with the transfer, a portion of the background has been estimated and subtracted (green line). Note the substantial increase in count-rate at low energies. In this case, for example, a detection model must be trained to identify this response and associate it with the appropriate radiation event type.

Radiation measurements are temporal, meaning they can be continuously measured and will vary statistically with time. One month of 1-minute measurements are plotted in Figure 2.4. A Probability Distribution Function (PDF) is plotted in each energy bin of the plot. That means that each vertical slice is a frequency of how often the measurement for that energy bin registered that magnitude of count-rate over the course of the month. The goal of this work is to identify anomalous measurements outside of the typical distribution represented in yellow and green. The low energy signatures associated with the transfers are visible as small purple dots, as well as other off-normal measurements elsewhere in the energy spectrum.

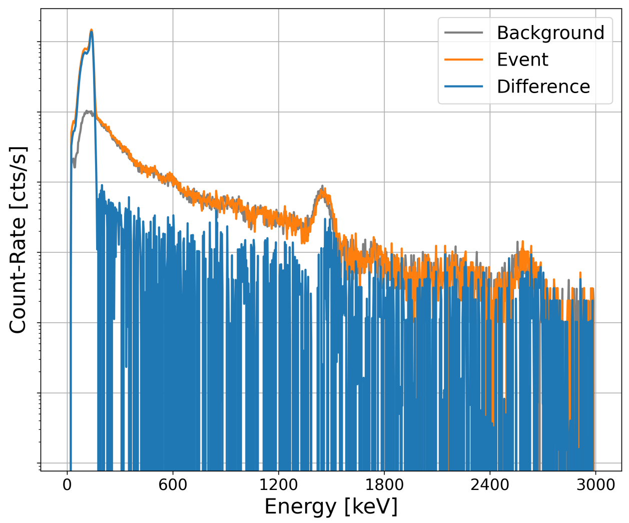


Figure 2.3: A radiation spectrum taken when a nuclear material transfer was occurring (orange). Note the high count-rate, low energy distribution associated with the transfer. In an attempt to accentuate this feature, an approximated background distribution (gray) is subtracted from the event spectrum to get a difference (blue) only associated with the anomalous features.

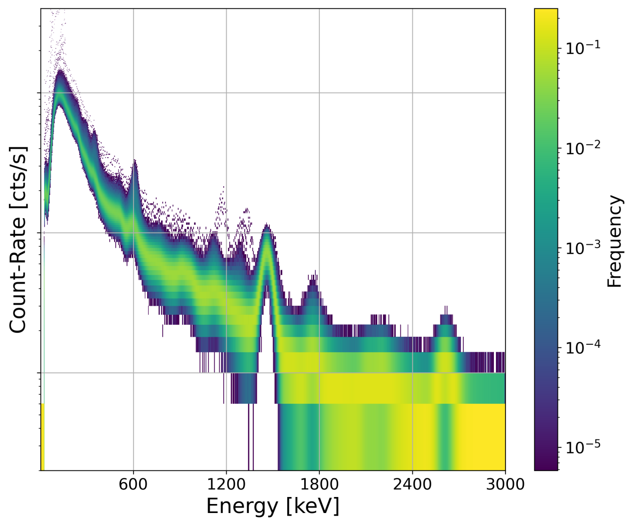


Figure 2.4: 1-minute gamma radiation spectrum measurements collected over one month, collected into one plot. Here, each vertical slice (energy bin) is a PDF. Color indicates the frequency at which the count-rate associated with each energy was measured at that specific magnitude.

2.1.2 Scintillation Detectors

Gamma radiation is the result of nuclei decaying from an excited energy state and releasing a photon with energy dependent on the series of allowable energy transitions during de-excitation and transition probabilities. These photons can be measured using instruments like a NaI detector. For a more detailed explanation of the physics underpinning nuclear radiation detection, refer to the textbook by Krane [9]. This detector will scintillate (i.e. emit light) when incident radiation interacts with the detection medium. The near-visible light will impinge on a detecting surface, producing and multiplying electrons that can be registered as an electronic pulse measureable using a data acquisition system. This pulse is proportional to the specific amount of energy imparted in a given acquisiton timing window.

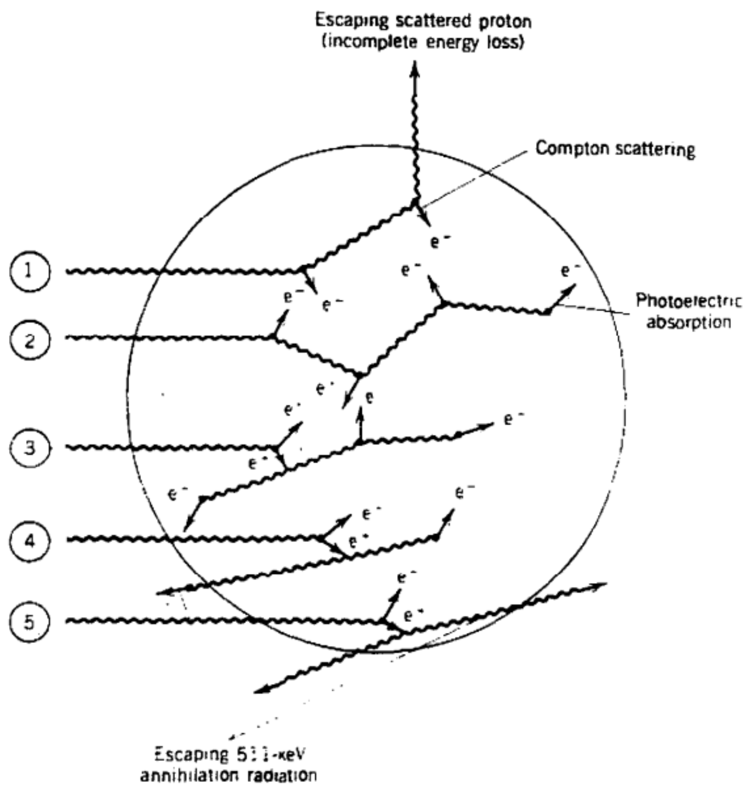


Figure 2.5: Each interaction that could occur when gamma radiation impinges on a detection medium is shown here: Compton scattering, photoelectric absorption, and pair production. (Image source: [9])

Several interaction mechanisms are possible between gamma radiation and a NaI detector.

Each is shown in Figure 2.5. Compton scattering can occur in which a photon interacts with a charged particle, e.g. electron. This collision imparts energy to the charged particle, reducing the energy of the incoming photon. The change in energy is a function of the scattering angle, θ, for the collision. With an initial wavelength of λ, the final wavelength (and therefore energy) after a collision will be:

$$\lambda' = \lambda + \frac{h}{m_e c}(1 - \cos\theta) \tag{2.1}$$

Where h is the Planck constant, m_e is the electron rest mass (511 keV), and c is the speed of light. Compton scattering can occur at a continuous number of scattering angles. A distribution of energies is registered by the detector for downscattered gamma rays that escape the detector imparting incomplete energy deposition. This distribution is defined as the Compton continuum and is spread across a NaI spectrum at energies less than the full gamma radiation energy level.

A photon can also impart the entirety of its energy to an electron via the photoelectric effect. In this case, the photon is absorbed by the electron, usually exciting it or freeing it from a bound state. When the full energy of the photon is recovered by the detector, this is registered as a photopeak, which is a spectral feature equal to the actual energy of the incident radiation. This is a useful feature for measuring the actual energy of the radiation in question.

Gamma radiation exceeding energy of two electron rest masses, or 1.022 MeV, is capable of pair production. In this case, a photon produces an electron-positron pair and is reduced by the requisite energy: 1.022 MeV. Positrons tend to annihilate with an electron, producing two photons each at an energy of 511 keV each. This photons in turn can interact with the detection medium, like the parent gamma radiation that produced them, or escape from the medium thus removing the energy from the system. When this occurs, the energy lost is not registered by the NaI detector and the resulting spectral response will be a count at the original photon energy less 511 keV (single escape peak) if one photon escapes or 1.022 MeV (double escape peak) if both leave the system.

2.2 An Introduction to Machine Learning

Machine learning—or more broadly, artificial intelligence—is a broad class of estimation techniques relying on observed data (measured or simulated) for numerically optimizing a function describing a relationship or physical system. Machine learning is occasionally used to generate new synthetic data that reflects the same nature of these experimentally derived instances. The primary purpose of this section is to provide uniform principles and terminology for the algorithms used in Chapter 3 and Chapter 6. This introduction reflects much of the reasoning, and further information can be found, in the textbook by Shalev-Shwartz and Ben-David [10]. Other overviews can be found in textbooks by Russell and Norvig [11] (which includes discussion on implementation) or by Devroye et al. [12] (with a slightly more statistical perspective).

In this work the scope is limited to machine learning used for classification. Observed instances, x, are sampled from a distribution, $P(\mathcal{X})$, determined by domain specific physical processes. An observation, x, is typically represented as a feature vector of discrete elements that could be sampled from continuous nonlinear systems. Examples include (but are certainly not limited to) images, mathematical variables, spectra, health data, etc. The set of possible labels or classed for the modeled system is defined as $\mathcal{Y}$. So, for a given $x \sim P(\mathcal{X}) \in \mathbb{R}^d$, find a function that relates this to a label, $y \sim \mathcal{Y}$. $\mathcal{X}$ is a set of datapoint instances that could take many forms depending on the input modality and the model being designed. For example, the model could be trying to classify good vs. bad fruit, in which case the labels may be good banana, bad banana, good apple, bad apple, etc.

The goal of machine learning is to find a relationship between x and y such that the model, f, can approximate that relationship: $f(x) \approx y \; \forall \; (x, y) \in (\mathcal{X}, \mathcal{Y})$. To do this, some observations must be used. Labeled instances, L, used in "supervised" machine learning, have an associated label and the total number of paired instances is $n = |L|$. If observations are present without labels, it is possible to do "unsupervised" machine learning on this unlabeled dataset, U, numbering $m = |U|$ instances. The dataset distribution in total consists of both forms:

$$\mathbb{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_{|L|}, y_{|L|}), x_{|L|+1}, \ldots, x_{|L|+|U|}\} \tag{2.2}$$

For supervised learning, this culminates in Empirical Risk Minimization (ERM) using a loss function. The loss function is some form of penalization against misclassification of known observations: $\frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i)$. There may be many, even infinite, models that describe the observed system. The goal is therefore to find the model that ultimately minimizes this loss function and finds the optimal model:

$$\hat{f} = \underset{f}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(x_i), y_i) \tag{2.3}$$

Of course, there are many ways to choose the model, the loss function, the method of finding $\hat{f}$, or even the instances (x_i, y_i) used. A simple model would be a linear regression system where each feature vector, $x_i = [x_1, x_2, \ldots, x_j]$, is connected to its label, y_i, by a system of weights, w. Then, the model becomes $f(x_i) = \hat{y} = \sum_{j=1}^{n} w_j x_j + b$ where b is a bias term. The loss could then be Mean Square Error (MSE) and ERM finds the optimal set of weights that describe all (x_i, y_i) with minimal loss:

$$\hat{w} = \underset{w}{\operatorname{argmin}} \sum_{i=1}^{n} (w^{\intercal} x_i - y)^2 \tag{2.4}$$

A more complicated model involves Artificial Neural Network (ANN)s, which are broadly one of the most popular methods of machine learning today. This model involves neurons that accept inputs, x, and a series of weights, w, and feed it through a nonlinear activation function, $\sigma(w^{\intercal} x + b)$. This activation function could take many forms like, for example, the sigmoid function: $\sigma(x') = \frac{1}{1+e^{-x'}}$. Ultimately, many layers of these functions can be connected ("feed-forward" if fully connected) creating a network that relates the input, x, to the eventual output, y.

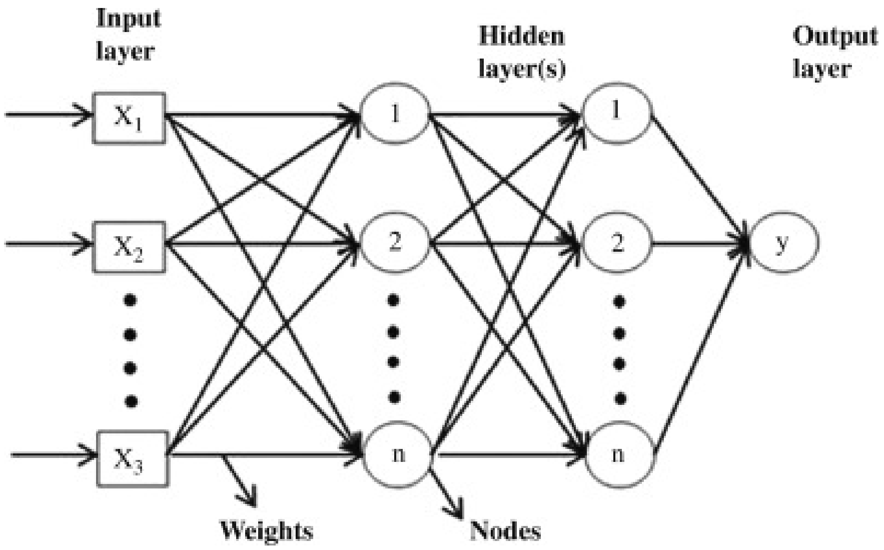


Figure 2.6: An abstract representation of a fully connected neural network architecture. Inputs are given by the vector x and passed to neurons, or nodes, through hidden layers with weights that are trained. The output is a collapsed prediction, y. (Image source: [13])

Take Figure 2.6 for example. Each circle in the hidden layer represents a neuron and each line is a weight linking the input or neuron to the next layer. The vast number of weights and neurons make these Deep Learning (DL) ANN models overdetermined, allowing them to capture nonlinear behavior in complex systems. Optimizing this massive system is possible thanks to the power of modern computing and involves methods like Stochastic Gradient Descent (SGD) via backpropagation (BP). A neuron in layer ℓ is calculated by its inputs and weights, which consists of the outputs from the previous neurons, which in turn is the result of its own inputs and weights, etc. This creates a chain of neurons and weights: $\sigma_\ell(w_\ell \sigma_{\ell-1}(w_{\ell-1} \ldots \sigma_1(w_1 x) \ldots))$. Calculating each neuron and optimizing weights is then a matter of calculating the chain rule derivatives starting with the last layer, ℓ, working backwards. Weights can be updated iteratively using gradient descent:

$$w_{t+1} = w_t - \mu \Delta_w f(w_t) \tag{2.5}$$

where μ is a hyperparameter, learning rate, that determines the magnitude of each update.

SGD is a stochastic process where the gradient Δ_w is approximated using randomly selected instances of x to calculate the update.

When training an ML model with an optimization algorithm like SGD, a training dataset of samples is used to compute loss (e.g. MSE in Equation (2.4)) and update network weights (e.g. Equation (2.5)) accordingly. This process is repeated for a specified number of epochs, which are full training iterations over all samples in the training dataset. Large training datasets can incur large losses and therefore lead to instabilities in weight adjustments. A system of batching can be implemented to mitigate this effect. Subdividing the training dataset into equal batches, the network's weights are then updated after each batch (and an epoch is completed after all batches have been processed once). The smaller optimization steps when batching introduce noise that would otherwise be averaged when using the whole dataset. This noise can help escape local minima in the commonly non-convex optimization spaces of ML (too small of a batch size and the noisy weight updates may be too gradual to converge). Optimal batch sizes are ill-defined and will depend on the dataset and model structures (multiple sizes can be tested or batch size can be applied as a hyperparameter).

2.3 A Survey of Semi-Supervised Machine Learning

SSML encompasses machine learning models that use both labeled and unlabeled data. This class of models sits between traditional supervised and unsupervised machine learning. The development of these methods was motivated by high-cost regimes for labeling. In some fields, labeling must be done manually and requires domain expertise. Both increase the cost of this time-consuming pre-processing step. When the cost is prohibitive, but data is plentiful, SSML attempts to alleviate this task. For a larger overview of SSML, refer to the textbook compiled by Chappelle et al. [14].

A pattern recognition model, $f(x, y)$, tries to approximate the joint distribution, $P(X = x, Y = y)$ for a given state space describe by the problem domain. The joint distribution is made of the conditional probability ($P(X = x \mid Y = y)$) and the marginal probability of the observable ($P(X)$). For an ideal model:

$$f(x, y) = P(X = x, Y = y) = P(X = x \mid Y = y) \times P(X) \tag{2.6}$$

Machine learning can broadly be defined under two target task groups. If the model only endeavors to understand the conditional probability ($P(X = x \mid Y = y)$) adequately enough to predict unlabeled patterns, or observations, in the state space, then it is a transductive method. In other words, a transductive method only concerns itself with the information necessary for classification, not the general rules governing data generation ($P(X)$). If, however, the model attempts to describe the joint probability ($P(X = x, Y = y)$) fully, it is an inductive method. Therefore, induction includes an intermediate task studying the marginal distribution ($P(X)$) for constructing a model to classify the entire space, not just the test or unlabeled points. Thus, an inductive method would be capable of generating new observables (x) given it successfully learned the general rules of generation from the marginal distribution. Either target task is valid for SSML.

To complete these tasks, machine learning employs two types of models. First, generative models try to learn the state space to accurately describe the statistical processes that produce the modeled samples. Also known as a regression, this is used to generate additional data instances for downstream modeling and simulation. This is equivalent to finding the classification function (i.e. inductive modeling) which captures the joint probability. Alternatively, statistical methods can be used for transductive tasks by creating discriminative models. These methods are not designed to generate new data instances but rather learn enough information to discriminate between classes of data in the state space. By creating a classification model, the conditional probability is measured instead of the joint probability.

2.3.1 Semi-Supervised Taxonomy

Several early surveys of SSML techniques [14] [15] focused on the distinction between generative and discriminative methods. More modern surveys have focused on the difference between inductive and transductive methods [16]. While significant improvements have been made to learning models that improve upon the methods mentioned above, methods that incorporate neural networks have shown increased interest in recent years. This is largely associated with the explosion of ANNs and DL in supervised regimes like image classification, Natural Language Processing (NLP), and computer vision. The organization of certain SSML models into buckets of inductive vs. transductive or generative vs. discriminative have blurred over time. Instead, advancements and gradual developments have made certain methods more flexible for different target tasks.

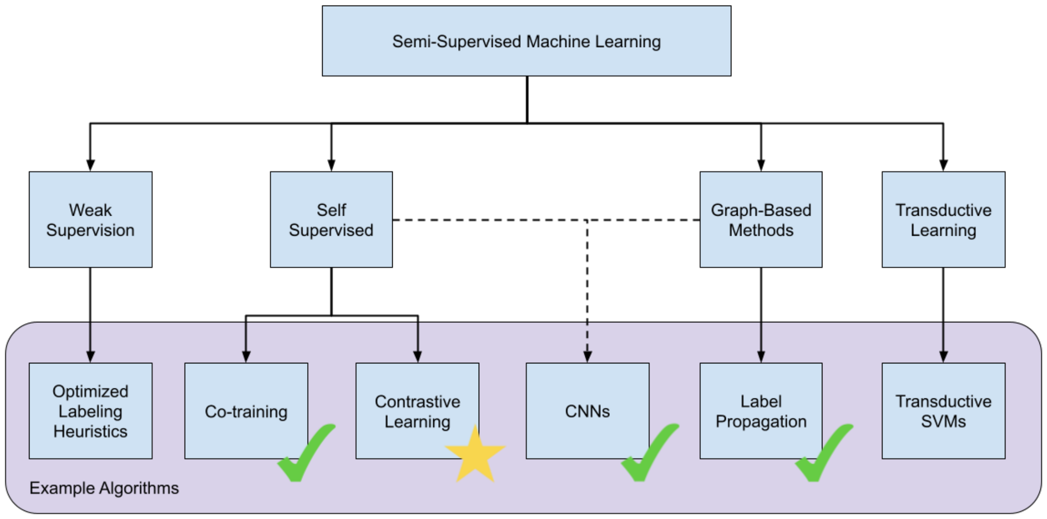


Figure 2.7: This is a non-exhaustive taxonomy of semi-supervised machine learning models grouped by their general algorithmic principles. Preliminary work includes a survey of models with potential applicability to nuclear nonproliferation (green check). Subsequent research focuses on contrastive learning, or self-supervision (gold star).

To avoid committing to these distinctions, a non-exhaustive SSML taxonomy is offered in Figure 2.7. Here, SSML principles are organized into broad categorizations with representative algorithms. For example, SSML methods include co-training and contrastive learning (more in Chapter 5) whereas modern CNN implementations are cross-cutting between many forms of SSML. Methods that have been implemented and studied below are marked with a green check. Contrastive learning, highlighted with a gold star, will be the focus of the following research. Weak supervision is a burgeoning class of methods that relies on domain-expertise to develop several noisy labeling heuristics and apply them to unlabeled data. Then, labeling techniques like majority vote can be used to apply labels to instances passed to downstream machine learning models. The whole pipeline is then optimized, including the voting importance of each heuristic.

Two methods listed in the taxonomy should be highlighted here: co-training and Label Propagation. Co-training serves as an early precursor to collaborating networks in which two (or more) models attempt to train on labeled data, passing this information on to unlabeled data and each other. Self-training then iteratively trains a model between steps of pseudo-labeling unlabeled instances. That is, train a model using labeled data ($f_1 : \mathcal{X}_L \to \mathcal{Y}_L$) and predict an unlabeled instance: $f_1 : x_{u_1} \to \hat{y}_{u_1}$. Then, include that newly labeled instance in the training data and train a new model, $f_2 : \mathcal{X}_{L+u_1} \to \mathcal{Y}_{L+u_1}$, repeat. For a more detailed overview of co-training, see Section 3.3.2.1.

Another example, Label Propagation, is an unsupervised technique in which graph-based relationships between instances are used to transductively label data: A notional distance between samples is computed with a distance metric. Unlabeled data are assigned a label from their closest labeled counterpart. Slowly, over iterations, labels are propagated from labeled to unlabeled data.

2.3.2 The Two Fundamental Semi-Supervised Assumptions

SSML relies on certain assumptions to learn a decision boundary. This is necessary for connecting information about classification from labeled data to knowledge about the underlying data distribution from unlabeled data. Without these assumptions, SSML is not guaranteed to benefit from additional unlabeled data over a traditional supervised model.

In fact, the relationship between labeled and unlabeled data may confound a model if labels are extrapolated to points that are uncorrelated, introducing further noise into the system.

The cluster assumption states that if two samples, x_1 and x_2, are close together, their labels should agree: $y_1 = y_2$. The notion of closeness is up to interpretation and depends on the unique data modality. This effectively relies on the reasonable assumption that the selected features do indeed describe the state space responsible for classification with some notion of smoothness, i.e. $f(x_0 \pm \epsilon)$ should exhibit the same response for some small noise, ϵ. A model cannot learn a decision boundary from a feature vector if the feature space has no correlation with class labels. This is connected to the smoothness assumption, which contends that classes of data should be clustered in a region of high-density, describable by that state space.

The manifold assumption maintains that the higher dimensional data lie on a lower dimensional manifold (a topological space). Therefore, the decision boundary that separates classes in this space may also be lower dimensional and thus not require the entire feature space to discern separation. For example, a 3-dimensional dataset may have a 2-dimensional representation that makes its classes separable. Both assumptions are illustrated in Figure 2.8. Each assumption supports scenarios in which unlabeled data can improve a learned decision boundary with added information.

T. Lu [17] argues that near certainty about "some non-trivial relationship between labels and the unlabeled distribution" is required for success with SSML. Otherwise, convergence will not be guaranteed. Singh, Nowak, and Zhu [18] take this further by quantifying the data distribution contexts in which SSML will converge faster or perform better than supervised methods. This is done within the perspective of the cluster assumption, but the definitive conclusion agrees with above: SSML will be more effective if a relationship exists, and classes are sufficiently distinguishable. Arguably this hold for nuclear cross-section data. Spectra that come from the same radiation source should exhibit the same radiation signature/photopeak. Any variation is the result of environmental effects and detector efficiencies

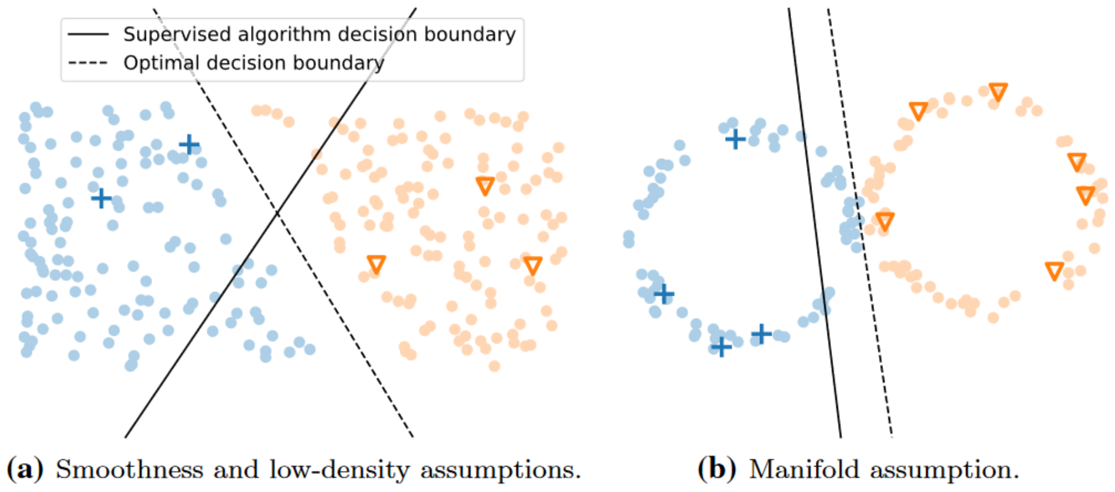


Figure 2.8: The two underlying SSML assumptions. In each plot, pluses and triangles are labeled instances, dots are unlabeled instances. Color (blue and orange) represent different classes. Note how the inclusion of unlabeled data in a ML model would improve its learned decision boundary. (Image source: [16])

like distance to source and the overall underlying background distribution. Spectra should contain the same labeling information regardless of whether they are labeled or unlabeled, excluding edge cases where the signature is barely discernible within a spectrum (like border samples from Singh et al. [18])

2.3.3 An Illustrative Example: Transductive Support Vector Machines

Consider, as an emblematic semi-supervised machine learning model, the Transductive Support Vector Machine (TSVM). First introduced by Gammerman, Vovk, and Vapnik [19] in 1998, this method has evolved to be called a Semi-Supervised Support Vector Machine (S^{3}VM) [20]. For binary classification, a given set of labeled pairs (x_i, y_i) can be separated by a hyperplane:

$$\begin{cases} wx_i - b \geq 1 & \text{if } y_i = 1 \\ wx_i - b \leq -1 & \text{if } y_i = -1 \end{cases} \tag{2.7}$$

For noisy data with a soft margin (no clear separation), a slack term is introduced to adjust the margin:

$$\begin{gathered} y_i((wx_i) - b) + \nu_i \geq 1 \\ \nu_i \geq 0 \end{gathered} \tag{2.8}$$

Therefore, the minimization problem involves regularizing the weights, w, and the slack term with a hyperparameter misclassification penalty, C:

$$\min_w \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\nu_i \tag{2.9}$$

Optimization results in a hyperplane that separates the labeled data along the maximum margin between the distributions: $\frac{1}{\|w\|}$. The hyperplane does not need to be linearly separable. Using a kernel as a feature map can lead to arbitrary dimensional planes of separation. To incorporate unlabeled data, numbering m, terms η and z are similarly used as soft margins that do not require a label, y_i. Instead, these are coupled constraints on unlabeled data, one for each binary classification, and minimization occurs alongside the original labeled data constraints:

$$\begin{aligned} \min_w C\big[\sum_{i=1}^{n}\nu_i + \sum_{j=1}^{m}\min(\eta_j, z_j)\big] + \|w\| \\ \text{s.t. } y_i(wx_i - b) + \nu_i \geq 1 \\ (wx_j - b) + \eta_j \geq 1 \\ -(wx_j - b) + z_j \geq 1 \end{aligned} \tag{2.10}$$

In summary for each soft margin, ν_i constrains labeled data (note it includes y_i), η_j constrains unlabeled data with a potentially positive label, and z_j constrains unlabeled data with a potentially negative label. The two additional constraints are soft margins on the unlabeled data and the eventual decision boundary applies labels on the unlabeled training data (i.e. transductive learning). This optimization problem is not necessarily convex, and significant research has been committed to improving this technique. Regardless, this semi-supervised model can consider both forms of data and possible reach a more optimal decision boundary. Take for example Figure 2.9, which is not an implementation of S^{3}VM but illustrates the point. The inclusion of unlabeled data may lead to a decision boundary that is more representative of the true statistical distribution even if those data lack explicit labeling information.

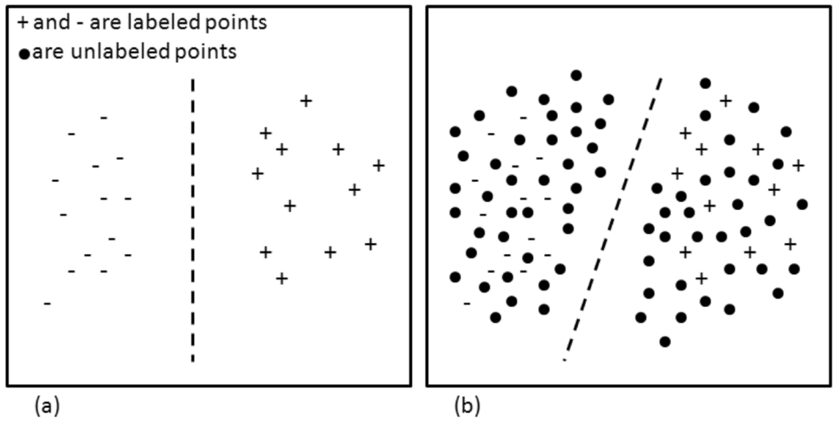


Figure 2.9: Nominal planes of separation for a supervised and semi-supervised Support Vector Machine (SVM). (a) Purely supervised SVM using only labeled points (positive and negative signs). (b) A TSVM that uses labeled points as well as unlabeled points (black dots). Note that including unlabeled points reveals more information about the plane of separation between the two classes of data. (Image source: [21])

2.4 Semi-Supervised Applications to Nuclear Engineering

Most applications of semi-supervised machine learning in nuclear engineering to date have been for fault diagnosis and transient identification. Ma and Jiang [22] utilize a graph-based SSML method. This is implemented in a self-training workflow, incorporating new data in future training iterations. Empirical success was shown in classifying experimental and simulated nuclear power plant (NPP) fault scenarios. Another example of SSML used for nuclear safety is Pinciroli et al. [23] where a more ad hoc system of feature extraction based on importance for characterization is used. Sun et al. [24] use a system of weak supervision to implement a CNN that applies pseudo-labels eventually included in future training. This results in a system for object detection that then is applied to nuclear waste.

Moshkbar-Bakhshayesh et al. [25, 26] use a two-step approach for identifying transients and then apply SSML to unknown transient classifications. First, a supervised machine learning model attempts to identify known transients. Unknown transients are then passed to a TSVM to predict unlabeled samples.

In general, these papers present implementations of SSML to nuclear engineering that demonstrate viability but lack context to nuclear nonproliferation. Implementations of SSML appear to be an emerging technology for nuclear nonproliferation applications. Most applications of machine learning in this space are supervised, which do not utilize the plethora of unlabeled data that may be available when labeling is costly. Therefore, SSML should be tested as a viable alternative to supervised machine learning techniques. This report, having identified the gap in nuclear engineering research, will study SSML applications to nuclear nonproliferation. In Chapter 3 and Chapter 6, SSML for nuclear nonproliferation applications will be developed, with a focus on data augmentations and SSL for nuclear radiation detection.

Chapter 3

Exploratory Work

This chapter introduces an experiment to demonstrate the value of semi-supervised machine learning to identifying SNM transfers using real-world data. Several SSML models are compared to a traditional supervised model. The machine learning methods used here discover underlying patterns in data useful for anomaly detection of SNM transfers. Previous implementations that utilize MINOS data have relied on anomaly detection using Sequential Probability Ratio Test (SPRT) with post-hoc data analysis [27] or physics-informed machine learning combined with data fusion [28].

Here, an alternative method with careful consideration of human and compute costs is offered. The model presented can detect anomalous radiation measurements and, when trained, uses a machine learning method to distinguish SNM transfers from other anomalous events. A hypothesis testing algorithm is implemented in order to identify anomalies. This test is also used to determine the domain-specific thresholds for anomaly detection. The results are fed into a labeling heuristic that applies a pseudo-label which is then used for training and testing various machine learning models. Overall, three SSML implementations are benchmarked against one supervised method. The results indicate an advantage in using an SSML method that utilizes unlabeled data when labeled data is limited.

3.1 Introduction

This work evaluates how semi-supervised machine learning can utilize information gained from both unlabeled and labeled data in pattern recognition algorithms for application in nuclear nonproliferation. In this application space, the goal is to optimally leverage large volumes of radiation data with limited ground-truth data. The efficacy of SSML models will be demonstrated on real-world data collected at ORNL containing transfers of shielded radiological material. Previous implementations that utilize MINOS data have relied on anomaly detection using a SPRT with post hoc data analysis [27] or physics-informed machine learning combined with data fusion [28].

Here, an alternative method with careful consideration of human and computing costs is offered. The model presented can detect anomalous radiation measurements and, when trained, uses a machine learning method to distinguish transfers from other anomalous events. A hypothesis testing algorithm is implemented in order to identify anomalies. This test is also used to determine the domain-specific thresholds for anomaly detection. The results are fed into a labeling heuristic that applies a pseudo-label that is then used for training and testing various machine learning models. Overall, three SSML implementations are benchmarked against one supervised method. The results indicate an advantage in using an SSML method that utilizes unlabeled data when labeled data are limited.

In summary, the main contribution of this work is an empirical demonstration of SSML as a tool for nuclear nonproliferation. The workflow introduced here utilizes all data collected, regardless of labeling status, without sacrificing performance but rather improving it over comparable supervised methods.

3.2 Methods

3.2.1 MINOS Data

The MINOS venture at ORNL collects multi-modal data streams relevant to nuclear nonproliferation. This is accomplished using a multimodal network of mature sensor technologies distributed at ORNL's campus surrounding two points of interest: Radiochemical Engineering Development Center (REDC) and High-Flux Isotope Reactor (HFIR). The reactor facility, HFIR, is used for scientific experiments (e.g., neutron scattering) and isotope production. Materials generated at HFIR are loaded into shielded casks and are transferred by a variety of methods (e.g. flatbed truck) to REDC. Once at REDC, the materials are unloaded, stored, and/or processed in, for example, hot cells.

Some of the material produced and processed at ORNL and detected by MINOS include [28]:

1. Unirradiated ^{237}Np targets used for ^{238}Pu production;
2. Irradiated ^{237}Np containing ^{238}Pu;
3. Unirradiated Cm targets used for ^{252}Cf production;
4. Irradiated Cm containing ^{252}Cf;
5. ^{225}Ac;
6. Activated metals;
7. Spent fuel.

The ultimate goal is to develop capabilities that distinguish and differentiate between shielded radiological material that might be present at the testbed. Material transportation can occur along several routes between facilities. Nodes in MINOS are distributed across the possible routes alongside the road. These nodes collect different forms of data including atmospheric conditions (e.g., temperature and pressure), video, audio, seismo-acoustic, and radiation.

Radiation data are collected using a network of NaI detectors—each designated as a node—that are distributed along the roads between HFIR and REDC (illustrated in Figure 3.1). This detector is capable of measuring gamma radiation emitted from nearby sources. Nodes are designed to take one measurement every second. This measurement is energy-dependent, binned in 1000 channels of 3 keV per bin. Energy calibration, which accounts

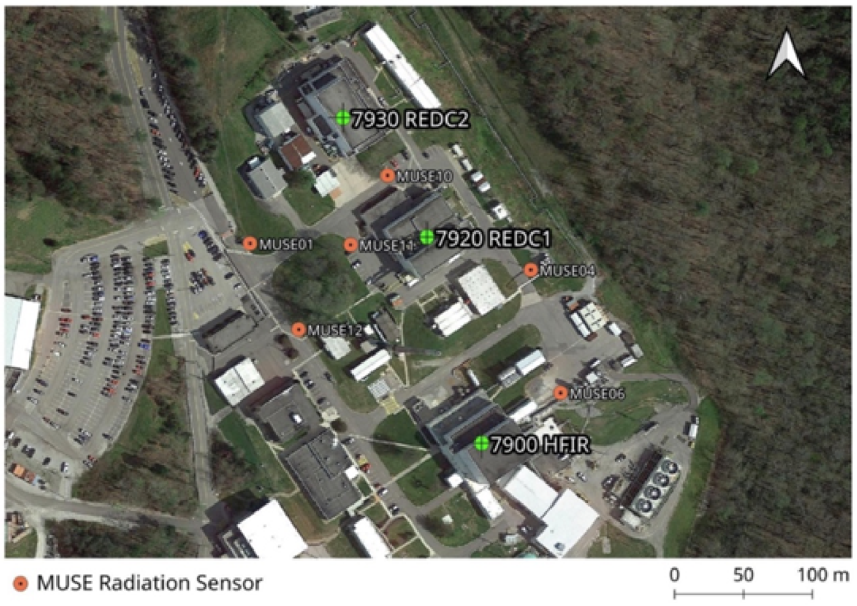


Figure 3.1: MINOS radiation sensor network distributed around HFIR and REDC, labeled. There are six radiation detectors in total (red), placed along roads used as routes for material transfers. (Image source: [28])

for gain drift in detector electronics, is completed before being shared for data analysis. Materials transferred around the MINOS testbed will serve as observables for developing and testing analysis methods.

3.2.2 Radiation Events

A transfer event occurs when a vehicle or source moves past a detector and thereby exhibits a response in that node. Material recently generated in HFIR will typically have high activity, which is why shielding is necessary for the safety of individuals handling the material. This shielding means that the signature's characteristic photopeak will not be observed by the detector. Instead, emitted gammas are scattered by the shielding material before reaching the detector. The NaI response will appear as a low-energy, downscattered continuum without the expected photopeak. If a background spectrum can be measured or approximated, this continuum should appear above the characteristic background. This is time dependent, and count rates will rapidly rise as the transportation vehicle approaches and fall as it drives past the detecting node.

Other radiation events can appear to be anomalous if they exhibit similar rapidity in count-rate change and must be accounted for in detection algorithms. For example, HFIR produces a large flux of neutrons when it is active. These neutrons can be captured by argon naturally in the air at a non-negligible rate. This produces ^{41}Ar, which is radioactive, undergoing β^- decay to stable ^{41}K while emitting gamma radiation with a photopeak energy of 1,294 keV. Another example occurs when it rains, which can "washout" radioactive ^{222}Rn with a half-life of approximately 3.8 days and is a result of the decay of ^{238}U. This weather pattern will exhibit elevated gamma radiation sourced from the radioactive daughters in this decay chain.

These unique environmental conditions will each contribute to the background radiation distribution with different intensities at different times. Exact characterization of the background spectrum is impossible without full knowledge of these conditions. Even a fully constructed background distribution for one measurement may not be transferable to a second measurement with different diurnal or seasonal variations. Here, an estimation of background is used if it is reasonable for a given time and state of a measurement.

3.2.3 Hypothesis Testing

First, a model is constructed to identify anomalous measurements from the temporally continuous radiation data stream. This model ingests energy-independent count rates for each minute of measurement. That is, the one-second, 1,000-bin measurements are integrated for all energies and every second in a one-minute window. This results in a "gross" count rate that is a raw measure of the number of gamma emissions counted by the NaI detector in that period. All of this data pre-processing occurs in RadClass—the software suite developed for this analysis—as shown in Figure 3.2.

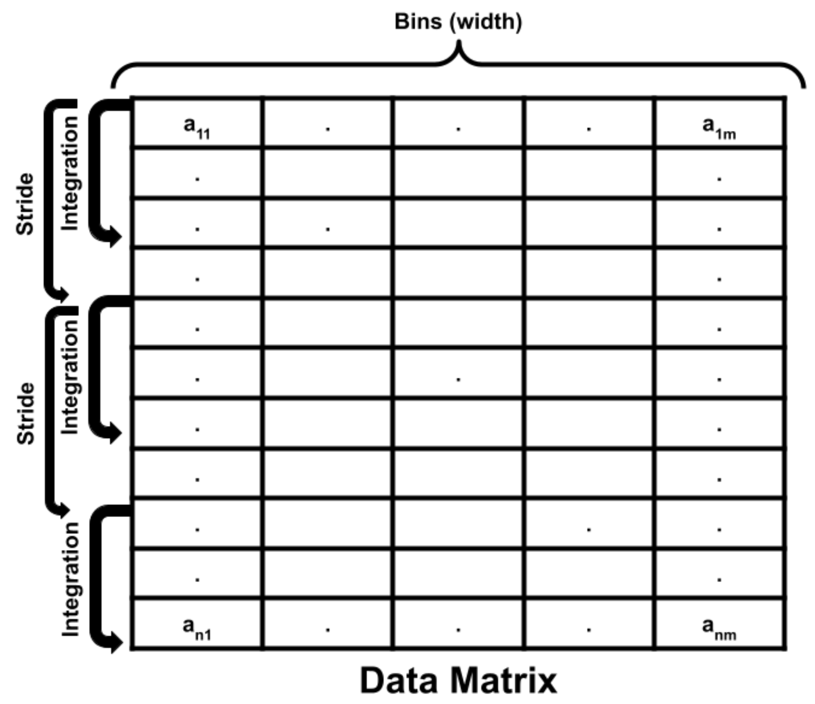


Figure 3.2: RadClass expects a standardized data format that can be abstracted to an n by m matrix with n temporal instances and m bins of data. For MINOS radiation measurements, this would be 1-second temporal instances and 1000 energy bins. For example, one month would be approximately a 43,200 $\times$ 1000 matrix. Given an integration time, RadClass will collapse and integrate a set of rows for every column. If the integration time is 60 seconds, every 60 rows will be integrated column-wise. An optional stride parameter could be defined to overlap or skip rows for integration.

A method of hypothesis testing is employed that was originally used in counting statistics [29]. Given two measurements, x_1 and x_2 (in this case, one-minute gross count-rates), taken at times t_1 and t_2, respectively, it is expected that they are each sampled from some statistical distribution. In the case of radiation counting statistics,

$$\begin{aligned} x_1 &\sim \text{Poisson}(\mu_1) \\ x_2 &\sim \text{Poisson}(\mu_2)\,, \end{aligned} \tag{3.1}$$

where μ_1 and μ_2 are the expected values of the Poisson distribution. If the magnitudes of x_1 and x_2 are approximately equivalent, then they likely come from the same distribution. For example, if that distribution is the background, then they can both confidently be labeled as coming from the background. However, if their magnitudes are appreciably different, then they should be labeled as anomalous, i.e., from different statistical distributions:

$$\begin{aligned} &H_0 : \mu_1 = \mu_2 \\ &H_1 : \mu_1 \neq \mu_2 \,. \end{aligned} \tag{3.2}$$

This anomalous behavior could be from a nuclear material transfer or from any other kind of radiation event off-normal for the defined background distribution. Anomalies depend on the rate of change between t_1 and t_2 to be recognized as having sufficiently varying magnitudes. The expected values, μ_1 and μ_2, are typically not measurable without enough unbiased samples. Suppose H_0 is true, then a mathematical translation can be made:

$$\begin{aligned} &n = x_1 + x_2 \\ &\mu = \mu_1 + \mu_2 \\ &p = \frac{\mu_1}{\mu_2 + \mu_2} = \frac{1}{2} \,. \end{aligned} \tag{3.3}$$

Therefore, comparing one measurement, x_1, with the sum of both measurements, n, behaves like a binomial distribution

$$x_1 \mid x_1 + x_2 \sim \text{Binomial}(x_1, n, \frac{1}{2}) \,. \tag{3.4}$$

If H_1 is true, then the binomial test above will fail to some significance level. This binomial test can be completed using MINOS data and identifies anomalous measurements in a temporal dataset. Each consecutive pair of measurements is tested, a p value is computed, and the null hypothesis is either accepted or rejected to some significance level. In practice, this means the testing framework evaluates by how much two measurements deviate from equivalence, illustrated in Figure 3.3. For a specified significance level, certain amounts of deviation are tolerated beyond which the null hypothesis is rejected, and an anomaly is identified. This significance level threshold acts as a hyperparameter and must be set via optimization using a ground-truth reference.

Using one node, one month of data (approximately 43,200 measurements) and ground truth for transfers, a Receiver Operating Characteristic (ROC) curve can be generated (see

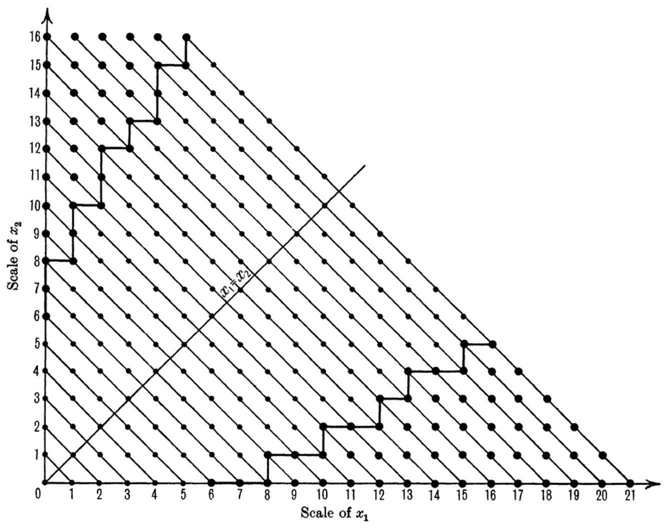


Figure 3.3: From [29], this represents the decision each hypothesis test applies for every x_1, x_2 pair. If they skew (i.e. one is larger than the other) by a significant amount, the null hypothesis is rejected. This is manifested by a significance level that sets a threshold on the resulting p-value, illustrated by the bolded step function on either edge of $x_1 = x_2$.

Figure 3.4). For an overview of ROC curves, see Appendix I. Here, a positive is defined as a prediction of a transfer event, and a negative is a prediction of some other anomalous event (for an introduction to confusion matrices, positives, and negatives, refer to Appendix I). First, the number of correct positive classifications can be expressed as the true positive rate: Recall $= TPR = \frac{TP}{TP+FN}$. The rate of positive misclassifications can be expressed as the false positive rate: $FPR = \frac{FP}{TN+FP}$. True positives are defined here as an anomalous measurement taken within 20 min of the timestamp for the event in ground truth or, absent a specific ground-truth timestamp, one anomalous measurement in a given day with a recorded transfer. This accounts for transportation and detector efficiencies that delay nuclear material transfers from being registered in a detector response for some time after the event may be recorded in ground truth.

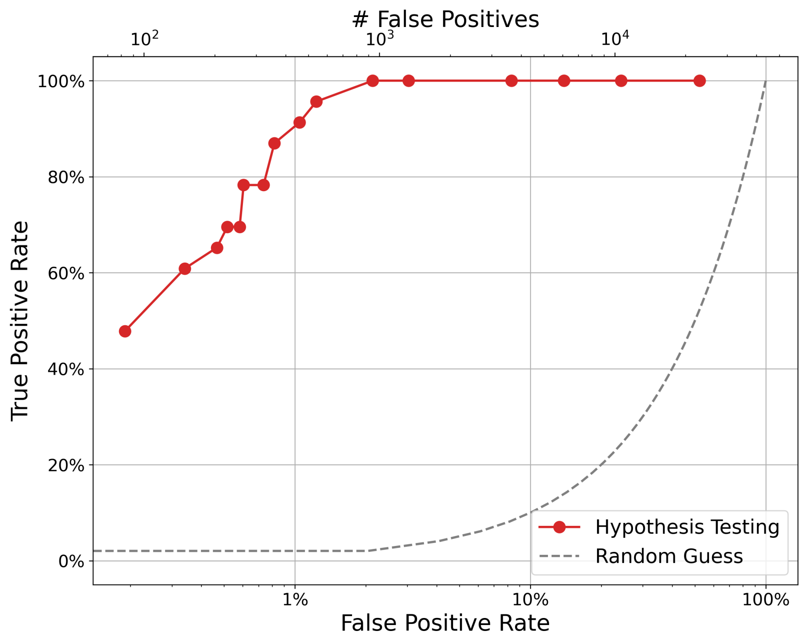


Figure 3.4: ROC for one node and one month of data using energy-independent, 1-minute count rates. Individual red points on the curve indicate different significance levels, which vary how strictly to enforce the measurement equivalence. As the significance level becomes larger, more and more null hypotheses are rejected, capturing more true and false positives. The gray dotted line indicates a 50–50 ratio, equivalent to an algorithm that makes a random guess for classification.

Note that the ROC curve sweeps over several values for significance level. An ideal significance level is one that maximizes true positives while minimizing false positives. However, true positives (i.e., material transfers) are rare in the dataset compared to the measurement frequency. This leads to an imbalance in the number of true positives and true negatives. That is, a percentage increase in the FPR is a larger increase in magnitude of false positives than a comparable percentage increase in the TPR.

For example, the tenth point (a significance level of 0.001) registers 22 out of 23 true positives ($TPR = 95.7\%$) with a FPR of only 1.2%. In gross terms, this false-positive rate corresponds to 540 one-minute false-positive measurements. The next significance level of 0.005 captures the last true positive at the cost of an additional 397 false positives ($TPR = 100\%$, $FPR = 2.1\%$). This imbalance must be considered in choosing a strict value for training and testing the models below.

The p value used for the analysis below is 10^{-20}, which is the strictest value tested with a true positive rate of only about 40% but is also the smallest false positive rate tested. Future experiments can loosen this restriction by using a larger significance level that increases the number of true positives and false positives and thereby furthers the need for event discrimination. The overall magnitude of false positives is very large in part because of noisy ground-truth labeling. A small significance level is chosen to avoid confounding the machine learning models tested below with a large number of false positives, especially since the ground truth will not be used moving forward. This further emphasizes the need to carefully discriminate between SNM transfers and other anomalous measurements in any downstream analyses.

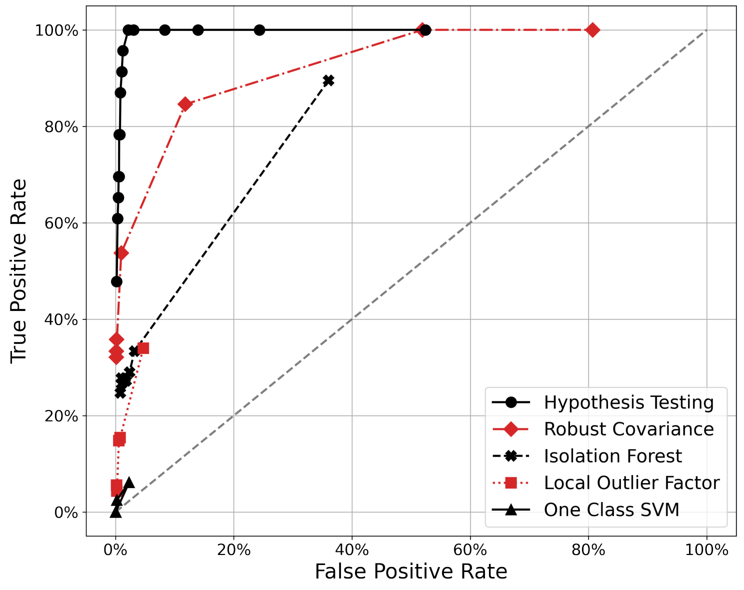


Figure 3.5: The hypothesis testing algorithm results from Figure 3.4 are compared to several outlier detections algorithms implemented in Scikit-learn. The gray dotted line indicates a 50-50 ratio, equivalent to an algorithm that makes a random guess for classification. Each algorithm is varied by one hyperparameter, with accuracies measured against the same ground-truth. Note that most algorithms capture significantly less true positives with increasing false positives.

The hypothesis testing algorithm can also be compared to other anomaly detection methods available in Scikit-learn. The results are shown in Figure 3.5. These models do not consider temporal dependencies in data like the algorithm above but instead attempt to quantify the typical response in the overall data distribution and draw a decision boundary for outliers. The models tested are One-Class SVM, Local Outlier Factor, Isolation Forest, and Robust Covariance. Each "blackbox" model generally performs worse than hypothesis testing across the ROC. Note that no hyperparameter optimization has been done on these models. Instead, one parameter is varied while the others remain fixed. Robust Covariance achieves the highest performance after hypothesis testing. Robust Covariance is the only model that assumes a Gaussian statistical distribution for the data. This likely explains the performance since this behavior mimics the Poisson expectation of hypothesis testing.

3.2.4 Labeling Heuristic

In evaluating the predictive accuracy of the machine learning models described below, three data subsets must be created: labeled training data, unlabeled training data, and labeled testing data. First, five months of energy-independent (gross count-rate) data from six different MINOS nodes are passed through the hypothesis-testing algorithm with a temporal integration time of one-minute. This provides a collection of spectra for timestamps at which the gross count rate was deemed anomalous (i.e., the null hypothesis was rejected). For this and all downstream tasks, the background is removed from the anomalous spectrum by estimation. It is assumed that for each event, a spectrum taken 20 min prior would constitute a typical background distribution absent any radiation events. This is subtracted from the event spectrum, resulting in a feature vector that notionally consists of only energy-dependent counts associated with the anomalous event (called the difference spectrum, illustrated in Figure 2.3).

Rather than using the ground truth to apply labels to the data, a labeling heuristic applies an automated "guess" that serves as a noisy label (i.e., with nonzero labeling error) for each sample. That guess could be a material transfer, ^{41}Ar event, Radon washout, or other anomalous behavior. The labeling heuristic uses Scikit-learn's FIND_PEAKS method to estimate the most prominent peaks in each spectrum. If one of those peaks appears within an energy range where signatures resulting from shielded radiological material transfers would be expected, and that peak is sufficiently prominent, the labeling heuristic assigns a material transfer guess to that sample. If not, a different label may apply. If none of the peaks are prominent enough to make a definitive guess, the spectrum is not labeled, and the sample is removed from the collection of anomalies.

In this way, the labeling heuristic applies a noisy label prediction to samples without relying on ground-truth information. This heuristic itself is not a sufficient model for discriminating between SNM transfers and other anomalous events because its accuracy is limited by an intrinsic error rate related to noisy labeling. The labeling heuristic is still better than random guessing because it applies limited domain knowledge related to expected behavior from material transfers and the detector response from MINOS. Therefore, these labels can appropriately be used for training and testing machine learning models because the uncertainty associated with noisy labeling is propagated to *each* model and is thus invariant when comparing models to each other. Any uncertainty associated with the labeling heuristic will be propagated to each model trained and tested on the resulting labels. Therefore, MINOS data can be used to train and compare ML models, avoiding reliance on ground truth for preparing training data.

The labels are organized for binary classification: material transfer or "other" anomalous measurement. The labeled corpus is then randomly split into small subsets that approximately maintain the binary class population ratio. The remaining majority are passed (as unlabeled data) with their label masked from any models or evaluation techniques. This simulates the real-world behavior of costly labeling in which it may be difficult to label a large amount of data, but a smaller amount is still feasible. Labeling these unlabeled data was not necessary (since the label was not stored) but it ensured that the labeled and unlabeled subsets had similar distributions of classes. The heuristic is summarized in Figure 3.6.

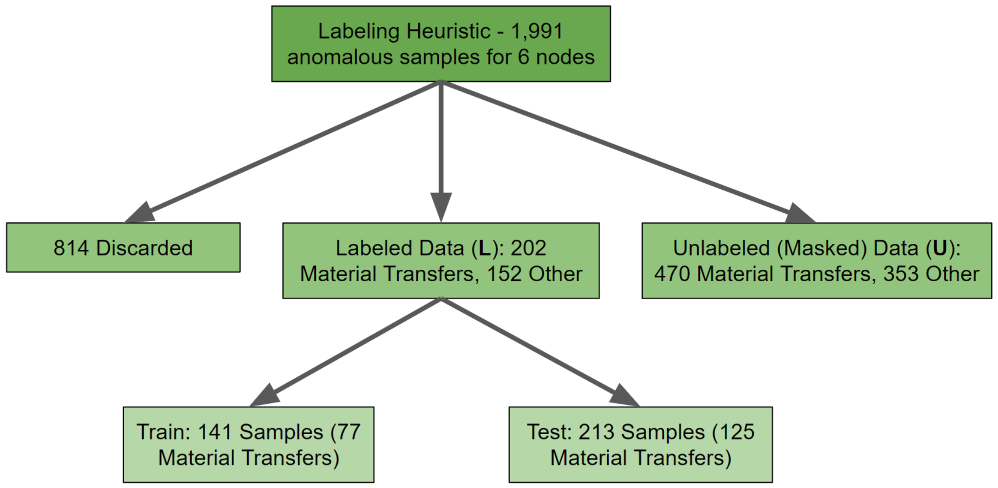

Labeling Heuristic - 1,991 anomalous samples for 6 nodes
814 Discarded
Labeled Data (L): 202 Material Transfers, 152 Other
Unlabeled (Masked) Data (U): 470 Material Transfers, 353 Other
Train: 141 Samples (77 Material Transfers)
Test: 213 Samples (125 Material Transfers)

Figure 3.6: The breakdown for sample splits between labeled training, labeled testing, and unlabeled data from the labeling heuristic. Note that using 5 months and 6 nodes worth of data results in 1991 anomalous measurements, where 814 are discarded since their label could not be resolved by the heuristic, and the rest are divided proportionally into train, test, and unlabeled subsets.

In summary, this provides three sets of data. The training data can be divided into two subsets: the labeled data, $\boldsymbol{L}$, and the unlabeled data, $\boldsymbol{U}$. A labeled test subset that is not used in training can be used for checking the precision of a trained model.

$$\mathbb{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_{|L|}, y_{|L|}), x_{|L|+1}, \ldots, x_{|L|+|U|}\} \tag{3.5}$$

3.3 Machine Learning Models

3.3.1 Supervised

Logistic Regression

Logistic regression serves as the baseline supervised model to be compared with semi-supervised machine learning. This model minimizes a loss function on labeled data:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(x_i^T w + c)) + 1)\,. \tag{3.6}$$

where w is a weight vector that predicts a label from some sample x_i with a bias term c. The model minimizes the disagreement between a predicted label $x_i^T w$ and the actual label y_i while minimizing the magnitude of weight values in w (L2 regularization). Logistic regression requires a fully labeled training corpus and cannot utilize unlabeled data in this form. Therefore, this model only trains on $\boldsymbol{L}$. This can be problematic in a costly labeling regime where labeled data are less voluminous.

3.3.2 Semi-Supervised

3.3.2.1 Co-Training with Logistic Regression

This first semi-supervised machine learning model extends the supervised logistic regression model above to handle unlabeled data. Two supervised logistic regression models are trained in tandem, passing information gained between each other, called co-training [30]. The logic is described in Algorithm 1. Each model learns some information from its training corpus so that it can predict an unlabeled sample, and it trades this information with its partner model. Ideally, the two models will converge to some increased level of accuracy because of shared information learned from each other (observed in Figure 3.7).

Algorithm 1 Co-training

```
Given: $\boldsymbol{L}$ and $\boldsymbol{U}$, split $\boldsymbol{L}$ between two logistic regression models ($h_1$ and $h_2$)
while len($\boldsymbol{U}$) $> 0$ do
    Train $h_1$ and $h_2$ on their portions of labeled data, $\boldsymbol{L_1}$ and $\boldsymbol{L_2}$
    Have $h_1$ predict $u_1$ and $h_2$ predict $u_2$, $\|u_1 + u_2\| \leq \|U\|/2$ sampled from $\boldsymbol{U}$.
    Include $u_1$ in $\boldsymbol{L_2}$ and $u_2$ in $\boldsymbol{L_1}$ with their predicted labels; remove from $\boldsymbol{U}$.
    Repeat training with this newly labeled unlabeled samples
end while
Evaluate on test set
```

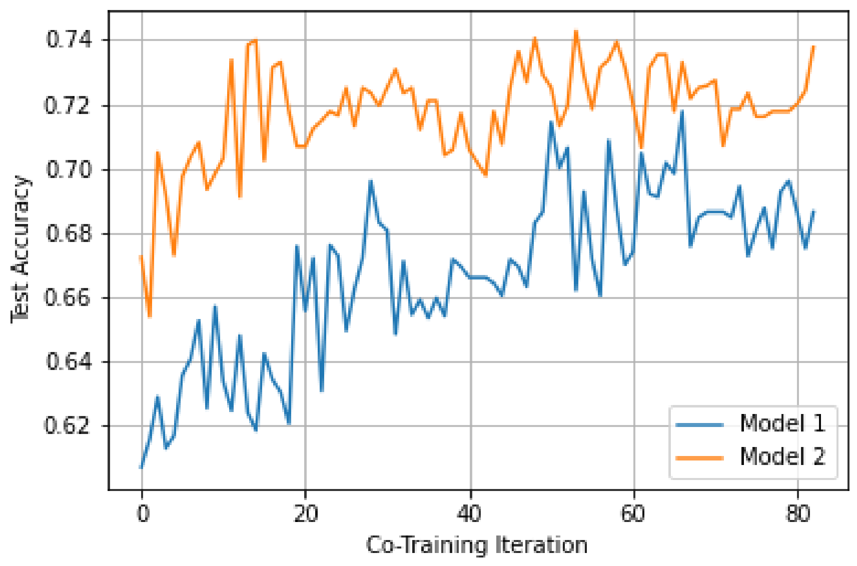


Figure 3.7: The test accuracy for each co-training model as trained using Algorithm 1. Test accuracy is defined as the percentage of correctly classified samples in a test set not used in training the models (refer to Equation (3.10)*). Ideally, both models would converge to a higher accuracy, indicating that information passed between models was helpful for learning and pattern recognition. A marginal increase is observed here.*

3.3.2.2 Label Propagation

Label propagation [31] is a transductive approach that acts like a semi-unsupervised model. This algorithm propagates labels from labeled data, $\boldsymbol{L}$, to unlabeled data, $\boldsymbol{U}$. Propagation is accomplished by comparing distances between data samples, whatever that notion of distance might be. First, data are arranged as a fully connected graph with edge weights defined by some kernel. In this experiment, a Radial Basis Function (RBF) is used:

$$w_{i,j} = \exp\left(-\frac{||x_i - x_j||^2}{\sigma^2}\right) . \tag{3.7}$$

where σ is essentially the standard deviation in the dataset or can be learned as a hyper-parameter. This is used to create a transition matrix, T, which updates the label matrix, Y, with dimensions $(|\boldsymbol{L}| + |\boldsymbol{U}|) \times C$ where C is the number of classes ($C = 2$ for binary classification). Each row is the probability of that sample being labeled with each respective class. If, for example, a row was associated with a labeled instance for the first class, its row would be $[1.0, 0.0]$. To reiterate, propagate using $Y \leftarrow TY$, normalize row-wise, and clamp any labeled rows to conform with their labeled classification instances. Afterward, an argmax can be applied to classify each row according to the most probable label.

There is a unique solution to this construction because it can be shown that the intialization of Y is inconsequential [31]. Therefore, there is no training per se if sufficient propagation iterations have occurred to converge (alternatively, the solution can be solved directly). This is also the only semi-supervised machine learning algorithm used in this work that is readily available in Scikit-learn. Only two other SSML models have been implemented: label spreading and a self-training classifier.

3.3.2.3 Convolutional Neural Network

An ANN is a form of machine learning model that is designed around networks of connected neurons consisting of unique activation functions that train for predictive classification by learning patterns in labeled data. These networks can be large or deep, which allows them to be flexible in pattern recognition but which requires copious amounts of data for training. In principle, CNNs are designed to pass filters over data and to convolve data structures for more robust feature representation. This operation is also shift invariant, making it effective for nonlinear patterns.

Shadow [32] is a Python package with loss functions that can be applied to various neural networks implemented in PyTorch. Originally designed for seismological data, these loss functions are data agnostic and include semi-supervised functions for processing unlabeled data in a neural network. Given some model, f_θ, the loss function is divided into two portions:

$$\mathcal{L}(f_\theta(x_l), y_l) + \alpha g(f_\theta, x)\,. \tag{3.8}$$

In this case, f_θ is a CNN and can predict a classification label for an instance, x_l, and $\mathcal{L}$ compares it to the actual label, y_l, where labeled data exist. The model can also be passed to a consistency-enforcing function, $g(\cdot)$, alongside any data instance (labeled or unlabeled) and penalizes differences given some perturbation. The consistency-enforcing function is weighted in the overall loss function by a hyperparameter, α. The loss function, $g(\cdot)$, used here is Exponential Averaging Adversarial Training (EAAT) [32]

$$g(f_\theta, x) = d(f_\theta(x + r_{\text{adv}}), f_\theta^{'}(x))\,, \tag{3.9}$$

where $d(\cdot)$ is a distance metric that describes the difference between its two inputs: the exponential moving average teacher, $f_\theta^{'}$, and its student, f_θ. Here, a small perturbation, r_{adv}, is added to the data. Utilizing the cluster assumption, a sufficiently small perturbation should not affect the predicted label since samples close together in state space should have similar classifications. The EAAT algorithm combines Virtual Adversarial Training (VAT) with Mean Teacher (MT). Virtual adversarial training chooses perturbations that produce the largest change in model output to enforce regularization. Mean teacher sets model weights using an exponential moving average, which is shown to produce more stable predictions.

In practice, both labeled and unlabeled data are processed by a CNN, optimized using SGD with a cross-entropy loss function, and employs the cluster assumption to better probe the decision boundary for classification. The initial neural network architecture for this model is based on an example structure for handling MNIST data [33], which comprise a collection of 2D images of handwritten digits. Instead of accepting 2D image inputs, the model here accepts 1D spectra. Figure 3.8 shows the initial CNN architecture prior to hyperparameter optimization. As the loss function is minimized, the model is optimized by maximizing the classification of labeled instances and enforcing consistency among unlabeled data (see Figure 3.9). The loss decreases over the training period. However, the variability—or noise—over epochs shows that certain instabilities may prevent further loss minimization or outright convergence.

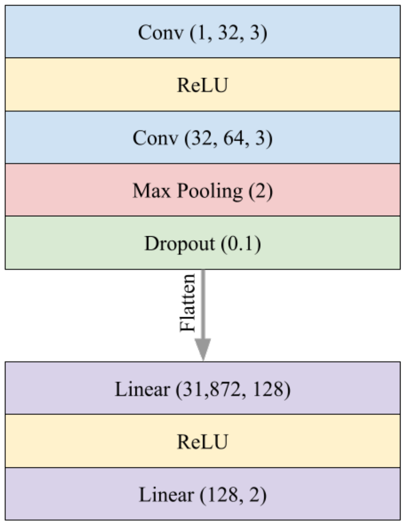
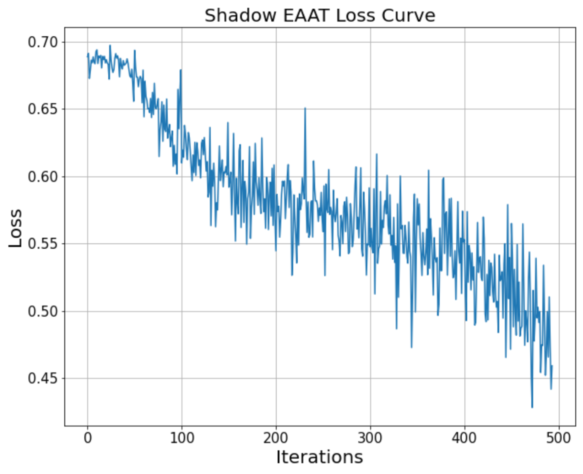
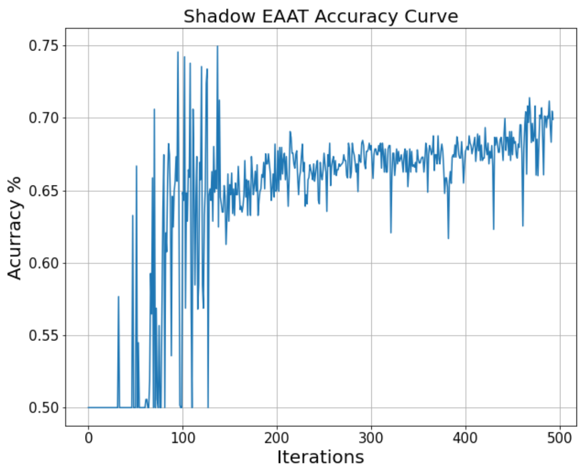


Figure 3.8: The base (prior to hyperparameter optimization) architecture for the CNN used by Shadow *EAAT. This consists of two convolution layers with max pooling and dropout, resulting in a representation that is passed to linear, connected layers ending in a binary classification prediction. This was offered as an example architecture for analyzing MNIST data [33] by* Shadow*, adjusted for accepting 1D spectra rather than 2D images.*



(a) Loss Curve



(b) Test Accuracy

Figure 3.9: The results of training for the best Shadow *model found via hyperparameter optimization. (**a**) The results of the loss function optimized during training: cross-entropy loss with EAAT optimized using SGD. (**b**) Accuracy on test data for every epoch. Accuracy notionally increases as the model is optimized, with early stopping resulting in a test accuracy greater than 70%.*

Table 3.1: Hyperparameter optimization balanced accuracy results.

Model	**Worst (%)**	**Best (%)**
Logistic Regression	63.9	67.7
Co-training	56.1	71.8
Label Propagation	55.1	77.1
Shadow EAAT	48.4	70.9

3.4 Results

Hyperparameter optimization was implemented using the Hyperopt [34] software package. This method uses Bayesian inference to explore the hyperparameter state space and to find the model parameters that maximize classification accuracy. Hyperparameters are chosen in a direction of expected accuracy increase. The loss function for optimization is the error rate, which Hyperopt attempts to minimize. Different hyperparameters are chosen over 100 tuning epochs. The resulting performance ranges are shown in Table 3.1. Local minima are possible for more complex models, but convergence typically occurs well before 100 tuning epochs.

The following metrics for binary classification—relating correctly classified instances, true positives (TP) and true negatives (TN), to misclassifications, false positives (FP) and false negatives (FN)—are used in evaluating a model's performance:

$$\begin{aligned}\text{Accuracy} &= \frac{TP+TN}{TP+FP+TN+FN}\\ \text{Balanced Accuracy} &= \frac{1}{2}\left(\frac{TP}{TP+FN}+\frac{TN}{TN+FP}\right)\\ \text{Precision} &= \frac{TP}{TP+FP}\\ \text{Recall} &= \frac{TP}{TP+FN}\,.\end{aligned} \tag{3.10}$$

Balanced accuracy (balanced for different class populations) will be used as a concise measure for comparing models. Several feature vector normalization options were also tested to maximize accuracy. Label propagation benefited from input features that were not normalized, since it measures distances between samples. Therefore, any notional distance exhibited by data samples would be affected by normalization, reducing label propagation's performance. Shadow significantly benefited from normalization against the distribution's mean and standard deviation, since the neural network's loss function is sensitive to large feature magnitudes. K-fold cross-validation was studied on logistic regression and co-training to ensure that there was no accuracy biasing because of random train–test splits. No appreciable difference was noted from different random splits.

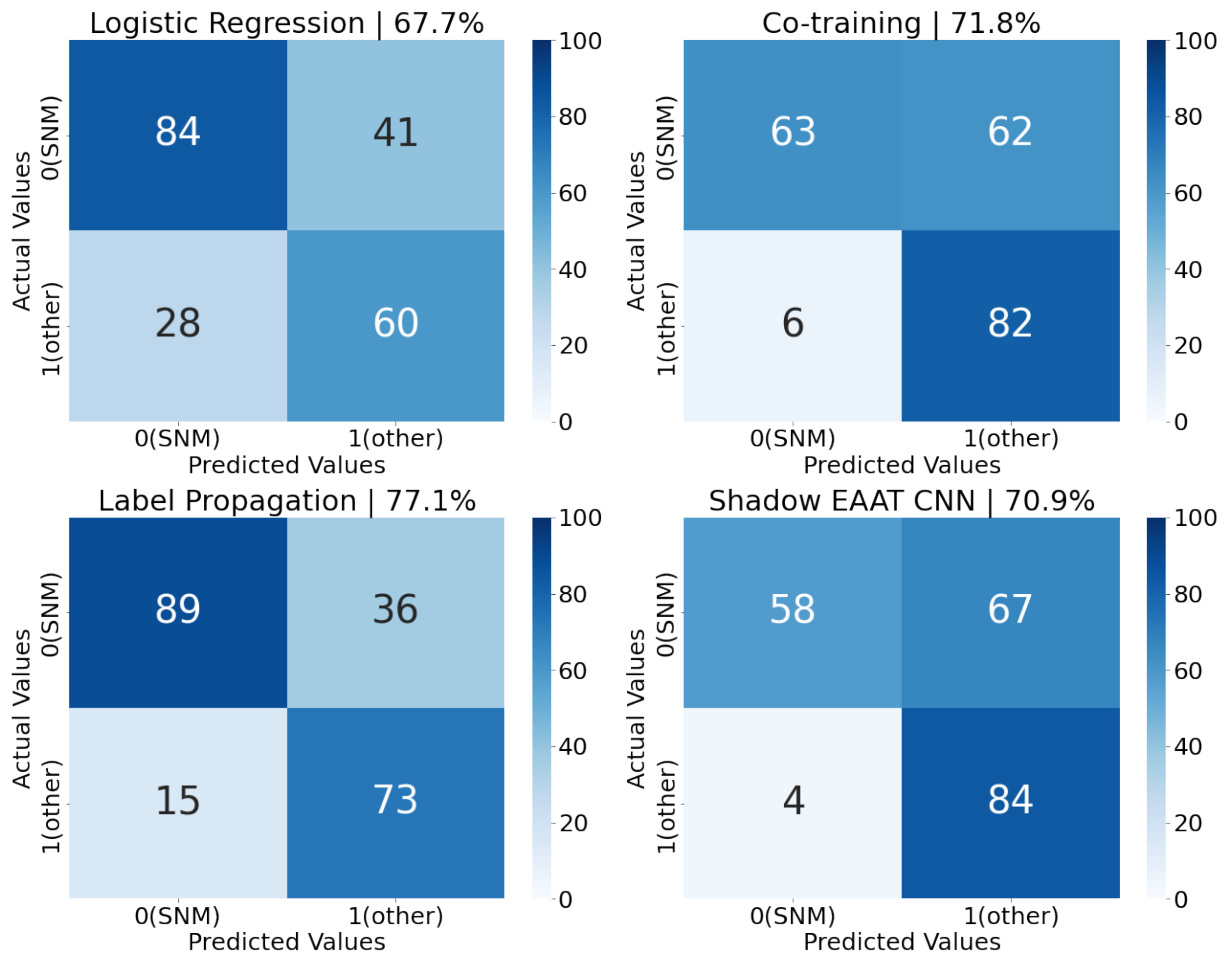


Figure 3.10: Confusion matrices on test datasets for each machine learning model. Note that the scores above for each confusion matrix are its respective balanced accuracy. SNM, class label 0, represents a positive transfer event of shielded radiological material, and other, class label 1, represents a negative.

Confusion matrices for each model's best balanced accuracy score (from Table 3.1) can be seen in Figure 3.10. The models, in order of increasing maximum balanced accuracy achieved, are: logistic regression, Shadow, co-training, and label propagation. None of the models tested predicted more false positives than false negatives, leading each to have a higher precision than recall. Co-training and Shadow, despite having comparable balanced accuracy scores, have exceptionally low recall (and very high precision) due to numerous false negatives, and only a handful of false positives. False positives are arguably less impactful than false negatives (i.e., recall is more important than precision) since detecting all instances of SNM transfers is the desired objective. In practice, the level of tolerance for false positives and false negatives is influenced by the needs of an end-user. A policy could be designed that weights these factors in model optimization or guides the deployment choice from a selection of trained models.

Several possible explanations exist for each model's accuracy. Logistic regression performs the worst of all four methods, suggesting that it is not generalizable to the test dataset. That is, this model has relatively simple complexity but is still capable of learning a decision boundary within the limited labeled training dataset. A more complex supervised model, such as a Multilayer Perceptron (MLP), could possibly generalize, but these methods are typically data hungry. This illustrates the challenge of using supervised models in this regime because a large, labeled data corpus is required to achieve high performance, but collecting such a dataset is costly (perhaps infeasible).

Co-training achieves a higher balanced accuracy because it utilizes previously unused unlabeled data. However, the increase in accuracy is likely limited by the structure of the data algorithm used. The authors of this method state that the two sets of labeled data, $\boldsymbol{L_1}$ and $\boldsymbol{L_2}$, must exhibit conditional independence. That is, $X_1 \perp X_2 \mid Y$, i.e., each subset contains predictive information for the classification but not for its counterpart. This is enforced so that each model learns different information to share. If $\boldsymbol{L_1}$ and $\boldsymbol{L_2}$ do not exhibit conditional independence, convergence to a higher accuracy is not guaranteed.

The data passed here are not necessarily conditionally independent. Only radiation data are used, and the measurements of all nodes are combined into one corpus. Two measurements from the same detector, and even for the same event, could be present in the training data for both models, breaking independence. Any increase in performance from training could be the result of information passed between models that is from different detectors and events. Co-training's accuracy could be increased by enforcing conditional independence and by carefully separating detectors between models. A more powerful implementation would utilize data fusion, a natural extension of MINOS. For example, one model could be trained on seismo-acoustic data, and one model could be trained on radiation measurements, both of which are coincident on SNM transfers but conditionally independent of each other.

Label propagation achieves the highest recall among the four models used (with the largest number of true positives classified). This is likely because label propagation is the only model to consider the geometric or distance relationship between data points, thus indicating that pattern information could be embedded in the data manifold. A manifold pattern also suggests some feature importance, or low-dimensional representation, which leads to eventual classifications. This agrees with physical intuition that radiation signatures are energy-dependent and should be unique between transfers of shielded radiological material and other types of radiation events.

Finally, SHADOW's performance was surprisingly the lowest among the semi-supervised models. SHADOW's CNN has many hyperparameters that can be further optimized, including the architecture of the network itself. The MNIST structure used here might not lend itself to spectral data. Rather, a network architecture such as those used for frequency/audio data could be used. The loss function may be too difficult to minimize for high-variance data such as radiation spectra. Training indicated that SHADOW can optimize to many local minima in its state space. Different loss functions could be applied, including different semi-supervised functions, that may train with more stability or converge to an optimal decision boundary. Overall, a CNN still requires significant amounts of data to train an accurate classifier. Despite the inclusion of unlabeled data, more labeled (and unlabeled) data could be required to find an effective decision boundary.

3.5 Conclusions

Nuclear material transfer detection methods that reduce computing and domain costs while maintaining detection performance can be an important component toward advancing nuclear nonproliferation. Models using SSML can alleviate the high cost of labeling radiation samples while still utilizing otherwise unused unlabeled data to build robust ML models for the detection of SNM transfers. The work presented here applied an anomaly-detection algorithm to analyze radiation data collected at the MINOS testbed and to distinguish radiation events from background. A labeling heuristic was designed to study the effects of noisy labeling on training data to reduce the reliance on ground truth or on unreliable prior knowledge. All the semi-supervised models tested performed with higher accuracy than a supervised logistic regression benchmark, indicating that it is possible to utilize labeled and unlabeled data in SSML models for detecting and characterizing SNM transfers. The increased accuracy of SSML models suggests that there is still valuable classification information about the data distribution present in unlabeled data. This warrants more detailed work to study the effects of learning on unlabeled data using more advanced systems of SSML techniques.

Chapter 4

Contrastive Machine Learning

This chapter provides the mathematical foundation for contrastive self-supervised machine learning underpinning the analytical method used in this work. A generic framework is presented which will be used in designing a contrastive machine learning model for identifying nuclear material transfers from radiation spectra. A derivation for the contrastive loss function—used to optimize a representation model's output—is detailed. Alternative contrastive machine methods and models are discussed. Finally, an important extension to contrastive ML, anomaly detection, and material transfers detection is explored: out-of-distribution detection. While not implemented in this work, out-of-distribution detection could be a powerful application of this analytical technique for nuclear nonproliferation.

4.1 Overview

Contrastive learning, or self-supervision more broadly, is a set of SSML methods that considers the similarity between data instances. Like other graph-based methods (e.g. Label Propagation), contrastive ML uses a notion of distances for making label predictions. Take the Simple Framework for Contrastive Learning of Visual Representations (SimCLR) framework [7] in Figure 4.1 for example. Given an instance, x, apply a transform $t \in \mathcal{T}$ to get an augmented instance $\tilde{x}$. These are fed into a model $f(x)$ that can produce some representation, h. A representation is similar to an encoded embedding from other forms of DL, but it is typically higher-dimensional to capture more of the abstract patterns and behaviors in $x, \tilde{x} \in \mathcal{X}$. $g(h)$ then uses the representation to output a prediction, z. This prediction depends on the target task. It could be another multivariate output like x (e.g. another image) or it could be a classification, assigning a predicted class to the input.

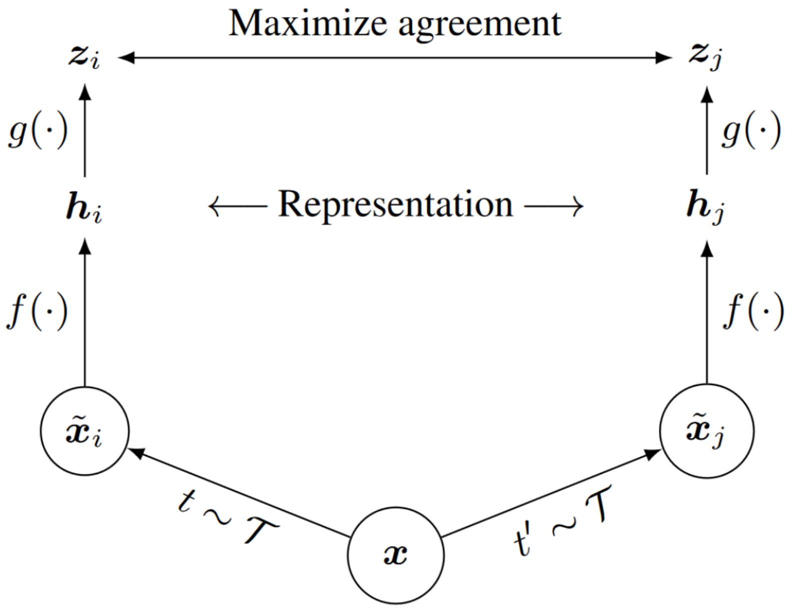


Figure 4.1: A contrastive learning model that uses two transformed instances of an original data sample x. Using a representation model $f(\tilde{x} = h)$ and a predictive model $g(h) = z$, positive pairs are trained to maximize agreement (i.e. contrastive learning). (Image source: [7])

Because each $\tilde{x}$ was created by transforming x, they should both have the same label because they contain the same intrinsic information and assuming the transformation is label invariant. The optimization task of contrastive learning is then to ensure that $z = g(h) = g(f(x))$ agrees for similar, or positive, pairs. These pairs could be the original instance, x, and its transformed value, $\tilde{x}$, or two augmented instances, $\tilde{x}_1, \tilde{x}_2$, from the same source instance. Negative pairs from dissimilar sources should have maximum disagreement. Therefore, the model contrasts between similar, augmented views of samples—which implies information about an instance's class—and dissimilar samples in a batch. Note that the true or observed label is not necessary for contrastive learning. The notion that two augmented instances are derived from the same original sample is all that is needed to define a positive pair since all three should agree on a subsequent label.

Dissimilar instances, or negative pairs, are only defined as such because they are not derived from the same original sample. However, it is possible that two instances, augmented from different samples, could share the same true label despite being defined as a negative pair. Contrastive training will attempt to construct representations generated by these two augmented instances to be maximally dissimilar since they are a negative pair. An oracle model (one with knowledge of true labels or the true decision boundary) would avoid making these representations dissimilar, but since the true label for augmented instances generated from unlabeled samples is unknown, these nominally negative pairs can occur and become a limitation on performance for contrastive models. Some work has attempted to address this directly with simultaneous contrastive attraction and repulsion [35]. Empirical success has been observed by using large batch sizes [7], which provide more augmented views of data in a single batch and thereby more (true) negative samples to contrast against.* For rare-event classes, the risk of contrasting samples with the same label would be rare for a large unlabeled dataset.

If labeled data is available, it can be used to regularize and fine tune the model (see Section 4.2 for more details). The models $f(x)$ and $g(h)$ are abstractly defined and could take on any structure that outputs a representation or prediction, neural network or otherwise.

The point of data augmentation is to allow a model to build intuition for valid class distributions and understand key features needed for classification. This process of data augmentation is typically defined as a pretext task. Four pretext tasks have been defined for contrastive learning on images [36]: color transformation, geometric transformation, context-based tasks, and cross-modal based tasks. Certain transformations are appropriate for certain pretext tasks [37]. Thus, it is important to consider similar pretext tasks for other data distributions. This work will will design transformations that naturally help more robust learning within a domain-specific pretext task that captures the principles of radiation detection.

Recent research [38] has shown that self-supervised models may have poorer performance

*This work uses a smaller batch size (512) than reported for SimCLR because the benefit of larger batch size was not well observed when tested (likely because the total unlabeled dataset size is smaller here than for SimCLR).

than supervised models when the downstream task is object recognition (e.g. image classification) but can outperform supervised methods in tasks for part-level recognition (i.e. any of retrieval, classification, or segmentation). While this success assumes sufficient labeled data for successful supervised training, the advantage of self-supervised models in part-level recognition reflects that representations highlight input feature patterns. Further, the projection head is shown to connect the part-aware representations produced by the representation model to the whole object. The distillation of part-aware representations to a lower dimensional representation that is used in contrastive training could successfully produce a model for either downstream task: object detection or part-level recognition [39]. More simply, the reduced dimensionality of projected representations reduces the curse of dimensionality associated with Euclidean and other distance metrics like cosine similarity. Even if self-supervised methods are weaker at object recognition (e.g. nuclear material transfer detection) with abundant labeled data, it seems reasonable to expect that meaningful representations accentuate features that would otherwise be occluded in raw spectral measurements and therefore difficult to learn when labeled data is scarce. This factor could be exploited in constructing and training downstream, task-specific classifiers.

4.2 Normalized Cross-Entropy Loss

The loss function used in SimCLR is a generalization of N-Pairs Loss, dubbed the normalized temperature-scaled cross-entropy loss (or NT-Xent). Elsewhere, this loss function is titled Info NCE [40, 41]. Given a set of data augmentations, samples are fed through a base encoder which produces a representation. There are no constraints on this encoder in SimCLR, varying strategies include producing higher-dimensional representations, an encoder/decoder pair, convolution followed by average pooling, etc. SimCLR observed that attaching a projecting head consisting of a single hidden layer with a ReLU activation function is beneficial for applying and optimizing contrastive loss.

The projection head reduces the dimensionality of the representations to what is fed to the contrastive loss function. The training algorithm for SimCLR is as follows. For a given batch of batch size N, produce two augmentations for each, resulting in $2N$ (augmented) samples. Each pair of augmented samples, $\{\tilde{z}_{i_1}, \tilde{z}_{i_2}\}$, deriving from the same sample (x_i) are considered a positive pair; the other $2(N-2)$ samples are assumed as negatives.

For two non-zero vectors, the Euclidean dot product is

$$\boldsymbol{A} \cdot \boldsymbol{B} = \|\boldsymbol{A}\|\|\boldsymbol{B}\| \cos(\theta) \tag{4.1}$$

where θ is the angle between the two vectors. Rearranging, the cosine similarity is then

$$\text{sim}(\boldsymbol{A}, \boldsymbol{B}) = \frac{\boldsymbol{A}^\intercal \boldsymbol{B}}{\|\boldsymbol{A}\|\|\boldsymbol{B}\|} \tag{4.2}$$

which varies in the range $[0, 1]$. This serves as a distance metric, or difference measure, between the two vectors (i.e. two input samples). NT-Xent loss is then the cross-entropy of cosine similarities for the projected representations of positive pairs (z_{i_1}, z_{i_2}), normalized over all pairs

$$\begin{aligned} \ell(z_{i_1}, z_{i_2}) &= -\log \frac{\exp(\text{sim}(z_{i_1}, z_{i_2})/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[z_k \neq z_{i_1}]} \exp(\text{sim}(z_{i_1}, z_k)/\tau)} \\ &= -\log \big[\exp(\text{sim}(z_{i_1}, z_{i_2})/\tau) \big] + \log \big[\sum_{k=1}^{2N} \mathbb{1}_{[z_k \neq z_{i_1}]} \exp(\text{sim}(z_{i_1}, z_k)/\tau) \big] \end{aligned} . \tag{4.3}$$

Note that the denominator sums over the first input, z_{i_1}, paired with all other samples but itself. If, for example, $z_{i_1} = z_{i_2}$, then their cosine similarity will be 1, and (assuming $\tau = 1$) the first term of Equation (4.3) will be -1. If $z_{i_1} \perp z_{i_2}$, then their cosine similarity will be 0, and the first term of Equation (4.3) will be 0. In practice, many negative pairs will not be exactly orthogonal to the anchor point (i.e. the first input) and therefore the second term will be nonzero, meaning the minimum will be nonzero even if the positive pair is optimally identical (remember that the objective is to minimize loss). However, the additional nonzero loss for negative pairs encourages them to be maximally dissimilar (i.e. assuming $\tau = 1$, an optimal solution would have $\text{sim}(z_{i_1}, z_{i_2}) = 1$ for positive pairs and $\text{sim}(z_{i_1}, z_{i_2}) = 0$ for negative pairs). Therefore, minimizing the loss function achieves two goals: maximizing similarity between positive pairs and minimizing similarity between negative pairs.

The temperature coefficient, τ, acts as a fixed hyperparameter scaling the magnitude of similarities. Finally, the loss is averaged for all positive pairs in each direction[†]

$$\mathcal{L}_N = \frac{1}{2N}\sum_{i=1}^{N}[\ell(z_{i_1}, z_{i_2}) + \ell(z_{i_2}, z_{i_1})]. \tag{4.4}$$

Even though cosine similarity is a symmetric distance metric (i.e. $\text{sim}(z_{i_1}, z_{i_2}) = \text{sim}(z_{i_2}, z_{i_1})$), $\ell(z_{i_1}, z_{i_2}) \neq \ell(z_{i_2}, z_{i_1})$ because the denominator of Equation (4.3) will change depending on the anchor point of $\ell(\ldots)$.

Semi-supervised contrastive loss computed for labeled data can be used to train a representation model if labeled data is included in the training corpus. Additional loss terms are computed when more than one sample of the same class is passed, since the labels are known for these samples and therefore multiple positive pairs can be generated. For example, two labeled spectra of the same class generate four augmented samples or six positive pairs, for a total of 12 loss terms ($\ell(\ldots)$), as opposed to the four loss terms from two positive pairs generated by two samples of different (or unknown) classes. Semi-supervised contrastive loss acts as regularization terms (scaled by a hyperparameter, α) that guides the representation model to produce representations corresponding to the labeled data, introducing context to the downstream domain specific task. Accounting for additional loss terms, semi-supervised contrastive loss loops over the labeled training corpus, M,

$$\begin{aligned}\mathcal{L}_M = \frac{\alpha}{2}\sum_{j=1}^{M}\frac{1}{2|K_j|-1}\Bigg[&[\ell(z_{j_1}, z_{j_2}) + \ell(z_{j_2}, z_{j_1})] \\ &+ \sum_{k\in K_j}\mathbb{1}_{[k\neq j]}[\ell(z_{j_1}, z_{k_1}) + \ell(z_{j_1}, z_{k_2}) + \ell(z_{j_2}, z_{k_1}) + \ell(z_{j_2}, z_{k_2})]\Bigg]\end{aligned} \tag{4.5}$$

where K_j is the set of indices for all training spectra that share the same label with z_j (i.e. $L_k = L_j$). The first two loss terms are the same as before: contrastive loss between the

[†]The software implementation conducts a sub-batching routine that computes the loss for 64 positive pairs in a batch at a time. This is functionally equivalent, but circumvents memory limitations for larger batch sizes.

two augmented samples from the same original training spectrum. The other loss terms compute the contrastive loss between (augmented) spectra sharing the same label. Note that because the denominator of Equation (4.3) sums over all augmented samples from the training dataset, the sum there would now be through $2(N + M)$.

For intuition, imagine a square matrix where every column and row correspond to an augmented sample, all of which share the same label. Elements of the matrix correspond to the contrastive loss for the positive pair of that row and column. Since augmented samples are not contrasted against themselves, the diagonal is zero. Therefore, the total number of loss terms, including the reverse of each positive pair, is $(2k)^2 - 2k = 2k(2k - 1)$. The total loss when labeled data are included is

$$\mathcal{L} = \mathcal{L}_N + \mathcal{L}_M. \tag{4.6}$$

4.3 Alternative Contrastive Methods

Several different contrastive and self-supervised models have already been proposed. SimCLR [7] does not use the original sample image (here, a spectrum) but rather trains a model using only augmented pairs. Alternatively, a student-teacher method [42] could compare an augmented sample versus its untransformed parent. Some implementations [43] train a smaller, distillation model to effectively generalize. Others [44] create a stop-gradient so that optimization does not occur along augmented instances. One implementation [45] includes spectral clustering to avoid a requirement for conditional independence and more closely consider class similarity.

Empirical results [46] suggest that self-supervised learning frameworks like contrastive learning above can be robust to class imbalance in select data streams. This is likely because the representations learned are strong enough to discriminate between different types of class distributions. Even though the experiments described above conduct binary classification as an ultimate downstream target task, these models would be beneficial for multinomial classification. SNM transfers tend to be rare in the MINOS data and potentially non-existent for certain real-world monitoring applications. If so, a model needs to be tuned and prepared for severe class imbalance to the point of anomaly detection.

Contrastive learning, and self-supervised machine learning broadly, does not intrinsically require labeled data to optimize a model. Contrastive loss functions typically maximize agreement between positive pairs, where positive pairs are defined either as two augmentations of the same data instance or one augmented instance compared with the original. Optimization in this way can be successfull if the augmentations—given a known set of transformations—are not severe enough to obscure labeling information present in the unaltered observation. If labeled data is available, fine-tuning can be used to "check answers" in model training or more importantly define the number of classes possible in the dataset. Semi-supervised has an advantage over supervised learning because it can handle unlabeled data and extrapolate labels via known classes.

How else could labeled data be used in SSL? Typical SSML models are rooted in supervised methods with consistency enforcing functions added to the loss term. Self-supervision is almost the opposite, reminiscent of unsupervised methods. One such example trains a larger model contrastively and then prunes the model, or makes it smaller, by a process of supervised fine-tuning [47]. Another example, supervised contrastive learning [48], uses labeled data as anchor points for the broader data distribution, further defining positive pairs and adding generalization weight to contrastive learning. Finally, another example uses the similarity between samples to gradually group classes: weakly supervised contrastive learning [49].

Chapter 5

Spectral Augmentations

The SimCLR framework utilizes mature augmentations that have been designed for image classification, illustrated in Figure 5.1. Each label-invariant transformation exploits certain properties of an image and image generation. For example, a cutout (g) of variable size could be applied to an image of a dog. If the augmentation is not overly severe, enough contextual information should still be present in the image to label it as a dog. If any augmentation is applied too heavily, it is possible that the labeling information becomes indistinguishable. The set of augmentations are arranged such that an image's perturbed counterpart could plausibly be observed by real-world measurement.

The combined use of (randomly applied) augmentations expands a finite training dataset and helps a contrastive model learn semantic labeling information that can be used in downstream detection tasks. While SimCLR focuses on image representation, these principles are not limited to a specific data modality. This chapter details data augmentations specifically designed for gamma spectra and their use in contrastive machine learning.

5.1 Mathematical Background

First, a formalized derivation of spectral properties should be established. Assume that an individual spectrum can be decomposed into three constituent parts

$$\boldsymbol{x} = \boldsymbol{s} + \boldsymbol{b} + \boldsymbol{p}. \tag{5.1}$$

That is, a spectrum ($\boldsymbol{x}$) consists of Gaussian-shaped signals ($\boldsymbol{s}$) specifically associated with various radioactive sources present during measurement, a persistent and characteristic back-

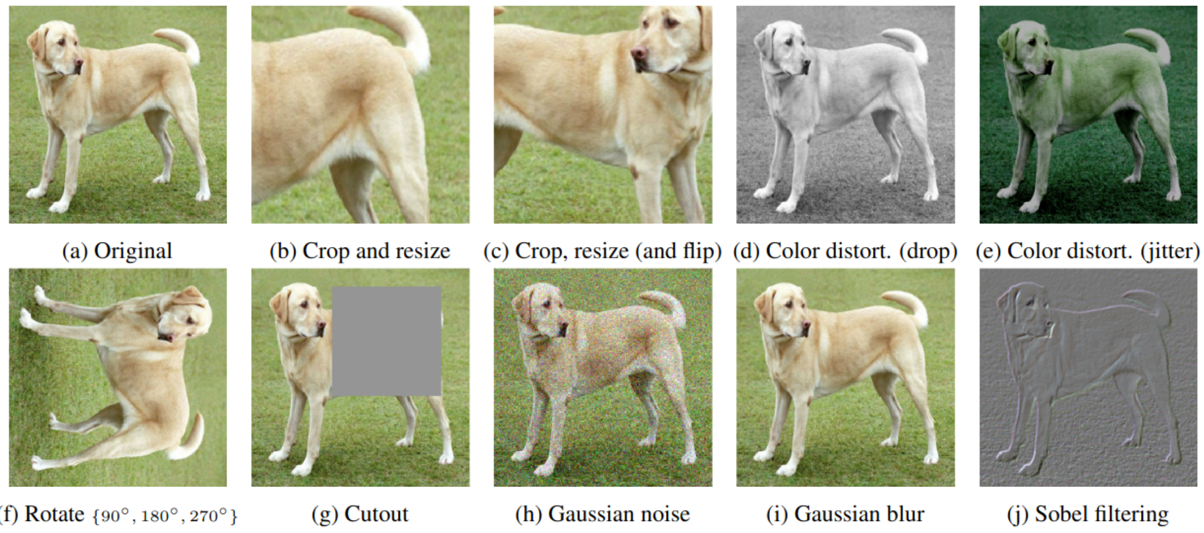

(a) Original
(b) Crop and resize
(c) Crop, resize (and flip)
(d) Color distort. (drop)
(e) Color distort. (jitter)
(f) Rotate {90°, 180°, 270°}
(g) Cutout
(h) Gaussian noise
(i) Gaussian blur
(j) Sobel filtering

Figure 5.1: A grouping of each type of valid image augmentation that will maintain labeling information for classification given a reasonable degree of transformation. (Image source: [7])

ground distribution ($\boldsymbol{b}$), and Poissonian noise ($\boldsymbol{p}$). The spectra in this analysis consist of vectors with 1000 components corresponding to the 1000 energy bins (3 keV each) collected during measurements. Each augmentation below manipulates at least one of a spectrum's underlying components or an individual feature in that component.

It is necessary for some augmentations to first estimate the background distribution ($\boldsymbol{b}$) before transforming the spectrum. This can be a difficult to manually extract—even for an SME—because background radiation is highly dependent on environmental factors that are unique for every measurement. For example, in a given Region of Interest (ROI) ($[x_i, x_j]$) where one signature is present, a common procedure is to fit the curve

$$\begin{aligned}\boldsymbol{x} &= \boldsymbol{s} + \boldsymbol{b} \\ &= \frac{A}{\sigma\sqrt{2\pi}} \exp\Big(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\Big) + mx + c\end{aligned} \tag{5.2}$$

where the additional fit parameters are the peak's amplitude (A), the peak's standard deviation (σ, related to the peak's width), the peak's centroid (μ), the background's linear slope (m), and the background's zero-point (x). This assumes an ROI sufficiently small enough to capture the peak and its shoulders, where the present background is approximately linear. Equation (5.2) omits the noise associated with counting statistics for detection setups

Table 5.1: Optimized BEADS Hyperparameters.

Parameter	Value
f_c	4.749×10^{-2}
r	4.1083
d_f	2
λ_0	3.9907×10^{-4}
λ_1	4.5105×10^{-3}
λ_2	3.3433×10^{-3}

($p_i \sim \text{Poisson}(x_i)$) because modeling this noise could result in overfitting *if* successful. This fit procedure is used below when estimating signal peak parameters for manipulation.

Instead of measuring ROI fit parameters directly, an entire spectrum could be decomposed using baseline estimation and denoising using sparsity (BEADS) [50]. BEADS uses a method of matrix decomposition—essentially a low-pass and high-pass filter—to separate a spectral measurement (originally intended for chromatography) into its components by modeling it as a sparse signal with sparse derivatives. In summary, the BEADS fit parameters for a typical NaI detector can be learned via hyperparameter optimization and defaults for this analysis are chosen as such (see Table 5.1 for reference values). An example decomposition of a spectrum using BEADS is shown in Figure 5.2. See below for a more detailed derivation of BEADS:

Derivation 1. *For an observed spectral vector $\boldsymbol{x}$, model it as consisting of three components,*

$$\begin{gathered} \boldsymbol{x} = \boldsymbol{s} + \boldsymbol{b} + \boldsymbol{p} \\ \boldsymbol{x} \in \mathbb{R}^N. \end{gathered} \tag{5.3}$$

Here, $\boldsymbol{s}$ is a vector consisting of numerous peaks (photopeaks and their associated mechanisms in nuclear radiation spectroscopy), $\boldsymbol{b}$ is a smoothly varying baseline signal (i.e. low-pass), and $\boldsymbol{p}$ is Poissonian noise. For a given estimate of $\boldsymbol{s}$ (i.e. $\hat{\boldsymbol{s}}$) a sparse-derivative signal can *be reconstructed by reconstructing the baseline*

$$\begin{aligned}\hat{\boldsymbol{x}} &= \hat{\boldsymbol{b}} + \hat{\boldsymbol{s}} \\ &= \boldsymbol{L}(\boldsymbol{x} - \hat{\boldsymbol{s}}) + \hat{\boldsymbol{s}} \\ &= \boldsymbol{L}\boldsymbol{x} + \boldsymbol{H}\hat{\boldsymbol{s}}.\end{aligned} \tag{5.4}$$

The high-pass filter, $\boldsymbol{H}$, is related to the low pass, $\boldsymbol{L}$, via an identity matrix: $\boldsymbol{H} = \boldsymbol{I} - \boldsymbol{L}$. Since the reconstructed sparse-derivative signal and baseline rely on the estimated peak signal, that can be found via the difference with the observed

$$\|\boldsymbol{x} - \boldsymbol{L}\boldsymbol{x} - \boldsymbol{H}\hat{\boldsymbol{s}}\|_2^2 = \|\boldsymbol{H}(\boldsymbol{x} - \hat{\boldsymbol{s}})\|_2^2. \tag{5.5}$$

The high-pass filter is defined as a combination of two banded convolution matrices [50]. Therefore, BEADS reconstructs the peak signal by optimizing a penalty function that encourages a positive reconstruction by penalizing negative $\boldsymbol{x}$ values

$$\hat{\boldsymbol{s}} = \underset{\boldsymbol{s}}{\operatorname{argmin}}\{F(\boldsymbol{s}) = \frac{1}{2}\|\boldsymbol{H}(\boldsymbol{x} - \boldsymbol{s})\|_2^2 \lambda_0 \sum_{n=0}^{N-1} \theta_\epsilon(s_n; r) + \sum_{i=1}^{M} \lambda_i \sum_{n=0}^{N_i - 1} \phi\|[\boldsymbol{D}_i \boldsymbol{s}]_n\|\}. \tag{5.6}$$

Here the penalty function, ϕ, is defined to be symmetric with a first derivative, λ's are regularization parameters, $[\boldsymbol{D}_i \boldsymbol{s}]$ are diagonalized versions of $\boldsymbol{x}$, θ is a convex function

$$\theta = \begin{cases} x, x > \epsilon \\ f(x), |x| \leq \epsilon \\ -rx, x < -\epsilon \end{cases} \tag{5.7}$$

and r is a positive parameter. With an estimated signal vector, $\hat{\boldsymbol{s}}$, a baseline can be estimated, $\hat{\boldsymbol{b}}$, and the residual noise is $\hat{\boldsymbol{p}} = \hat{\boldsymbol{x}} - \boldsymbol{x}$.

BEADS does not necessarily converge to a correct decomposition, and in fact can be troubled by the low signal-to-noise ratios exhibited by NaI detectors. This means that the peak shapes and amplitudes and the precise baseline distribution can be quantitatively incorrect when compared to traditional radiation detection methods. This system is sufficient for a method of rough background estimation and requires less computation compared to a manual analysis.

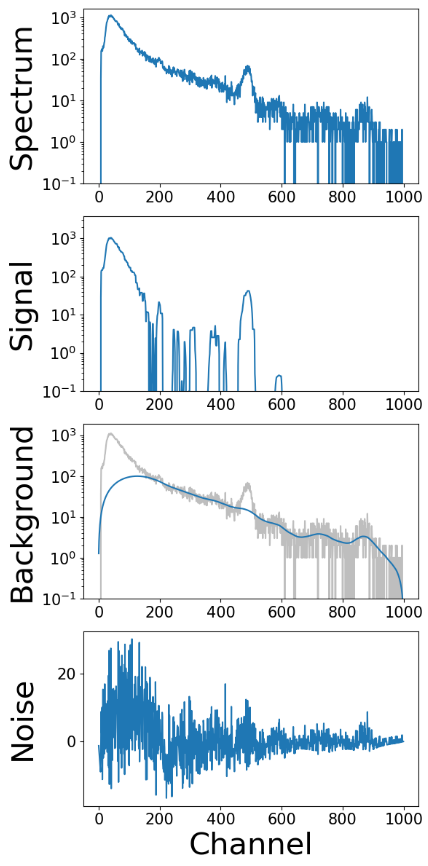


Figure 5.2: An example decomposition of a spectrum using BEADS. The original (top) consists of peaks (signal) resting on a slowly varying baseline distribution (background) perturbed by some fluctuations (noise) associated with counting statistics.

5.2 Data Augmentations

The first step in a self-supervised pipeline is data augmentation, which employs transformations that will be used to learn representations relevant for downstream tasks, like classification. While appropriate data augmentations might be mature for image classification, valid transformations for spectroscopic data must be developed. If physically viable transformations can be constructed, then self-supervision can be used to process unlabeled spectra and create a classifier with low labeling cost. Detailed here are transformations designed to naturally aid in robust contrastive learning based on physical principles of nuclear radiation detection. A summary of the augmentations described below is illustrated in Figure 5.3.

Each augmentation is implemented in a manner such that a user can specifically control the characteristics and magnitude of each spectral augmentation, or it can be automatically generated using a set of default parameters and estimates for each spectrum. A set of manually labeled, one-minute material transfer spectra from MINOS consisting of seven transfer classes is used to demonstrate augmentations: unirradiated ^{237}Np targets used for ^{238}Pu production, irradiated ^{237}Np containing ^{238}Pu, unirradiated Cm targets used for ^{252}Cf production, irradiated Cm containing ^{252}Cf, ^{225}Ac, activated metals, and spent fuel [28].

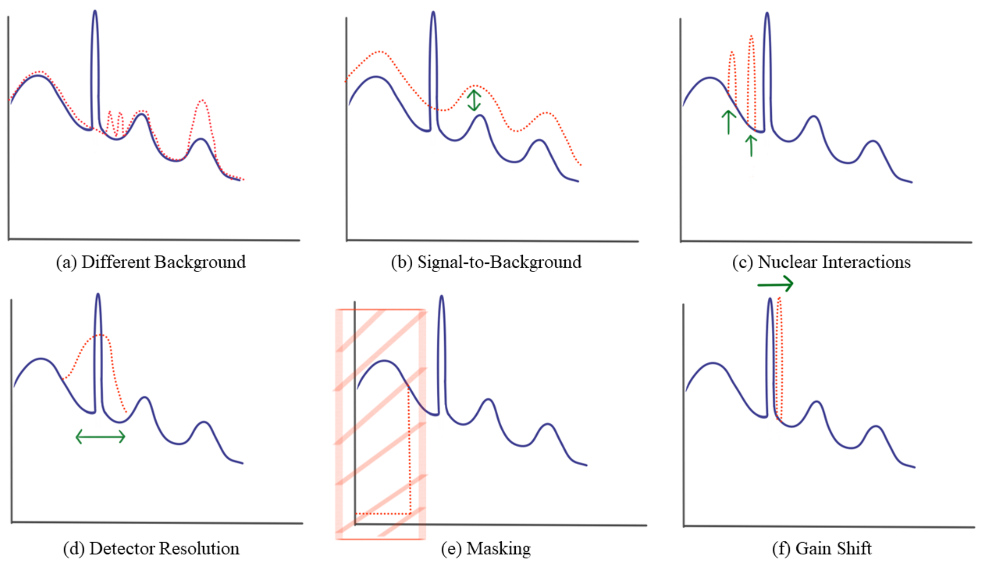


Figure 5.3: These augmentations are presented as valid transformations for gamma spectra. They are physically derived and maintain labeling information (granted that the transformation is not severe). The transformation (red-dotted) upon the original spectrum (dark blue-solid) is indicated with arrows (green).

5.2.1 Channel Resampling

Arguably the simplest transformation involves randomly resampling each bin in a spectrum. Due to counting statistics in detection physics, the response in a bin as a result of detecting various radiation signatures will be inexact by some amount of noise, p_i. The new augmented spectrum is synthetic—because it is randomly resampled from a statistical distribution—but is derived from a physical measurement. See Figure 5.4 as an example. Note that the final plot, showing the ratio of augmented over original counts per bin, indicates where significant changes were made in the spectrum. For channel resampling, the change looks like Poissonian noise as expected (with the relative change increasing as the absolute count magnitude decreases for higher energy bins).

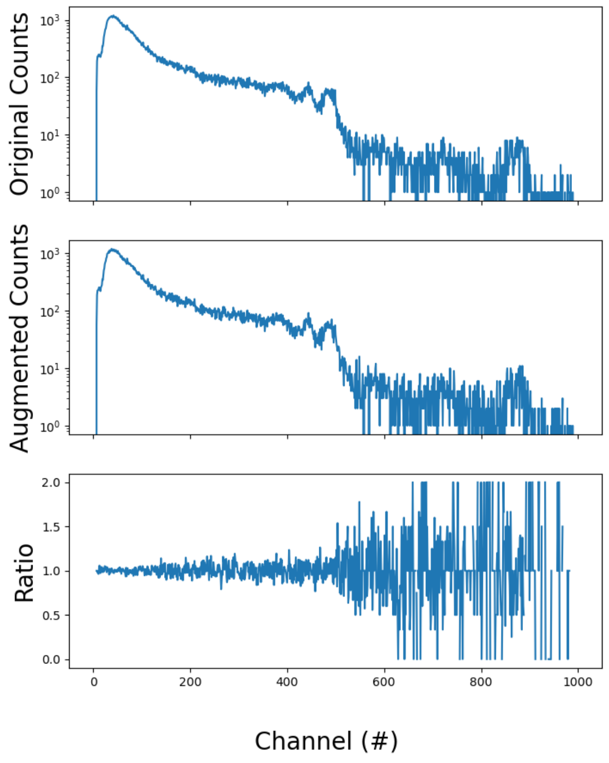


Figure 5.4: Channel resampling augmentation example: original transfer spectrum of activated metal (top) compared to an augmented version (center), and the ratio of augmented over original (bottom).

Rather than extracting $\boldsymbol{p}$ from $\boldsymbol{x}$, each been can be randomly resampled directly. Assume the counts of a given channel, x_i, represent the expected value (λ) of that channel's distribution: $x_i \sim \text{Poisson}\lambda = \text{Poisson}x_i$. To avoid unrealistic counting statistics at low count-rates (particularly at higher energies), resampling is taken from a Binomial distribution. Arbitrarily fixing the Binomial distribution's probability of success, $p = 0.5$, the "number of trials",

n, are derived as:

$$\begin{aligned} x_i = \lambda_i &= n_i p_i \\ p_i &= 0.5 \\ n_i = x_i/p_i &= x_i/2. \end{aligned} \tag{5.8}$$

Then, randomly resample: $x_i = \text{Binomial}(\frac{1}{2}, \frac{x_i}{2})$.

5.2.2 Different Backgrounds

Learning to recognize the underlying background continuum may help a model distinguish it from event signatures. Characteristic event signatures ($\boldsymbol{s}$) are typically superimposed onto a different background distribution ($\boldsymbol{b}$) constructed from the KUT decay chain. The background distribution—though fairly consistent for consecutive measurements—can change significantly for different detection environments (e.g. altitude, time of day, geographic location, weather, etc.). By replacing $\boldsymbol{b}$ in a spectrum, different environmental scenarios can be simulated.

A library of replacement background distributions can be generated using BEADS. A dataset of weakly anomalous spectra collected from August 2019 to December 2021 (see Section 6.1) are used. The three components of each spectrum are extracted, and $\boldsymbol{b}$ is saved. This creates a large library (≈45k) of representative background samples that could be used in a background transformation. When a spectrum is selected for augmentation, $\boldsymbol{s}$ and $\boldsymbol{p}$ are estimated using BEADS, and $\boldsymbol{b}$ is randomly replaced with a different distribution from the library. See Figure 5.5 as an example.

Not all background measurements in the library are truly a background measurement or have a shape representative for typical background that might be useful for learning information useful in downstream detection tasks. Artifacts as a result of repalcement or BEADS decomposition are possible and can result in unrepresentative shapes for typical background. These backgrounds do not appear to limit contrastive training to the point of

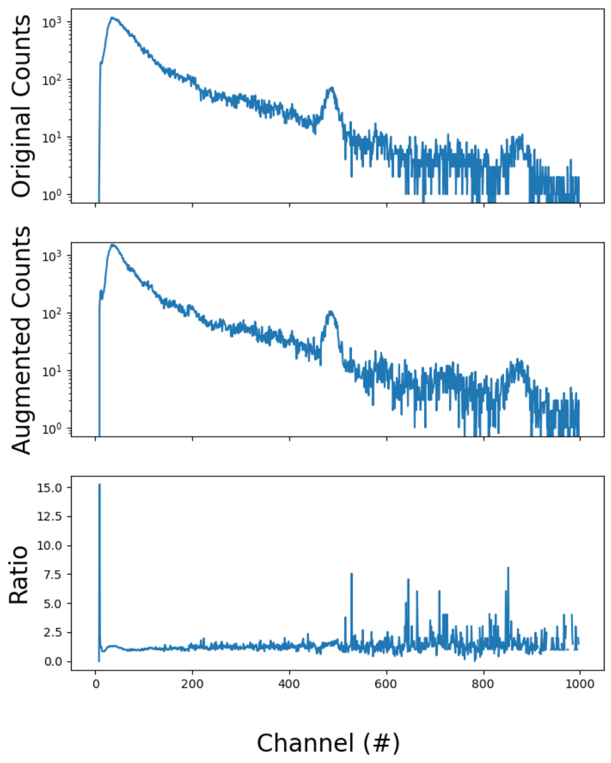


Figure 5.5: Different background augmentation example: original transfer spectrum of ^{252}Cf (top) compared to an augmented version (center), and the ratio of augmented over original (bottom).

requiring manual inspection and removal. If samples from the background corpus are, on average, representative of a typical background response, the downstream model should be able to learn from this augmentation.

5.2.3 Signal-to-Background

Detecting a radiation event is dependent upon proximity. As a detector moves away from a source (or vice versa), the count-rate intensity of radiation associated with the source decreases on the order of $1/r^2$ (r being the radius, or distance, between the detector and the source). Therefore, the strength of an event could be varied to coincide with different geometric efficiencies, or proximity to the source. This is a natural extension of differing the background above since it amounts to a complementary change in the signal-to-background ratio. Here, the signal ($\boldsymbol{s}$) is being transformed rather than the underlying background distribution ($\boldsymbol{b}$) it lies upon.

The signal extracted from BEADS is scaled by some ratio, r, randomly upscaling or downscaling by up to two-times the original magnitude. For simplicity of use, r must be positive. A value less than 1 indicates a downscaling by r, vice versa for greater than 1 and upscaling. Once perturbed, $\boldsymbol{s}$ is re-added to the other components of $\boldsymbol{x}$.

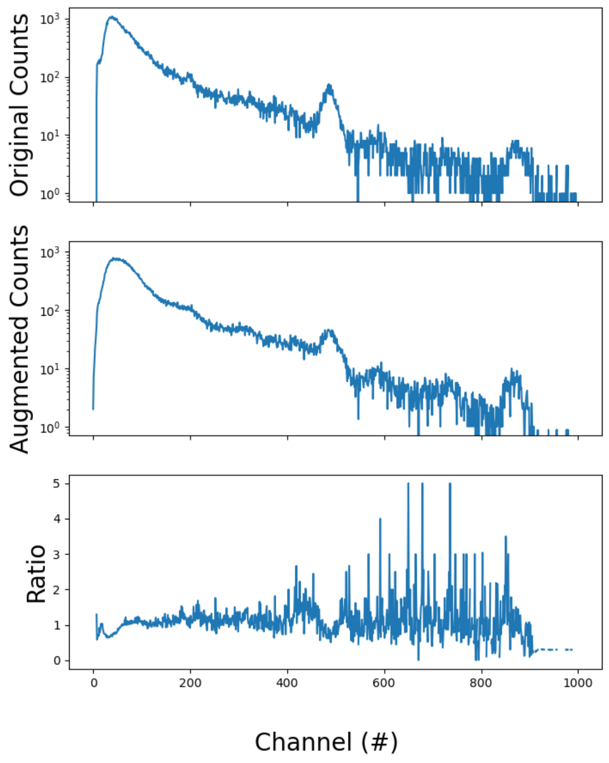


Figure 5.6: Signal-to-background augmentation example: original transfer spectrum of an unirradiated Np target (top) compared to an augmented version (center), and the ratio of augmented over original (bottom).

Two NaI detectors could have similar detector responses, but the unique circumstances of their measurement may result in varying background distributions or signature intensities, especially if they are different distances to the source. Therefore, perturbing spectra by manipulating their signal-to-background ratios could amount to the largest variation between realistic spectral measurements. Note an important limitation: BEADS can misattribute persistent peaks in a spectrum as part of its signal ($\boldsymbol{s}$) that would ordinarily be associated with background ($\boldsymbol{b}$). In this case, the background photopeak would be scaled along with other peaks in the spectrum. See Figure 5.6 as an example.

5.2.4 Signal/Photopeak Insertion

This augmentation involves artificially introducing detector specific phenomena, peaks, and distributions for radiation signatures ($\boldsymbol{s}$) that could realistically expected to be present in a spectrum. Other detection mechanisms could also be introduced into spectra. For example, a photon with energy greater than 1.022 MeV will register a photopeak in a detector and can also undergo pair production. This will produce two photons of energy 511 keV imparted from the incident gamma radiation. If these photons are not absorbed by the detector, the response will be the photopeak energy less 511 keV (single escape peak) or 1,022 keV (double escape peak). Sometimes these peaks do not appear in a spectrum because they have a low detection probability and therefore do not appear above the baseline distribution. However, these spectral features could give further indications of a signatures presence if their pattern can be learned.

First, BEADS is used to isolate peaks ($\boldsymbol{s}$) from the baseline distribution ($\boldsymbol{b}$) in a spectrum. Even if the peak shapes extracted with BEADS are inaccurate, their centroids are typically trustworthy, which is all that is necessary for identifying peaks to manipulate. Centroids are estimated from the BEADS-extracted peaks using Scipy's `find_peaks` method. Requiring a width of 4 bins, measured at the FWHM, appears to give an adequate selection of the most important peaks in a spectrum. The fit ROI around a peak is 1% of the total number of bins in the spectrum. That means that a spectrum with 1,000 bins would identify peaks in an ROI of 10 bins. The ROI is so small because the only fit parameter that is used is the estimated centroid (μ) and avoids other peaks that might be close or overlapping.

For this analysis, the list of possible peak insertions consists of radiation events that occur in MINOS but are not transfers: the KUT decay chain (^{40}K: 1460 keV, ^{232}Th: 2614 keV), ^{214}Bi: 609, 1764, and 2204 keV, ^{214}Pb: 295 and 352 keV, and ^{41}Ar: 1294 keV. This augmentation randomly inserts a nuclide's signatures from the list at its requisite energy. Amplitude and peak width are approximated by existing peaks in the spectrum. That is, all peaks beyond the first 100 bins (to avoid fitting the low-energy baseline distribution as a peak in itself) are fit using Scipy's `find_peaks` algorithm with a minimum prominence of 20 counts above baseline and a width of at least 4 bins as measured by Full-Width Half-Maximum (FWHM). The augmented peak's parameters are estimated using the values of the most prominent peak in the spectrum, assuming a linear relationship for amplitude and width as a function of peak energy. Note that the prominence threshold tends to result in high-energy peaks greater than 2 MeV being ignored, since they have a low count-rate (< 20 counts in some cases) when the background is essentially 0. In the future, if these peaks are of interest, the `find_peaks` thresholds should be adjusted. This crude approximation means that artificial peaks tend to have the same shape as the most recognizable peaks in a spectrum, but it will vary by spectrum independent of the underlying physics that would generate the augmented signals.

The artificial peak is inserted into the spectrum along with the approximated background near the peak shoulders and lower magnitude escape peaks if the inserted peak has an energy greater than 1.022 MeV. See Figure 5.7 as an example. The value of inserting different nuclear interactions is in teaching a model which radiation signatures that are not relevant to the class it is training on. If an augmentation inserts a radiation signature that is irrelevant to the characteristic signatures of a label class, this encourages the model to learn that this optional signature is not needed for a classification decision if it is present in a test spectrum. Then, if a transfer or event occurs on top of some other anomalous or underlying radiation event, the model can distinguish between the two. In practice, a user could specify a list of nuclides and gamma emissions that could occur in the data collection environment but are not necessary for classification.

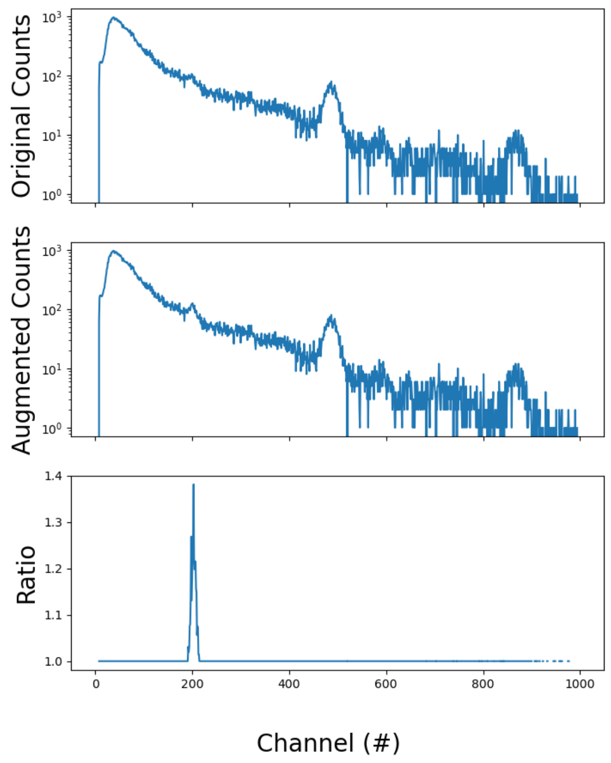


Figure 5.7: Signal augmentation example: original transfer spectrum of an irradiated Np target (top) compared to an augmented version with an ^{214}Bi 609 keV photopeak inserted (center), and the ratio of augmented over original (bottom).

5.2.5 Detector Resolution

Detection efficiency can affect photopeak energy resolution leading a peak to appear sharper or wider in a spectrum. This is typically measured using FWHM, the width of a peak at half its maximum height divided by the centroid energy. Gamma signatures are discrete in that they have a finite energy associated with the nuclear decay that produces them. Nuclear physics interactions within the detector (like Compton scattering or pair production) results in imperfect detector responses. This widens the discrete response to approximately a Gaussian distribution whose width is determined by individual detector properties (see Equation (5.2)). Widening or narrowing a peak would result in a spectrum that still maintains the same labeling information but appears with a different FWHM.

Peaks ($\boldsymbol{s}$) in a spectrum are first identified using Scipy's `find_peaks` algorithm. The model then randomly selects a peak and an ROI of 3% around that peak. The peak is fit to a Gaussian distribution, and a peak width (i.e. standard deviation) is estimated. Note that the width of the ROI matters greatly here. A small ROI may not capture the approximately linear background on either shoulder of the peak or the full peak distribution itself. A large ROI may include other peaks or spectral shapes that can confuse a fitting routine. Too large of an ROI could also erase peaks close to the one inserted and replace it with a crudely fit linear representation of the local background distribution.

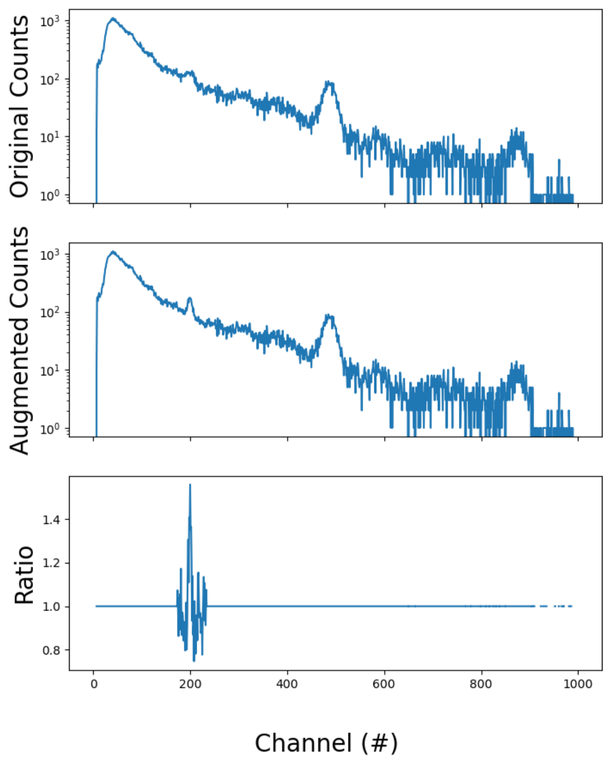


Figure 5.8: Resolution augmentation example: original transfer spectrum of ^{225}Ac (top) compared to an augmented version with a perturbed 200 keV photopeak (center), and the ratio of augmented over original (bottom).

The peak width is multiplied by a random scalar between 0.5 and 1.5. The original signature is overwritten by adding the new Gaussian peak with scaled width. The new peak is perturbed using channel resampling above to give it a realistic detection shape. See

Figure 5.8 as an example.

Altering detector resolution would be valuable because it simulates mild variations in real-world detector applications. If a successful peak cannot be obtained (i.e. the peak finding and fitting procedure returns an error after 100 attempts) or its fit parameters are nonsensical (e.g. negative amplitude), the augmentation is abandoned and the original spectrum is returned. This almost never occurs, but can slow the augmentation routine for spectra that are particularly difficult to estimate.

5.2.6 Masking

In contrastive learning on images, masking (i.e. setting a regions values to some null point) is a commonly used augmentation. By omitting the features in a masked region, especially when applied randomly, the model cannot rely on that semantic information for generating meaningful representations. For example, a masked image of a dog would encourage a model to recognize that a dog's entire body will not always be present in an image, so it should recognize individual features for object detection. A spectrum could be masked, but care must be taken to select an appropriate mask that reflects ways in which a detector may be unable to resolve certain energy regions.

Masking is applied here by setting a random region from 20 to 100 of the first bins in a spectrum to zero. Detection systems commonly apply a low energy mask to avoid the overwhelming response associated with low energy radiation, typically in the first couple bins of a spectrum. Just like with images, care must be taken to select appropriate mask properties. Larger masking regions could mean masking classification information. Conversely, too small of a masked region and it becomes a meaningless augmentation for representation learning. A variable mask region between 20 and 100 strikes a balance that sufficiently perturbs spectra for contrastive learning. See Figure 5.9 as an example.

While this masking strategy is aligned with radiation detection practices, other masking strategies could theoretically be applied. It may not be physically realistic, but masks could

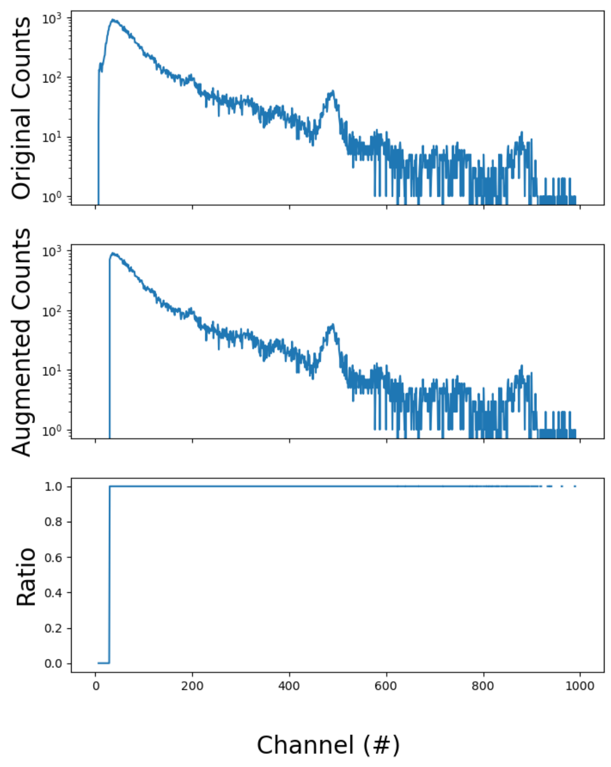


Figure 5.9: Masking augmentation example: original transfer spectrum of spent fuel (top) compared to an augmented version (center), and the ratio of augmented over original (bottom).

be applied anywhere in a spectrum. Masking a mid-energy region with notable spectral features would encourage a model to infer typical responses in that region through contextual information elsewhere in a spectrum (contrasted with the augmented spectrum's positive pair). These masks could help with pattern recognition, but risk losing label-invariance if labeling features only span a handful of energy bins.

5.2.7 Gain Shift

Energy versus channel gain calibration is a post-processing step in detection data acquisition. If the calibration in pulse-height discrimination is off, the entire spectrum can be shifted by several units of energy (as determined by the multi-channel binning). If severe, photopeaks could be misclassified as the wrong energy even if they are a true event signature, confounding detection and classification. Severe gain shift can stretch or squeeze the features and shapes in a spectrum. Mild gain shift is typical for long-running measurements, so calibration often occurs over the course of a measurement sequence.

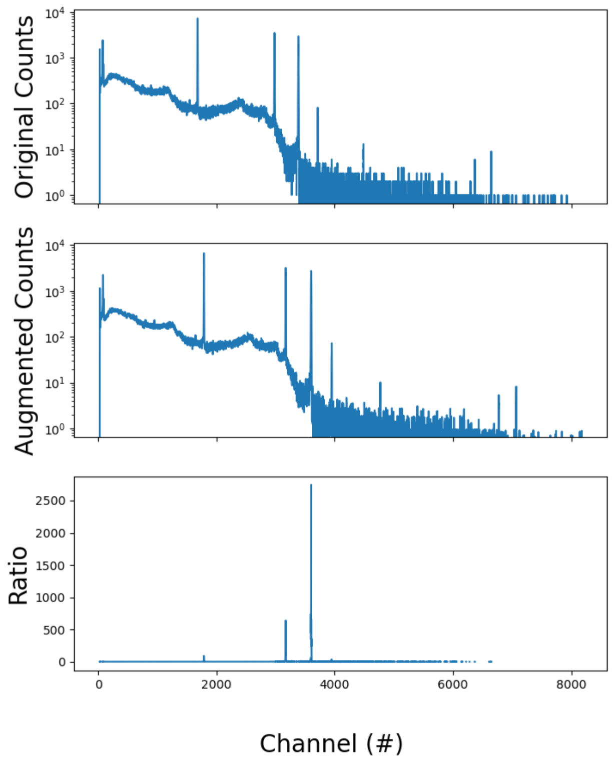


Figure 5.10: Gain shift augmentation example: original laboratory measurement spectrum of ^{137}Cs & ^{60}Co (top) compared to an augmented version (center), and the ratio of augmented over original (bottom).

Gain shift is applied via rebinning a spectrum by linearly interpolating. That is, the spectrum is randomly shifted up to 10 bins in either direction by adding these bins to the beginning or end of the spectrum. Then, the spectrum's counts are linearly interpolated through this new binning length, the original total count magnitude is conserved, and the spectrum is clamped to the original spectral length (1000 bins for MINOS data). This procedure results in a similar shaping behavior to physical gain shift in a detection system where spectra that otherwise would be well calibrated are stretched or squeezed. See Figure 5.10 as an example.

Gain shift introduced as a data augmentation may make a contrastive learning model more robust to shifts in energy and encourage tolerance for uncertainty in a photopeak's centroid. Note that Figure 5.10 perturbs a laboratory measurement of ^{137}Cs & ^{60}Co. This is to emphasize that these augmentations are not specialized for an individual dataset but generalized to principles of gamma spectroscopy. The behavior is comparable across augmented spectra even if the detection systems and target sources change.

5.3 Conclusion

A set of valid augmentations for gamma spectra based on principles of radiation detection have been presented and described here. Each augmentation exploits at least one component of any given $\boldsymbol{x}$. A contrastive learning model would be able to learn information for constructing meaningful representations from label-invariant perturbations in observed training samples. The next chapter illustrates an example of using these augmentations for contrastive learning, resulting in a model useful for detecting and characterizing nuclear material transfers even when trained on a limited amount of labeled data.

Chapter 6

Detecting Nuclear Material Transfers

This chapter constitutes a wide ranging study of contrastive learning applications to the detection and characterization of shielded radiological material transfers in support of nuclear nonproliferation. Labeled and unlabeled data collected from the MINOS testbed at ORNL are used to train and test a contrastive learning framework. An example model is presented, where the supervised classifier is trained on an extremely limited set of labeled spectra. The reported results are compared to other popular methods for classification. Several analysis tools are presented to evaluate the performance of a given contrastive model, including PCA and integrated gradients for feature importance. The value of individual augmentation types are studied. Finally a discussion is offered for optimal allocation strategies of limited labeled data and how noisy labels can effect the performance of SSL and SSML models.

6.1 MINOS Data and Detection Task

The data used for all experiments in this chapter were collected at the MINOS testbed (see Section 3.2.1 for more details). A breakdown of all data, unlabeled data, and labeled data can be seen in Figure 6.1. At a one-minute integration time, there are approximately 9.5 million uncharacterized spectra collected over six nodes and three years at the testbed. Since most of the spectra measured over time will be of background, down-sampling will provide a training set more reflective of nuclear material transfers. A training subset is collected using the hypothesis testing anomaly detection algorithm detailed in Section 3.2.4.

Here, a larger p-value captures more anomalous spectra by applying a looser threshold to changes in gross count-rate over time. A p-value of 1×10^{-5} yields 68,124 weakly anoma-

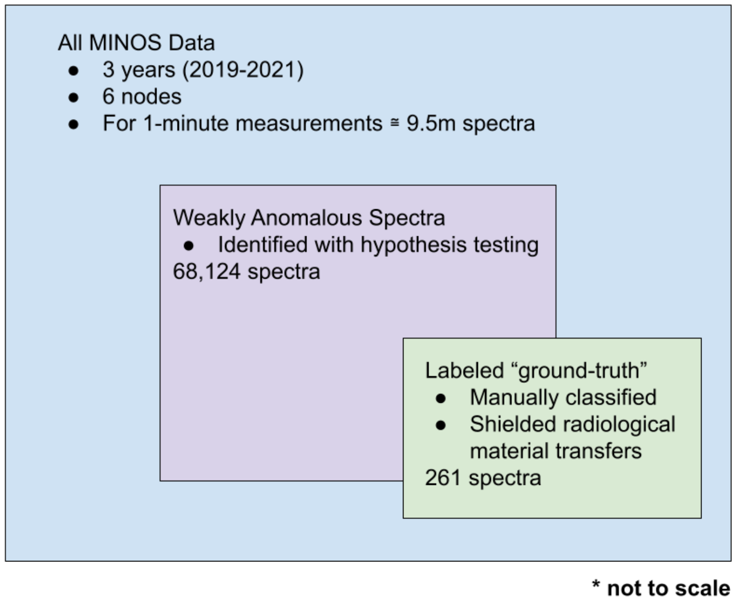


Figure 6.1: A breakdown of all MINOS data used in contrastive machine learning. A weakly anomalous subset of the entire dataset (blue) is generated using hypothesis testing anomaly detection with $p = 1 \times 10^{-5}$ (purple). A smaller subset (green) with an unknown amount of overlap with the unlabeled subset was manually labeled using ground-truth as a reference by an SME.

Table 6.1: Shielded radiological material transfers present in MINOS data.

Byproducts	Nuclear Material
^{225}Ac	Fresh Cm
Activated Metals	^{252}Cf
Spent Fuel	Fresh Np
	Irradiated Np

lous spectra. This subset will include some unknown number of transfers, background, and other anomalous event spectra. An even larger p-value could be used, but as p increases, more and more of the entire dataset is captured as "anomalous." For example, $p = 0.01$, captures 252,854 spectra, but the increase from $p = 1 \times 10^{-5}$ only includes spectra that are less anomalous and therefore less likely—but not impossible—to be from material transfers. Initial contrastive models that were trained and tested indicated no added benefit to classification accuracy by using $p = 0.01$ over $p = 1 \times 10^{-5}$. Likewise, a smaller p-value would capture fewer anomalies and therefore have less unlabeled data to contrastively train from. Therefore, $p = 1 \times 10^{-5}$ strikes a balance of a large enough training dataset for contrastive learning.

The classification task here is to identify shielded radiological material transfers. Seven types of transfers occur in the MINOS dataset that can roughly be subdivided into two binary classes (see Table 6.1). To simulate nuclear nonproliferation scenarios, there are byproducts of the nuclear fuel cycle that may be radioactive or have alternative applications but are not necessarily material of interest. Contrast these with nuclear material that might be tracked or monitored for proliferation resistance. To demonstrate the utility of contrastive machine learning, the model attempts to distinguish between these two classes. Many other nonproliferation detection tasks could be designed from this dataset (e.g. multiclass classification for identifying each type of material transfer). The task chosen here is more challenging than the original task of Chapter 3 (i.e. identifying material transfers from other anomalies) but is not so granular as to stretch the limited number of labeled samples per transfer type.

An even smaller subset of labeled data was prepared for training a supervised classifier. Labeled data was prepared and shared by an SME cross-referenced with ground-truth transfer records. In total, 261 spectra spanning the seven transfer subcategories were labeled. Timestamps are not available for these spectra and the degree of overlap with the unlabeled subset is unknown. Crucially, some spectra may not qualitatively exhibit characteristic features (e.g. photopeaks) associated with their labeled transfer type. This can occur when ground-truth specifies a transfer but various factors (e.g. background) obfuscate features. A contrastive learning framework and classifier may be able to classify these spectra if it is sensitive enough to transfer type, or these spectra may result in misclassifications or out-of-distribution (OOD) behavior, resulting in limited classification performance.

6.2 Experimental Design

The following sections offer details on an initial machine learning model design (prior to hyperparameter optimization in Chapter 7), semi-supervised and self-supervised frameworks, and individual model architectures.

6.2.1 Contrastive Learning Framework

The model framework for using MINOS data for training a representation model based on contrastive machine learning is illustrated in Figure 6.2. From the entire MINOS dataset, radiation data (as described in Section 6.1) is used as the sole data modality for this model.

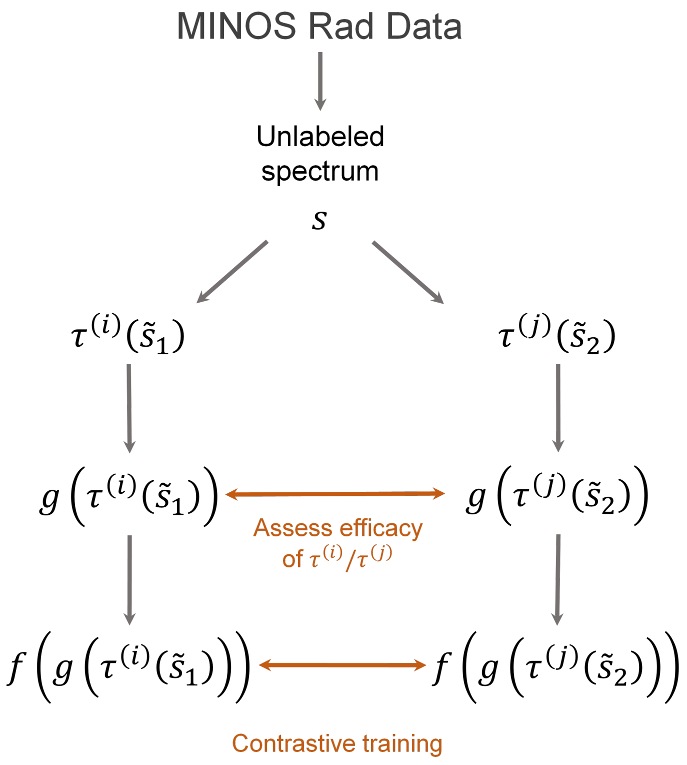

MINOS Rad Data
Unlabeled spectrum
s
$\tau^{(i)}(\tilde{s}_1)$
$\tau^{(j)}(\tilde{s}_2)$
$g\left(\tau^{(i)}(\tilde{s}_1)\right)$
$g\left(\tau^{(j)}(\tilde{s}_2)\right)$
Assess efficacy of $\tau^{(i)}/\tau^{(j)}$
$f\left(g\left(\tau^{(i)}(\tilde{s}_1)\right)\right)$
$f\left(g\left(\tau^{(j)}(\tilde{s}_2)\right)\right)$
Contrastive training

Figure 6.2: A contrastive learning framework defined for use with MINOS radiation data. Spectra, s, are perturbed using a transformation, τ, into augmented spectra, $\tilde{s}$. Augmented spectra serve as input to a representation model, $g(\cdot)$, which provides representations to a projection head, $f(\cdot)$, upon which contrastive learning is ultimately conducted.

Each unlabeled (or labeled) spectrum, s, in a batch is augmented twice using two random transformations, with replacement, from the set of transformations discussed in Chapter 5 (i.e. $\tau^{(i)}, \tau^{(j)} \in \mathcal{T}$). The two transformations randomly generate (within a degree of severity) two augmented spectra, $\tilde{s}_1, \tilde{s}_2$. The representation model, $g(\cdot)$, takes each augmented spectrum as input and outputs a higher dimensional representation. These representations become the input of a projection head, $f(\cdot)$, which reduces the dimensionality (thereby distilling significant features or patterns in representations) to an output that is directly used in computing and optimizing a contrastive loss (see Section 4.2).

6.2.2 Model Architecture

A pictorial representation of the contrastive learning framework can be seen in Figure 6.3. A representation model, or encoder, produces representations from spectra. The projection head produces projections from the representations with reduced dimensionality.

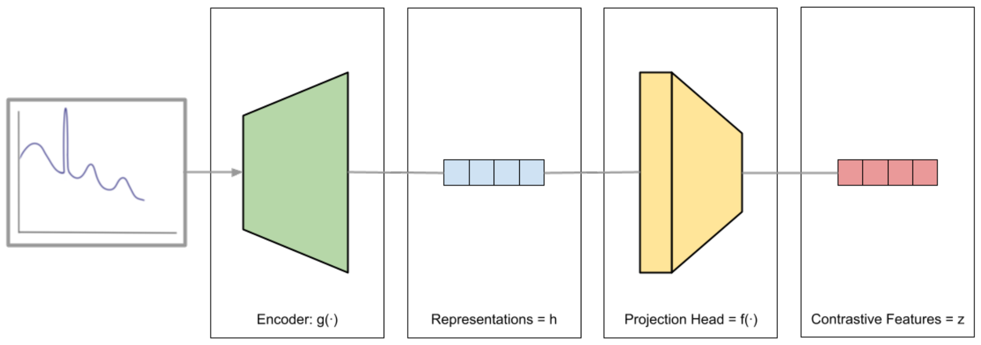


Figure 6.3: Pictorial depiction of the framework during contrastive machine learning. Spectra passed through an encoder generate higher dimensional representations which, when projected, can be used to compute and minimize a contrastive loss.

Nothing about the above contrastive learning framework specifies model design or architecture. This serves as an advantage in that, besides the set of augmentations $\mathcal{T}$, the framework is data and model agnostic. After contrastive training, the projection head is discarded. Instead, a classifier is trained directly on the representations (see Figure 6.4). The classifier used here is a linear model, containing no hidden layers, that takes the representation features and produces two class probabilities: one for byproducts and one for nuclear material transfers. This simple classifier in itself may not be complex enough to capture the relationships between representations and labels, but it allows observed increases in accuracy to be the result of contrastively learned information by the encoder. The largest class probability is the designated label for any spectrum in the training or testing dataset. The classifier is trained using the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm in PyTorch via supervision by a labeled subset with cross-entropy loss penalizing disagreement between estimated and expected labels.

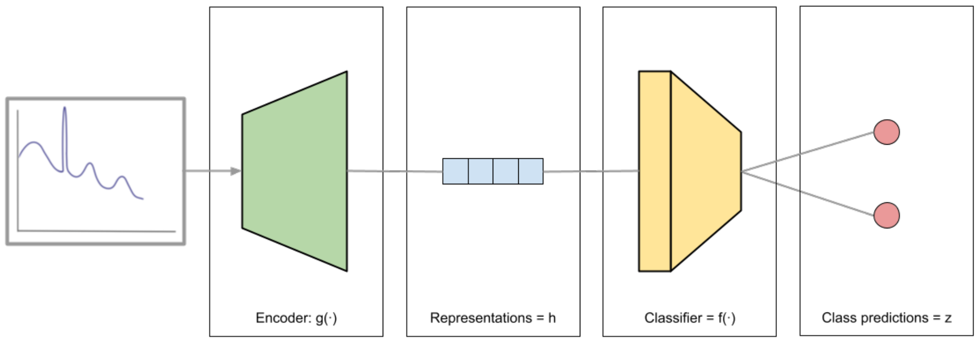


Figure 6.4: Pictorial depiction of the framework after contrastive machine learning, during training and testing of a supervised classifier. The projection head is replaced by a classifier which takes high dimensional representations and produces class probability. In this case: binary class probabilities.

Before hyperparameter optimization, the studies below will use the same general model architecture. The basic building block will be a fully-connected neural network, or MLP. From now on, neural networks will be denoted by their type (MLP or CNN) and their hidden layers: $\text{MLP}[\text{hidden\_layer}_1, \ldots, \text{hidden\_layer}_n]$. For example, an MLP with three hidden layers with 128 neurons each is $\text{MLP}[128, 128, 128]$. The encoder is $\text{MLP}[4096, 4096, 4096, 4096]$. Through four fully-connected layers, it produces representations that are approximately four times larger than the original spectral dimensionality of 1000 channels. The projection head consists of one hidden layer and reduces the dimensions to 128: $\text{MLP}[4096, 128]$. The basic classifier is a linear model that takes the representations of dimension 4096 and outputs a two-dimensional (binary classification) class probability.

6.3 Example Results

The following section offers a systematic demonstration of training and testing this contrastive learning framework using the data and neural architectures described above.

Table 6.2: Hyperparameters for self-supervised contrastive learning.

Parameter	Value
α	1.0
Temperature (τ)	0.5
Base Learning Rate	0.01
Batch Size	512
Epochs	33
Batches per Epoch	31

6.3.1 Representation Model

For a list of hyperparameters used in this example, refer to Table 6.2. The actual learning rate is determined by skewing the base learning rate, dependent on the batch size,

$$\mathrm{lr} = \begin{cases} \mathrm{base\_lr} \times \frac{\sqrt{\mathrm{batch\_size}}}{256}, \mathrm{batch\_size} \leq 1024 \\ \mathrm{base\_lr} \times \frac{\mathrm{batch\_size}}{256}, \mathrm{batch\_size} > 1024 \end{cases} . \tag{6.1}$$

This scaling helps stabilize training when batch size is very large [39]. For 33 epochs, 100 batches of 512 unlabeled spectra (identified as weakly anomalous) are augmented and contrasted. The representation training curve can be seen in Figure 6.5. Note that the model minimizes to a minimum loss after only a handful of epochs. The spikes in training loss observed at later epochs is a result of the optimization algorithm attempting to escape local minima.*

6.3.2 Classifier

With the representation model optimized by self-supervision, a supervised classifier can be trained to take encoded representations of spectra and output a label for tracked material transfer detection. Here, the labeled subset of 261 spectra can be used for training and testing. Only seven labeled spectra will be used for training the classifier to demonstrate the

*This particular attempt uses SGD for training. As is shown in Chapter 7, using Adam for loss minimization results in a more stable training curve.

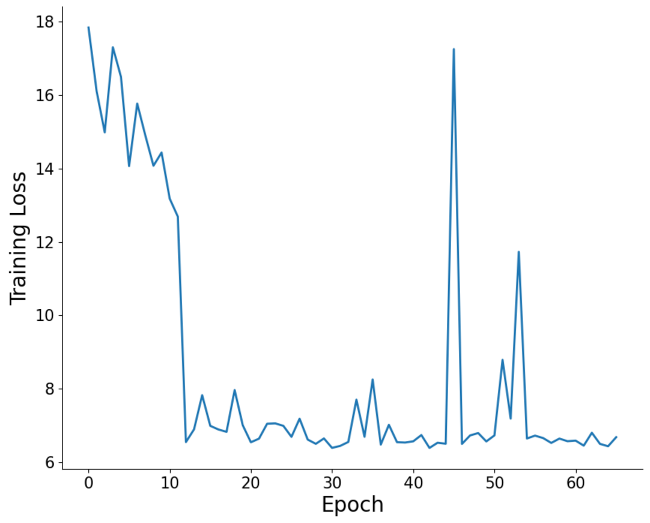


Figure 6.5: The training loss curve during self-supervised contrastive learning for the representation model. Note the rapid convergence to a local minimum loss. The spikes in training loss at later epochs is the result from the optimizer attempting to escape local minima.

Table 6.3: Ground-truth label for material transfers in classifier training set.

Transfer Type	Total
Byproduct	1
^{225}Ac	1
Nuclear Material	6
^{252}Cf	1
Fresh Np	1
Irradiated Cm	2
Irradiated Np	2
Total Transfers	7

efficacy of this method when labeled data is extremely limited. Each spectrum and transfer type used in the training set are listed in Table 6.3.

Note that this training set is very asymmetric. There is only one transfer for the byproduct classification label and six for the tracked nuclear material label. The classifier will therefore only have feature space information from the one byproduct spectrum to determine a decision boundary for the two classes. In practice, this type of training regime is likely to be plagued by suboptimal local minima that do not characterize the test space properly.

Training the classifier minimizes the cross-entropy loss between the predicted label given (with representations encoded from spectra passed to the classifier) compared to the true label in the training set. The L2 norm regularization weight is 1×10^{-3} and the learning rate is 1×10^{-2} for the L-BFGS optimizer. A decision boundary is reached for a loss of 0.0 and a training accuracy of 100%. The testing accuracy—computed on the remaining 254 labeled spectra—is 75.197%, for a balanced accuracy of 79.25%.

The resulting confusion matrix is shown in Figure 6.6. Whereas most of the byproducts are correctly identified (a false positive rate of 10.26%), nearly a third of the tracked nuclear material transferred are misclassified as byproducts (a false negative rate of 31.25%). This model as it stands could not feasibly be deployed in a monitoring scenario with, for example, the IAEA because their thresholds for false alarm rates (false positives) and detection probability ($1 - \text{FNR}$) are 5% and 90%, respectively [51]. However, the model demonstrates progress on developing a methodology for utilizing semi-supervised machine learning in nuclear material transfer detection.

Note that the accuracy and balanced accuracy scores achieved here are higher than was achieved in Chapter 3. Better metrics are achieved even though the target task of distinguishing between types of material transfers is more challenging than the task there, which was to distinguish transfers wholesale from other anomalous radiation events. Chapter 3 demonstrated that transfer detection information can be gained from unlabeled data. Here it has been demonstrated that a system of gamma radiation augmentations designed to generate encoded representations—combined with even limited labeled data—can learn information needed to successfully characterize between types of material transfers.

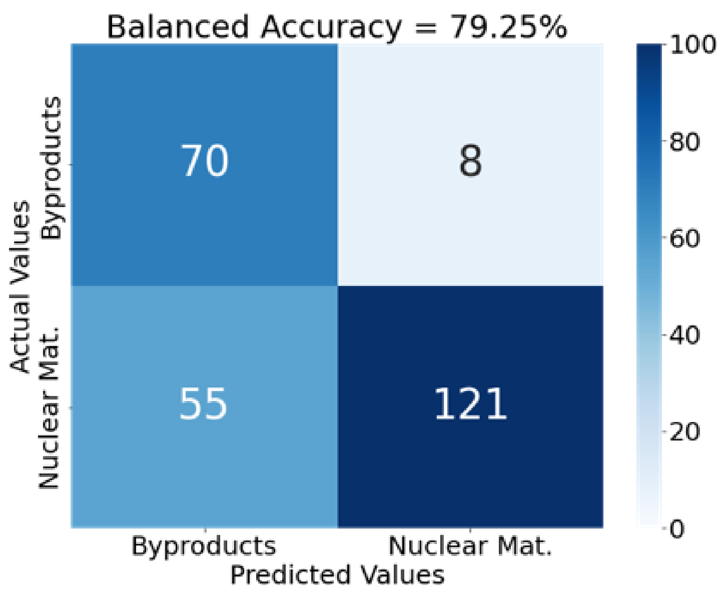


Figure 6.6: The confusion matrix for a binary classifier trained and tested on labeled spectra passed through a representation encoder. Since the two classes (byproducts and nuclear material) are imbalanced, the balanced accuracy score is reported. Note that this model has competitive performance with other models tested despite the nontrivial false negative rate.

6.3.3 Systematic Misclassification

The model successfully assigns each subclass of the seven training spectra to their respective binary class. That is, the model optimizes to a training accuracy of 100% on the training data. This is rather trivial when trained on such a small number of spectra. But investigating which subclasses are systematically misclassified by a given model helps diagnose strengths or weaknesses in the model's ability to identify specific transfer types. Figure 6.7 is a confusion matrix for all test spectra in the byproduct class broken down by their respective subclasses, Figure 6.8 provides a similar summary for the tracked nuclear material transfers.

Whereas ^{225}Ac and spent fuel transfers are largely labeled correctly by the model, activated metals are less successful although largely still labeled correctly. Both subclasses, activated metals and spent fuel, have the same number of test samples, which are significantly less than ^{225}Ac. Consider also that the one byproduct training sample used was of an ^{225}Ac transfer. This may suggest that activated metals have features and patterns more distinct from ^{225}Ac and spent fuel, causing poorer detection accuracy. Or perhaps the sin-

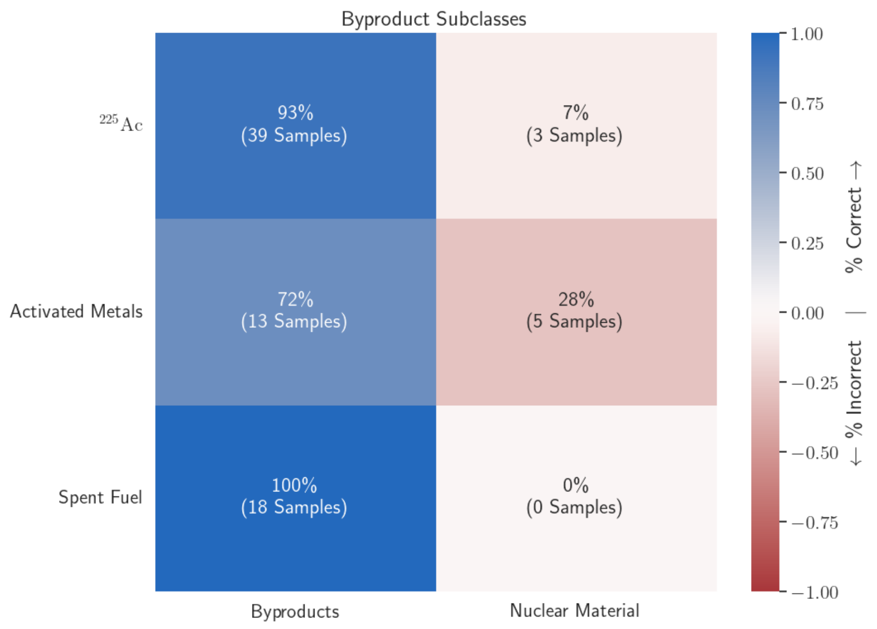


Figure 6.7: Classification accuracy summary for each subclass in the byproduct class (i.e. all instances have a true binary label of byproduct). In each cell, the percent (and gross number) of test spectra labeled a given binary class are listed. A higher true negative rate is shaded blue, a higher false positive rate is shaded red. The model appears to have better detection accuracy for ^{225}Ac and spent fuel than it does activated metals.

gle training sample provided insufficiently generalizable information for drawing a decision boundary capturing all the activated metal transfers in the testset.

The encoder and classifier model had worse performance in capturing the behavior necessary for labeling tracked nuclear material transfers. No subclass in the nuclear material testset was perfectly classified, which may be a limitation of the classifier or a demonstration of the noisiness in the labeled data. The model particularly struggled to successfully identify unirradiated transfers: ^{252}Cf and fresh Np. This is contrasted with better success at detecting irradiated material transfers: irradiated Cm and irradiated Np. This may be unsurprising because the recently irradiated samples would have stronger radioactive signatures (even if they are more shielded).

It is possible that binary classes constructed may be ill-suited for transfer detection. That is, perhaps the irradiated and unirradiated transfers are sufficiently different such that detection accuracy would increase by separating them into separate classes and conducting

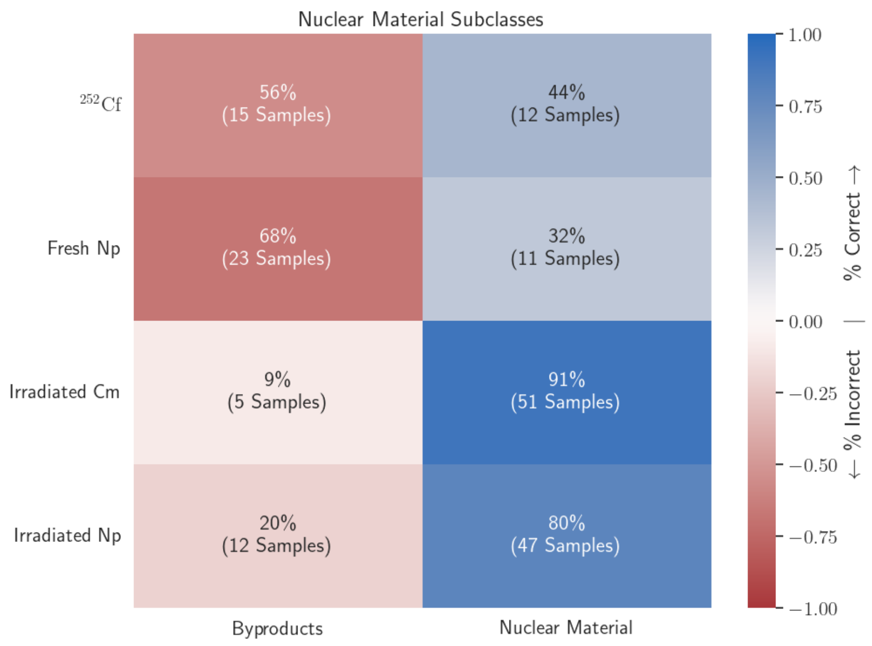


Figure 6.8: Classification accuracy summary for each subclass in the nuclear material class (i.e. all instances have a true binary label of nuclear material). In each cell, the percent (and gross number) of test spectra labeled a given binary class are listed. A higher true positive rate is shaded blue, a higher false negative rate is shaded red. The model appears to have better detection accuracy for irradiated Cm and irradiated Np transfers than it does for unirradiated transfers (^{252}Cf and fresh Np).

multiclass classification instead. While not tested in the scope of this work, it is possible that the arbitrary restraint of binary classification lowers detection accuracy by grouping dissimilar classes. However, given the available labeled data, a multiclass machine learning model would have less labeled samples to train on per class, which could introduce different limitations on detection accuracy.

6.4 Comparison to Scikit-Learn

A critical question should be considered: why bother with this complex framework of contrastive learning and semi-supervision when more simple, blackbox models are available through, say, Scikit-Learn. The reason these available methods are popular is because they are lightweight and performant for well curated, labeled data (they are often used to demonstrate Artificial Intelligence (AI) functionality for educational purposes). Available

Table 6.4: Cross-validated scores for logistic regression on encoded representations.

Trials	**Logistic Regression**	
	Accuracy	Balanced Accuracy
1	81.25%	81.68%
2	75.60%	72.65%
3	75.12%	71.16%
4	78.47%	75.93%
5	74.16%	69.51%
Average:	76.92%	74.19%
STD:	2.60%	4.30%

methods include logistic regression, supervised MLP, SVM, and even semi-supervised Label Propagation which was used in Chapter 3. Unfortunately, what these methods gain in ease-of-use, they lack in generalizability and robustness to poorly characterized, real-world, and unconventional data modalities (e.g. radiation vs. images).

For example, each of the methods mentioned from SCIKIT-LEARN is trained and tested using the 261 labeled spectra from MINOS. These methods only used the labeled dataset because they are purely supervised (apart from Label Propagation) and therefore incapable of handling unlabeled data. The training subset consists of 52 spectra (20%) and the testing subset consists of 209 spectra (80%)[†].Five-fold cross-validation can be used to average the resulting accuracy and avoid training bias associated with a specific cut of the labeled dataset. The results can be seen in Table 6.5.

Even with 52 spectra, each model is incapable of reaching competitive accuracy scores compared to the contrastive framework above. That is, they are not robust to some of the shortcomings in labeled data that have been observed so far: noisy labeling, obscured spectral features, and limited labeled data.

One of the advantages of contrastive learning is the development of the representation model, which provides a richer state space for determining a classification decision boundary. The encoder learns and accentuates spectral features associated with radiation sources in a

[†]Typical practice for cross-validation is to use a leave-one-out strategy. The opposite was used here to accentuate the effects of conducting supervised learning with limited labeled data.

Table 6.5: Cross-validation using five folds on various Scikit-Learn *algorithms.*

Trials	**Support Vector Machine**		**Logistic Regression**		**Multilayer Perceptron**		**Label Propagation**	
	Accuracy	Balanced Accuracy	Accuracy	Balanced Accuracy	Accuracy	Balanced Accuracy	Accuracy	Balanced Accuracy
0	69.71%	50.00%	71.15%	62.70%	76.44%	67.84%	65.38%	61.26%
1	69.86%	50.00%	78.47%	68.35%	76.56%	70.59%	75.60%	64.94%
2	69.86%	50.00%	75.12%	67.30%	77.99%	68.00%	74.16%	72.03%
3	69.86%	50.00%	79.90%	72.08%	77.03%	70.48%	75.12%	67.75%
4	69.38%	50.00%	78.95%	73.92%	77.99%	71.05%	72.73%	65.07%
Average:	69.73%	50.00%	76.72%	68.87%	77.20%	69.59%	72.60%	66.21%
STD:	0.21%	0.00%	3.60%	4.37%	0.75%	1.54%	4.18%	3.99%

task-agnostic manner. In fact, the linear classifier used in the contrastive framework has limited complexity and could be replaced by any of the supervised methods used here to capture more labeling information present in encoded representations. For example, cross-validation can be conducted on representations generated using the hyperparameter optimized encoder from Chapter 7 instead of training and testing the logistic regression model directly on spectra. The results are reported in Table 6.4. Cross-validated scores for a supervised logistic regression model using encoded representations learned from contrastive learning are clearly stronger than the same model trained and tested on the same data but as spectra instead. The combined utility of a self-supervised contrastive encoder and a complex supervised classifier results in a powerful model capable of distinguishing between types of material transfers.

6.5 Understanding Information Gained

An increase in detection accuracy has been observed, but this information learned could be the result of the augmentations used, the encoder which produces higher-dimensional representations, or the application of neural networks. This section attempts to qualify each component in kind.

6.5.1 Using PCA

One should ponder what benefit to transfer identification the representation model provides. Although it is demonstrated in Section 6.4 that alternative models trained directly on spectra are largely unsuccessful, how does the representation space generated by the encoder lead to success? Applying PCA can help to probe this question. For a background overview of PCA, refer to Appendix I.1.

A PCA is conducted on the normalized spectra directly and on the normalized representations produced by the encoder from input spectra. The entire labeled dataset (261

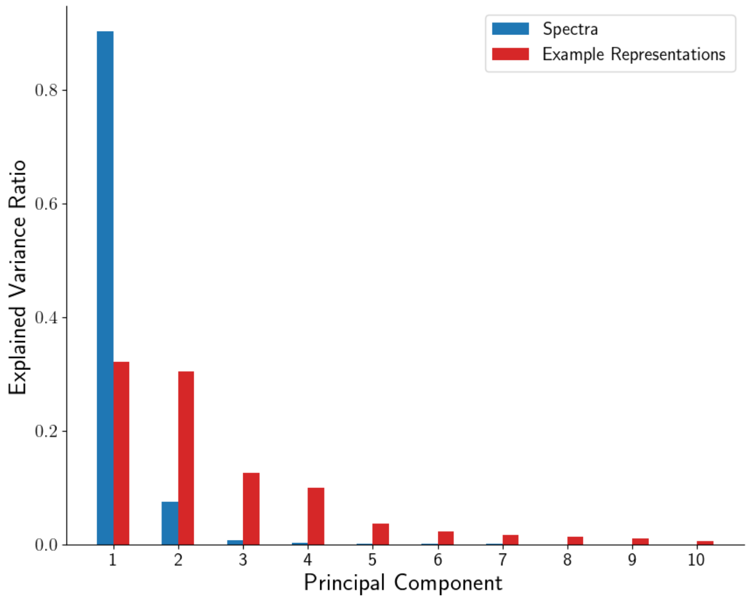


Figure 6.9: The first ten explained variance ratios in a PCA for spectra (blue) and encoded representations (red). The ratios represent the percentage of distributional variance that can be explained by a given principal component. Note that the variance is spread across many more components for representations than for spectra.

spectra) is used to compute each PCA. The variance ratios for the first ten principal components are plotted in Figure 6.9. The components are ordered in decreasing share of the dataset's variance. Each ratio represents the percentage of total variance in the dataset that lies in a given component's dimension.

For the spectral PCA, almost all the variance in the dataset can be explained by two components. This means that, generally, no other degrees of freedom can be used for separating the binary classes. A decision boundary would effectively have to be drawn in this two-dimensional feature space. While some classes appear distinct, separating labeled transfer spectra—even when reduced to a binary classification task—is visually difficult to accomplish (refer to the two-dimensional PCA plot in Figure 6.10). The limited degrees of freedom in spectral feature space helps explain why the models in Section 6.4 have limited detection accuracy.

Contrast this to the PCA created for encoded representations generated from the trained

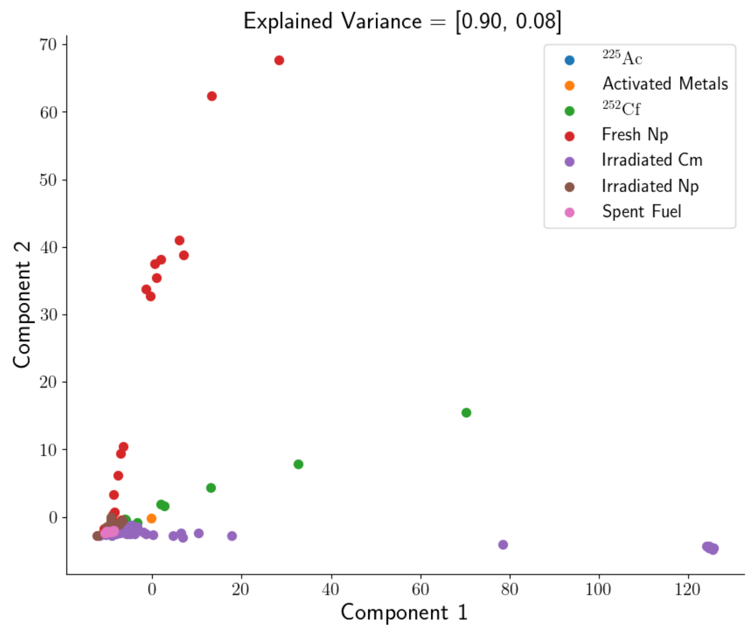


Figure 6.10: A two-dimensional PCA is conducted on the spectra from the labeled dataset. Each subclass is uniquely colored. Note any visual separation between classes and the explained variance covered by these two components.

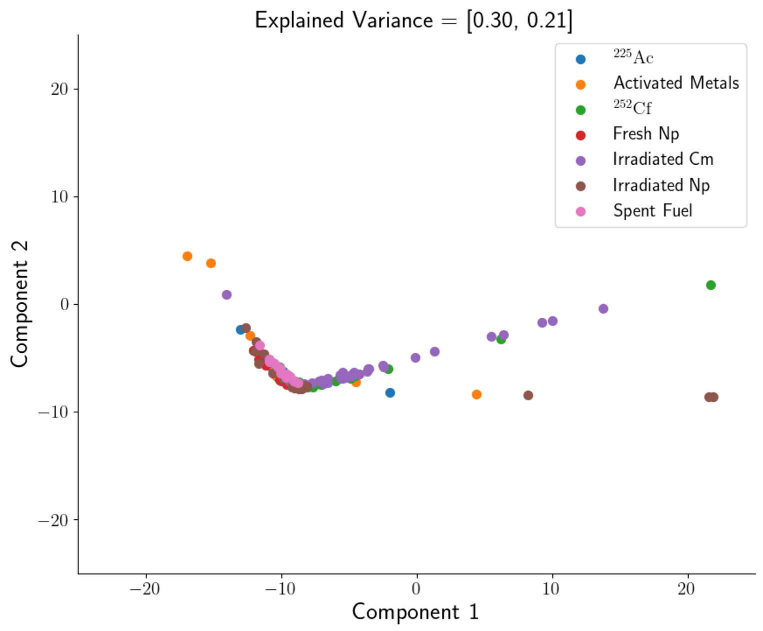


Figure 6.11: A two-dimensional PCA is conducted on the representations generated by the encoder using the spectra from the labeled dataset. Each subclass is uniquely colored. Note any visual separation between classes and the explained variance covered by these two components.

encoder presented in Section 6.3. Explained variance is more spread out across the top ten components, and far less is explained by two-dimensions than with spectra directly. Whereas binary separation is just as unlikely to be found in two components with representations (refer to Figure 6.11), the extra degrees of freedom with explained variance suggests more feature space for drawing a decision boundary. A classifier that uses encoded representations instead of spectra directly has more distributional directions to separate classes, supporting the observation that the example model in Section 6.3 is more successful.

6.5.2 Integrated Gradient for Feature Importance

Can the contrastive learning framework be expected to learn spectral features for different classes that are physically realistic for the signatures of associated gamma radiation? This question can be explored using a system for quantifying feature importance: integrated gradients [52]. Unlike other methods like shapley values that use game theoretic approaches, integrated gradients directly exploits the differentiability of neural networks. That is, feature importance is constructed by computing the integral of the gradients of the neural network with respects to a given input or baseline. This method assumes that (a) the gradient is sensitive to important features and that (b) functionally equivalent neural networks should have identical feature importance. The CAPTUM package from PYTORCH will be used to compute integrated gradients.

In each of the figures below, the original spectrum (blue) is plotted above its normalized variant (black, which is calculated by subtracting the mean and dividing the standard deviation of the weakly anomalous training data) and the integrated gradient (i.e. feature importance, red). This analysis should demonstrate how integrated gradients could be used alongside a successful detection model to guide manual analysis of spectra where typical or expected features may be difficult to perceive.

Figure 6.12 shows an example of a correctly labeled byproduct spectrum from the test dataset. For this spent fuel spectrum, the low-energy continuum appears to be smaller than

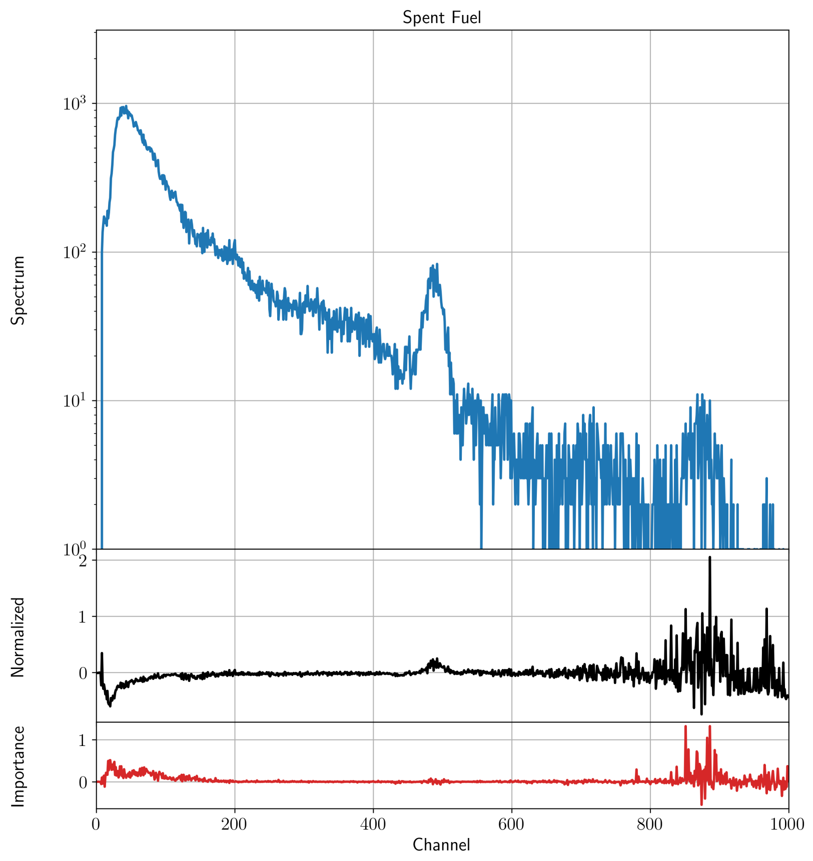


Figure 6.12: An original spectrum (blue) for a spent fuel transfer (byproduct), the normalized feature vector passed to the representation model (black), and the integrated gradient, or feature importance (red). Note that the model uses deviations in the spectrum at low and high energies to label the spectrum as a byproduct.

the training normal. The 2614 keV ^{208}Tl photopeak and surrounding region also appear to be highly variable compared to the training normal. These two features—and their deviations from the typified anomalies of the contrastive training dataset—appear to contribute most to the integrated gradient and the ultimately correct labeling of byproduct.

Figure 6.13 shows an example of a correctly labeled nuclear material transfer spectrum from the test dataset. This is a spectrum for a transfer of irradiated Np, which contains unknown amounts of ^{238}Pu. No photopeaks or signatures of ^{238}Pu and its decay chain are easily identified visually from the original spectra. This is likely because the transfer was shielded when it was transported. However, the integrated gradient exhibits at least two low-energy distributions responsible for correctly labeling this spectrum as a nuclear material

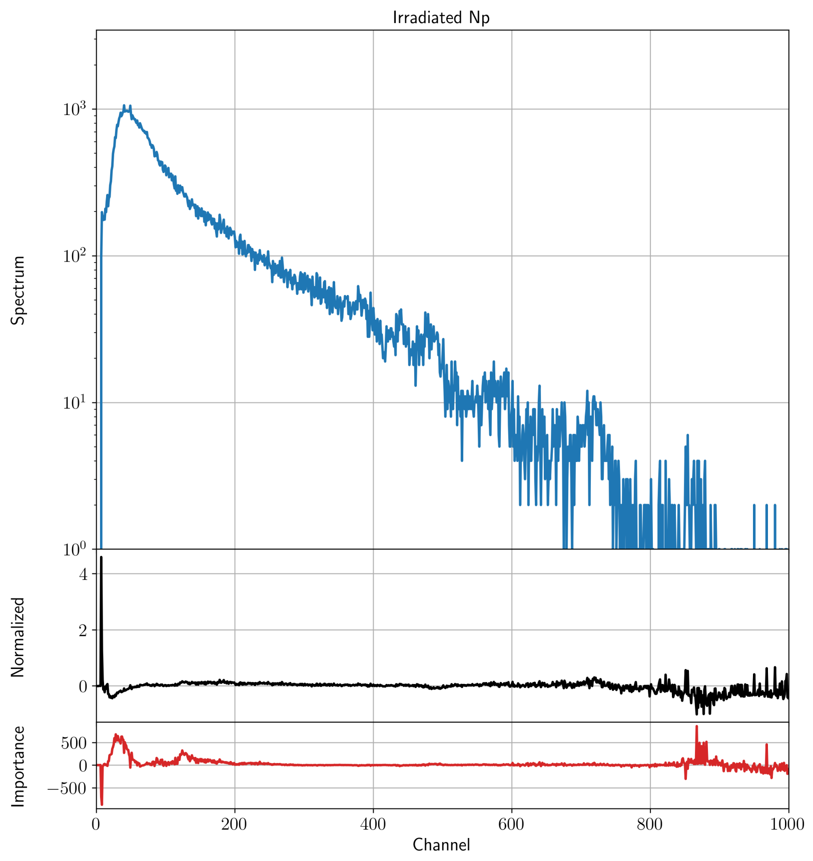


Figure 6.13: An original spectrum (blue) for an irradiated Np transfer (nuclear material), the normalized feature vector passed to the representation model (black), and the integrated gradient, or feature importance (red). Note that the model correctly identifies low-energy signatures expected of an irradiated Np transfer containing ^{238}Pu.

transfer. Low-energy gamma radiation is expected when ^{238}Pu is present. These distributions likely correspond to physically derived signatures associated with the underlying irradiated NP transfer, or at least a nuclear material transfer in general.

Not all spectra are correctly labeled/identified, and integrated gradient provides a diagnostic tool for investigating what features or patterns led to a particular misclassification. When reading integrated gradients for a misclassified example, the magnitude intuition is reversed. Now, a positive magnitude indicates that feature has a stronger contribution to mislabeling the sample, and vice versa for negative magnitudes. For example, Figure 6.14 shows an ^{225}Ac transfer with higher-than-average counts for channels in the first half of the spectrum. Whereas the shape and magnitude of the signatures between channels 100 and

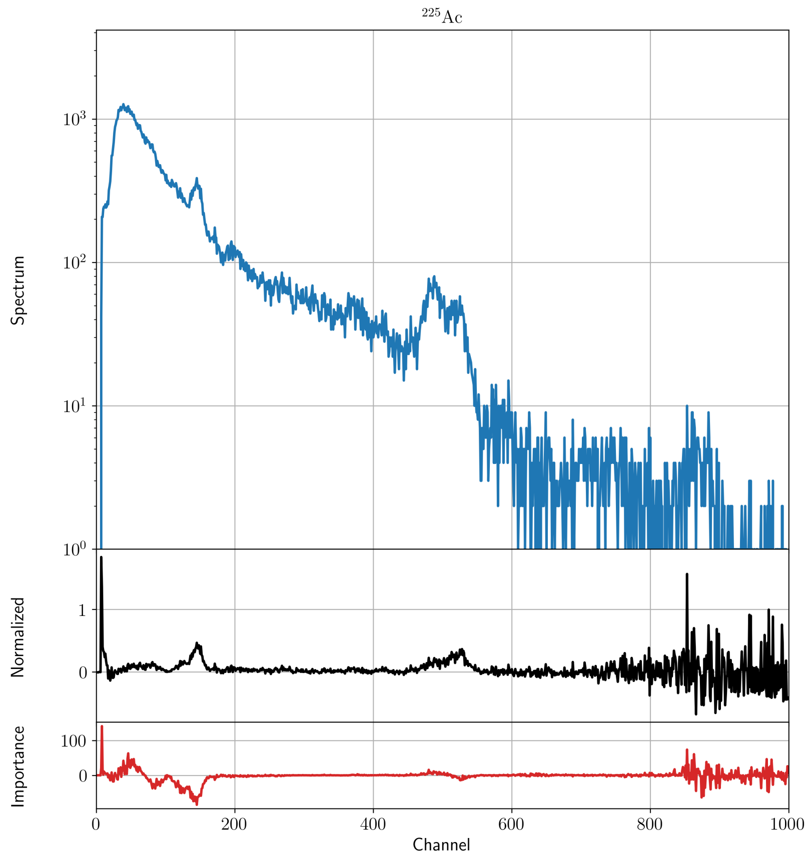


Figure 6.14: An original spectrum (blue) for an ^{225}Ac transfer (byproduct), the normalized feature vector passed to the representation model (black), and the integrated gradient, or feature importance (red). Note that the model misinterprets the low-energy background continuum leading to a false positive labeling.

200 would have correctly labeled this spectrum as a byproduct transfer, it is outweighed by peculiarities elsewhere. Channels on the extreme ends of the spectrum are highly sensitive to large differences in counts expected from the observed anomalous training set. In particular, the large differences in counts for the first few channels leads the model to incorrectly label this spectrum as a nuclear material transfer falsely derived on the low-energy background continuum.

Note that the normalized spectra for the labeled subset can often be dominated by outlier bins. These are bins with magnitudes largely unobserved by the training dataset. For more obvious material transfers (i.e. more prominent photopeaks and signatures), the spectra can be so unique as to be virtually OOD compared to what the model expects from the weakly

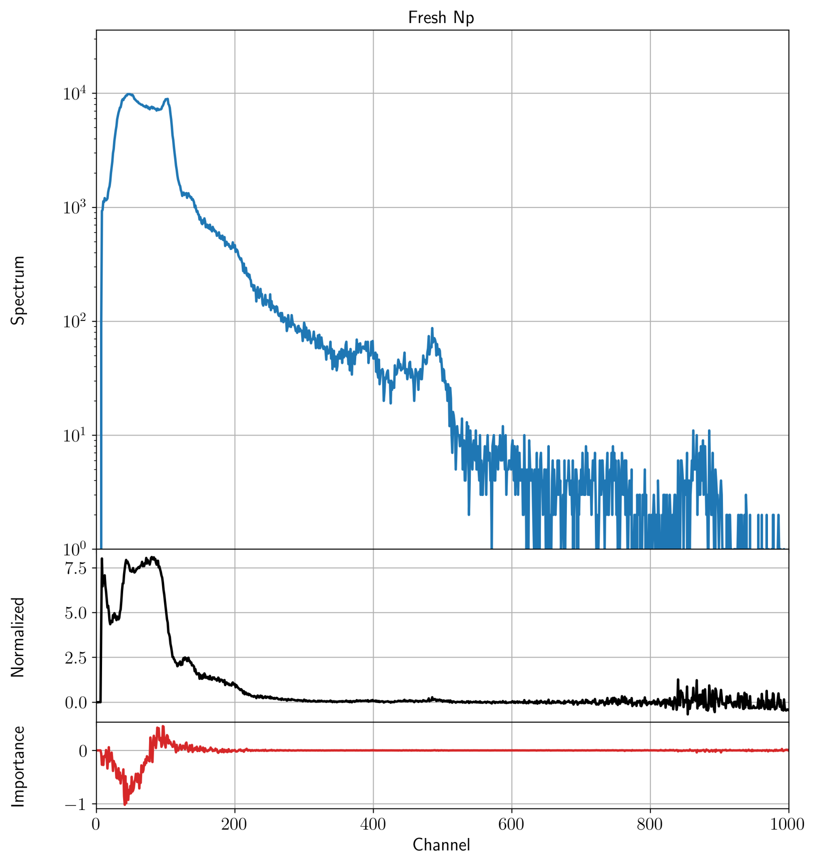


Figure 6.15: An original spectrum (blue) for a fresh Np transfer (nuclear material), the normalized feature vector passed to the representation model (black), and the integrated gradient, or feature importance (red). Note that the model struggles to identify the OOD features of this spectrum, leading to a false negative labeling.

anomalous training data. This affects the resulting representation produced by the encoder, with an unclear effect on the resulting label classification. In Figure 6.15, this spectrum's foreign features result in a misclassification as a byproduct. One possible solution is to include example spectra with these features in the self-supervised training set to reduce the OOD likelihood. Alternatively, it is possible these extremely deviated spectra are not characteristic of the distribution for their given label, and should be considered an "other" class.

6.5.3 Augmentation Importance Study

So far, the augmentations detailed in Chapter 5 are used with the assumption that they are label-invariant and provide some level of meaningful learning to the contrastive training process. Are any augmentations harmful to successful contrastive training (as measured by transfer detection accuracy), in which case they should be removed from the transformation pool? Alternatively, are any of the augmentations predominantly responsible for contrastive learning, in which case they should be preferentially utilized?

To quantify the value of each designed augmentation, models were trained on a specified combination of augmentations. First, a model was trained using an individual augmentation contrasted against the original spectrum. Second, a model was trained on pairs of augmentations for all enumerated combinations. In all iterations, an MLP[4096, 4096, 4096, 4096] was trained for 100 epochs, with 100 batches of 512 samples per epoch. Adam [53] was used as an optimizer and the classifier was trained and validated every 10 epochs. No labeled data was used in representation learning (i.e. representation learning was purely self-supervised). Note that the final balanced accuracy results are taken as-is after training a model one time (i.e. no averaging occurs). It is possible (and likely) that these results suffer from local optimization minima and could be stabilized by averaging over several model training iterations. The resources for such an experiment were not available, and the results presented here should be sufficient for lending a sense of how these augmentations compare to each other, even if they are not quantitatively precise.

Training loss for each augmentation regime is illustrated in Figure 6.16. The largest magnitude augmentations are consistently Gain Shift, Background, Signal-to-Background, and Resolution (in no particular order). This may be unsurprising because each of these transformations can significantly affect a spectrum across many—if not all—energy bins. The only one that is concerning is Resolution, since this does not affect the spectrum more than by an individual photopeak to the same degree as, say, the Signal augmentation. This suggests that Resolution might be incorrectly estimating the peak parameters and overwriting too

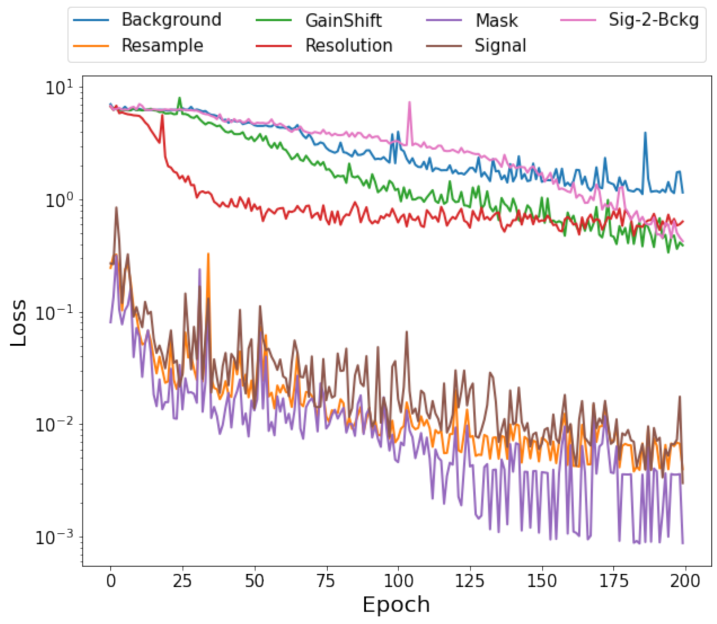


Figure 6.16: Each curve represents the contrastive loss of a self-supervised representation model using an individual augmentation to contrast an augmented spectrum from the original version.

Table 6.6: Relative change in loss over training epochs.

Mask	-98.92%
Signal	-98.89%
Resample	-98.38%
GainShift	-94.21%
Sig2Bckg	-93.75%
Resolution	-90.52%
Background	-83.60%

much of the spectrum, causing the representation model to learn incorrect information from a bad augmentation. The other augmentations (Signal, Resample, and Mask) have a smaller loss magnitude because they do not affect the spectrum as widely.

Each experiment results in nearly one full order of magnitude decrease in loss magnitude. In decreasing order of optimization extent (see Table 6.6): Mask, Signal, Resample, Gain Shift, Signal-to-Background, Resolution, and Background. Notice that the augmentations that have the smallest magnitudes result in the largest decrease in relative magnitude.

Compare these numbers to the test accuracy results for the individual augmentations as shown in the table below. In decreasing order of best balanced accuracy score (see

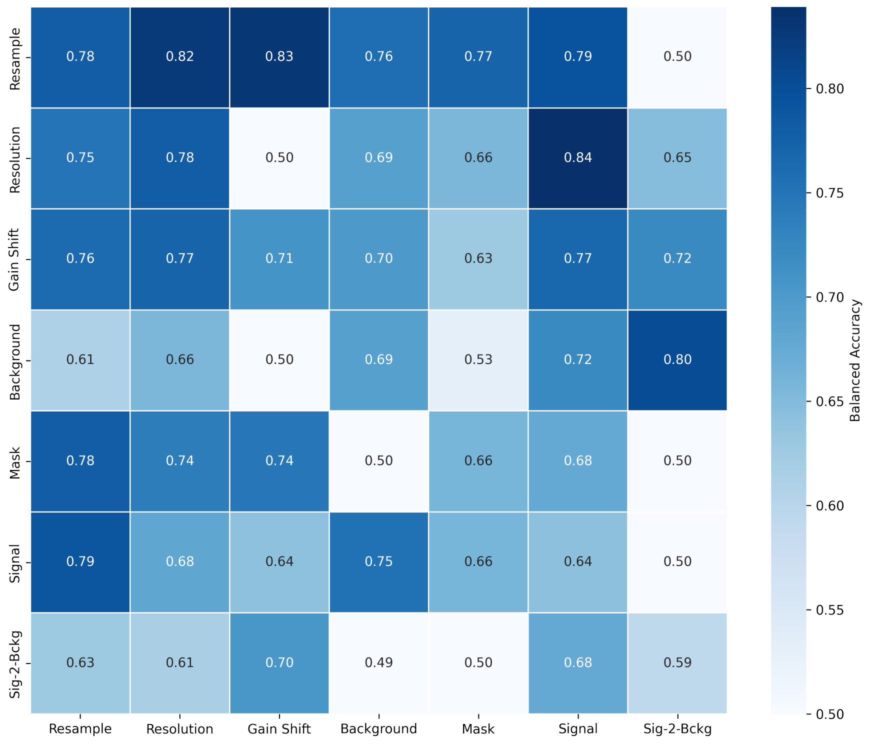


Figure 6.17: Summary results for studying the importance of individual augmentations. Each value corresponds to the final balanced accuracy of a classifier using a trained representation model with the specified augmentations. Iterations along the diagonal involve one augmentation contrasted again the original spectrum. Iterations off the diagonal contrast two augmentations against each other.

Figure 6.17): Resample, Resolution, Gain Shift, Background, Mask, Signal, and Signal-to-Background (this is approximately commensurate with the raw accuracies). Signal and Mask, despite having the highest decrease in relative loss, have comparatively poor balanced accuracies (contrasted with Resample, which has relatively high optimization and balanced accuracy scores).

It perhaps makes sense that Resample, Resolution, Gain Shift, and Background are the top four augmentations (in that order) for balanced accuracy since they represent the largest changes in a measured spectrum for varying environmental and detection scenarios: Resample for different counting statistics, Resolution for different detectors, Gain Shift for different post-processing calibrations, and Background for different detection environments. Mask may not be physically realistic or provide insight on labeling information, Signal may not provide a diverse enough set of new photopeaks, and Signal-to-Background may be redundant with Background.

Recent research has shown that successful SSL used for anomaly detection synergizes augmentation type and the physical mechanisms that generate anomalies [54]. That is, data augmentations either serve to bolster normal unlabeled data or accentuate expected anomalies in the training data. Here, the unlabeled data used for contrastive training contains weakly anomalous background spectra, other anomalous events, and unlabeled material transfers. Because each augmentation type is designed with detector physics in mind (see Chapter 5), they are aligned with the environmental characteristics that would generate background and/or transfers/anomalies. The captured physical behavior should result in a deeper representation state space that improves accuracy over classification on spectral dimensions directly.

Other studies [55] have demonstrated that examples that have small contrastive loss (i.e. their augmented instances are most similar to other augmented instances and their counterpart in a batch) contribute more overall to training a representation model and classification performance. Samples with small contrastive loss act as anchors for latent and generalized classes in optimizing similarity and dissimilarity of representations. In a data-constrained environment, similar samples can be used for contrastive learning and complex examples can be used for supervised training of a downstream classifier. This reasoning could be extended to augmentation types, where augmentations that lead to more loss minimization are less effective for downstream classification. The results here are inconclusive, however, since the top four augmentations for change in loss over training are evenly split between higher and lower accuracy compared to other augmentations. A study using cleaner, less variable, and more highly characterized data may be able to accurately identify the relationship between augmentation type and degree of contrastive loss.

In summary, while some augmentations appear to be strong candidates for use in contrastive machine learning on radiation spectra (e.g. Resample), no individual augmentation appears to be self-sufficient for success. For example, Resample combined with Gain Shift achieves a higher balanced accuracy score (82.90%) than Resample alone. The interplay between augmentations can be difficult to quantify. Background-Sig2Bckg has really good balanced accuracy, but Sig2Bckg-Background is less than 50%. Further analysis may suggest preference for augmentations. For example, even if Resample and Resolution are both valuable augmentations, maybe it is preferable to use Resample 80% of the time.

6.6 Semi-Supervised Contrastive Learning

How can labeled data be effectively leveraged by a contrastive learning framework? Labeled data can be used in contrastive learning by using the additional loss term in the contrastive loss function (see Equation (4.4)). Loss for those labeled samples included in the contrastive training corpus will penalize difference in representations generated from augmented instance of those labeled data. Multiple positive pairs can be present in a batch if multiple spectra for the same class are included as labeled data, leading to losses penalizing difference in output for all samples of the same class. The value added by including labeled data in representation learning must be evaluated, especially when labeled data is limited in nonproliferation regimes.

For all the experiments discussed here, there are three bodies of labeled data used in a training experiment: representation learning, training a classifier, and testing a classifier. Each experiment below is denoted with these splits. For example, an experiment that uses 5% of labeled data for representation learning, 10% for training a classifier, and 85% for testing is denoted as 5/10/85. All splits are in addition to the unlabeled data typically used for self-supervised contrastive learning, spanning several nodes and months' worth of weakly anomalous spectra identified with anomaly detection. The available labeled data remains

Table 6.7: Labeled data for representation learning.

Trial	**70/10/20**		α
	Accuracy	Balanced Accuracy	
1	67.92	71.71	1
2	69.81	71.28	
3	64.15	67.23	
4	69.81	73.06	
Average:	67.92	70.82	■

the 261 manually labeled spectra generated from the ground-truth of MINOS.

6.6.1 Initial Observations

Labeled data is expected to act as a regularization term (with hyperparameter α) by guiding representation output towards the task-specific context associated with the labels provided. Representations produced by semi-supervised contrastive learning should have accentuated features related to task-specific labels. Representations with these patterns should be more valuable input to a classifier attempting to draw decision boundaries for identifying different types of material transfers. For example, initial testing primarily using labeled data for representation learning (70/10/20) showed final balanced accuracy scores greater than accuracy (see Table 6.7). However, plentiful labeled data is not typically available in nonproliferation, therefore assigning the majority of labeled data to training a representation model may not be optimal.

6.6.2 Allocating Limited Labeled Data

Since this work is primarily motivated by limited labeled data in nuclear nonproliferation applications, should labeled data be prioritized in guiding contrastive learning or training a supervised classifier? Several experiments were conducted for varying training data splits and values of α. For all experiments, training occurred with default parameters (see Table 6.8). For all splits, ten trials were executed. 0/20/80 had an extra trial and trial 9 failed for

Table 6.8: Semi-supervised contrastive learning experiment parameters.

Parameter	**Value**
Learning Rate	1×10^{-2}
Batch Size	512
Epochs	100
Batches per Epoch	100
Hidden Layers	4
Hidden Layer Size	4096

5/15/80. The accuracy, balanced accuracy score, average, and difference from the baseline of not using semi-supervised contrastive learning are listed in Table 6.9.

Perhaps surprisingly, not using any labels in representation learning achieves the highest accuracy. The 10/10/80 and 15/5/80 splits have significantly worse performance metrics on average than 0/20/80. Using an asymmetric amount of labeled data in semi-supervised contrastive learning compared to training a classifier (5/15/80) achieves higher accuracy than 15/5/80 or 10/10/80 and is only marginally worse than the baseline experiment. Almost none of the trials exhibit the greater balanced accuracy score than accuracy behavior observed in Table 6.7. There also appears to be no discernible effect on final accuracy when changing the value of α.

If it was, then a larger α would lead to higher accuracy as the representation model becomes increasingly aligned with the downstream classification task.

When labeled data is scarce, it is crucial to prioritize training the classifier over consistency regularization in representation learning. Without sufficient data, a classifier struggles to determine a decision boundary regardless of how useful representations are for the specific task. For example, most of the trials in Table 6.7, with only 10% of labeled data used for training a classifier, have worse accuracy than the baseline trials in Table 6.9 that use 20% despite using most of the labeled data (70%) for semi-supervised contrastive learning. According to these results, semi-supervised contrastive learning is not advantageous when labeled data is limited but should instead be fully used in building a classifier. Unlabeled data is sufficient for constructing a representation model. See Section 6.4 for a discussion on

Table 6.9: Semi-supervised contrastive learning with limited labeled data experimental results.

Trial	**0/20/80**		**10/10/80**		**15/5/80**		**5/15/80**		α
	Accuracy	Balance Accuracy	Accuracy	Balanced Accuracy	Accuracy	Balanced Accuracy	Accuracy	Balanced Accuracy	
1	73.21	67.74	64.59	55.71	76.56	71.94	79.9	80.65	0.5
2	72.73	71.91	67.94	63.07	61.24	55.57	66.99	62.38	
3	74.16	72.03	71.77	68.97	66.51	57.08	68.42	66.57	
4	78.95	74.55	49.76	39.23	74.16	76.09	78.95	76.81	1
5	77.99	75.22	64.11	59.88	69.38	61.84	73.68	70.34	
6	74.64	71.92	69.86	66.69	64.59	59.77	71.29	68.17	
7	73.68	73.49	63.64	65.4	75.12	73.17	70.81	71.89	
8	78.95	72.3	71.29	62.31	70.81	58.36	75.6	68.1	1.5
9	73.68	66.73	72.73	73.26	65.55	57.75	-	-	
10	65.07	67.78	56.94	58.35	79.9	82.91	74.64	66.06	
11	77.99	72.52	-	-	-	-	-	-	
Average:	74.64	71.47	65.26	61.28	70.38	65.45	73.36	70.11	
Difference from 0/20/80:			-12.56%	-14.25%	-5.71%	-8.43%	-1.71%	-1.91%	

the benefit of using a representation model over the original spectra as inputs for detecting material transfers.

6.6.3 Allocating Noisy Labeled Data

If available labeled data are noisy (i.e. mislabeled or containing obfuscated labeling information), can they be used in training SSML models? The value of labeled data allocated to contrastive learning versus training a classifier in Section 6.6.2 above is difficult to assess because there are two independent variables. Furthermore, an optimal allocation strategy is difficult to determine because the amount of labeled data for training is so limited (and therefore marginal changes in allocation result in only marginal effects). The following experiment attempts to amplify the effects of labeled data on contrastive learning, using more of the subset for training.

Here, the model architecture remains the same, but the number of batches per epoch of contrastive learning is reduced. Instead of 100 batches per epoch, there are approximately ten batches per epoch, with the number of unlabeled data being reduced to 7.5% of the original dataset: approximately 5100 spectra. Of the labeled data, 50% is allocated to testing the model, the other 50% is allocated to training the classifier and representation model, enumerating between all options in steps of 10%. Since there are 261 labeled spectra in total, a step of 10% corresponds to a change of about 26 spectra in a training subset. Because each positive pair produces one contrastive loss pair, there is less unlabeled data, and passing multiple labeled spectra of the same class (n) produces $2n(2n-1)$ positive pairs, the effect of labeled data on contrastive learning should be more pronounced in every iteration. For every allocation strategy, $\alpha = 1$. The results of ten iterations for every combination can be seen in Table II.1.

A summary of the average accuracies and average balanced accuracies can be found in Table 6.10 and Table 6.11, respectively. The individual trials are reported in Appendix II. If the notion that more labeled data always helps a classifier achieve higher accuracy is true,

Table 6.10: Accuracy results for semi-supervised contrastive learning with different allocation strategies.

Average Accuracy		**Representation Model**				
		0.00%	10.00%	20.00%	30.00%	40.00%
	10.00%	75.651	72.827	72.977	76.185	75.496
	20.00%	77.1	76.871	76.795	75.575	-
Classifier	30.00%	75.039	77.863	75.42	-	-
	40.00%	78.398	78.932	-	-	-
	50.00%	77.253	-	-	-	-

Table 6.11: Balanced accuracy scores for semi-supervised contrastive learning with different allocation strategies.

Average Accuracy		**Representation Model**				
		0.00%	10.00%	20.00%	30.00%	40.00%
	10.00%	71.475	68.809	68.149	71.579	72.976
	20.00%	72.799	72.774	73.209	71.14	-
Classifier	30.00%	72.155	72.736	71.659	-	-
	40.00%	73.452	74.256	-	-	-
	50.00%	71.786	-	-	-	-

then the accuracies in Table 6.10 and Table 6.11 should be expected to increase monotonically down a column, across a row, or otherwise diagonally. These behaviors are not strictly observed, and in some cases increasing labeled data can be harmful to accuracy (see allocations where the classifier is trained on 10% of the labeled dataset). Although it is possible that some allocation strategies suffer from outlier iterations, this suggest some limitations in using labeled data.

Unlike the highly characterized and curated toy datasets used in fundamental machine learning research, the collection of labeled data used here are noisy. Noise in labeled data is either the result of mislabeling (unlikely given that an SME prepared this dataset with reference to ground-truth) or spectra that do not exhibit prominent features (e.g. photopeaks) expected for a given label. For example, a transfer spectra labeled for irradiated Np should exhibit signatures associated with the decay products and radioactive isotopes present in such a nuclear material transfer. If these features are not present, or occluded, in a labeled spectrum due to a variety of reasons (e.g. shielding, distance between detector and source), then contrastive learning on its positive pairs will incorrectly make representations similar or dissimilar.

Recent work has shown that contrastive machine learning can be harmed by semi-supervision with a subset of labeled data if the degree of noisy labeling is nontrivial [56]. This is likely behind the limitation in the study here. The labeled data used here has some degree of noisy labeling, or at least labeling without defined class features in an observed spectrum. Weakly labeled samples passed to the representation model could have an outsized effect on regularizing representations to a form less amenable to a decision boundary that outcompetes the value added in the context of correctly labeled samples. The noise rate for this labeled dataset is unknown and permeates both training and testing data. Even when no labeled data is used for contrastive learning, allocating more labeled data to training the classifier does not necessarily increase accuracy. This variability indicates that the additional 26 spectra added to the classifier's training subset are introducing more noise, thereby decreasing transfer detection accuracy.

The representation model is perhaps less generalizable because it is training on less unlabeled data. The variability in Table 6.10 and Table 6.11 is likely explained by a nontrivial noise rate in labeled data that outcompetes the added benefit of context introduced by labeled data used in contrastive training. Therefore, unless the labeled data of transfer spectra are highly characterized—or curated in such a way as to limit noise—semi-supervised contrastive learning can actually harm transfer detection, and the optimal allocation strategy remains committing labeled data to supervised training of a classifier. Particularly when labeled data is limited (as in Section 6.6.2), self-supervised contrastive learning appears to be most advantageous, as the benefit of context is outweighed by the confusion of noise.

6.7 Conclusion

This chapter represents a summation of several key components researched in designing a contrastive learning framework for detecting and characterizing nuclear material transfers. Unlabeled and labeled spectra from MINOS are used in the self-supervised contrastive training of an encoder and supervised training of a transfer classifier, respectively. This framework uses a set of spectral augmentations that aide a model in learning features and signatures associated with different types of (transfer) spectra.

An example model is presented. The task-specific classifier is trained on an extremely small sample of seven labeled spectra to conduct binary classification of transfers: byproduct or tracked nuclear material. Promising balanced accuracy is achieved, with the model showing acute sensitivity for detecting irradiated material transfers. The encoder produces representations that improve the separability of transfer subclasses, providing a richer state space for an ML model to create a decision boundary. While it is difficult to assess whether the model is learning classification patterns that are associated with distinct radioactive sources, a method of feature importance is used as a diagnostic for relating input spectra to a model's output classification.

Overall, this model decisively outperforms other traditional ML methods and advocates for a more systematic approach to AI approaches for nuclear nonproliferation. The augmentations designed and implemented here all contribute to a performant detection model, and generally capture the behavior expected in radiation detection. One important challenge in nuclear security is optimal allocation of limited labeled data and limited contextual, or ground-truth, information. When assigning labeled data, the level of noisy labeling is crucial and must be considered. If training data is non-negligibly misclassified, or labeling features/information is obfuscated, increasing the amount of labeled training data can hinder model performance rather than help it. When labeled data is limited, it is more important to allocate data to training a robust classifier than encouraging task-specific representations, especially when noise can confuse representation outputs. The question of efficient resource allocation is explored further in the next chapter, where hyperparameter optimization is used to create a model that is defensible to variability in data and model architecture.

Chapter 7

Hyperparameter Optimization

The results presented in Section 6.3 represent an individual model trained with a unique split of the labeled data used for training and testing. While the performance—as measured by accuracy—is quite good for this model, there is no guarantee that this is reproducible for different combinations of labeled training and testing data. Further, the architecture was chosen carefully, but arbitrarily, and there is no guarantee that a repeated training of the framework with the given hyperparameters but a different initialization would result in an equally performant model. The model in Section 6.3 ultimately represents a serendipitous result. Hyperparameter optimization and cross-validation are required to test the robustness of the system and measure accuracy metrics that account for these generalizable variabilities. In this section, hyperparameter optimization will be conducted using a system of Bayesian inference for selecting and optimizing hyperparameter values, just as in Chapter 3.

7.1 Method for Hyperparameter Optimization

Hyperparameters are iteratively optimized over the three main models of the framework: the encoder, projection head, and classifier. After each component, the respective hyperparameters are fixed and a fresh model is trained from scratch. That new model—trained with the best hyperparameters from its respective hyperparameter optimization trials—is used in optimizing all subsequent components/trials.

Table 7.1: Hyperparameters for encoder optimization.

Hyperparameter	Minimum	Maximum
Base Learning Rate	1×10^{-5}	0.5
# Layers	1	7
Temperature (τ)	0.1	0.9
β_1	0.7	0.99
β_2	0.8	0.999
Weight Decay	1×10^{-7}	1×10^{-2}
Architecture (Choice)	MLP	CNN

7.1.1 Encoder

First, the encoder's architecture and individual hyperparameters are optimized. A table of hyperparameters and the range of values tested are listed in Table 7.1. All hyperparameters except the layer size and architecture choice are assigned by the optimization algorithm: Adam [53]. Hyperparameter values are selected from a uniform distribution within the range of specified values. The minimization objective is the final contrastive loss after contrastive, self-supervised training. That is, the best model will have the smallest contrastive loss with the default training criteria (see table Table 7.2). In total, 351 trials of hyperparameter optimization were conducted.

In determining the encoder's architecture, first the number of hidden layers and whether an MLP or CNN will be used is chosen. Once these hyperparameters are selected for a given trial, the size of each hidden layer is randomly selected from a pre-selected list of values. For MLP, the possible hidden layer sizes are: 512, 1024, 2048, and 4096. For CNN, the possible hidden layer sizes are: 8, 16, 32, 64, and 128. These values are chosen based on typical convention in ML research (that is, values of 2^n). Not all combinations of number of layers and hidden layer sizes will result in an encoded representation that is larger than the length of the original spectra, so this is implicitly tested during hyperparameter optimization. Certainly, more combinations or possible hyperparameter values could be tested given more time and computing resources.

The best hyperparameters can be seen in Table 7.5. The best overall model is a CNN,

Table 7.2: Hyperparameter optimization training criteria.

Parameter	Value
Epochs	100
Batch Size	512
Batches per Epoch	66

Table 7.3: Hyperparameters for projection head optimization.

Hyperparameter	Minimum	Maximum
Base Learning Rate	1×10^{-5}	0.5
Temperature	0.1	0.9
β_1	0.7	0.99
β_2	0.8	0.999
Weight Decay	1×10^{-7}	1×10^{-2}
Projection Dimension	8	1024

for a final loss of 1.5253 and an architecture of CNN[64, 32, 64, 8, 128, 64] leading to a final representation length of 832. The best MLP had a final loss of 1.8635 and an architecture of MLP[4096, 2048, 4096, 2048]. Note that this is very similar to the original architecture used in Section 6.3, just with fewer neurons in two layers and half the final representation length.

7.1.2 Projection Head

Next, the projection head and dimensions are optimized. Remember, the projection head is used to reduce the dimensions of the encoded representations for computing a more stable contrastive loss. The hyperparameters optimized here are listed in Table 7.3.

This optimization is conducted using the best CNN and the best MLP encoder. For each case, 50 trials are completed. The minimization objective here is now a summation of the final contrastive loss and one minus the final balanced accuracy score computed on a freshly trained validation classifier. This classifier is still a linear model, trained on 20% and tested on 80% of the labeled data. This combination of self-supervised training and domain-specific task evaluation in the minimization objective gently starts to encourage models that can accurately detect nuclear material transfers, not just models that are contrastively

Table 7.4: Hyperparameters for classifier optimization.

Hyperparameter	Minimum	Maximum
Learning Rate	1×10^{-6}	1×10^{-2}
Regularization Weight	0.0	1×10^{-6}

robust.

The best hyperparameters can be seen in Table 7.5. After 50 trials each, the best CNN has a contrastive loss of 1.069 and a balanced accuracy of 67.07%, with a projected dimension of 155. The best MLP has a contrastive loss of 1.2155 and a balanced accuracy of 60.36% with a projected dimension of 527.

7.1.3 Classifier

From this point onward, only the MLP encoder and its accompanying projection head will be used in optimization. Even though the best CNN has smaller contrastive loss and higher balanced accuracy, the MLP is more directly comparable to the model used in Section 6.3 since they only differ by a few hidden layers and hyperparameter values. Future work could repeat this exercise for the best CNN, although the results are expected to be similar.

The two hyperparameters optimized are listed in Table 7.4 and belong to the Adam optimization algorithm from PyTorch with L2 regularization. Simultaneously, 20 parallel iterations of 1000 serial trials are conducted and then merged for a total of 20000 trials. Training and testing a linear classifier is much less computationally expensive than training the representation model, hence the significantly larger number of trials despite less hyperparameters. The best hyperparameters can be seen in Table 7.5.

In each trial, 5-fold cross-validation is conducted. That is, the labeled data is shuffled and then split into five subsets. For each fold, one of the subsets is used to train the classifier while the other four are used to test it. One minus the average balanced accuracy score for the five folds is used as the minimization objective. Cross-validation avoids the reported performance metrics from being biased towards specific split in the labeled dataset. Note

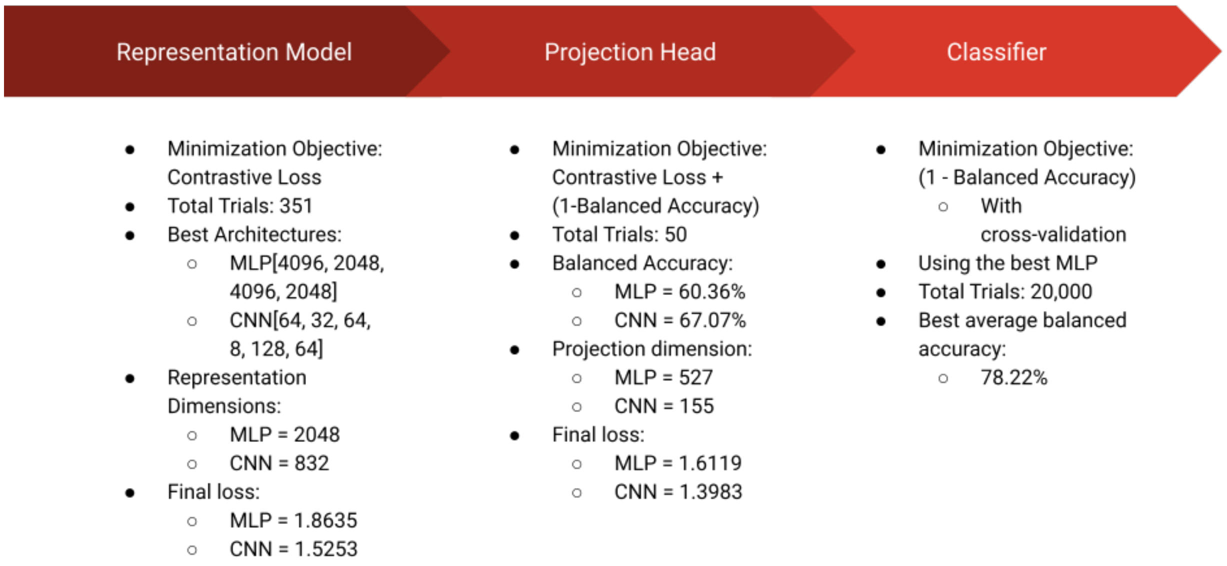
Table 7.5: Optimized hyperparameters.

Hyperparameter	Encoder (CNN)	Encoder (MLP)	Projector (CNN)	Projection (MLP)	Classifier
Base Learning Rate	0.3953	0.3663	0.09770	0.4412	
# Layers	6	4			
Temperature (τ)	0.1073	0.1080	0.1418	0.1263	
β_1	0.7792	0.9280	0.8841	0.7835	
β_2	0.9562	0.8659	0.8492	0.9462	
Weight Decay	0.0003536	0.008326	0.008738	0.002930	
Projection Dimension			155	527	
Learning Rate					1.581×10^{-4}
Regularization Weight					7.939×10^{-7}

that the 20-80 split still maintains a limited training data regime.

7.1.4 Summary

A summary of the workflow and results are also visualized in Figure 7.1. After 351 trials optimizing the encoder, the best CNN had an output representation length of 832. The best MLP had an output representation length of 2048. Using this architecture and the best (now fixed) hyperparameters, a new CNN and MLP encoder were freshly trained.



- Minimization Objective: Contrastive Loss
- Total Trials: 351
- Best Architectures:
  - MLP[4096, 2048, 4096, 2048]
  - CNN[64, 32, 64, 8, 128, 64]
- Representation Dimensions:
  - MLP = 2048
  - CNN = 832
- Final loss:
  - MLP = 1.8635
  - CNN = 1.5253

- Minimization Objective: Contrastive Loss + (1-Balanced Accuracy)
- Total Trials: 50
- Balanced Accuracy:
  - MLP = 60.36%
  - CNN = 67.07%
- Projection dimension:
  - MLP = 527
  - CNN = 155
- Final loss:
  - MLP = 1.6119
  - CNN = 1.3983

- Minimization Objective: (1 - Balanced Accuracy)
  - With cross-validation
- Using the best MLP
- Total Trials: 20,000
- Best average balanced accuracy:
  - 78.22%

Figure 7.1: A summary of the three components optimized sequentially for the contrastive learning framework. Each component conducted hyperparameter optimization using Bayesian inference to select given hyperparameters over a number of trials. The best parameters were selected and fixed, and a new model was freshly trained with the given parameters before the next sequence was executed.

These representation models were then used as the basis for optimizing the projection head. After 50 trials, the best projection head for the best CNN had a projected length of 155. The best projection head for the best MLP had a projected length of 527. For both types, this architecture and hyperparameters (now fixed) were used to freshly train an encoder and projection head.

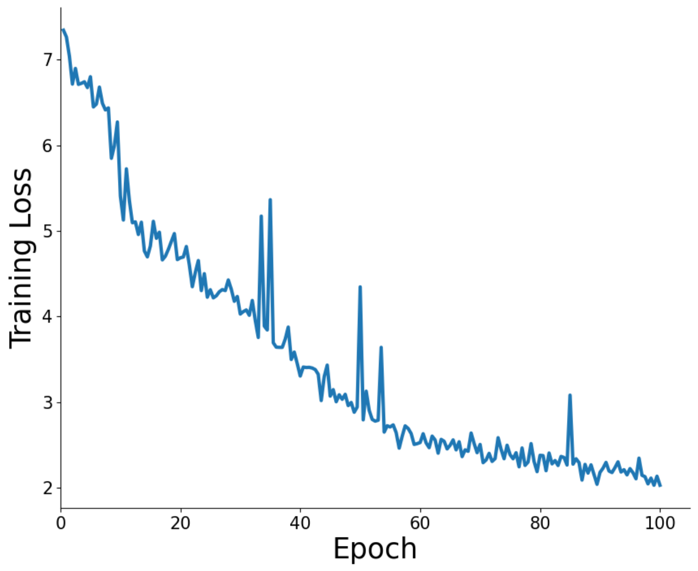


Figure 7.2: Contrastive training loss curve for the MLP encoder trained after optimizing the encoder and after optimizing the projector.

The training loss curve for the MLP model freshly trained after hyperparameter optimization (for the encoder and projection head) can be seen in Figure 7.2. As observed, the model has a steady convergence to a lower contrastive training loss over the training criteria. This MLP encoder trained after hyperparameter optimization is the representation model that will be used in optimizing the classifier and the detection performance results presented below.

The new MLP model is then used in optimizing the linear classifier, which has only two training hyperparameters (its architecture is fixed as a linear model). After 20000 trials, training a classifier for five different slices of the labeled dataset (i.e. cross-validation), the best average balanced accuracy score was 78.22%. This represents the best accuracy achieved for the given number of trials, conditioned for variabilities in hyperparameters and bias in data slices. In reality, the performance can vary significantly based on these values, but the best average balanced accuracy indicates the level of learning achievable by this system beyond the most rudimentary detection systems (e.g. a coin-flip).

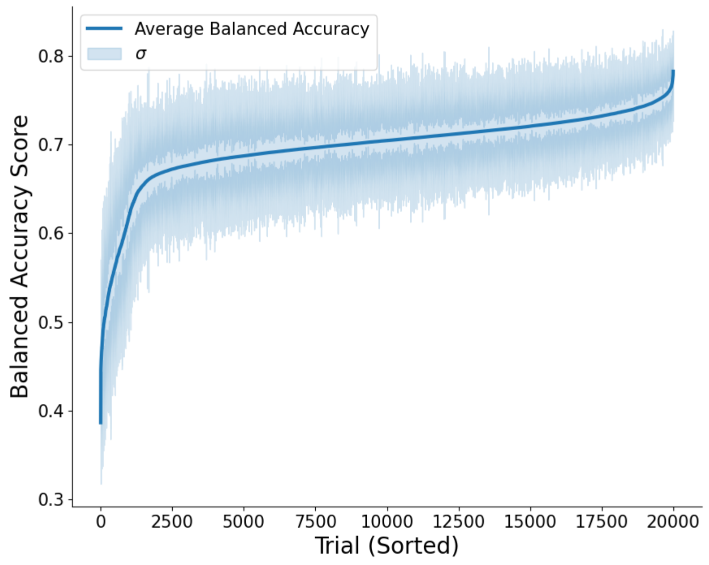


Figure 7.3: The classifier test results for all 20000 trials conducted in hyperparameter optimization for the classifier. Since each trial conducted 5-fold cross-validation, the average balanced accuracy (blue) is reported, and the trials are sorted according to this magnitude (i.e. this is not the serial order in which trials were executed). One standard deviation—as calculated by the five balanced accuracy scores reported for each trial—is plotted (blue shaded region). Note that the standard deviation is noisy, even for trials with comparable average balanced accuracies.

7.2 Transfer Detection Performance

The best model—as determined by maximum balanced accuracy score achieved—is benchmarked for comparison to the example result presented in Section 6.3. The accuracies of all classifiers trained during hyperparameter optimization of the classifier model are presented in Figure 7.3. The trials are sorted according to their average balanced accuracy. Since each trial also conducts cross-validation, meaning five separate classifiers are trained and tested on five separate folds of labeled data, the standard deviation is also plotted. Note that the standard deviation is quite noisy trial to trial. That is, the ceiling and floor can change drastically even for trials with similar average balanced accuracy (this causes the darker and lighter hue in shaded regions around the average curve).

The average balanced accuracy curve achieved by cross-validation for any given trial appears to have a range of values centered around 70%. A small portion of trials achieved

Table 7.6: Cross-validation balanced accuracy scores for the highest average achieved.

Fold	Balanced Accuracy
1	77.13
2	78.21
3	77.90
4	77.57
5	80.30

very poor performance, with a balanced accuracy score less than 50%. As is illustrated later in Figure 7.12, these trials correspond to extreme hyperparameter choices. The best average balanced accuracy achieved was 78.22%. The balanced accuracy score for every cross-validation fold for this best iteration are reported in Table 7.6. As a benchmark, the best model of the five trained and tested in cross-validation for this best trial will be used.

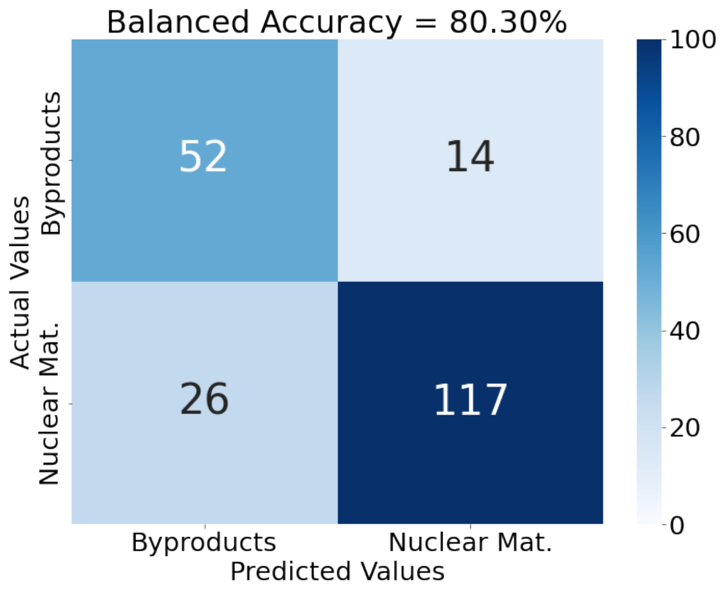


Figure 7.4: A confusion matrix for test spectra passed to the best model from the best trial trained after all sequences of hyperparameter optimization (encoder, projector, and classifier). This model's balanced accuracy score is 80.30%, tested on 80% of the labeled dataset. Compared to the previous model tested, this model seems more successful at detecting nuclear material transfers and less successful at detecting byproducts.

For the best model from the best trial, which achieved a balanced accuracy score of 80.30%, the confusion matrix for spectra tested is plotted in Figure 7.4. Note that whereas

97% of the labeled data is tested in Figure 6.6, only 80% are tested here. Whereas less of the tracked nuclear material transfers are misclassified as byproducts here, more of the byproducts tested are misclassified as tracked nuclear material transfers. The value of this tradeoff between false positives and false negatives will depend on several factors for an end user. Ultimately, a policymaker's unique tolerance for false positives and false negatives will lead to different adoption preferences for either of these two models (if any).

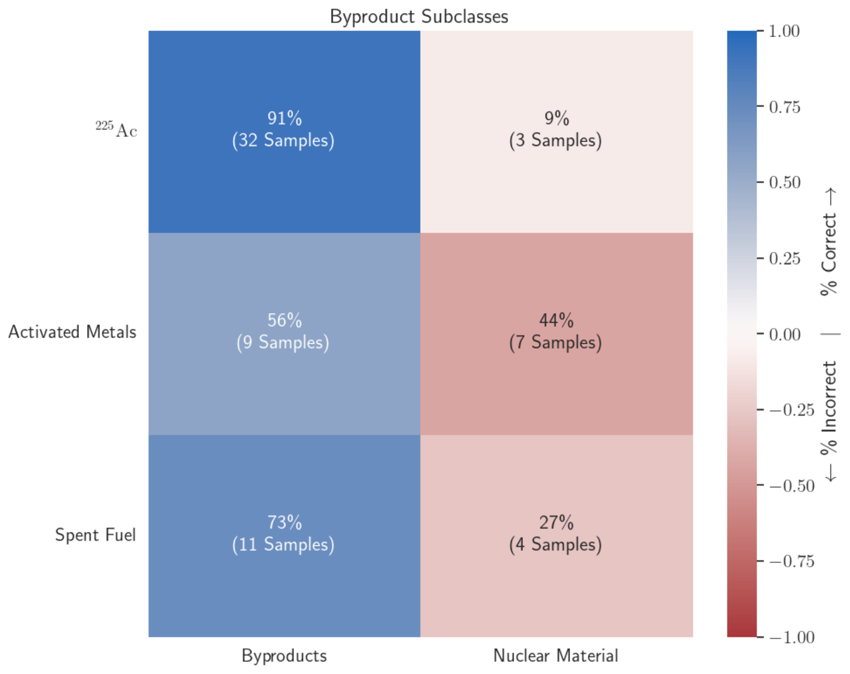


Figure 7.5: Classification accuracy summary for each subclass in the byproduct class (i.e. all instances have a true binary label of byproduct). In each cell, the percent (and gross number) of test spectra labeled a given binary class are listed. A higher true negative rate is shaded blue, a higher false positive rate is shaded red. The model appears to have the best success at detecting ^{225}Ac.

To analyze systematic misclassifications further, the test results categorized for each subclass are presented in Figure 7.5 and Figure 7.6 for byproducts and tracked nuclear material transfers, respectively. Before, the model appeared to have learned features that led to more successful detection of irradiated nuclear material transfers than unirradiated nuclear material transfers. Here, it appears that the model was more successful at detecting ^{252}Cf and irradiated Cm transfers in particular. This suggests some form of pattern recognition that is advantageous for these types of transfers.

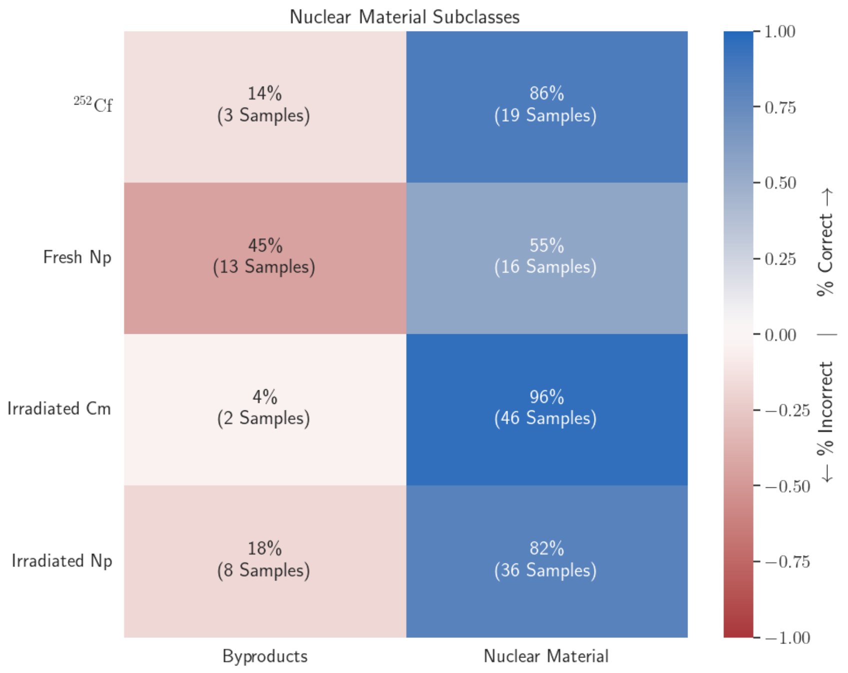


Figure 7.6: Classification accuracy summary for each subclass in the nuclear material class (i.e. all instances have a true binary label of nuclear material). In each cell, the percent (and gross number) of test spectra labeled a given binary class are listed. A higher true positive rate is shaded blue, a higher false negative rate is shaded red. The model appears to have better detection accuracy for ^{252}Cf and irradiated Cm than it does for Np transfers (unirradiated or irradiated).

It is important to note that unlike the model in Section 6.3, this classifier did not achieve 100% training accuracy, as shown in Figure 7.7 and Figure 7.8 for byproducts and nuclear material transfers, respectively. Instead, the model appears to achieve the best training detection accuracy for activated metals and spent fuel transfers despite (or perhaps because of) only having a handful of training spectra for those transfer types. The relation between the training misclassifications, testing misclassifications, and a definitive decision boundary are difficult to assess. The lower performance on training data may be the cause of success in testing data, or it could be to its detriment. Perhaps another version of this model—trained on the same data but at a different initialization—may perform better or worse. At the very least, this best model of the best trial from a large sample of hyperparameter optimization iterations presents a unique comparison with strengths and weaknesses that contrast it with the model presented in Section 6.3.

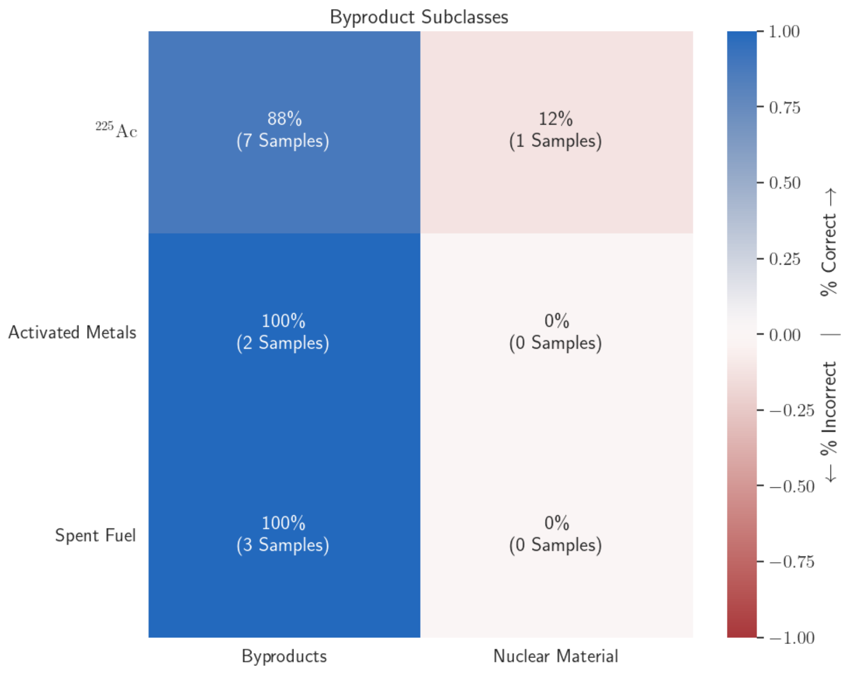


Figure 7.7: Training classification accuracy summary for each subclass in the byproduct class (i.e. all instances have a true binary label of byproduct). In each cell, the percent (and gross number) of test spectra labeled a given binary class are listed. A higher true negative rate is shaded blue, a higher false positive rate is shaded red. The model only has perfect training accuracy for activated metals and spent fuel, not ^{225}Ac.

The PCA conducted on representations produced by the encoder used in this analysis show promising results in Figure 7.9. Again, PCA is applied using the entire labeled dataset of 261 spectra. Distinct clusters of classes are starting to emerge when compared to Figure 6.10 and Figure 6.11. In Figure 7.10, these representations have explained variance ratios that are more distributed across many component dimensions when compared to the spectra directly or the encoded representations of Section 6.3. Further separation between classes and more feature space dimensionality both help to explain the increase in balanced accuracy. The classifier is likely taking advantage of the separation between representations in feature space to draw a decision boundary. Perhaps a more complex classifier than the linear model used here could exploit these dimensions for more complicated tasks, like distinguishing between all subclasses.

The features learned by this model are also exemplified by using integrated gradients

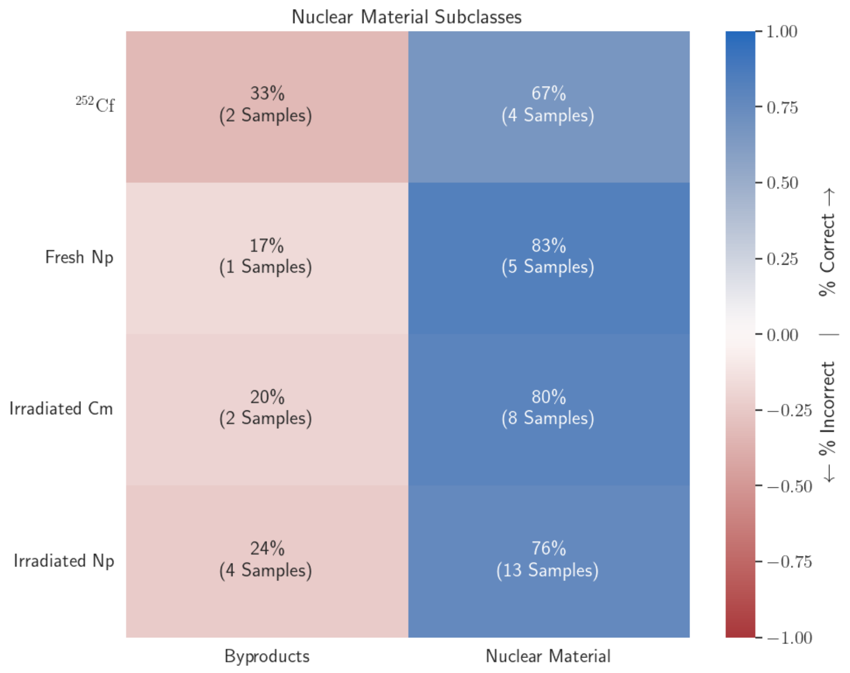


Figure 7.8: Training classification accuracy summary for each subclass in the nuclear material class (i.e. all instances have a true binary label of nuclear material). In each cell, the percent (and gross number) of test spectra labeled a given binary class are listed. A higher true positive rate is shaded blue, a higher false negative rate is shaded red. The model does not have perfect training accuracy for any of the tracked material transfer types.

for feature importance. Figure 7.11 shows a ^{252}Cf transfer spectrum that was correctly classified by the model as a tracked nuclear material transfer. Even though this spectrum deviates significantly from the distribution observed by the encoder, distinguishing features and photopeaks are difficult to identify in the original spectrum. The integrated gradient clearly accentuates signatures and shapes that it has learned for evaluating a spectrum for transfer detection (with the exception of channels at the extreme ends of the spectrum, where normalization and the model are most sensitive to changes in counts). This complicated structure appears to not only consider the presence or absence of features but also the shape of features (consider how the integrated gradient between channels 700 and 800 appears to be focused on a specific portion of the broader peak at that location). Directly relating each of these features and patterns to a radioactive signature would be difficult, but this example demonstrates how an increasingly optimized ML model is capable of learning features for

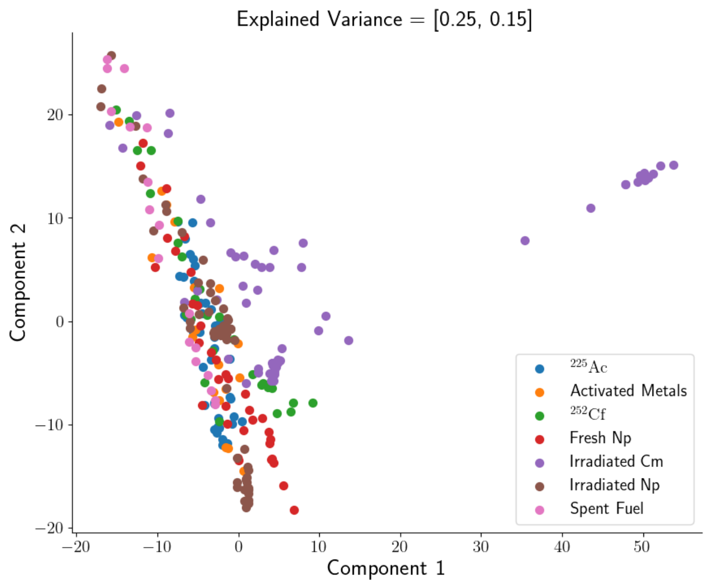


Figure 7.9: A two-dimensional PCA conducted on the representations generated by the encoder using the spectra from the labeled dataset. Each subclass is uniquely colored. Note that the difference transfer subclasses are visually distinct, and could be separable given more principal components (and therefore more explained distributional variance).

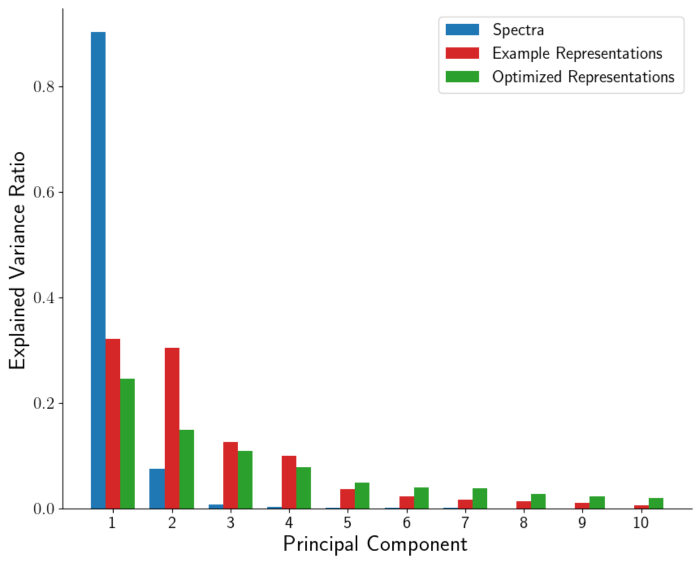


Figure 7.10: The first ten explained variance ratios in a PCA for spectra (blue), encoded representations from the example model in Section 6.3 (red), and encoded representations from the best hyperparameter optimized model via cross-validation (green). The ratios represent the percentage of distributional variance that can be explained by a given principal component. Note that the variance is spread across more components for representations than for spectra, most so for the optimized encoder model.

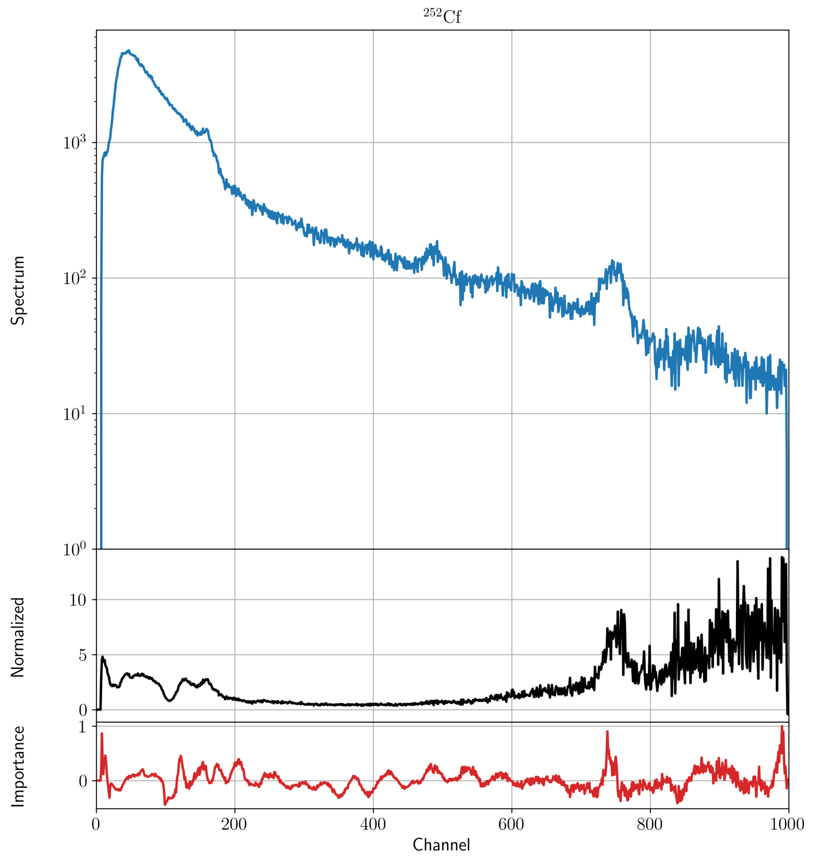


Figure 7.11: An original spectrum (blue) for a ^{252}Cf transfer (nuclear material), the normalized feature vector passed to the representation model (black), and the integrated gradient, or feature importance (red). Note that the model recognizes several complex structures and patterns in the spectrum and uses them for evaluating what kind of transfer this sample is.

detection and characterization.

7.3 Optimally Optimizing

As a final analysis, this section considers the role hyperparameters serve in creating an ML model. Hyperparameters are controls used in the training and construction of models, but do not generally have a physically realizable relationship with data. Setting these parameters typically requires a form of state space search because there is not an evidently optimal choice (except in trivial or toy scenarios). In fact, finding an optimal selection of hyperparameter values becomes infeasibly hard as the number of parameters increase, the model becomes increasingly complex, and the data used for training and testing becomes highly variable/noisy.

For example, even in the simplest cases, choosing a neural network architecture can be a challenging decision. There are few heuristics that guide the width and depth of a network, and they are rarely transferable to other domain data and tasks. Many network architectures are chosen by domain convention. The limits placed on architecture choices in the hyperparameters tested above were an attempt to limit the unbounded combinations for neural network architectures. If the optimization space is extremely non-smooth, covered in local minima with no indication of an optimal direction, how then can hyperparameters be assigned and when can maximizing performance (as measured by balanced accuracy) be considered sufficient?

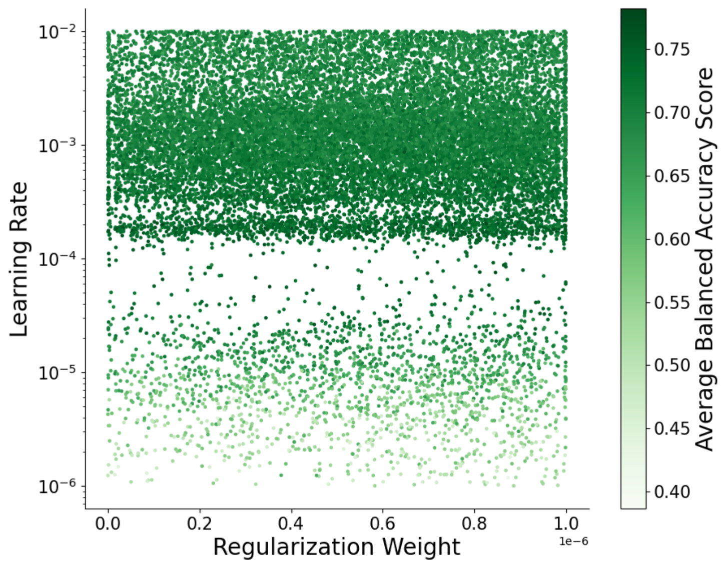


Figure 7.12: All classifier sequence trials plotted for the learning rate and regularization weight tested by hyperparameter optimization. Green hue is controlled by the average balanced accuracy score from cross-validation for an individual trial. Extreme learning rate values prevent the model from converging to a performant model in 100 epochs, discouraging Bayesian inference from choosing in that range. Within a reasonable range of hyperparameter values, there is no discernable correlation with resulting average balanced accuracy score.

Consider the final two hyperparameters optimized for the classifier, plotted in Figure 7.12.

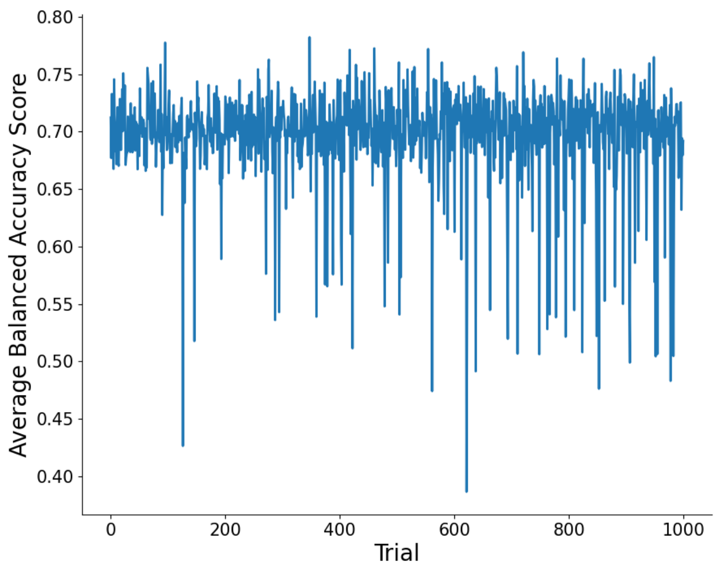


Figure 7.13: Average balanced accuracy score from cross-validation for every trial in order of completion. This particular run achieved the highest overall average balanced accuracy score (78.22%) out of 20 parallel processes that ran 1000 trials each. No systematic increase in balanced accuracy score is observed, although the highest average balanced accuracy score is achieved within the first half of the trials.

Extremely small values for learning rate have significantly poorer balanced accuracy score. This is likely because a smaller learning rate means the model converges more slowly during training. The model likely did not converge within the 100 training epochs, hence the worse performance. The Bayesian inference algorithm would therefore be discouraged from selecting smaller learning rates for subsequent trials.

Outside of these extremes but within a wide range of hyperparameter values, there does not appear to be a discernible correlation with higher average balanced accuracy score. That is, balanced accuracy score does not appear to be dependent on the value of these hyperparameters for a large range of values tested. This stochasticity means that even hyperparameter optimization algorithms that use Bayesian inference will be essentially random. For another view, see Figure 7.13, where the average balanced accuracy score is plotted in the order trials were completed. Even though hyperparameter optimization is attempting to maximize balanced accuracy score, no definitive increase is observed over time. So, how can one be sure to maximize model accuracy if it is stochastic with respect to hyperparameters?

First, if accuracy systematically depended on hyperparameter optimization, then very few trials would be necessary. The extreme value for each hyperparameter that maximizes accuracy could be selected by quickly following the gradient of optimization. Instead, hyperparameter optimization provides a mechanism by which a state space can broadly be sampled, and the best model selected from that sample. Due to all these variabilities (in data split, neural architectures, hyperparameters, etc.), the best model for one set of optimization trials may differ significantly from the best model of a different set of optimization trials. The optimization task then becomes a matter of how many trials are required to achieve a desired level of accuracy, as chosen by a user.

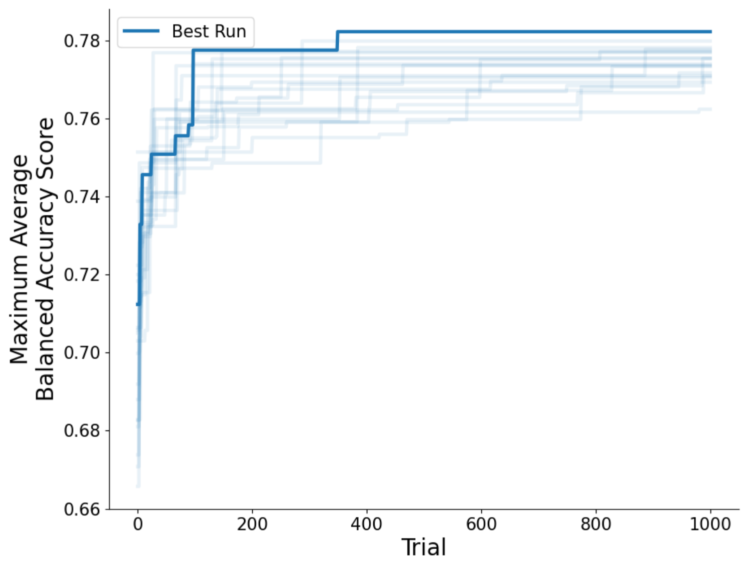


Figure 7.14: The maximum (blue) average balanced accuracy score from cross-validation achieved over all trials in order of completion. Each line represents a different parallel process—20 in total—completing 1000 trials each. The bolded line represents the process with the highest final average balanced accuracy score (78.22%) overall. The maximum accuracy achieved quickly converges to some asymptote that only varies by a couple percent between processes.

If a desired maximum level of accuracy is achievable (which is difficult to know definitively but related to the noisiness and separability of data), then how many trials are necessary to achieve it? See Figure 7.14, where the maximum average balanced accuracy scores achieved is tracked over all trials. The maximum accuracy achieved monotonically increases to some asymptote, which is rapidly reached in the first 500 trials; no change is observed in the remaining trials. This curve suggests a specific number of trials necessary to find a model that reaches a certain accuracy, given a set of training/testing data, but this is determined empirically and could be misleading for a different dataset or model construction. Perhaps an even higher balanced accuracy can be achieved with twice as many trials, or perhaps 78.22% is the absolute best possible average balanced accuracy score that can be achieved with cross-validation given a linear classifier, and a trained encoder.

A sequential system of hyperparameter optimization could be a suboptimal procedure for finding the optimal model and parameters. Given unlimited computing resources and time, a better system might be to optimize all hyperparameters and models simultaneously. However, additional hyperparameters included in an optimization algorithm are not necessarily linearly complex. In fact, because neural network architecture is so unbounded, finding an optimal model structure with optimized hyperparameters is likely intractable. Further, minimizing contrastive loss is not perfectly correlated to maximizing balanced accuracy (and is highly dependent on combinations of labeled training and testing data). Sufficiently balancing these two objectives would require experimentation for any given dataset (like here).

7.4 Conclusion

In summary, hyperparameter optimization was conducted in three sequential steps. First, hyperparameters controlling the encoder—including rudimentary choices of neural network architecture—were tuned by minimizing the final contrastive training loss after 100 epochs. Second, the projection head, which determines the dimensionality of representations when computing contrastive loss, is optimized by minimizing the final contrastive training loss after 100 epochs and maximizing the balanced accuracy score achieved with a linear classifier. Finally, the two hyperparameters controlling the linear classifier model are tuned by maximizing the average balanced accuracy score achieved by cross-validation using 20% of the labeled dataset for training and 80% for testing.

The best model from the best trial (as measured by balanced accuracy score) is selected for comparison to the model trained and tested with limited labeled data in Chapter 6. This model shows improved signs of transfer detection and characterization, despite a weaker balanced accuracy score. The model seems suited for detecting transfers of unirradiated and irradiated Np transfers. The encoder is particularly stronger in producing robust representations, which indicate increased separability in a PCA analysis. Integrated gradient feature importance suggests certain patterns are learned and used by the model for evaluating a spectrum (and its subsequent representation) for transfer detection, although physical correlations are hard to connect.

In determining computing resource allocation for achieving desired success in hyperparameter optimization, a few lessons and heuristics are discussed. Presumably, as the number of trials approaches infinity, the maximum achieved balanced accuracy approaches the global maximum. (This is the global maximum for the given training/testing dataset, there is no guarantee this will be the global optimum for another dataset for a field-deployed model!) How many trials are necessary in the search for an optimally performant model will depend on a given policymaker's need for accuracy, tolerance for computation time, and supply of computational resources. The purpose of hyperparameter optimization is not to find *the* optimal assignment of hyperparameters, but generate a set with a tolerably optimal model that is acceptable given resources available.

Chapter 8

Conclusion

This chapter offers some concluding remarks, including a summary of the research conducted, conclusions and insights that were reached, and a brief overview of future work this research can precipitate.

8.1 Summary

The work conducted here has been primarily motivated by a challenge of limited labeled data in nuclear nonproliferation applications. That is, even when data generation is feasible, carefully characterizing the necessary volumes of labeled data for use in machine learning applications can be challenging. Gamma spectra, despite being an essential observation modality, often require manual analysis to determine present nuclear sources. Manual analysis can be time consuming, highly variable, and dependent on subject-matter expertise. These problems might become intolerably resource intensive when the number of spectra needed to adjudicate alarms or anomalies increases, either because of an increasingly large sensor network or longer collection periods.

In essence, the problem is circular. Artificial intelligence has potential to alleviate the cost from the manual analysis of spectra, but the current regime of manual analysis is too costly to generate the necessary training data to implement artificial intelligence systems. Therefore, this work explored modern machine learning techniques—specifically semi-supervised machine learning—to develop a methodology for making the analysis of radiation spectra for nuclear security more efficient. To demonstrate this capability, semi-supervised machine learning is used on gamma spectra from the MINOS testbed at ORNL (collected with limited ground-truth) to detect and characterize the transfer of shielded radiological material. Summary accomplishments and conclusions are highlighted below.

If persistent monitoring can generate significant amounts of unlabeled data, is there value in using unlabeled data for detection tasks?

Preliminary work demonstrated that unlabeled data used in semi-supervised machine learning methods can increase detection accuracy compared to purely supervised methods relying on only small amounts of labeled data. Three established SSML techniques were trained using a combination of data types, and tested to outperform supervised logistic regression when characterizing material transfers from other types of anomalous radiation events. The distributional context provided by unlabeled data allows SSML methods to utilize assumptions about the relationship between observations, regardless of their labeling status. That is, the SSML methods perform better by either enforcing consistency in model output over the feature space or by extrapolating manifold information from the data distribution. This experiment demonstrated that SSML methods can be successful even when labeled data is limited (where traditional methods would normally fail) and can alleviate the otherwise required cost of labeling training data.

How can contrastive learning be used with radiation spectra?

In preparing a contrastive learning framework, several unique data augmentations were designed and implemented for gamma spectra. These augmentations capture physical phenomena that occur in radiation detection and are used to stochastically perturb measure spectra. For example, replacing the underlying background distribution (which first requires an estimation of the baseline with BEADS) simulates different detection environments. Data augmentations encourage a model to isolate radiation signatures specifically associated with material transfers and be robust in identifying them for different spatio-temporal environments. No individual data augmentation is responsible for success in contrastive learning, but specific augmentations spanning the principles of radiation detection help a model with pattern recognition.

The broader contrastive learning framework has been fitted for use with radiation data. For every measured spectrum from MINOS, two augmented samples are generated. The representations produced by an encoder from these perturbed instances are encouraged to be similar, contrasted to any other augmented spectrum's representation regardless of an underlying label. This self-supervision generalizes the encoder, producing representations that are task-agnostic. The higher dimensional representations can be used as inputs to a classifier designed for a specific task like detecting material transfers. The semi-supervised framework, consisting of an encoder trained on unlabeled data and a classifier trained on labeled data, efficiently learn necessary labeling information for an accurate analysis model.

What is an efficient way to utilize unlabeled and labeled data for detecting shielded radiological material transfers?

A self-supervised contrastive encoder and a supervised linear classifier are capable of distinguishing between types of nuclear material transfers even when trained on only seven labeled spectra (i.e. few-shot learning). The classifier, using encoded representations as input, was able to learn necessary labeling information for successfully characterizing between transfers of byproducts from the nuclear fuel cycle at the MINOS testbed and nuclear material transfers that might be tracked in a nonproliferation scenario. The asymmetric training dataset of seven spectra (consisting of 1 byproduct and 6 nuclear material transfer spectra) was enough for the classifier to achieve 79.25% balanced accuracy on the test dataset. The model showed particular strengths for detecting ^{225}Ac and spent fuel transfers but struggled with detected activated metal transfers. Likewise, the classifier was stronger at detecting irradiated nuclear material than unirradiated targets (likely because irradiated transfers had more pronounced signals, less occluded by shielding and background radiation). Overall, the accuracy of this model was high for a challenging detection task despite various false positives and negatives.

Can a model be trusted to learn relevant detection information and how can it be evaluated for explainability?

Contrastive machine learning efficiently learns labeling information from unlabeled and labeled data, something out-of-the-box supervised methods struggle to do. Using cross-validation to negate effects of biased splitting between training and testing data, several supervised and semi-supervised machine learning methods were tested from the package Scikit-Learn. When trained on 20% of the labeled dataset and tested on the other 80%, none of the implementations could achieve the same balanced accuracy. Many models did not even learn a decision boundary at all, failing to move beyond labeling all spectra as one type of transfer. This behavior would only be further pronounced in low-shot learning, when less and less labeled spectra are used for training a classifier.

A trained contrastive encoder produces representations in a feature space that is more amenable for downstream detection tasks. Using PCA, representations showed higher dimensionality for explaining data distributional variance. That is, more components are available for drawing a decision boundary for representations, unlike the spectral feature space which could be almost fully explained by two components. Where subclasses of nuclear material transfers are hardly separable as spectra, an encoder provides a more separable feature space that can be learned by a classifier.

Integrated gradients can be used as a diagnostic tool for evaluating whether a model is learning appropriate features in radiation spectra for transfer detection. This method directly relates individual channels of a spectrum to its relative importance to a classifiers output label. It can be hard to evaluate heuristics a trained model follows in detecting transfer type. Fortunately, there are clear patterns in example integrated gradients that are related to expected signatures in a transfer spectrum, even if those signatures are difficult to visually identify.

How can scarce labeled data be effectively allocated to training tasks?

Labeled data is best suited for training a classifier for detection tasks, but it can help guide contrastive learning if noise is limited. That is, labeled data can act as a regularization term when used in contrastive learning. Training the encoder results in representations that are encouraged to take forms that better accentuate features associated with the downstream task. For example, if the labels denote types of transfers, representations for similar transfers should be similar, rather than representations being similar only when spectra are similar. However, if labeled data are noisy—either because of mislabeling or occluding labeling features—including them in contrastive learning can confuse a model more than it provides in context. Unless the rate of noise is negligible (which is not trivial for manually labeled, real-world spectra), more labeled training data can have deleterious effects on performance.

When can a contrastive model and its hyperparameters expected to be sufficiently optimized?

Hyperparameter optimization is used to quantify the average information gained when controlling for variabilities in model structure and data splitting for training and testing. An optimization sequence was conducted for empirically determining and assigning neural architectures and hyperparameters for the encoder, projection head, and then classifier. Five-fold cross-validation was used to benchmark the optimized model's performance. The best trial of cross-validation and subsequent hyperparameters achieved an average balanced accuracy of 78.22%. The best of five models (as determined by its tested balanced accuracy) achieved a score of 80.30% and showed promising pattern recognition and a strength for detecting

Cm transfers. The encoder model indicated continued strength at producing a rich representation space for drawing decision boundaries, as observed by PCA, feature importance, and subclass performance.

Assuming a desired detection accuracy is achievable, hyperparameter optimization can be used to span a state space in search of a performant or generalizable model. As should be clear, consistently finding a performant transfer detection model is challenging when the state space is non-smooth, training data is noisy and limited, and model design choice is unbounded. When resources are constrained, the methodology detailed here efficiently uses data and can be a powerful data analytical tool for nuclear security. Assuming a hyperparameter optimization goal of finding a model with maximized balanced accuracy score (for a given training and testing data), separate classifier optimization trials appear to converge to a consistent accuracy, even if the needed number of trials is ambiguous.

8.2 Future Work

Several avenues exist for further development of this model and broader applications for nuclear nonproliferation scenarios. Some of these have established research efforts in other domains and could show promising synergy when applied here.

8.2.1 Uncertainty Quantification

While several methods were explored to aid in explainability (e.g. PCA and integrated gradients), the models demonstrated here are notably lacking uncertainty quantification. Measuring the uncertainty of machine learning methods is a burgeoning field of study in computer science because uncertainty is difficult to measure for nonlinear systems like neural networks. One method in particular uses Monte Carlo dropout (that is, randomly zeroing neuron weights) to approximate uncertainties in a neural network [57]. Implementing this would require minimal refactoring, as dropout is already part of PyTorch networks.

Estimating uncertainty with dropout would also decompose model uncertainty between aleatoric and epistemic. Aleatoric uncertainty is that which is determined by the accuracy of data generated, whereas epistemic uncertainty dictates a model's limitations for OOD observations. This could be very useful in assessing whether a model is inaccurately ascribing confidence to labeled test samples. Epistemic uncertainty could be useful for determining a model's generalizability strengths and whether it is robust to previously unseen anomalous spectra.

8.2.2 Novel Class Discovery

The response of an ML detection model can be unpredictable when presented with OOD spectra. For example, if a model has been trained to learn the patterns of background spectra, spent fuel transfers, and unirradiated Np transfers to detect between types of transfers, what happens if it is presented with an irradiated Cm transfer spectrum? Depending on whether the model learned generalizable information during training, it may recognize this as another type of tracked nuclear material transfer with similarities to other irradiated transfers, or it may wildly fail and label the transfer as a byproduct. If, however, a model has been trained with functionality for OOD detection [58], it would be more robust to unknown transfer types and could learn new transfer types once recognized. This overlaps with few-shot learning, where the limited labeled training data may be missing samples for subclasses (e.g. irradiated Cm) that are present in the test distribution. Fortunately, contrastive learning's system of augmentations may intrinsically contribute to OOD robustness by perturbing inputs (akin to adverarial methods [59]).

In practice, especially with large volumes of unlabeled data, the number of classes may be unbounded, unknown, or at least larger than specified. These novel classes may be the result of previously unobserved environmental characteristics and may or may not be equally interesting classes from a nonproliferation detection perspective. Open-world contrastive learning considers the opportunity for novel classes to exist and be discovered during training.

The first implementation [60] has a loss function consisting of three parts. First, a supervised term to handle previously seen classes and form novel classes if not seen. Novel classes are balanced against biasing towards known classes using a system of adaptive uncertainty. Second, a pairwise term that considers similarity between pairs which generates pseudo-labels for unlabeled data. Third, a regularization term.

The next evolution in open-world contrastive learning was produced in OpenCon [61]. This multistep process includes an out-of-distribution detection operation for novel discovery. Prototypical samples are used in clustering data based on similarity. Sufficiently dissimilar clusters can be considered different classes and labeled as such. This produces more distinguishable and compact classes, known and novel.

8.2.3 Multiclass Classification

This work has purely focused on binary classification, between types of transfer spectra: byproducts and tracked nuclear material transfers. This task construction implicitly assumes a relationship between the subclasses that are grouped together in the dataset. Even though irradiated Cm and irradiated Np may both be materials of interest for detection and characterization, it may not be appropriate to group these two together if their relevant labeling information is different in a spectrum. In fact, because different radioactive sources have (sometimes subtly) different gamma emission patterns, grouping different types of transfers together expects a model to recognize spectral features that can sometimes be mutually exclusive. By constructing a model to predict between several classes of transfers, it may make the detection task easier, by separating the number of features that must be learned by a model between classes. However, two limitations arise from a multiclass design for detecting nuclear material transfers.

First, separating subclasses into separating labels reduces the number of labeled spectra to train on per class. For example, if there were five irradiated Cm transfers and five irradiated Np transfers that were originally used to train a binary classifier, now the model only has five instances for each to learn from in a multiclass setting. This can exacerbate challenges with training on limited labeled data when labeled data is already scarce, as explored here.

Second, multiclass systems assume an enumeration of all the possible transfer types expected. Whereas tracked nuclear material transfers is a potentially broad and ambiguous class, more granular labels risk omitting potentially unseen transfer types. These unseen transfer types will have unpredictable responses in a multiclass setup, returning to the principal challenges of OOD detection.

8.2.4 Multi-modal Data Fusion

Prior work utilized the multi-modal data collected by MINOS in tracking nuclear material transfers. For example, alarms were designed to identify transportation vehicles from seismo-acoustic measurements and detect the coincident presence of nuclear material with radiation measurements [28]. The contrastive learning framework used here was exclusively applied to radiation data, but the framework—except augmentations—is data-agnostic. Augmentations can be similarly applied to different data modalities and produce encoders that could be used in tandem for a more robust detection system. The original contrastive model in SimCLR was originally used on images with their respective augmentations set. Frequency domain augmentations could feasibly be designed for, say, seismo-acoustic data.

This work has also isolated radiation measurements static in time (each sample is integrated for a minute measurement length), irrespective of detection location at the MINOS testbed. It is possible, particularly with data fusion, that this framework could be used to detect the spatio-temporal transportation of nuclear material if deployed as part of a broader sensor network. That is, a contrastive model could individually detect transfers at individual detectors and raise time-dependent alarms when material is present.

Time could also be included in the data structure used by the contrastive learning framework. Again, because all spectra used so far have been static in time, the feature space given to the model consists of one-dimensional spectra vectors. Instead, two-dimensional “images” could be constructed, consisting of energy/detection channels as a function of time. The same augmentations could apply to “slices” in the snapshot or could be combined with time-dependent augmentations. Considering time-dependency in tracking nuclear material transfers offers a more complex state space of observations but could generate strong and robust models for detection and characterization.

Bibliography

[1] IAEA, "Treaty on the Non-Proliferation of Nuclear Weapons (NPT)," Text (2014)URL `https://www.iaea.org/publications/documents/treaties/npt`.

[2] IAEA, "The Statute of the IAEA," Text (2014)URL `https://www.iaea.org/about/statute`.

[3] NNSA, "National Nuclear Security Administration Mission and Prorities," URL `https://www.energy.gov/nnsa/missions`.

[4] 112th Congress; 2nd Session, "Nuclear Regulatory Legislation," 10.

[5] IAEA, "Annual Report for 2020," (2020)Note: Compare the A2.MP4 budget utilization and A5 safeguards facilities to previous years.

[6] IAEA, "The Agency's Budget Update for 2023," (2022).

[7] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A Simple Framework for Contrastive Learning of Visual Representations," (2020); 10.48550/arXiv.2002.05709.

[8] U. N. R. Commission, "Stages of the nuclear fuel cycle," URL `https://www.nrc.gov/materials/fuel-cycle-fac/stages-fuel-cycle.html`.

[9] K. S. Krane, *Introductory nuclear physics*, Wiley, New York, NY (1988)URL `https://cds.cern.ch/record/359790`.

[10] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press (2014).

[11] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall (2020).

[12] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, vol. 31 of *Stochastic Modelling and Applied Probability*, Springer (1996); 10.1007/978-1-4612-0711-5.

[13] N. J. Sairamya, L. Susmitha, S. Thomas George, and M. S. P. Subathra, "Chapter 12 - Hybrid Approach for Classification of Electroencephalographic Signals Using Time–Frequency Images With Wavelets and Texture Features," *Intelligent Data Analysis for Biomedical Applications*, Intelligent Data-Centric Systems, 253–273, Academic Press; 10.1016/B978-0-12-815553-0.00013-6., URL `https://www.sciencedirect.com/science/article/pii/B9780128155530000136`.

[14] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*, The MIT Press (2006)URL `http://dblp.uni-trier.de/db/books/collections/CSZ2006.html`.

[15] X. Zhu and A. B. Goldberg, "Introduction to Semi-Supervised Learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, **3**, *1*, 1 (2009); 10.1007/978-3-031-01548-9.

[16] J. E. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, **109**, *2*, 373 (2020); 10.1007/s10994-019-05855-6.

[17] Lu, Tyler (Tian), "Fundamental Limitations of Semi-Supervised Learning," Master's Thesis (2009)URL `http://hdl.handle.net/10012/4387`.

[18] A. Singh, R. Nowak, and J. Zhu, "Unlabeled data: Now it helps, now it doesn't," *Advances in Neural Information Processing Systems*, vol. 21, Curran Associates, Inc. (2008)URL `https://proceedings.neurips.cc/paper/2008/file/07871915a8107172b3b5dc15a6574ad3-Paper.pdf`.

[19] A. Gammerman, V. Vovk, and V. Vapnik, "Learning by Transduction," *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, 148–155, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1998).

[20] K. Bennett and A. Demiriz, "Semi-Supervised Support Vector Machines," *Advances in Neural Information Processing Systems*, vol. 11, MIT Press (1998)URL `https://proceedings.neurips.cc/paper/1998/file/b710915795b9e9c02cf10d6d2bdb688c-Paper.pdf`.

[21] M. Peikari, S. Salama, S. Nofech-Mozes, and A. L. Martel, "A Cluster-then-label Semi-supervised Learning Approach for Pathology Image Classification," *Scientific Reports*, **8**, *1*, 7193 (2018); 10.1038/s41598-018-24876-0.

[22] J. Ma and J. Jiang, "Semisupervised classification for fault diagnosis in nuclear power plants," *Nuclear Engineering and Technology*, **47**, *2*, 176 (2015); 10.1016/j.net.2014.12.005.

[23] L. Pinciroli, P. Baraldi, A. Shokry, E. Zio, R. Seraoui, and C. Mai, "A semi-supervised method for the characterization of degradation of nuclear power plants steam generators," *Progress in Nuclear Energy*, **131**, 103580 (2021); 10.1016/j.pnucene.2020.103580.

[24] L. Sun, C. Zhao, Z. Yan, P. Liu, T. Duckett, and R. Stolkin, "A Novel Weakly-Supervised Approach for RGB-D-Based Nuclear Waste Object Detection," *IEEE Sensors Journal*, **19**, *9*, 3487 (2019); 10.1109/JSEN.2018.2888815.

[25] K. Moshkbar-Bakhshayesh and M. B. Ghofrani, "Combining Supervised and Semi-Supervised Learning in the Design of a New Identifier for NPPs Transients," *IEEE Transactions on Nuclear Science*, **63**, *3*, 1882 (2016); 10.1109/TNS.2016.2547866.

[26] K. Moshkbar-Bakhshayesh and S. Mohtashami, "Classification of NPPs transients using change of representation technique: A hybrid of unsupervised MSOM and supervised SVM," *Progress in Nuclear Energy*, **117**, 103100 (2019); 10.1016/j.pnucene.2019.103100.

[27] A. D. Nicholson, D. E. Archer, I. Garishvili, I. R. Stewart, and M. J. Willis, "Characterization of gamma-ray background outside of the High Flux Isotope Reactor," *Journal of Radioanalytical and Nuclear Chemistry*, **318**, *1*, 361 (2018); 10.1007/s10967-018-6097-5.

[28] K. Dayman, J. Hite, R. Hunley, N. S. V. Rao, C. Geulich, M. Willis, J. Ghawaly, D. Archer, and J. Johnson, "Tracking Material Transfers at a Nuclear Facility with Physics-Informed Machine Learning and Data Fusion," Institute of Nuclear Materials Management (2021)URL `https://resources.inmm.org/annual-meeting-proceedings/tracking-material-transfers-nuclear-facility-physics-informed-machine`.

[29] J. Przyborowski and H. Wilenski, "Homogeneity of Results in Testing Samples from Poisson Series: With an Application to Testing Clover Seed for Dodder," *Biometrika*, **31**, *3/4*, 313 (1940); 10.2307/2332612.

[30] A. Blum and T. Mitchell, "Combining Labeled and Unlabeled Data with Co-Training," *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, 92–100, Association for Computing Machinery (1998); 10.1145/279943.279962.

[31] X. Zhu and Z. Ghahramani, "Learning from Labeled and Unlabeled Data with Label Propagation," (2003).

[32] L. Linville, D. Anderson, J. Michalenko, J. Galasso, and T. Draelos, "Semisupervised Learning for Seismic Monitoring Applications," *Seismological Research Letters*, **92**, *1*, 388 (2020); 10.1785/0220200195.

[33] Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database," *ATT Labs*, **2** (2010)URL `http://yann.lecun.com/exdb/mnist`.

[34] J. Bergstra, D. Yamins, and D. Cox, "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures," vol. 28 of *Proceedings of Machine Learning Research*, 115–123, PMLR, Atlanta, Georgia, USA (2013)URL `https://proceedings.mlr.press/v28/bergstra13.html`.

[35] H. Zheng, X. Chen, J. Yao, H. Yang, C. Li, Y. Zhang, H. Zhang, I. Tsang, J. Zhou, and M. Zhou, "Contrastive Attraction and Contrastive Repulsion for Representation Learning," *Transactions on Machine Learning Research* (2023)URL `https://openreview.net/forum?id=f39UIDkwwc`.

[36] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A Survey on Contrastive Self-Supervised Learning," *Technologies*, **9**, *1* (2021); 10.3390/technologies9010002., URL `https://www.mdpi.com/2227-7080/9/1/2`.

[37] S. Yamaguchi, S. Kanai, T. Shioda, and S. Takeda, "Image Enhanced Rotation Prediction for Self-Supervised Learning," *2021 IEEE International Conference on Image Processing (ICIP)*, 489–493 (2021); 10.1109/ICIP42928.2021.9506132.

[38] J. Zhu, J. Qi, M. Ding, X. Chen, P. Luo, X. Wang, W. Liu, L. Wang, and J. Wang, "Understanding Self-Supervised Pretraining with Part-Aware Representation Learning," *Transactions on Machine Learning Research* (2023)URL https://openreview.net/forum?id=HP7Qpui5YE.

[39] R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian, A. Schwarzschild, A. G. Wilson, J. Geiping, Q. Garrido, P. Fernandez, A. Bar, H. Pirsiavash, Y. LeCun, and M. Goldblum, "A Cookbook of Self-Supervised Learning," (2023).

[40] A. van den Oord, Y. Li, and O. Vinyals, "Representation Learning with Contrastive Predictive Coding," *CoRR*, **abs/1807.03748** (2018)URL http://arxiv.org/abs/1807.03748.

[41] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum Contrast for Unsupervised Visual Representation Learning," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

[42] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow Twins: Self-Supervised Learning via Redundancy Reduction," *Proceedings of the 38th International Conference on Machine Learning*, vol. 139, 12,310–12,320, PMLR (2021)URL https://proceedings.mlr.press/v139/zbontar21a.html.

[43] Z. Fang, J. Wang, L. Wang, L. Zhang, Y. Yang, and Z. Liu, "{SEED}: Self-supervised Distillation For Visual Representation," *International Conference on Learning Representations* (2021)URL https://openreview.net/forum?id=AHm3dbp7D1D.

[44] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, k. kavukcuoglu, R. Munos, and M. Valko, "Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning," *Advances in Neural Information Processing Systems*, vol. 33, 21,271–21,284, Curran Associates, Inc. (2020)URL https://proceedings.neurips.cc/paper/2020/file/f3ada80d5c4ee70142b17b8192b2958e-Paper.pdf.

[45] J. Z. HaoChen, C. Wei, A. Gaidon, and T. Ma, "Provable Guarantees for Self-Supervised Deep Learning with Spectral Contrastive Loss," *Advances in Neural Information Processing Systems* (2021)URL https://openreview.net/forum?id=mjyMGFL8N2.

[46] H. Liu, J. Z. HaoChen, A. Gaidon, and T. Ma, "Self-supervised Learning is More Robust to Dataset Imbalance," *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications* (2021)URL https://openreview.net/forum?id=vUz4JPRLpGx.

[47] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big Self-Supervised Models are Strong Semi-Supervised Learners," H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin (Editors), *Advances*

in Neural Information Processing Systems, vol. 33, 22,243–22,255, Curran Associates, Inc. (2020)URL https://proceedings.neurips.cc/paper_files/paper/2020/file/fcbc95ccdd551da181207c0c1400c655-Paper.pdf.

[48] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised Contrastive Learning," *Advances in Neural Information Processing Systems*, vol. 33, 18,661–18,673, Curran Associates, Inc. (2020)URL https://proceedings.neurips.cc/paper/2020/hash/d89a66c7c80a29b1bdbab0f2a1a94af8-Abstract.html.

[49] M. Zheng, F. Wang, S. You, C. Qian, C. Zhang, X. Wang, and C. Xu, "Weakly Supervised Contrastive Learning," *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10,042–10,051 (2021).

[50] X. Ning, I. W. Selesnick, and L. Duval, "Chromatogram baseline estimation and denoising using sparsity (BEADS)," *Chemometrics and Intelligent Laboratory Systems*, **139**, 156 (2014); 10.1016/j.chemolab.2014.09.014., URL https://linkinghub.elsevier.com/retrieve/pii/S0169743914002032.

[51] IAEA, "IAEA Safeguards Glossary," Text (2019)URL https://www.iaea.org/publications/6663/iaea-safeguards-glossary.

[52] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic Attribution for Deep Networks," D. Precup and Y. W. Teh (Editors), *Proceedings of the 34th International Conference on Machine Learning*, vol. 70 of *Proceedings of Machine Learning Research*, 3319–3328, PMLR (2017)URL https://proceedings.mlr.press/v70/sundararajan17a.html.

[53] I. Loshchilov and F. Hutter, "Fixing Weight Decay Regularization in Adam," *CoRR*, **abs/1711.05101** (2017)URL http://arxiv.org/abs/1711.05101.

[54] J. Yoo, T. Zhao, and L. Akoglu, "Data Augmentation is a Hyperparameter: Cherry-picked Self-Supervision for Unsupervised Anomaly Detection is Creating the Illusion of Success," *Transactions on Machine Learning Research* (2023)URL https://openreview.net/forum?id=HyzCuCV1jH.

[55] S. Joshi and B. Mirzasoleiman, "Data-Efficient Contrastive Self-supervised Learning: Most Beneficial Examples for Supervised Learning Contribute the Least," *Proceedings of the 40th International Conference on Machine Learning*, 15,356–15,370, PMLR (2023)URL https://proceedings.mlr.press/v202/joshi23b.html, iSSN: 2640-3498.

[56] J. Cui, W. Huang, Y. Wang, and Y. Wang, "Rethinking Weak Supervision in Helping Contrastive Learning," (2023); 10.48550/arXiv.2306.04160.

[57] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, 1050–1059, JMLR.org (2016).

[58] J. Katz-Samuels, J. B. Nakhleh, R. Nowak, and Y. Li, "Training OOD Detectors in their Natural Habitats," *Proceedings of the 39th International Conference on Machine Learning*, vol. 162 of *Proceedings of Machine Learning Research*, 10,848–10,865, PMLR (2022)URL `https://proceedings.mlr.press/v162/katz-samuels22a.html`.

[59] J. Chen, Y. Li, X. Wu, Y. Liang, and S. Jha, "Robust Out-of-distribution Detection for Neural Networks," *arXiv:2003.09711 [cs, stat]* (2020)URL `http://arxiv.org/abs/2003.09711`.

[60] K. Cao, M. Brbic, and J. Leskovec, "Open-World Semi-Supervised Learning," *CoRR*, **abs/2102.03526** (2021)URL `https://arxiv.org/abs/2102.03526`.

[61] Y. Sun and Y. Li, "OpenCon: Open-world Contrastive Learning," *Transactions on Machine Learning Research* (2023)URL `https://openreview.net/forum?id=2wWJxtpFer`.

[62] M. Hesami and A. M. P. Jones, "Application of artificial intelligence models and optimization algorithms in plant cell and tissue culture," *Applied Microbiology and Biotechnology*, **104**, *22*, 9449 (2020); 10.1007/s00253-020-10888-2.

[63] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, **27**, *8*, 861 (2006); 10.1016/j.patrec.2005.10.010.

Appendix I

Tools for Data Analytics

This appendix provides an introduction of three important tools for data sceince and analytics: PCA, confusion matrices, and ROC curves.

I.1 Principal Component Analysis

Consider a data distribution of feature vectors:

$$\begin{aligned} x_i = \{x_1, x_2, \ldots, x_d\} \\ x_i \in \mathcal{X} \\ i = 1, \ldots, k \end{aligned} \tag{I.1}$$

A matrix can be constructed from this dataset where each row is an instance of $\mathcal{X}$, x_i, numbering some data subset k and the columns are features, length d, from each vector.

$$X = \begin{bmatrix} x_{11} & \ldots & x_{1d} \\ \vdots & \ddots & \\ x_{k1} & & x_{kd} \end{bmatrix} \tag{I.2}$$

Using the Gram matrix, $X^\intercal X$, a representation can be made that defines the data distribution along its axes of largest variance, i.e. principal components. The first principal component vector of X would be the vector that lies along the direction of maximum variance:

$$\begin{aligned} w_1 &= \underset{w \in \mathbb{R}^k}{\operatorname{argmax}} \frac{w^\intercal X^\intercal X w}{w^\intercal w} \\ &= \underset{w}{\operatorname{argmax}} \frac{\|Xw\|_2^2}{\|w\|_2^2} \end{aligned} \tag{I.3}$$

As an aside, this can also be represented using the Singular Value Decomposition (SVD) of X. In this case, the first principal component would have length σ_1^2, i.e. the magnitude of the first singular value of X. For each subsequent axes of variance, in decreasing order, a similar process can be used. Each subsequent principal component vector would also have to be orthogonal to all previous principal component vectors:

$$w_j^\intercal w_{j-1} = w_j^\intercal w_{j-2} = \ldots = w_j^\intercal w_1 = 0 \tag{I.4}$$

In summary, these principal component vectors provide a matrix transform that converts the original data instances in X to different axes. Take Figure I.1, for example. The original data, as orange dots, lie along a traditional two-dimensional place. After finding the directions of maximum variance, this data can be transformed along the new axes. The first principal component direction is along the longer black arrow. The second dimension is along the second, smaller, black arrow.

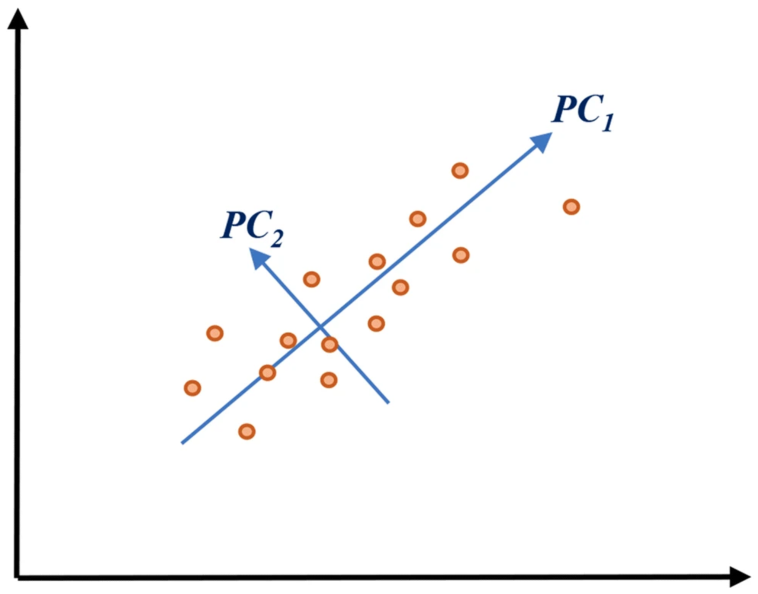


Figure I.1: PCA is demonstrated here using two-dimensional data (orange dots). The direction of maximum variance is the first principal component (PC_1). The second principal component is in the direction of the second largest variance (PC_2). (Image source: [62])

I.2 Confusion Matrices

Three sets of data are used during the construction and development of a machine learning model. First, training data is used to train a model and determine its parameters. Of the total data partition, this subset is usually the largest, as a sizeable portion is required as the number of model parameters increase. To determine the final accuracy of a trained model, a test set is used. Sometimes, as a model is trained or if several models are trained simultaneously, a validation set is used. This validation set is also used to test the accuracy of a model but is used as part of the model selection process during training, not for evaluation of a trained model. For example, K-fold cross-validation partitions a dataset into several iterations (K) between training and validation and averages to avoid biasing along any particular data partition. Neither the validation nor test set is used by the model to train its parameters.

Several metrics are available to test a model's performance. Although occasionally used interchangeably, accuracy and precision are slightly different measurements. Accuracy refers to how often the model was accurate at predictions (continuous or discrete) overall. Precision refers to a model's accuracy at predicting a specific class (especially important for multiclass

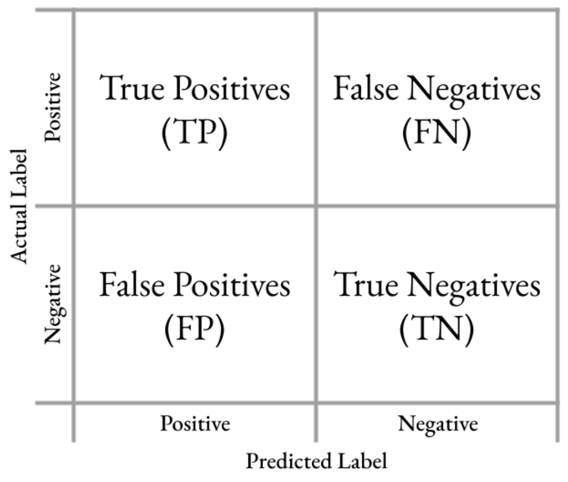


Figure I.2: A diagram indicating what each quadrant of a binary classification confusion matrix represents. This can be extended to n number of classes, with the eventual confusion matrix expanding to size $n \times n$.

problems). Further, a balanced accuracy score accounts for class imbalance in a dataset. Using Figure I.2, each metric can be calculated. This matrix divides any given model's results into four quadrants by comparing a test instance's true label to the predicted label. Note that this implicitly requires test data that is pre-labeled, or data in which the actual label is assumed to be true. The measures are then:

$$\begin{aligned}\text{Accuracy} &= \frac{TP+TN}{TP+FP+TN+FN}\\ \text{Balanced Accuracy} &= \frac{1}{2}\left(\frac{TP}{TP+FN}+\frac{TN}{TN+FP}\right)\\ \text{Precision} &= \frac{TP}{TP+FP}\end{aligned} \tag{I.5}$$

Accuracy is a metric that penalizes any misclassifications, false positives or false negatives. Precision is a metric that penalizes false positives specifically. This metric may be useful when limiting the number of predicted positives is important. Another metric often used is recall. Recall is opposite of precision in that it penalizes false negatives specifically. This may be valuable when classifying all true positives is important.

$$\text{Recall} = \frac{TP}{TP+FN} \tag{I.6}$$

Finally, sometimes a summary score is given to a model that attempts to account for all intricacies in the confusion matrix. A common metric would be the F_1 score, which strikes a balance between precision and recall. The F_1 score has a range of possible values between 0 and 1. A perfectly performing model, where there are no false positives or false negatives, would have an F_1 score of 1.

$$\begin{aligned} F_1 &= 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \\ &= \frac{2TP}{2TP + FP + FN} \end{aligned} \tag{I.7}$$

I.3 Receiver Operating Characteristics Curve

Another method of model diagnostics using test results like true positives and false positives is a ROC. First, the number of correct positive classifications can be expressed as the true positive rate: $TPR = \frac{TP}{TP+FN} = \text{Recall}$. Alternatively, the rate of positive misclassification can be expressed as the false positive rate: $FPR = 1 - TNR = \frac{FP}{TN+FP}$.

Any given model, trained and evaluated with a validation set, will have a given TPR and FPR. Different models can be compared using these different metrics in a ROC. This curve can be used to answer the question, "what predictive threshold should be used for a probabilistic classifier?" In this scenario, each instance in the validation set will have a true label and a predicted probability of having a positive label (for binary classification). A threshold can be applied that acts as a cutoff for accepting predicted labels.

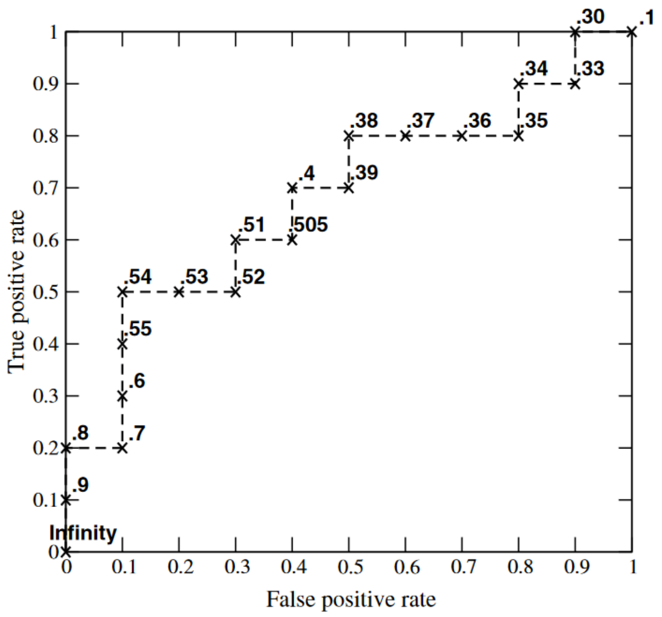


Figure I.3: An example ROC curve for a given test dataset ranging from [inf, 1.] *for a given model parameter. (Image source: [63])*

For example, for a threshold of 0.9, any probability for a positive label with value greater than or equal to 0.9 will be labeled as a positive instance. From this, TPR and FPR can be computed. Sweeping over different values of this threshold will result in Figure I.3. Then, an optimal model, or threshold, would be one that maximizes the true positive rate while minimizing the false positive rate. Here, that appears to be approximately 0.4. Thus, ROC was used to choose a model parameter as a method of semi-manual optimization. This can be extended to other forms of hyperparameters not limited to probabilistic models. For discrete classifiers, TPR and FPR can be calculated outright and an input hyperparameter can be varied as part of training.

Appendix II

Trials of Labeled Data Allocation Strategies

The results of various labeled data allocation strategies used in semi-supervised contrastive learning, Section 6.6.3, are reported here. For each allocation of labeled data, 10 trials are conducted and the final accuracy and balanced accuracy are computed.

Table II.1: Results for all iterations of semi-supervised contrastive learning with different allocation strategies.

Trials	**0/10/50**		**0/20/50**		**0/30/50**		**0/40/50**	
	Accuracy	**Balanced Accuracy**	**Accuracy**	**Balanced Accuracy**	**Accuracy**	**Balanced Accuracy**	**Accuracy**	**Balanced Accuracy**
1	69.47	63.31	74.81	67.16	74.05	72.21	82.44	79.66
2	74.81	71.36	77.1	75.81	74.05	72.21	84.73	80.6
3	77.86	70.76	75.57	69.11	78.63	74.81	78.63	74.11
4	69.47	68.21	78.63	74.11	82.44	78.96	71.76	66.36
5	74.05	72.21	80.15	78	77.86	70.76	75.57	68.41
6	74.81	69.26	82.44	80.36	70.99	69.31	76.34	73.16
7	80.92	83.46	77.1	70.91	71.76	67.76	78.63	70.6
8	72.52	63.41	69.47	62.61	71.76	71.26	83.97	79.35
9	80.92	75.76	77.1	77.21	73.28	71.66	75.57	69.81
10	81.68	77.01	78.63	72.71	75.57	72.61	76.34	72.46
Average:	75.65	71.48	77.10	72.80	75.04	72.16	78.40	73.45
STD:	4.56	6.16	3.47	5.45	3.63	3.05	4.16	4.98

Trials	**0/50/50**		**10/10/50**		**10/20/50**		**10/30/50**	
	Accuracy	**Balanced Accuracy**	**Accuracy**	**Balanced Accuracy**	**Accuracy**	**Balanced Accuracy**	**Accuracy**	**Balanced Accuracy**
1	77.1	73.01	76.34	76.66	74.81	69.96	77.86	68.65
2	80.92	74.35	77.1	71.61	72.52	69.01	78.63	69.9
3	78.63	73.4	69.47	62.61	77.1	73.01	80.92	75.05
4	80.15	76.61	67.18	64.46	74.81	68.56	77.86	73.56
5	73.28	66.06	80.15	71.7	79.39	77.46	75.57	71.91
6	76.34	72.46	78.63	80.41	78.63	69.2	77.86	82.16
7	71.76	62.16	65.65	62.66	80.15	78.71	79.39	74.66
8	80.92	72.95	77.1	73.01	74.81	66.46	74.05	70.11
9	80.15	72.4	60.31	53.91	77.1	74.41	77.1	68.1
10	73.28	74.46	76.34	71.06	79.39	80.96	79.39	73.26
Average:	77.25	71.79	72.83	68.81	76.87	72.77	77.86	72.74
STD:	3.47	4.33	6.67	7.83	2.55	4.94	1.97	4.12

Trials	10/40/50		20/10/50		20/20/50		20/30/50	
	Accuracy	**Balanced Accuracy**	**Accuracy**	**Balanced Accuracy**	**Accuracy**	**Balanced Accuracy**	**Accuracy**	**Balanced Accuracy**
1	80.15	75.21	73.28	70.96	83.97	78.65	77.1	69.51
2	83.21	78.8	74.81	74.16	75.57	69.81	75.57	72.61
3	77.86	70.05	74.81	72.76	83.21	78.8	77.86	73.56
4	78.63	74.81	70.99	62.31	74.81	72.76	68.7	67.66
5	74.05	68.01	77.1	67.4	75.57	72.61	79.39	75.36
6	80.15	77.31	66.41	70.22	74.05	69.41	67.94	66.41
7	82.44	77.55	72.52	66.21	71.76	64.96	79.39	80.26
8	80.15	73.1	73.28	60.45	63.36	60.32	74.05	72.21
9	78.63	76.21	72.52	66.21	83.21	83.01	72.52	64.11
10	74.05	71.51	74.05	70.81	82.44	81.76	81.68	74.9
Average:	78.93	74.26	72.98	68.15	76.80	73.21	75.42	71.66
STD:	3.06	3.52	2.84	4.45	6.55	7.38	4.61	4.81

Trials	30/10/50		30/20/50		40/10/50		[redacted]	
	Accuracy	Balanced Accuracy	Accuracy	Balanced Accuracy	Accuracy	Balanced Accuracy	[redacted]	[redacted]
1	72.52	68.31	74.05	65.91	73.28	68.16	[redacted]	[redacted]
2	69.47	69.62	75.57	69.11	77.1	75.81	[redacted]	[redacted]
3	76.34	70.36	71.76	69.86	75.57	69.81	[redacted]	[redacted]
4	76.34	77.36	80.92	76.46	74.05	71.51	[redacted]	[redacted]
5	77.1	66.7	76.34	70.36	80.15	74.51	[redacted]	[redacted]
6	73.28	68.86	74.05	72.21	78.63	75.51	[redacted]	[redacted]
7	83.21	80.91	76.34	68.96	67.18	67.27	[redacted]	[redacted]
8	78.63	72.01	77.86	76.36	80.15	74.51	[redacted]	[redacted]
9	74.81	67.86	74.05	71.51	73.28	74.46	[redacted]	[redacted]
10	80.15	73.8	74.81	70.66	75.57	78.21	[redacted]	[redacted]
Average:	76.19	71.58	75.58	71.14	75.50	72.98	[redacted]	[redacted]
STD:	3.96	4.55	2.52	3.26	3.90	3.60	[redacted]	[redacted]