A Semi-Supervised Model for Multi-Label Radioisotope Classification and Out-of-Distribution Detection

by

Alan Van Omen

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Master of Science
(Electrical and Computer Engineering)
in The University of Michigan
2023

Committee:

Professor Clayton Scott, Chair
Professor Alfred Hero
Assistant Professor Qing Qu

Alan J. Van Omen

avanomen@umich.edu

Dedication

This thesis is dedicated to my wife and daughter, Moriah and Amelia.

Acknowledgements

My thesis advisor Dr. Clayton Scott provided invaluable perspective on statistical methods as they apply to this problem. Tyler Morrow, Elliott Leonard, and Paul Thelen from Sandia National Laboratories provided invaluable support and expertise on this project in software development, machine learning, radiation detection, and gamma spectroscopy. Lastly, financial support was provided by both Sandia National Laboratories via the Critical Skills Master's Program as well as the U.S. Department of Energy, National Nuclear Security Administration, Office of Defense Nuclear Nonproliferation Research and Development (DNN R&D).

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. SAND number: SAND2023-02705T.

Thank you to everyone.

Table of Contents

List of Tables

List of Figures

List of Appendices

List of Acronyms

GADRAS	Gamma Detector Response and Analysis Software
GNLL	Gaussian Negative Log-Likelihood
HPGe	High Purity Germanium
i.i.d.	Independently and Identically Distributed
ID	In-Distribution
LPE	Label Proportion Estimation
MAE	Mean Absolute Error
OOD	Out-of-Distribution
PNLL	Poisson Negative Log-Likelihood
RIID	Radioisotope Identification
SNR	Signal-to-Noise Ratio
SSE	Sum of Squared Errors

Abstract

In the machine learning problem of multi-label classification, the objective is to determine for each test instance which classes the instance belongs to. In this work, we consider multi-label classification in the context of multi-label radioisotope classification for gamma spectra data. By viewing spectra as discrete distributions we tackle a more challenging variant of multi-label classification where the goal is to ascribe a proportion to each class label, not just a binary variable. Motivated by this application to radioisotope identification, we aim to simultaneously predict label proportions while also performing out-of-distribution (OOD) detection. To achieve this goal, we introduce a novel semi-supervised loss function that combines a traditional supervised loss with an unsupervised reconstruction error penalty. This work demonstrates that the proposed model can successfully perform radioisotope identification in a realistic test scenario. We also show how to extend this approach to perform OOD detection which can determine when the model prediction should not be trusted due to the presence of an anomalous source. The semi-supervised model, trained on gamma spectra based on a measurement of a real fission source containing a mixture of 30 distinct radioisotopes (labeled by a spectroscopist), learned to estimate in-distribution (ID) samples with about 39% error while simultaneously being able to differentiate (with 95% confidence) between ID and OOD samples with an anomaly contribution as small as 10%.

Chapter 1 Introduction

The unique signatures manifested in gamma spectra by each contributing radioisotope have long been analyzed by pattern recognition algorithms. Utilizing modern software tools which can simulate spectra, researchers are well-equipped to try and construct synthetic datasets which represent their problem space. Although radioisotope identification (RIID) has traditionally relied on manual analysis by subject matters experts, the availability of synthetic data has opened the problem space to more modern techniques. Machine learning methods are especially suited for this problem space as they excel at generalizing useful models from large amounts of diverse data.

The simplest RIID problem would be to identify a single, dominant isotope in each observed gamma spectrum. This is a multi-class classification problem where the output is a single positive class. Unlike multi-class classification, multi-label radioisotope classification involves identifying the presence of multiple radioisotopes in a single gamma spectrum.

The goal of this thesis is two-fold. We investigate a novel formulation of multi-label classification where the objective is to predict the proportion of each isotope present in a spectrum (LPE). In addition, as a second goal, we aim to design a classifier that can detect the presence of a novel source (OOD detection). To address these problems a neural network is trained on a semi-supervised loss function which simultaneously optimizing for these two objectives. This semi-supervised loss combines a traditional supervised loss with an unsupervised reconstruction error penalty. In particular, the reconstruction error penalty uses a dictionary of pure gamma spectra (corresponding with each class label) along with the model's predicted proportions to generate a reconstruction of the input spectrum. The resulting reconstruction error represents a metric that can be used for determining confidence in proportion estimates or in conjunction with a threshold to identify OOD samples.

Both this problem formulation and our approach leverage the fact that these classes and spectral measurements are probability distributions. Thus, the proposed technique should extend to other applications where the feature vector can be viewed as a mixture of distributions.

1.1 Contributions

The key contributions of the proposed learning framework are as follows:

- A novel semi-supervised loss function is proposed for LPE which combines a traditional supervised loss with an unsupervised reconstruction error term.
- A method for performing OOD detection using the unsupervised portion of the loss function is proposed. This allows the model to learn both LPE and OOD detection simultaneously using shared network parameters.
- A probabilistic interpretation of the proposed semi-supervised loss function is offered, which explains the learning objective as a paired maximum likelihood estimator.
- The effectiveness of the proposed model is tested on synthetic gamma spectra based on a fission source with a large mixture size of 30 radioisotopes.

1.2 Organization of Paper

Before discussing the technique used to create the proposed semi-supervised model, chapter 2 presents a review of current machine learning methods for radioisotope identification and related topics. In chapter 3, the problem statement, assumptions, and goals for this work are formally defined. In chapter 4, the proposed method label proportion estimation with OOD detection is given, along with a probabilistic interpretation of the loss function. Chapter 5 details the steps taken to generate a dataset of representative, synthetic gamma spectra, along with an evaluation of the dataset. Chapter 6 contains information regarding the training setup, network architecture, and experimental results. Finally, conclusions and future work are discussed in chapters 7 and 8, respectively.

Chapter 2 Background

In this chapter, radioisotope identification is motivated and a review of recent approaches for radioisotope identification with machine learning is presented in Section 2.1. Dictionary-based signal modeling and sparse coding are reviewed in Section 2.2, as this forms the basis for the unsupervised term of the proposed loss function. Finally, in Section 2.3, out-of-distribution detection is motivated and defined.

2.1 Gamma Spectra and Fission Sources

Radioactive decay occurs when an unstable atomic nucleus emits radioactive electromagnetic energy, and subsequently transforming into a different atomic nucleus. This decay gives off energy in various forms including alpha, beta, and gamma radiation. The energy of emitted gamma ray particles, which are the highest energy form of electromagnetic radiation (i.e., have the shortest wavelength), can be individually resolved via a gamma-ray spectrometer. Each radioisotope emits gamma particles during this radioactive decay at unique, characterizable energy levels. As a gamma-ray spectrometer detects individual gamma photons, they can be binned by energy level, producing a histogram of counts known as a gamma spectrum. This histogram can then be normalized and viewed as a discrete probability distribution. Since radioisotopes tend to produce unique and characterizable features in gamma spectra, gamma spectroscopists take advantage of this fact to identify radioactive sources. All these processes are well described in [1].

The data used for this work is based on a gamma spectrum taken of a fission source in [2]. The fission source, in this context, is spent nuclear fuel removed from a reactor and subsequently measured with a High-Purity Germanium (HPGe) detector. The source itself, which has been bombarded with neutrons, is a chaotic soup of radioisotopes. Despite this, to obtain ground truth proportions, a spectroscopist performed a peak-based analysis of the spectrum to determine the individual activities of each radioisotope present. These ground truth activities form the basis of the synthesis process described in chapter 5.

Unfortunately, spectra are not always easy to analyze. A myriad of real-world, physical effects related to both the sources, detector, and environment can alter a gamma spectrum from what is expected. Often, the resolution of a detector prevents characteristic peaks of two sources from being distinguishable, and sometimes resolution worsens as a detector ages. Simultaneously, a specific configuration of a source can produce an unexpected scattering effect making it look less like what one would expect. Moreover, environmental temperatures change, and this can cause the spectrum to shift without employing automatic gain stabilization at the detector level. These are but a few of the challenges faced by spectroscopists. While fission source data could face the real-world challenges mentioned above, it is less likely as the process by which such data is collected is typically more controlled.

2.2 Radioisotope Identification

The goal of identifying and quantifying the radioisotope contributors in measured gamma spectra is of major importance to national security, especially in areas such as nuclear device detection, nuclear material identification, nuclear treaty verification, and emergency response. Traditionally this task is performed manually by a subject matter expert (SME), namely a spectroscopist, who follows a series of steps to perform the analysis. These time-consuming steps can include photo-peak identification, background subtraction, and software-assisted template matching [3], and often rely heavily on the intuition/experience of the spectroscopist.

Although conventional methods will likely continue to be used, machine learning techniques have been shown as viable alternatives in a variety of limited problem spaces. In the last few decades, many off-the-shelf machine learning methods have been applied to this problem space with the goal of performing radioisotope identification faster and more accurately [4], [5], [6], [7], [8], [9], [10], [11], [12], [13].

Machine learning techniques were first applied to radioisotope identification by Olmos et al. in 1991 [4]. In their work they applied a simple neural network to single-isotope determination by training from limited experimental data. Although the model did not outperform conventional techniques, such as peak analysis, it gave promising results and demonstrated that machine learning could provide a simple and fast approach for gamma spectrum analysis, which would only improve with better model and data availability.

More recently, there has been significant progress in this area. For example, in 2017 Kamuda et al. [8] studied using artificial neural networks for identifying the relative proportions of radioisotopes from low-resolution mixture gamma spectra. Unlike some previous works, this paper focused on training with the entire normalized spectral shape rather than counts in regions of interest and utilized a large library of sources (32 radioisotopes). They predicted relative class contributions by training a two-layer neural network with softmax activation and cross-entropy loss. They demonstrated that the neural network could correctly identify the presence and approximate proportions for small mixtures of two radioisotopes. For larger mixtures of 5 radioisotopes, the model did not successfully predict their relative proportions, but could generally identify the largest contributors to the spectrum.

In 2020, Daniel et al. [11] use convolutional neural networks for radioisotope identification with gamma spectra. They trained a separate neural network model for each of the

6 sources they aimed to identify, where each model consisted of convolutional feature learning layers followed by fully connected layers and finally a binary output. Each model took the log-normalized gamma spectra as an input and minimized the binary-cross-entropy loss. For each model they found that they could achieve 90% accuracy with at least 1000 counts in the gamma spectrum and a relative proportion of at least 5%. Although this demonstrates the effectiveness of convolutional neural networks for radioisotope identification, this method does not allow for estimated mixture proportions of detected sources.

Recently there have been several papers that compare a variety of machine-learning techniques for radioisotope identification. For example, Qi et al. in 2022 [12] compare six different machine learning algorithms for single radioisotope identification including the support vector machine, k-nearest neighbor, logistic regression, naïve Bayes, decision tree, and multilayer perceptron methods. They compared these algorithms on two groups of datasets with 5 and 14 target nuclides and trained the algorithms on simulated data generated via Monte Carlo simulations. They demonstrated that all the methods were able to achieve similar performance (in terms of accuracy) on the simulated test datasets. They also found that although all the methods had slightly higher accuracies on the simulated data, the naïve Bayes and decision trees models performed significantly worse on experimental data.

Another recent work by Khatiwada et al. in 2023 [13] also compares several machine learning techniques for gamma ray radioisotope identification, including decision trees, gradient-boosted trees, k-nearest neighbors, Gaussian process regression, multi-layer perceptron (MLP), and convolutional neural networks. These model were trained on simulated data with various proportions of Uranium and Plutonium under various levels of shielding. They found that a fully connected neural network (i.e., MLP) achieved the best results in terms of mean error compared to the other methods.

2.3 Dictionary-Based Modeling and Sparse Coding

We say that a signal $\boldsymbol{x} \in \mathbb{R}^p$, where p is the length of the signal, has a dictionary decomposition if it can be closely represented as a linear combination of the columns (atoms, d) of a dictionary $\boldsymbol{D} \in \mathbb{R}^{p \times d}$. Dictionary learning methods involve simultaneously learning a dictionary as well as the corresponding coefficients. Dictionary learning techniques have led to state-of-the-art results in a variety of different tasks such as image denoising [14] [15] and facial recognition [16] [17]. In some applications the signal being estimated is sparse, in which case this is known as sparse coding. Sparse coding, an unsupervised dictionary learning method, involves learning a sparse representation of an input signal using a predefined dictionary. Sparse coding techniques are also known to be resilient to noisy and corrupted data, even in cases where a small amount of training data is available.

Although sparse coding methods are generally resilient to noisy data, they have been shown to be susceptible to cases where noise represents natural variation in training data, for example if an image of a face has different scales or poses for facial recognition [18]. This leads to disappointing performance on large datasets where variation like this is common. Several recent methods have aimed to mitigate these issues by pairing sparse coding techniques with neural networks which conversely perform well with large amounts of data [19] [20] [21] [22]. For example, Sun et al. [23] proposed a novel method for extending sparse coding to deep multi-layer networks by developing a sparse coding bottleneck module which pairs two sparse coding layers with wide and slim respective dictionaries to generate an intermediate lower-dimensional feature space.

To realize sparse coding on neural networks, rather than incorporating it into the network layers, sparse coding objectives can also be included in the loss function. The idea of using a dictionary learning method with a supervised loss function is not new. For example, Mairel et al. [24] propose a semi-supervised learning objective which combines two learning cost functions together with a trade-off parameter. The unsupervised term minimizes a sparse coding objective while the supervised term uses the learned sparse representation of the input signal as an input. This concept is very similar to the technique used in the supervised loss function, except both the unsupervised and supervised term use the model inputs (i.e., a dictionary-based representation is not used to learn the supervised task).

2.4 Out-Of-Distribution Detection

Typically, machine learning models are trained under the assumption that test data is drawn *i.i.d.* from the same distribution of training data. This is known as the closed-world (or closed-set) assumption and can be dangerous in practice, as models are generally applied in an open-world scenario [25]. In their natural habitats, models will often encounter test data that was unseen in training data and thus is out-of-distribution (OOD) [26]. Neural networks have been shown to assign over-confident predictions to OOD inputs, which could be especially dangerous in high-consequence scenarios. For example, Nguyen et. al. demonstrate that state-of-the-art deep neural networks will assign high probability predictions (> 99%) to completely unrecognizable images [18]. For a machine learning model to be trustworthy it should not only have a high performance in terms of the known classes, but it should also be able to identify OOD samples.

The field of OOD detection is not the only field which operates on the open-world assumption and is closely related to similar problems including outlier detection, anomaly detection, novelty detection, and open set recognition. All these fields are closely related, and the terms are often used interchangeably. There have been several recent surveys which aim to unify these fields under a common framework [25] [27]. These fields can be generally defined as detecting when a test sample comes from a different distribution than the in-distribution (ID) training data due to some sort of distribution shift (shift in the label space, shift in sensory conditions, etc.).

According to Yang et al. [25] OOD detection techniques can be grouped into four main categories: classification-based detection, distance-based detection, density-based detection, and reconstruction-based detection. Classification-based methods rely on the output of classifiers to identify OOD samples and originate from simply using the softmax probabilities as OOD indicators [28]. Several post-hoc methods [29], [30], [31] fall into this category with various techniques including input perturbation, data augmentation, and adversarial training to expand the separability between ID and OOD samples. A popular approach called outlier exposure is a classification-based OOD detection method which involves pre-training (or exposing) a model on an auxiliary OOD dataset which has been shown to result in better differentiation between ID and OOD inputs [32], [33]. Distance-based OOD detection methods operate on the assumption that OOD samples should be relatively far from ID prototypes in terms of some distance metric such as the Mahalanobis distance [34], [35], cosine similarity [36], radial basis function kernel [37], and Euclidean distance [38]. Density-based methods involve explicitly modeling the ID data under some probability distribution and then declaring OOD samples in the low-density regions [39], [40]. Finally, reconstruction-based methods assume that the reconstruction of ID and OOD samples based on some generally smaller-dimensional latent representation will yield different values. This is typically done with an encoder-decoder model [41], [42], [43], [44].

The approach taken in this work for OOD detection would best be categorized as a reconstruction-based method because the difference between the reconstruction input signal and observed input signal is used as an OOD indicator. However, the OOD detector developed here is realized by declaring a sample OOD if it falls in the low-density region of reconstruction errors for ID samples (see 6.4), so it could also be viewed as a density-based OOD detection method.

The approach taken in this work was partly motivated by Katz-Samuels et al. in [26], who explored a similar setting for OOD detection as in this paper. Inspiration was taken from their feedforward model structure which used shared network parameters to perform both multi-class classification and OOD detection simultaneously.

Chapter 3 Problem Setup

In this chapter we formally define LPE and OOD detection in the context of the multi-label gamma spectra classification problem. First, the necessary condition for classes to be represented by probability distributions is discussed. This is followed by a formulation of the training data and goals in terms of quantifiable metrics. Finally, we address the case where label proportions are known to be sparse.

3.1 Distributional Assumption

This approach for LPE and OOD detection is based on the idea that each class can be viewed as a unique probability distribution. With this in mind, label proportions can be viewed as mixture proportions, and the observations can be viewed as a mixture of distributions. Moreover, if the class distributions are assumed to be discrete, then the true distribution for each class can be represented as a normalized histogram. This approach relies on the assumption that these histograms (the true distribution of each class) are known a priori.

For this problem in particular, these discrete distributions represent the pure spectral shape of each radioisotope the model predicts. This assumption is reasonable in real-world applications as most only care about a specific set of sources, and the spectral shape of those sources can be determined by measurement or simulation for the particular detector being used.

Formally, the model assumes access to a dictionary $\boldsymbol{D} \in \mathbb{R}^{p \times d}$ a priori, where each column (atom) contains the pure, normalized (in this case, normalization means divided through by total counts) spectrum for a specific radioisotope. Training samples are then synthesized from this dictionary as a random mixtures of columns, collectively and randomly scaled by signal-to-noise ratio (SNR), and with noise included for both background subtraction and Poisson statistics (more details in chapter 5). Each input is therefore approximately representable as a linear combination of dictionary columns,

$$\boldsymbol{x} = \boldsymbol{D} * \boldsymbol{y} + \boldsymbol{n},$$

where $\boldsymbol{n} \in \mathbb{R}^p$ is a vector representing noise. Based on this interpretation of the problem, a dictionary-based modeling approach, where the predicted isotope proportions can be viewed as dictionary coefficients, seemed justified.

3.2 Training Data

With this distributional assumption in mind, the inputs to this model will be measured a mixture of distributions, and in this case a mixture gamma spectra. The corresponding labels will be the true mixture proportions of each contributing source and must be in the probability simplex (i.e., the proportions must be non-negative and sum to one). Formally, let $\mathcal{X} \in \mathbb{R}^p$ denote the input space (i.e., input spectra have p channels) and $\mathcal{Y} \in \Delta^{d-1}$ (where d is the number of sources targeted) denote the label space. Then assume access to a labeled, training dataset $\mathcal{D} :$ $= \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$ where n samples are drawn *i.i.d.* from the joint distribution $\mathbb{P}_{\mathcal{X}\mathcal{Y}}$.

3.3 Goal

Suppose we are given a new observed gamma spectrum $\boldsymbol{x} \in \mathbb{R}^p$ from an unknown mixture of isotopes. The goal is to accurately predict the proportion of known sources in the mixture (LPE) while simultaneously detecting whether the observation is OOD. Let $f_\theta: \mathcal{X} \rightarrow \Delta^{d-1}$ denote a model for the label proportion estimation task, in this case a neural network with a single hidden layer. Let $g_\theta: \mathcal{X} \rightarrow \{in, out\}$ be a binary classification function which maps each test input $\boldsymbol{x} \in \mathbb{R}^p$ to either ID or OOD. Note that for this work only the case where samples are

OOD due to a semantic distribution shift (i.e., some proportion of the counts of a spectrum come from a novel source) is considered. Also notice that both f and g are parameterized by θ to indicate that they use shared parameters.

Formally, suppose we are given an unlabeled test dataset which may contain both ID and OOD samples, drawn from $\mathbb{P}_{\text{in}}$ and $\mathbb{P}_{\text{out}}$, respectively. The goal will be to minimize the mean absolute error (MAE) of the label proportion estimator on ID samples while maximizing the F1 score of the OOD detector. The priority given to each of these objectives can be controlled by the user via a hyperparameter referred to as beta (β).

3.4 Optional Condition of Sparsity

In some applications the mixtures proportions are known to be a sparse mixtures where only a small number of classes are present at one time. This is an important case to consider. In this case, Section 4.1.2 offers a variation of the proposed approach that accounts for this alternative assumption.

Chapter 4 Method

The main idea of the proposed method is to train a neural network by minimizing the empirical risk associated with a custom loss function which leverages the assumption mentioned in the previous chapter. A supervised loss term is paired with an unsupervised loss term to promote consistency with the assumed dictionary $\boldsymbol{D}$ via a reconstruction error objective. The reconstruction error objective will compare the input signal to a reconstructed input signal created using predicted proportion estimates and the dictionary.

For the training dataset $\mathcal{D} := \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$, the semi-supervised learning objective takes the form of,

$$L = \sum_{i=1}^{n} L_s\,(\widehat{\boldsymbol{y}}_i, \boldsymbol{y}_i) + \beta L_{us}(\widehat{\boldsymbol{x}}_i, \boldsymbol{x}_i),$$

where L_s, L_{us} are the supervised and unsupervised loss terms, respectively, $\beta \geq 0$ is a trade-off parameter, $\widehat{\boldsymbol{y}}_i = f_\theta(\boldsymbol{x}_i)$ are the predicted proportions, and $\widehat{\boldsymbol{x}}_i = \boldsymbol{D} * \widehat{\boldsymbol{y}}_i$ is the reconstructed input signal. The trade-off parameter β controls the priority of the unsupervised term.

4.1 Supervised Term – A Loss Function for Label Proportion Estimation

In this section the traditional supervised loss function for estimating label proportions in this model is introduced. An alternative is also discussed for the case when the true label proportions are known to be sparse.

4.1.1 Cross Entropy Loss

For the supervised term, L_s, in the loss function, the categorical cross-entropy loss (softmax activation + cross-entropy loss) naturally fits. Typically, cross-entropy loss is used for multi-class classification with a one-hot encoding and can be used to estimate class probabilities as it outputs a distribution. Cross-entropy can also be used for predicting label proportions. For example, Kamuda et al. use cross-entropy for LPE in the context of radioisotope identification like ours [8]. Let the predicted proportions $\hat{\boldsymbol{y}} = f_\theta(\boldsymbol{x})$ be the already softmax-activated logits from the model with corresponding true label proportions $\boldsymbol{y}$. Then the cross-entropy loss can simply be defined as,

$$L_s^{CE}(f_\theta(\boldsymbol{x}), \boldsymbol{y}) = -\sum_{j=1}^{d} \boldsymbol{y}_j \log(f_\theta(\boldsymbol{x})_j) \,.$$

4.1.2 Sparsemax Loss

This section introduces an alternative supervised loss function to the cross-entropy loss if the true labels proportions are known to be sparse. To promote sparsity, ℓ_1 regularization is typically used as a convex relaxation of the ℓ_0 norm. However, this is not ideal for truly sparse data when paired with the categorical cross-entropy loss as this approach relies on softmax activation to map the model scores to the probability simplex, and softmax will always map to a dense distribution (i.e., distribution with full support). Thus, to obtain truly sparse proportions, the softmax output must be thresholded at some arbitrarily chosen value. Moreover, as label proportions are being predicted, the predictions will already sum to one, so ℓ_1 regularization would not make sense.

In this case, the sparsemax loss function for sparse label proportion estimation, proposed by Martin et al. [45], would be a better option for the supervised term in the learning objective. The sparsemax loss is based on a the sparsemax activation function, which is like softmax but outputs truly sparse probability distributions. For some vector $\boldsymbol{x} \in \mathbb{R}^d$, the sparsemax activation is defined as,

$$\text{sparsemax}(\boldsymbol{x}) = \underset{\boldsymbol{p}\in\Delta^{d-1}}{\text{argmin}} \; ||\boldsymbol{p} - \boldsymbol{x}||_2^2,$$

which is the Euclidean projection of the input vector $\boldsymbol{x}$ onto the probability simplex. In other words, the sparsemax activation function is the closest valid probability distribution in terms of the ℓ_2 distance. Since this projection is often on the boundary of the probability simplex, the sparsemax will nearly always output a sparse probability distribution, with some of the terms close to or exactly zero. The sparsemax has a closed form solution based on soft thresholding,

$$\text{sparsemax}_i(\boldsymbol{x}) = [x_i - \tau(\boldsymbol{x})]_+,$$

$$\tau(\boldsymbol{x}) = \frac{\left(\sum_{j\in S(\boldsymbol{x})} x_j\right) - 1}{|S(\boldsymbol{x})|},$$

where $S(\boldsymbol{x})$ is the support of the sparsemax activation: $S(\boldsymbol{x}) \coloneqq \{j \in [d] \mid \text{sparsemax}_j(\boldsymbol{x}) > 0\}$. This can be computed in $O(d \log d)$ time.

Martins et al. [45] propose a loss function based on the sparsemax activation for estimating sparse label proportions. Let $\widehat{\boldsymbol{y}} = f_\theta(\boldsymbol{x})$ be the model prediction for some input $\boldsymbol{x}$ with corresponding true label proportions $\boldsymbol{y}$. Then the sparsemax loss is defined as,

$$L_{\text{sparsemax}}(\widehat{\boldsymbol{y}};\ \boldsymbol{y}) = -\boldsymbol{y}^T\widehat{\boldsymbol{y}} + \frac{1}{2}\sum_{j \in S(\widehat{\boldsymbol{y}})} \left(\widehat{\boldsymbol{y}}_j^2 - \tau(\widehat{\boldsymbol{y}})\right) + \frac{1}{2}\|\boldsymbol{y}\|_2^2.$$

The gradient for this loss simplifies nicely to,

$$\nabla_{\widehat{\boldsymbol{y}}} L_{\text{sparsemax}}(\widehat{\boldsymbol{y}};\ \boldsymbol{y}) = -\boldsymbol{y} + \text{sparsemax}(\widehat{\boldsymbol{y}}).$$

This is identical to the gradient for the categorical cross-entropy loss, except the softmax activation is replaced with sparsemax activation.

4.2 Unsupervised Term – Penalizing Dictionary-Based Reconstruction Error

The main idea behind the unsupervised term in the loss function is to minimize the difference between the input signal and the estimated reconstruction of the input signal. This estimated reconstruction can be found as a linear combination of the dictionary columns with the predicted coefficients: $\widehat{\boldsymbol{x}} = \boldsymbol{D} * f_\theta(\boldsymbol{x})$. Including this in the loss function will encourage the model to make predictions which are consistent with the predefined dictionary. The quantity resulting from a comparison between the input spectrum and the reconstructed spectrum is referred to as *reconstruction error*. In the following sections several different reconstruction error functions are considered.

4.2.1 Poisson Negative Log-Likelihood (PNLL)

It has been shown that each channel of a gamma spectrum can be modeled as a Poisson random variable with a mean equal to the channel count and variance equal to the square root of the mean (standard deviation equals variance) [1]. The idea here is instead of minimizing the differences between the input and reconstruction, the reconstruction probability in terms of this Poisson noise model is being maximized. This is equivalent to minimizing the negative log-likelihood, which is described as,

$$L_{us}^{poi} = -\log \Pr(\hat{\boldsymbol{x}} \mid \boldsymbol{X} = \boldsymbol{x}),$$

where $\hat{\boldsymbol{x}} = \boldsymbol{D} * f_\theta(\boldsymbol{x})$ is the reconstructed signal. In other words, this is the probability of measuring a reconstructed signal $\hat{\boldsymbol{x}} \in \mathbb{R}^p$ given an input signal $\boldsymbol{x} \in \mathbb{R}^p$. Expanding this for each energy bin in the spectrum gives,

$$\Pr(\hat{\boldsymbol{x}} \mid \boldsymbol{X} = \boldsymbol{x}) = \Pr(\hat{X}_1 = \hat{x}_1, \hat{X}_2 = \hat{x}_2, \ldots \hat{X}_p = \hat{x}_p \mid \boldsymbol{X} = \boldsymbol{x}),$$

and assuming that each channel is measured independently,

$$\Pr(\hat{\boldsymbol{x}} \mid \boldsymbol{X} = \boldsymbol{x}) = \Pr\left(\hat{X}_1 = \hat{x}_1 \left| X_1 = x_1\right.\right) * \Pr\left(\hat{X}_2 = \hat{x}_2 \left| X_2 = x_2\right.\right) * \ldots * \Pr\left(\hat{X}_p = \hat{x}_p \left| X_p = x_p\right.\right).$$

Because each channel of a gamma spectrum can be modelled as a Poisson random variable, this can be simplified to,

$$\Pr(\hat{\boldsymbol{x}} \mid \boldsymbol{X} = \boldsymbol{x}) = \prod_{j=1}^{p} Poi(\hat{x}_j; \lambda_j = x_j) = \prod_{j=1}^{p} \frac{x_j^{\hat{x}_j} e^{-x_j}}{\hat{x}_j!}.$$

Then the negative log-likelihood can be written as a summation,

$$-\log \Pr(\hat{\boldsymbol{x}} \mid \boldsymbol{X} = \boldsymbol{x}) = \sum_{j=1}^{p} (x_j - \hat{x}_j \log x_j + \log \hat{x}_j!).$$

Thus, the unsupervised loss term can be written as,

$$L_{us}^{poi}(\boldsymbol{D} * f_\theta(\boldsymbol{x}), \boldsymbol{x}) = \sum_{j=1}^{p} (x_j - [\boldsymbol{D} * f_\theta(\boldsymbol{x})]_j \log x_j + \log[\boldsymbol{D} * f_\theta(\boldsymbol{x})]_j!).$$

4.2.2 Gaussian Negative Log-Likelihood (GNLL)

It is well-known that for sufficiently large count rates a Gaussian random variable can be used as an approximation of a Poisson random variable. In particular, a Poisson random variable $\mathcal{X} \sim Poisson(\lambda)$ can be approximated as

$$\mathcal{X} \approx \mathcal{N}(\mu = \lambda, \sigma^2 = \lambda).$$

Although for high count rates this approximation is excellent, for low count rates a Poisson model is more precise, especially as the number of counts approaches zero [46]. For this reason, directly using the GNLL as an unsupervised loss function would not be expected give better performance than using the PNLL, but it is considered for comparison. And in the next section the GNLL is further simplified into a new unsupervised loss function with some nice optimization properties for this problem space.

Just like the unsupervised Poisson negative log-likelihood shown the in previous section, the Gaussian negative log-likelihood can also be derived in a similar way. This will estimate the probability of an input reconstruction in terms of a Gaussian noise model. Using this Gaussian model, the probability can be expressed similar to before as,

$$\Pr(\hat{\boldsymbol{x}} \mid \boldsymbol{X} = \boldsymbol{x}) = \prod_{j=1}^{p} \mathcal{N}(\hat{x}_j; \mu_j = x_j, \sigma_j^2 = x_j) = \prod_{j=1}^{p} (\frac{1}{\sqrt{2\pi x_j}} \cdot \exp\left(-\frac{1}{2}\frac{\left(\hat{x}_j - x_j\right)^2}{x_j}\right)).$$

Then the negative log-likelihood can be given as,

$$-\log \Pr(\hat{\boldsymbol{x}} \mid \boldsymbol{X} = \boldsymbol{x}) = \frac{1}{2}\sum_{j=1}^{p} (\log\left(2\pi x_j\right) + \frac{\left(\hat{x}_j - x_j\right)^2}{x_j}).$$

Then this unsupervised term can be written as,

$$-\log \Pr(\hat{\boldsymbol{x}} \mid \boldsymbol{X} = \boldsymbol{x}) = \frac{1}{2}\sum_{j=1}^{p} (\log\left(2\pi x_j\right) + \frac{\left([\boldsymbol{D} * f_\theta(\boldsymbol{x})]_j - x_j\right)^2}{x_j}).$$

In practice, however, issues arise for low-count channels, especially when there are exactly zero counts (which is common at high-energies) where this unsupervised loss breaks down. At the zero-count limit, the Gaussian approximation is defined with zero uncertainty which does not make sense statistically and runs into a divide-by-zero problem. Like the approach used by Lass et al. in [47], this is resolved by setting a lower threshold for the variance at one. Then the GNLL can be adjusted to,

$$L_{us}^{\mathcal{N}}(\boldsymbol{D} * f_\theta(\boldsymbol{x}), \boldsymbol{x}) = \frac{1}{2}\sum_{j=1}^{p}(\log\big(2\pi \max(x_j, 1)\big) + \frac{\big([\boldsymbol{D} * f_\theta(\boldsymbol{x})]_j - x_j\big)^2}{\max(x_j, 1)}).$$

4.2.3 Sum of Squared Errors (SSE)

A new unsupervised loss function can be obtained by forcing the variance to remain constant in the GNLL loss (a homoscedastic GNLL). Then the GNLL loss for one sample can be simplified to,

$$\frac{1}{2}\sum_{j=1}^{p}(\log(2\pi\sigma^2) + \frac{\big([\boldsymbol{D} * f_\theta(\boldsymbol{x})]_j - x_j\big)^2}{\sigma^2})$$

Notice that as the variance no longer affects the minimization problem, this is equivalent to simply minimizing the sum of squared errors (SSE) or the squared ℓ_2-norm difference between the input and reconstruction ($\|\boldsymbol{y} - \boldsymbol{D} * f_\theta(\boldsymbol{x})\|_2^2$).

Although this simplification no longer accurately models the conditional probability of measuring a reconstructed spectrum, it does have several nice properties for optimization. For one, it eliminates the zero-count issue that is present when computing the GNLL loss. Also, by assuming a constant variance across all the energy channels, this loss will exponentially penalize channels which have higher errors. For example, in typical gamma spectra high-energy channels generally contain very few counts, often exactly zero (of course depending on SNR and the detector live-time), while spectral peaks in lower energy channels will generally have thousands of counts. By assuming constant variance, the error from being a few counts off in low-count channels will be very small compared to the error from being a few hundred counts off in high-count channels. In other words, this SSE unsupervised loss should encourage peak fitting more than the GNLL and PNLL which judge each energy channel with equal importance.

For this study, the SSE loss is not fully simplified and the constant variance terms in the unsupervised loss function are left in place, with a constant variance selected based on the sample variance to obtain loss values on a similar scale as the PNLL and GNLL reconstruction error functions. Using $\sigma^2 = Var(\boldsymbol{x})$ would make the most sense statistically, where $Var(\boldsymbol{x})$ refers to the sample variance over all the counts in a gamma spectrum. However, as the SSE is a less accurate probabilistic model of count spectra than the PNLL and GNLL, the reconstruction errors are on a much higher scale. To combat this issue, $\sigma^2 = \sqrt{Var(\boldsymbol{x})}$ was chosen as the constant variance term in the SSE reconstruction error function, based on empirical observations of the range of reconstruction errors. We chose to apply the square root as this adjusted the scale of the reconstruction errors for this unsupervised loss function to be on the same scale as the other reconstruction error loss functions (PNLL and GNLL). As this value is just a constant it does not affect the solution of the minimization problem but was chosen simply to allow us to set the same β value when comparing performance later between the different unsupervised loss functions. In particular,

$$L_{us}^{SSE}(\boldsymbol{D} * f_\theta(\boldsymbol{x}), \boldsymbol{x}) = \frac{1}{2}\sum_{j=1}^{p}(\log\left(2\pi\sqrt{Var(\boldsymbol{x})}\right) + \frac{\left([\boldsymbol{D} * f_\theta(\boldsymbol{x})]_j - x_j\right)^2}{\sqrt{Var(\boldsymbol{x})}}).$$

An alternative method for balancing loss terms would be to use a method for adaptively adjusting the weighting between the supervised and unsupervised loss terms while training (see Future Work).

4.3 Tying it All Together

Combining the three unsupervised losses with the supervised cross-entropy loss gives three semi-supervised loss functions which were used to train the model. Formally, for the labelled training dataset $\mathcal{D} := \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$, the three learning objectives can be written as:

$$L_{poi} = \min_{\theta} \sum_{i=1}^{n} [- \sum_{j=1}^{d} [\boldsymbol{y}_i]_j \log(f_\theta(\boldsymbol{x}_i)_j) + \beta \sum_{j=1}^{p} ([\boldsymbol{x}_i]_j - [\boldsymbol{D} * f_\theta(\boldsymbol{x}_i)]_j \log[\boldsymbol{x}_i]_j + \log[\boldsymbol{D} * f_\theta(\boldsymbol{x}_i)]_j!)]$$

$$L_{\mathcal{N}} = \min_{\theta} \sum_{i=1}^{n} [- \sum_{j=1}^{d} [\boldsymbol{y}_i]_j \log(f_\theta(\boldsymbol{x}_i)_j) + \beta \frac{1}{2} \sum_{j=1}^{p} (\log\left(2\pi \max([\boldsymbol{x}_i]_j, 1)\right) + \frac{\left([\boldsymbol{D} * f_\theta(\boldsymbol{x})]_j - [\boldsymbol{x}_i]_j\right)^2}{\max([\boldsymbol{x}_i]_j, 1)})]$$

$$L_{SSE} = \min_{\theta} \sum_{i=1}^{n} [- \sum_{j=1}^{d} [\boldsymbol{y}_i]_j \log(f_\theta(\boldsymbol{x}_i)_j) + \beta \frac{1}{2} \sum_{j=1}^{p} (\log\left(2\pi \sqrt{Var(\boldsymbol{x}_i)}\right) + \frac{\left([\boldsymbol{D} * f_\theta(\boldsymbol{x}_i)]_j - [\boldsymbol{x}_i]_j\right)^2}{\sqrt{Var(\boldsymbol{x}_i)}})]$$

4.4 OOD Detection

The proposed OOD detector is predicated on the fact the unsupervised losses (PNLL, GNLL, and SSE) do not depend on the true label proportions. Thus, these losses can be computed even for test cases where the true label proportions are unknown. The idea is that these metrics should not only improve the model by encouraging predictions consistent with the dictionary, but that they should also provide a built-in metric for OOD detection. When a novel/anomalous source is sufficiently present in an observed spectrum (i.e., some nonzero proportion of the measured counts come from a novel isotope), the model will not be able to generate an accurate reconstruction as the dictionary is missing the column associated with the novel source. Even if the model correctly predicts the proportions of known isotopes, the presence of the novel source should result in an inferior reconstruction. In other words, the stronger and more distinct a novel source is, the harder it will be for the model to explain itself.

To force the model to generate poor reconstructions on out-of-distribution (OOD) samples, the prioritization of in-distribution reconstruction is encouraged by increasing β.

The idea of using reconstruction error as a means of OOD detection is not new. Recently Ghawaly et al. in 2022 [44] proposed a method called ARAD for detecting anomalies in gamma ray spectra using a deep convolution neural network. Their technique uses a deep convolutional autoencoder to learn a lower dimensional latent representation of input spectra. A decoder convolutional neural network then attempts to recreate the input spectra from the latent representation. They then use the Jenson Shannon distance as a reconstruction metric to compare their input and reconstructed input. Their approach, however, is different than this method in several aspects. Their reconstructed spectra are generated using a deep convolution auto-encoder neural network, whereas the method presented in this thesis generates a reconstruction simply as a linear combination of the dictionary columns. Their proposed ARAD model has a single objective of anomaly detection, while this approach pursues the detection of anomalies alongside proportion estimation of known sources. Furthermore, this work considers different metrics when computing reconstruction errors. The ARAD model uses the Jenson Shannon distance to compare the reconstruction to the input, while this paper proposes a Poisson probability model to measure the likelihood of the reconstruction compared to the input.

This thesis performs OOD detection as a binary classification problem and realizes the classifier $g_\theta(\boldsymbol{x})$ as a decision function using the unsupervised term of the learning objective,

$$g_\theta(\boldsymbol{x}) = \begin{cases} "in", & L_{us}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) < \gamma \\ "out", & L_{us}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) \geq \gamma' \end{cases}$$

where $\gamma \in \mathbb{R}$ is a threshold for detection. In practice the detection threshold γ must be set manually based on the typical reconstruction error in the model’s natural habitat for ID inputs. A sample is declared OOD if it lies in the low-density region of the ID reconstruction errors. More detail on this is provided in chapter 6.

4.5 A Probabilistic Interpretation

Earlier in this chapter, each of the unsupervised loss functions (PNLL, GNLL, and SSE) was shown to be derived from directly computing the conditional likelihood of a reconstruction given an input signal under different statistical models for the data. For example, the PNLL reconstruction error assumes that each channel in a measured gamma spectrum can be modeled as a Poisson random variable. Similarly, the GNLL and SSE reconstruction errors assume a Gaussian and homoscedastic Gaussian model for gamma spectrum channels. As minimizing the negative log-likelihood of the reconstructions is equivalent to maximizing their likelihoods, each of the unsupervised loss terms can be viewed as maximum likelihood estimators. This makes sense considering it is well known that minimizing the sum of squared errors is equivalent to performing maximum likelihood estimation under a Gaussian prior.

The supervised loss in the proposed learning objective can also be considered a maximum likelihood estimator. Cross-entropy is a function that measures the difference between two probability distributions. When used as a loss function, in the case of this model, it measures the difference between the true and predicted distribution of class labels. It is also known that minimizing the cross-entropy loss is equivalent to maximizing the likelihood of the predicted class distributions over the model parameters.

Thus, the overall semi-supervised loss function can be viewed as a paired maximum likelihood estimation problem, where the priority between the two maximum likelihood estimators can be controlled via a scalar parameter β.

Chapter 5 Generating Synthetic Training Data

This section describes the training dataset and how it was synthesized from assumptions made about the response of a high purity germanium (HPGe) detector.

5.1 Gamma Spectra Sources

The training data for this paper consists of synthetic fission source gamma spectra based on an 8 in^3 handheld HPGe detector using GADRAS [48]. Each training spectrum contains a random mixture of the following 30 sources: ^{112}Ag, ^{78}As, ^{139}Ba, ^{140}Ba, ^{143}Ce, ^{124}I, ^{131}I, ^{132}I, ^{133}I, ^{134}I, ^{135}I, ^{85m}Kr, ^{87}Kr, ^{140}La, ^{142}La, ^{99}Mo, ^{149}Nd, ^{150}Pm, ^{105}Rh, ^{105}Ru, ^{115}Sb, ^{129}Sb, ^{91}Sr, ^{92}Sr, ^{132}Te, ^{235}U, ^{88}Y, ^{93}Y, ^{91m}Y, and ^{95}Zr. A pure spectral signature for each source, known as a seed, is obtained via GADRAS Inject based on the detector at 100 cm distance, 100 cm height, and a dead time of 23 μs. Each seed spectrum consists of 16,384 energy channels spanning an 8 MeV energy range. The energy ranges of the seed spectra are cut to span ~30keV to 4 MeV, as all the meaningful features are contained in that range. And to reduce computational costs and model

Source	Expected Proportion	Source	Expected Proportion	Source	Expected Proportion
Ag112	0.0267	I135	0.0868	Sb115	0.0303
As78	0.0403	Kr85m	0.0227	Sb129	0.0506
Ba139	0.0197	Kr87	0.0252	Sr91	0.06
Ba140	0.0235	La140	0.0214	Sr92	0.0128
Ce143	0.0266	La142	0.0372	Te132	0.0328
I124	0.0377	Mo99	0.0067	U235	0.0337
I131	0.0217	Nd149	0.0952	Y88	0.0261
I132	0.0399	Pm150	0.0402	Y93	0.0257
I133	0.0168	Rh105	0.0204	Y91m	0.0157
I134	0.0429	Ru105	0.0369	Zr95	0.024

Table 5.1: list of foreground sources, along with count proportion in typical gamma fission spectrum

complexity, the resulting seed spectra are uniformly down binned to 2048 channels. Plots of the seed spectra are shown in Appendix A.

5.2 Generating Random Mixtures of Foreground Seeds

With seeds obtained from GADRAS, the next step in simulating realistic mixture spectra is to create pure mixture spectra by combining the seed spectra with random proportions. This is accomplished with PyRIID (Python-based Radioisotope Identification) [49], an open-source Python package for synthetically generating gamma spectra. PyRIID's methods use the Dirichlet distribution to randomly sample the mixture proportions based on a provided parameter. The Dirichlet distribution, a multivariate generalization of the beta distribution, can be used to randomly sample a vector of positive proportions which sum to one. The Dirichlet distribution can be thought of as a distribution of distributions. It has support over the probability simplex (i.e., $\boldsymbol{x} \in \mathbb{R}^k$, where $x_i \in [0,1]$ and $\sum_i x_i = 1$), where it is defined as,

$$Dir(\boldsymbol{x}|\boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{k} x_i^{\alpha_i - 1},$$

where $B(\cdot)$ is the multivariate beta function, and $\boldsymbol{\alpha}$ is a vector parameter controlling the expected shape of the distribution. If $\alpha_1 = \alpha_2 = \cdots = \alpha_k$, then the all the mixture proportions will be sampled uniformly, whereas $\alpha_1 = 2, \alpha_2 = 1, \dots, \alpha_k = 1$ would generate a skewed distribution with the first proportion greater than all others on average. The magnitude of $\boldsymbol{\alpha}$ controls the strength or variation of the distribution. In other words, $\alpha_1 = \alpha_2 = \cdots = \alpha_k \to \infty$ would generate completely uniform distributions with no variance (i.e., all proportions equal). For $\alpha_1 = \alpha_2 = \cdots = \alpha_k \to 0$, the Dirichlet distribution would converge on outputting a vector with a single non-negative proportion equal to 1, thus maximizing the variance.

The $\boldsymbol{\alpha}$ parameter of the Dirichlet distribution provides an opportunity to utilize prior information when sampling random mixture proportions, specifically the expected proportions of sources present in a measurement taken of a fission source. The expected proportions for each fission source, determined through consultation with a subject matter expert, are shown in Table 5.1. These proportions can then be multiplied by a large scalar, in this case 300, to create variation of each proportion centered on the expectation and usable as the vector parameter $\boldsymbol{\alpha}$ when generating random mixtures. For the training dataset, 1e4 random mixtures were sampled.

5.3 Static Synthesis of Mixture Spectra

With the randomly sampled mixtures of foreground sources in hand, PyRIID is used to simulate additional types of noise, most notably SNR variation and imperfect background subtraction. Before discussing how this is accomplished, it is necessary to define some terms used by PyRIID. In practice, every detected gamma spectrum will be a gross spectrum, where measured counts come from both the foreground (or net) sources (the targets of interest) and background sources (noise always present in the environment). Thus, a measured spectrum can be thought of as the summed foreground and background spectra. Let the gross, foreground, and background counts be defined as,

$$gross\text{ counts } \equiv G$$

$$\text{foreground counts} \equiv F$$

$$\text{background counts } \equiv B.$$

From this, PyRIID defines the signal-to-noise ratio (SNR) as,

$$snr = \frac{F}{B}.$$

With a collection time (also referred to as “live time”) over which the detector was counting, t, the count rate in counts per second (cps) is obtained as follows,

$$\text{gross counting rate (cps)} \equiv g = \frac{G}{t}$$

$$\text{foreground counting rate (cps)} \equiv f = \frac{F}{t}$$

$$\text{background counting rate (cps)} \equiv b = \frac{B}{t}.$$

Synthetically generating a gamma spectrum begins by modeling the number of counts in each channel (or energy bin). As described in [1], the Binomial distribution can be used to model the number of counts in each energy bin. And because the number of nuclei is generally very large and the collection times are relatively short compared to half-life of the measured nuclei, the Poisson distribution can be used in place of the Binomial distribution, which is parameterized by a single parameter equal to the expected number of counts for that energy bin [46]. Although for expected values above 25, the Poisson distribution can be approximated with a Gaussian, the Poisson distribution is used as it is applicable to channels with both low and high counts.

To synthesize a realistic p-channel gamma spectrum, the following information must be known a priori: a target SNR value ($\widetilde{snr}$), a target live time ($\tilde{t}$), a target background count rate ($\tilde{b}$), a known distribution of foreground counts across all channels ($\boldsymbol{f}$), and a known distribution of background counts across all channels ($\boldsymbol{b}$). In particular, the known foreground and background distributions must be p-dimensional, non-negative vectors which sum to one. The foreground distributions used here are the normalized mixture spectra described in the previous section. A single, normalized representative background spectrum (taken as a combination of K, U, Th, and cosmic radiation) is used as the target background distribution for all the training samples.

Then the process to synthesize a single gross gamma spectrum is as follows. First, expected foreground and background counts can be found as,

$$\tilde{B} = \tilde{t} \times \tilde{b}$$

$$\tilde{F} = \tilde{B} \times \widetilde{snr}.$$

Then the expected gross spectrum can be found as,

$$\widehat{\boldsymbol{G}} = \tilde{B} \times \boldsymbol{b} + \tilde{F} \times \boldsymbol{f}.$$

The final synthetic gross spectrum, $\boldsymbol{G}$**,** is obtained by sampling the counts in each channel of $\widehat{\boldsymbol{G}}$ from a Poisson distribution, where each mean is the expected number of counts for that channel,

$$G_i \leftarrow Poisson\left(\hat{G}_i\right).$$

All these synthesis steps are performed automatically using the *StaticSynthesizer* in PyRIID. For the fission source dataset in this study, the $\widetilde{snr}$ is randomly sampled 50 times for each mixture from a uniform distribution ranging from 1 to 100. The live time is held constant at 600 seconds.

5.4 Additional Preprocessing Steps

Before the model is trained, the following preprocessing steps are applied to the synthetic dataset. First, background is subtracted from each Poisson-sampled gross spectrum by removing a 600-second, Poisson-sampled background spectrum. By using a Poisson-sampled background spectrum, instead of just the expectation, the intent is to represent the imperfect subtraction in the real world. As a result, it is possible for low-SNR samples to have negative counts in some channels. Such negative values are clipped to zero to enable the calculation of the Poisson Negative Log-Likelihood. Finally, the resulting background-subtracted spectra are normalized by dividing through by total counts such that each spectrum sums to one, equivalent to an L_1 norm.

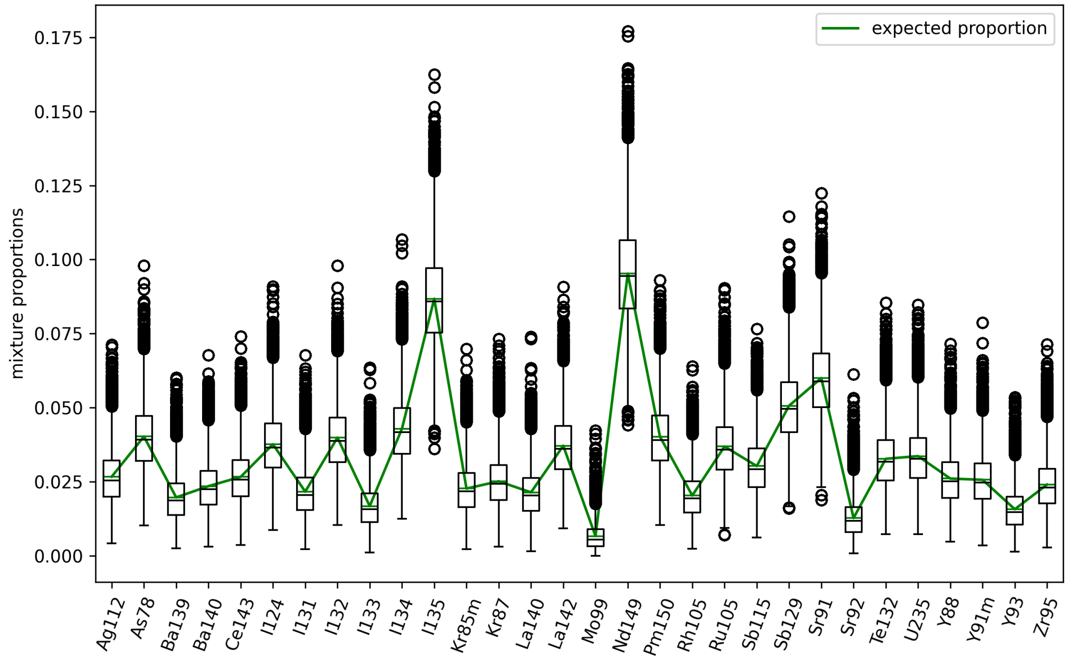


Figure 5.1: a boxplot showing the randomly sampled distributions of mixture proportions for each source in the training dataset, along with the expected proportion obtained from analysis of a measured fissions source spectrum.

5.5 Evaluating the Training Dataset

The training dataset used in this study, whose generation is described in the previous subsections, contains 5×10^5 samples (10k random mixtures with 50 random samples/mixture varying SNR). As mentioned previously, the expected source proportions from a measured fission source spectrum are collectively used as the alpha parameter of the Dirichlet distribution when generating random mixture proportions. The distribution of actual mixture proportions of the training dataset, along with the provided expected proportions, are shown in Figure 5.1. To double-check the behavior of the mixer, the expected proportion for each source falls well-within the inner quartile of the synthetic proportions and is very close to the median value. Also, the distribution of proportions for each isotope empirically follows a Gaussian distribution (truncated at 0 of course), which is shown for two select isotopes in Figure 5.2. The plotted distributions across the entire training set for each isotope are including in Appendix B. Utilizing this expected distribution when generating training data makes the problem more tractable to solve while also encouraging the model to predict more realistic proportions.

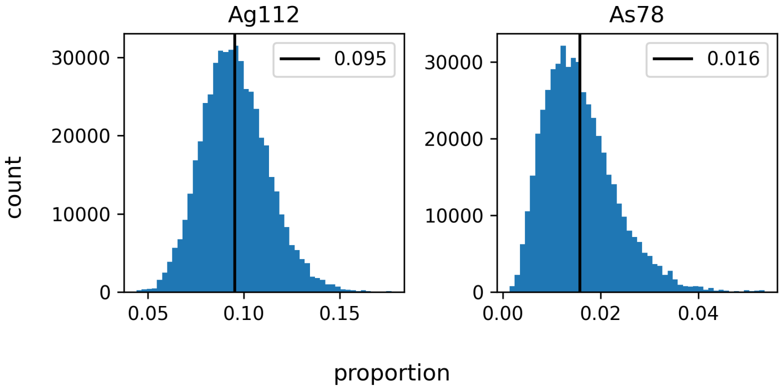


Figure 5.2: distribution of mixture proportions for Ag112 and As78, the black line indicates the expected measured proportion obtained from SME analysis

Chapter 6 Experimental Results

6.1 Neural Network Model

The proposed semi-supervised learning objective is minimized on a shallow neural network using TensorFlow [50], an open-source Python library for machine learning. In particular, the neural network accepts the preprocessed gamma spectra (2048×1) as inputs, has a single hidden layer (512 nodes), and outputs a target vector containing the relative contributions of each radioisotope (30×1). When exploring different models to minimize the loss function, we also tested other deep neural network architectures with additional hidden layers but found that a simple shallow neural network performed the best. The loss is minimized using the built-in Adam optimizer in TensorFlow with an initial learning rate of 0.001. Before training, the training dataset is split into training and validation data (80/20), and during training the overall semi-supervised loss, supervised loss, unsupervised loss, and mean absolute error (MAE) are tracked on both the training and validation datasets. Each model is trained for 50 epochs with a batch size of 100 and a dropout of 0.05 applied to the hidden layer. An overview of learning framework, including training data generation and reconstruction error is shown in Figure 6.1.

In the following experiments, the performance of models training on the proposed supervised loss is tested for both ID and OOD data. In particular, the effect of two key parameters on model performance are studied: (1) the unsupervised loss function (either PNLL, GNLL, or SSE), and (2) the priority trade-off between supervised and unsupervised loss (β).

Model performance is considered both in terms of how the model performs for the LPE task on ID samples, measured with the MAE, and how the model performs for the OOD detection task, measured with the F1 score.

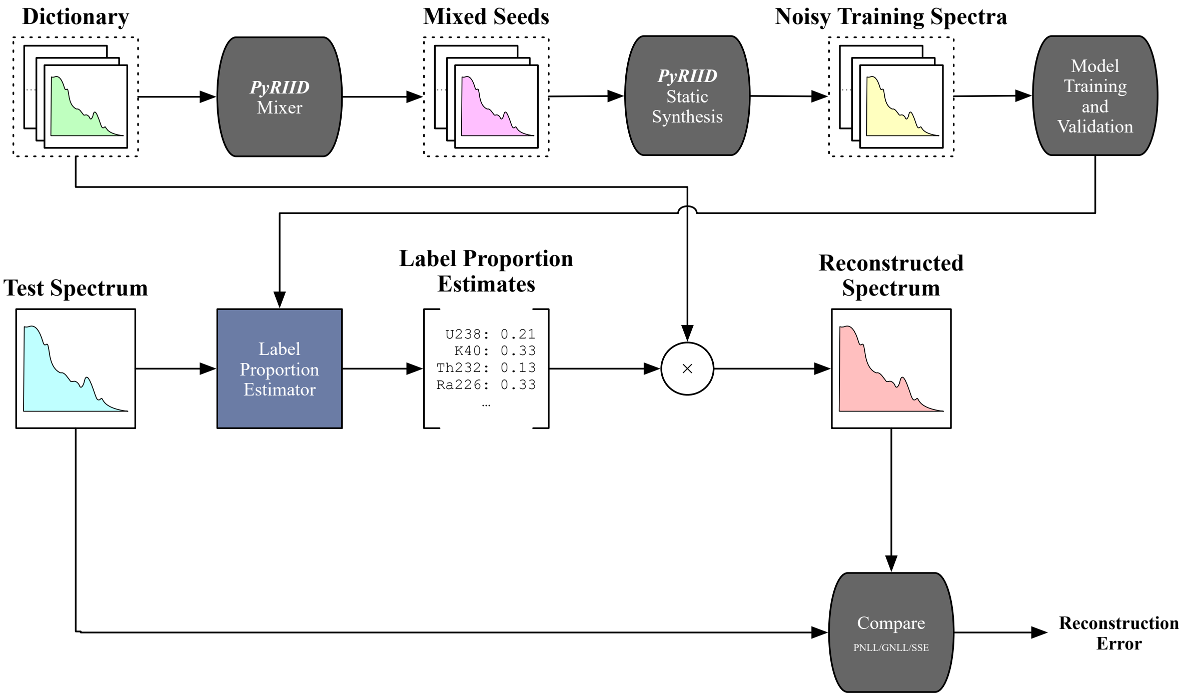


Figure 6.1: overview of model learning framework including training and testing

6.2 Effect of β on Model Performance

The results shown in the following tables and figures are obtained from applying the model to an unseen ID test dataset which was randomly generated in the same way as the training data. In particular the test dataset consists of 10^5 random samples (10k mixtures with 100 random samples/mixture varying SNR).

Setting the value of β to be non-zero gives priority to the unsupervised loss, and consequently affects the performance of the model in terms of both objectives as well as changing the scale of the expected loss values. Throughout this section, the role of β, when paired with various unsupervised losses (PNLL, GNLL, and SSE), is studied. For each of the unsupervised losses, models were trained on the ID training dataset with the following β values: 0, 5e-8, 1e-7, 5e-7, 1e-6, 5e-6, 1e-5, 5e-5, 1e-4, and 5e-4. Thus 10 models were trained using each unsupervised loss term, giving a total of 30 trained models. The training curves for each model can be found in Appendix C.

Each model was then tested on the ID test set. In Figure 6.2, the effect of β on ID test data is observed by measuring both the average MAE and the average reconstruction error on the test set. As expected, increasing β, and by extension the priority of the unsupervised learning objective, results in increasingly better reconstructions. However, this also results in a lower MAE for the LPE task.

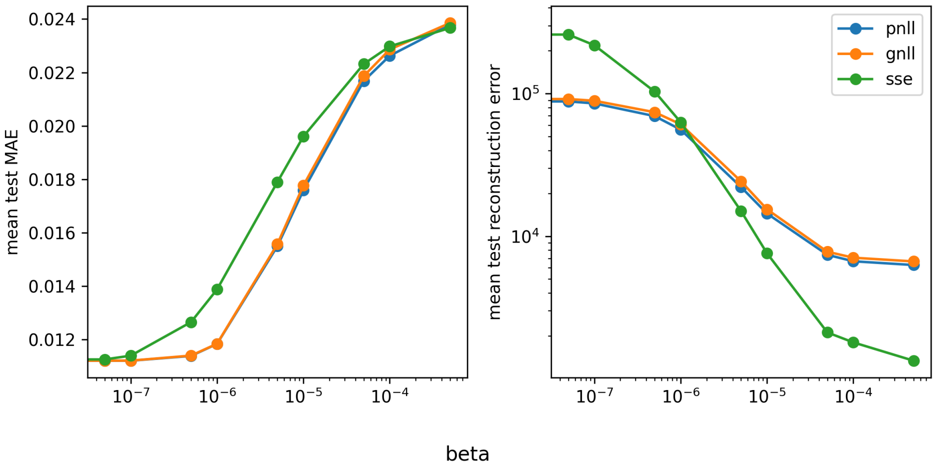


Figure 6.2: average test MAE and average test reconstruction error for models trained with different unsupervised losses (PNLL, GNLL, SSE) on a range of betas

6.3 OOD Detection Results

6.3.1 Testing the Effect of Anomaly Contribution on Reconstruction Error

To test the effectiveness of this method on OOD data, an OOD test dataset was created by adding an anomalous source spectrum that was not present in training to the ID test dataset, in this case ^{57}Co, with various proportions ranging from 0 to 1.0.

In Figure 6.3, the reconstruction errors for models trained with the different unsupervised losses are shown as the anomaly contribution in the OOD data increases. The ID test data was used to create the reconstruction errors with anomaly contribution equal to 0.0. This demonstrates that increasing β will significantly reduce the reconstruction error for ID test samples, while OOD test samples will be less affected by a change in β. This can also be shown by plotting the distributions of the reconstruction errors for ID test samples, which is shown in Figure 6.4.

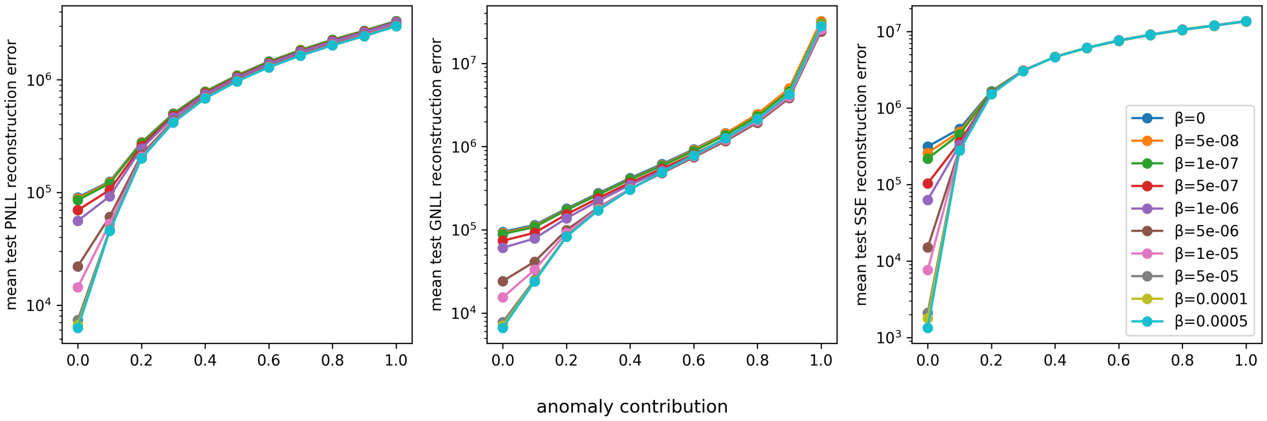


Figure 6.3: these plots show the average test reconstruction error when training the PNLL, GNLL, and SSE unsupervised losses, respectively, depending on the anomaly contribution level

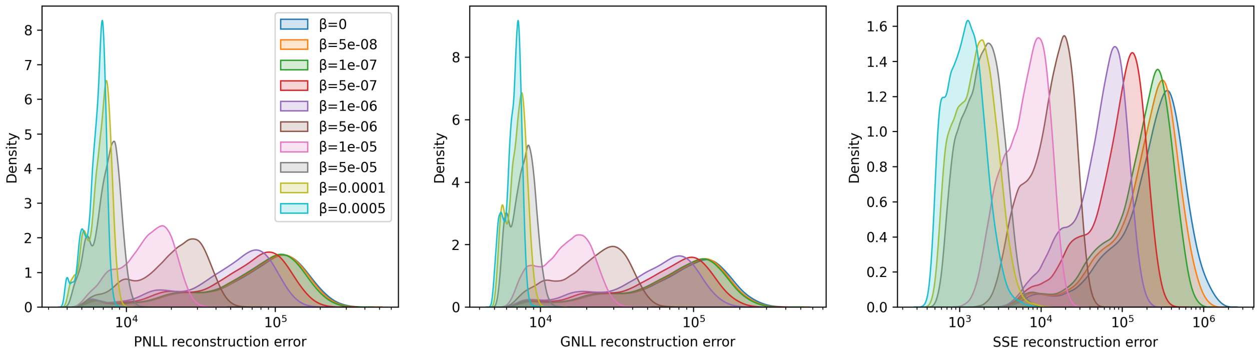


Figure 6.4: distribution of reconstruction errors on ID test dataset for models trained with the PNLL, GNLL, and SSE unsupervised losses, respectively, generated using seaborn kernel density estimation

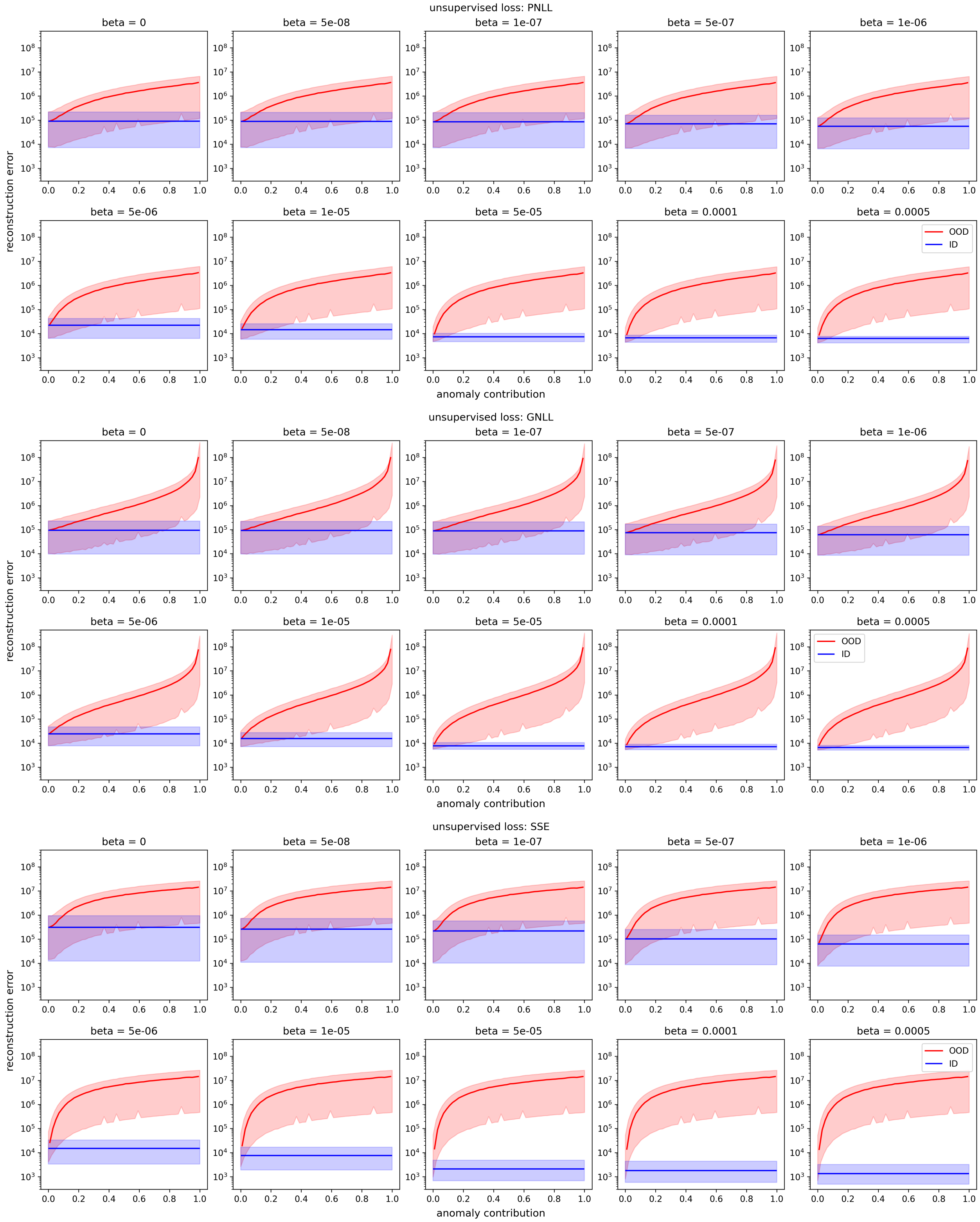


Figure 6.5: these plots show the mean reconstruction error as a function of anomaly contribution for each of the 30 trained models; the top, middle, and bottoms rows correspond to models trained with the PNLL, GNLL, and SSE unsupervised losses, respectively; the shaded bands represent the 95% confidence interval

For each of the 30 models trained, Figure 6.5 shows the average OOD reconstruction error as a function of anomaly contribution, along with the ID reconstruction error. The shaded bands on these plots represent the 95% empirical confidence interval of the data. These plots provide additional empirical evidence that reconstruction errors for ID test samples steadily decrease (in terms of both mean and variance) as β increases, while the OOD reconstruction errors are mostly unaffected.

6.3.2 Implementing and Testing an OOD Detector

The previous plots demonstrate the effect of anomaly contribution on the reconstruction error and ID MAE. To eliminate the effects of varying anomaly contribution when measuring OOD detection performance, the following experiments show how an OOD detector can be implemented along with its expected performance for a set anomaly contribution level of 10%. In particular, a new OOD dataset was created for this section by adding in the same anomalous source (^{57}Co) to the ID test data at a constant proportion. This way 10% of the counts in every test spectrum come from an OOD source.

Ideally, in the sense of maximizing the F1 score for anomaly detection, the threshold for a positive OOD detection would be set at the intersection between the distribution of ID reconstruction errors and the distribution of OOD reconstruction errors. For example, the distribution of ID and OOD reconstruction errors for two values of β are shown on the same axis in Figure 6.6. To have a "perfect" OOD detector there should no overlap between these distributions. The comparison of the distribution of reconstruction errors for the ID and OOD test samples for each model can be found in Appendix D.

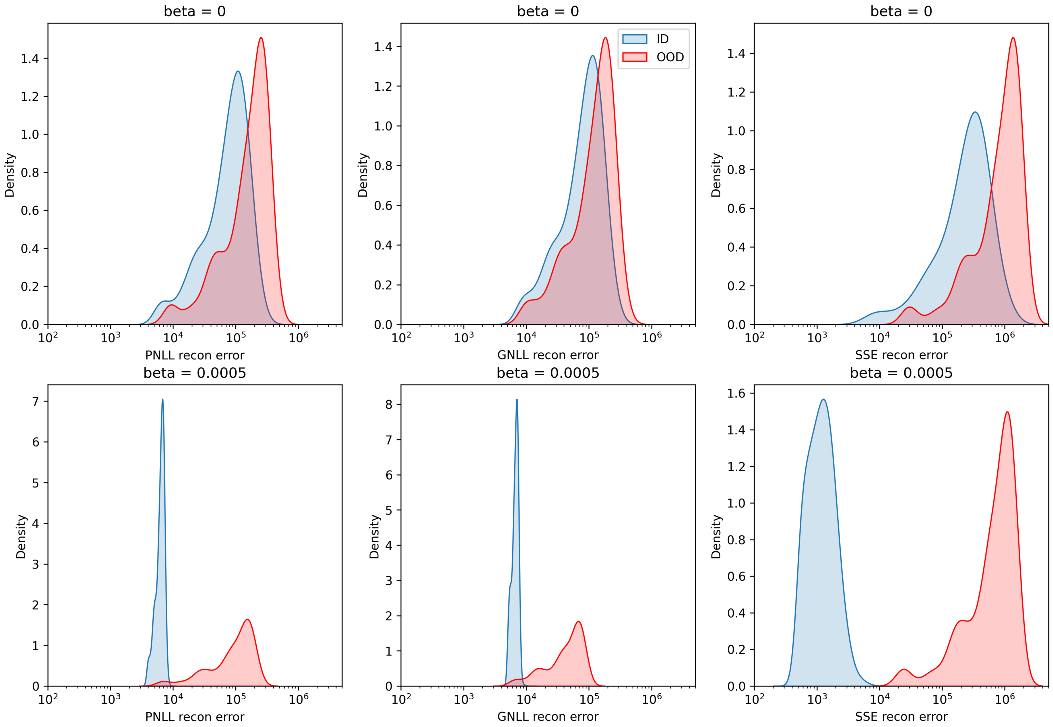


Figure 6.6: comparison of distribution of ID and OOD reconstruction errors for each unsupervised loss (PNLL, GNLL, and SSE, respectively) for beta equal to 0.0 and 5e-4; anomaly contribution at 10%

However, in practice, only the ID reconstruction error distribution can be known. The shape of the distribution of OOD reconstruction errors is unknown and can be affected by numerous factors which may not be known including the spectral shape of the OOD/anomalous source, the contribution of the anomaly, the overall SNR of the sample, and the value of β with which the model was trained.

A threshold for an OOD decision function must be chosen based only on the known distribution of ID reconstruction errors. This is realized by selecting a specific empirical tail density which will correspond to a reconstruction error value to be used as a threshold. For this OOD detector, the detection threshold was chosen to be set at the 99% quantile of the empirical distribution of ID reconstruction errors. An example of this is shown in Figure 6.7. For each model trained, the detection threshold is computed in this same way, and the plot for each model can be found in Appendix E.

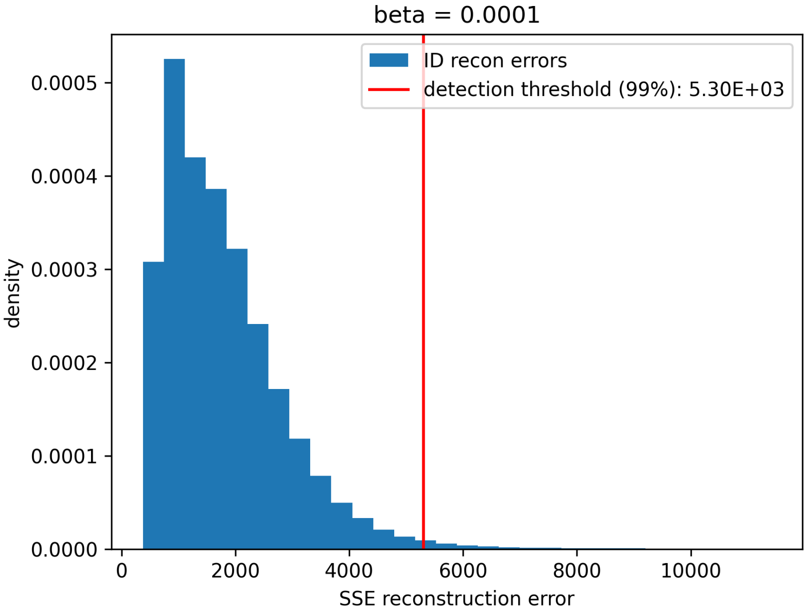


Figure 6.7: example of detection threshold setting for model trained on SSE unsupervised loss with $\beta = 0.0001$

A sample is then declared OOD if its reconstruction error is greater than the detection threshold (i.e., falls in the low-density region of ID reconstruction errors). Then the F1 score can be computed on the combined ID and OOD test dataset (with anomaly contribution held constant at 0.1). These results are given in Table 6.1, and displayed in Figure 6.8. These demonstrate that, as expected, increasing the priority of the unsupervised term results in a lower detection threshold and a higher F1 score for OOD detection. However, there is also a tradeoff to consider as a higher β results in poorer performance on ID data for the LPE task in terms of the MAE (see future work).

Beta	PNLL			GNLL			SSE		
	Detection threshold	F1 Score	ID MAE	Detection threshold	F1 Score	ID MAE	Detection threshold	F1 Score	ID MAE
0	2.474e5	0.873	0.011	2.601e5	0.815	0.011	1.166e6	0.886	0.011
5e-8	2.375e5	0.877	0.011	2.474e5	0.82	0.011	8.56e5	0.907	0.011
1e-7	2.282e5	0.88	0.011	2.401e5	0.822	0.011	6.475e5	0.922	0.011
5e-7	1.777e5	0.897	0.011	1.923e5	0.839	0.011	2.841e5	0.952	0.013
1e-6	1.371e5	0.911	0.012	1.53e5	0.857	0.012	1.675e5	0.966	0.014
5e-6	4.654e4	0.956	0.015	5.21e4	0.927	0.016	3.699e4	0.983	0.018
1e-5	2.747e4	0.969	0.018	2.974e4	0.951	0.018	1.882e4	0.987	0.02
5e-5	1.07e4	0.984	0.022	1.1e4	0.978	0.022	5.59e3	0.991	0.022
1e-4	8.85e3	0.986	0.023	9.298e3	0.981	0.023	5.302e3	0.991	0.023
5e-4	7.951e3	0.987	0.024	8.2e3	0.984	0.024	4.067e3	0.991	0.024

Table 6.1: table of detection thresholds, F1 scores, and ID MAEs for each of the trained models; the F1 score is computed on the combined ID test dataset and OOD test dataset with an anomaly contribution of 10%

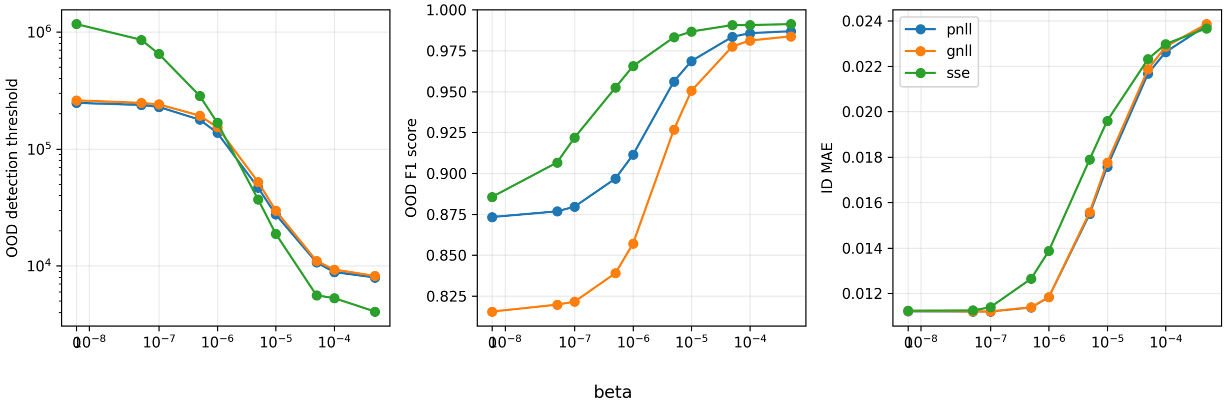


Figure 6.8: table of detection thresholds, F1 scores, and ID MAEs for each of the trained models as a function of beta with an anomaly contribution of 10%

Chapter 7 Conclusion

In conclusion, this work demonstrates a novel approach for identifying and predicting the proportions of 30 radioisotopes present in gamma spectra using a custom loss function, minimized on a neural network, which prioritizes explainable estimates. The semi-supervised model is tested on synthetic fission spectra based on a measured spectrum, which contains various compositions of 30 sources. The results show that the proposed model can estimate radioisotope proportions that are simultaneously suitable for use in an OOD detector. Accomplishing these tasks with a single neural network model allows for quick LPE and OOD detection without having to train separate models. By paring the supervised cross-entropy loss with the SSE unsupervised loss with a beta of 5e-7, the model reached a test MAE 0.013 for the LPE task, while achieving a 0.952 F1 score for the OOD detection tasks with an anomaly contribution of just 10%. Note that while an MAE of 0.013 seems low, it represents a cumulative error of ~39% across all 30 radioisotopes. While this error is higher than desired for applications requiring high accuracy, the model shows the ability to somewhat learn the task while still conveying the challenge of explaining these types of spectra in terms of 30 distinct radioisotopes. Perhaps incorporating more subject-matter expertise into the creation of the dictionary can help constrain assumptions and improve LPE. Alternatively, and in conjunction with SME consultation, one might find an alternative learning architecture that can utilize contextual information to narrow the problem space. Lastly, to easily share the developed technology with others, the model architecture and learning framework have been added to the PyRIID package.

Chapter 8 Future Work

From studying this problem space and considering the results of semi-supervised model, several weaknesses of this model were observed. This section discusses some avenues for future improvement that could be used to address these issues and make the model more robust and intuitive.

1. In the GNLL unsupervised loss term, the Gaussian approximation of a Poisson breaks down for low count channels. This issue could be sidestepped by applying some variance stabilizing transformation to the spectra, such as a Freeman-Tukey transform or an Anscombe transform, that would cause the spectra to converge to a Gaussian distribution more quickly and would make the loss function bounded (avoiding the divide by zero issue). Alternatively, one could just use the PNLL to handle low count samples.
2. In test cases there could be different detector and environmental parameters than in training, which could result in a distributional shift or change in the spectral shape of the target sources. The proposed model, which relies on knowledge of a dictionary of the expected sources, would be susceptible to shifts like these if not detected as OOD. To make the model more robust in different test settings we could initially adjust the dictionary columns to account for these changes by utilizing a calibration measurement in the test environment.
3. This work focuses on OOD detection in the case where OOD inputs are the result of a semantic distribution shift (i.e., some of the counts of the spectrum are from a novel/anomalous source not seen in training data). In practice, however, this model

would likely also consider a sample as OOD due to a covariate distribution shift (i.e., a change in the detector response), or other reasons. A direction for future work would be to improve on the OOD detection method so it can also differentiate which type of OOD input it has detected whether from the presence of a novel source, or from a change in the sensory conditions.

4. Currently, the supervised and unsupervised loss terms are combined as a weighted summation in the objective function, but as noted previously, they exist on completely different scales (differing by several orders of magnitude). As a result, β must either be very small or very large depending on the unsupervised loss function used, which is not intuitive. Ideally, a $\beta = 0.5$ would indicate approximately equal priority given to the two objectives in the loss function. Moreover, the scale of the loss values for each term can also change at different rates throughout training, which changes the relative priority of the objectives throughout training. For example, the learning curves show that the model can quickly minimize the unsupervised loss function resulting in a decrease in priority relative to the supervised loss term. As the model improves on the supervised task, the unsupervised loss in some cases will begin to marginally increase. Malkiel et. al. [51] recently proposed a method for adaptively balancing multiple loss terms while training. Using a technique such as this could alleviate some of the negative side effects of combining different losses in the model.
5. When formulating the Poisson negative log-likelihood for the unsupervised learning objective, it is assumed that the energy counts in each gamma spectrum channel are measured independently. This allowed the PNLL unsupervised loss function to simplify nicely to use it as a convex learning objective. However, in practice, there may be some

residual effects between energy channels resulting from certain decay characteristics of the measured nuclei as well as certain detector characteristics. In the future, the covariance between the energy channel counts could be used to better model the likelihood of a sample given an LPE-based reconstruction.

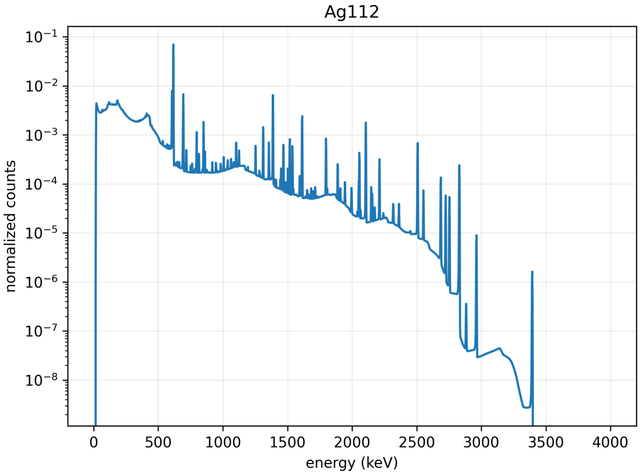
6. The unsupervised term in the learning objective compares the difference between the input signal and the model's reconstruction of the input signal. For this work, three different comparison metrics were tested. However, there are many other metrics that would be interesting to compare. For example, as the normalized spectra can be viewed as a probability distribution, it would be interesting to test the performance using KL divergence [44], Mahalanobis distance [52], or some other metric.
7. Recently, Blondel et. al. in [53] generalize the sparsemax loss in a continuous parametric family of loss functions called the Tsallis α-entropies, which are parameterized by a sparsity-controlling parameter α. For $\alpha = 1, 2, \infty$ the Tsallis entropy recovers the softmax, sparsemax, and argmax functions, respectively. Using the Tsallis α-entropies in the supervised portion of the loss function would give an additional hyperparameter which could be tuned based on the expected level of sparsity to yield better performance.
8. From these results the two problems being solved (LPE and OOD detection) are somewhat adversarial, at least in this problem space. Assigning too high a priority to the OOD detection task when training (i.e., setting a large β) negatively affects the performance of the model for LPE in terms of the mean absolute error. A direction for future work would be to learn an OOD detector without negatively affecting the LPE performance. One approach to this may be to decouple the tasks into two specialized

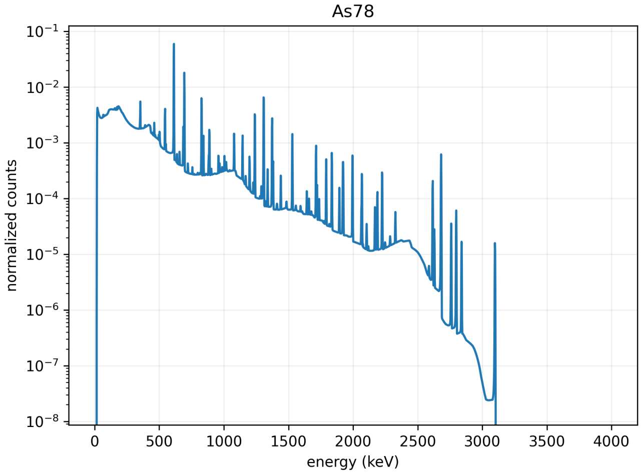
models, one which prioritizes LPE and the other which prioritizes OOD detection, and then use the OOD detection model as a gatekeeper for the LPE model.

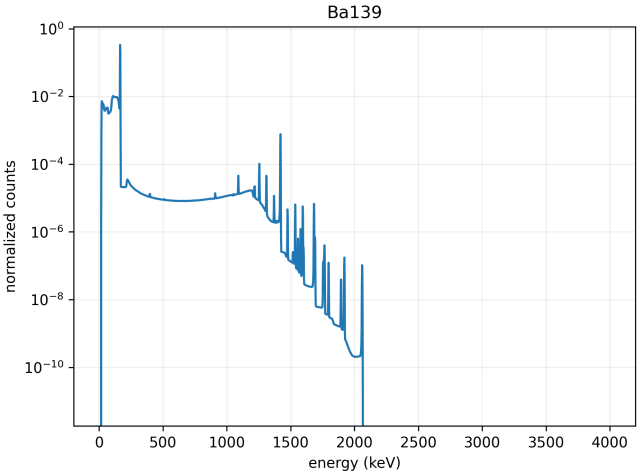
Appendices

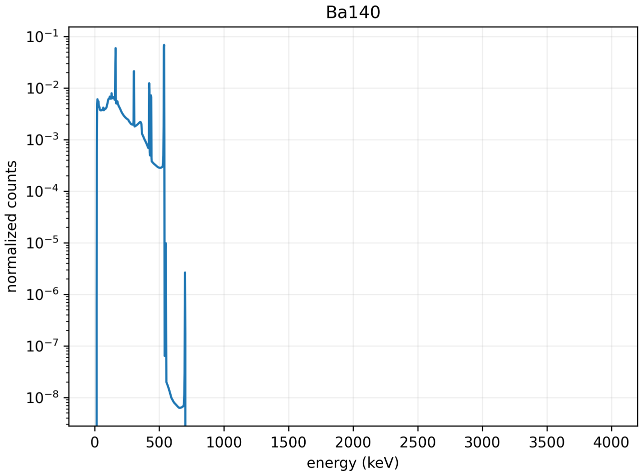
Appendix A Pure Spectral Signatures

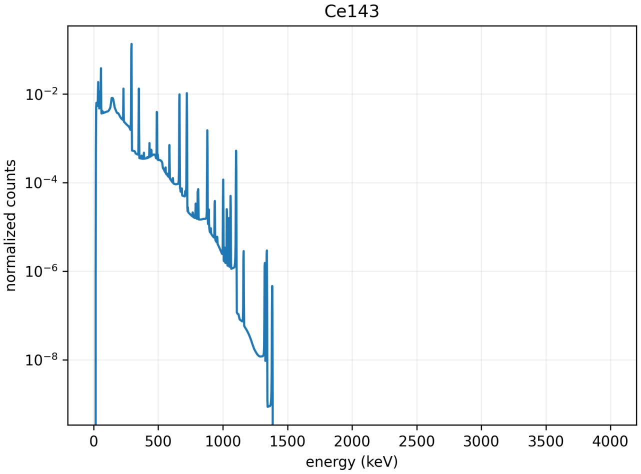
The spectral signature for each source used to generate the data of this work are shown below.

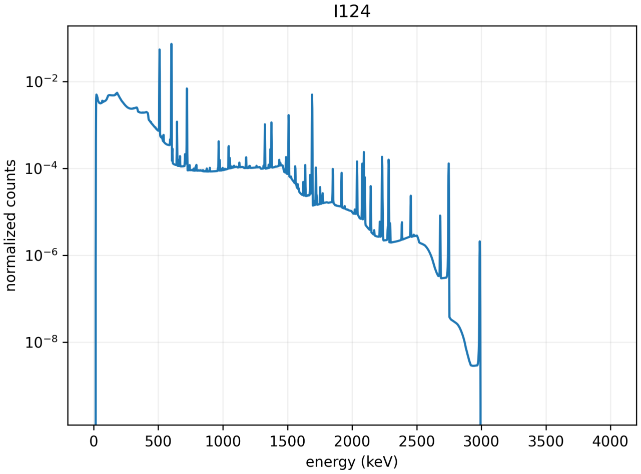


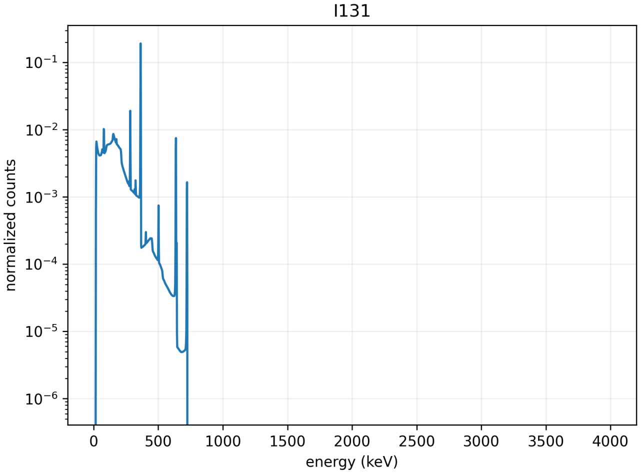


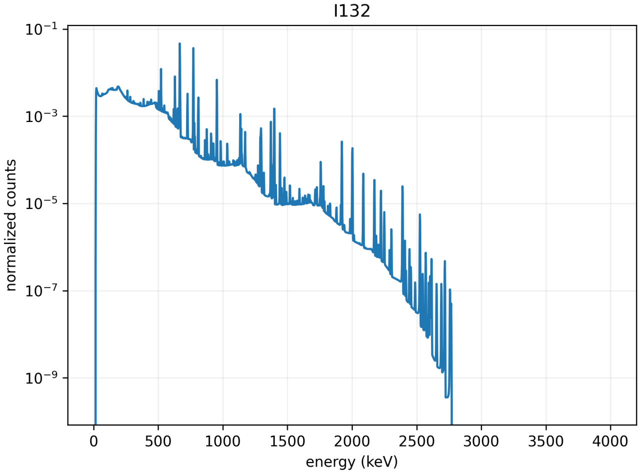


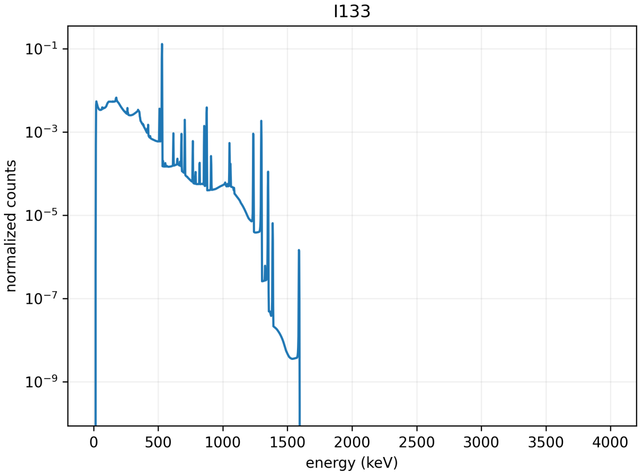


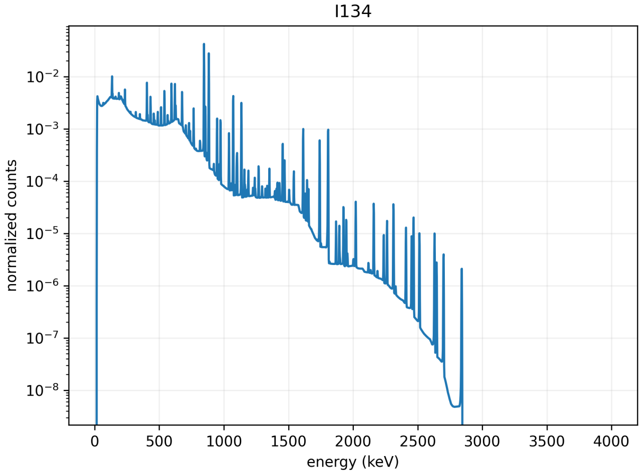

Ce143

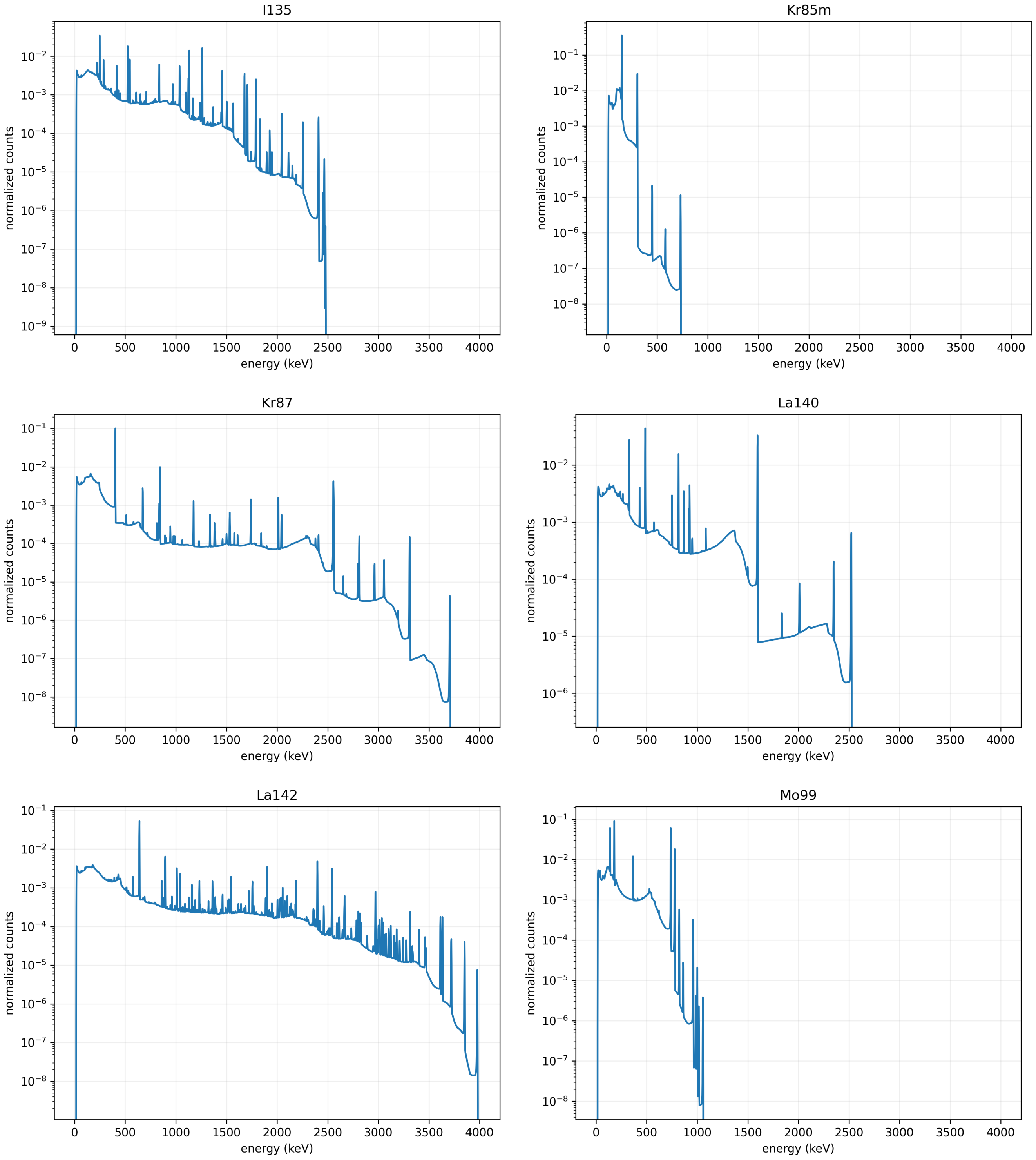

I124

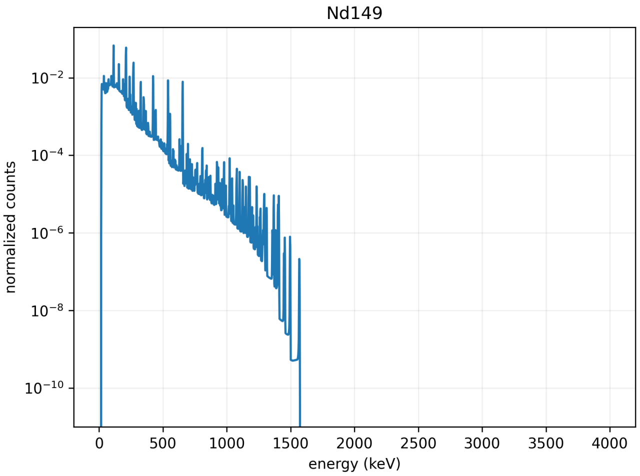

I131

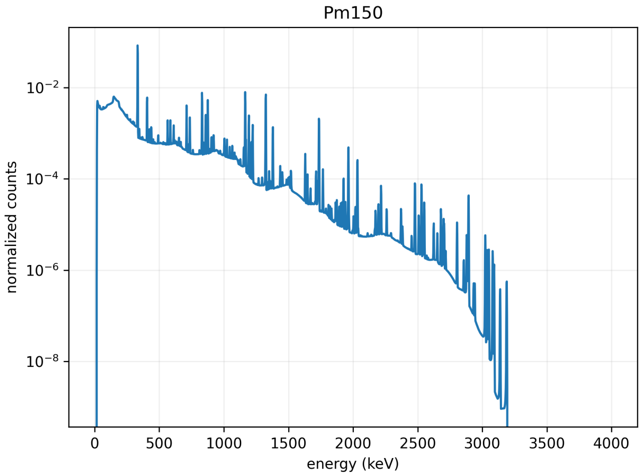

I132

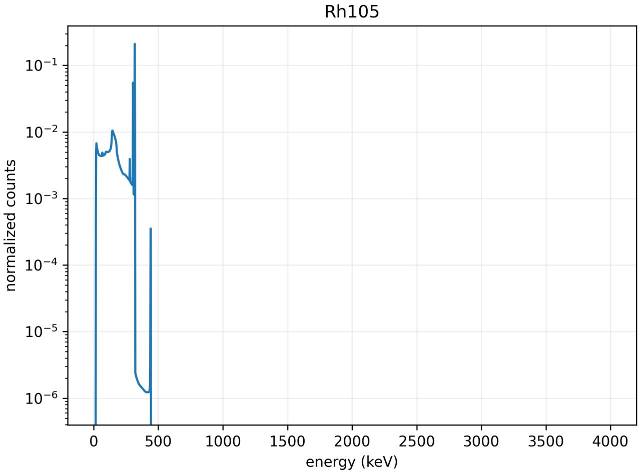

I133

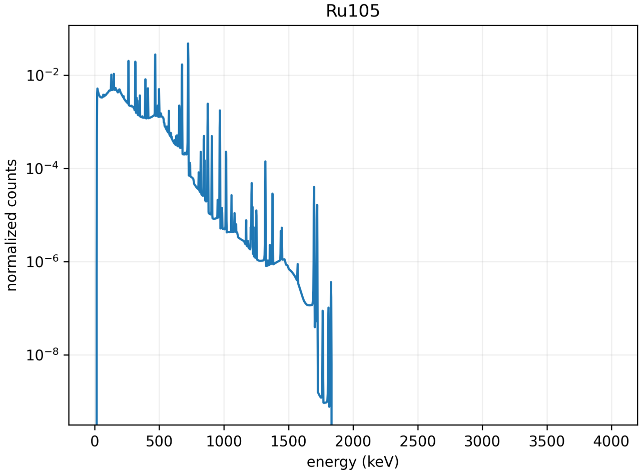

I134

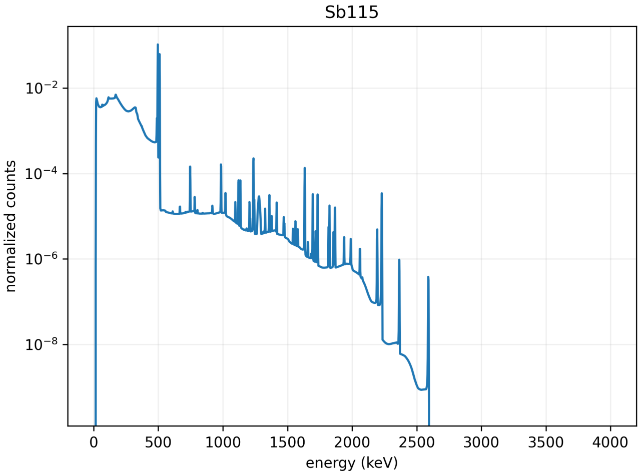

I135
Kr85m
Kr87
La140
La142
Mo99
normalized counts
energy (keV)
0
500
1000
1500
2000
2500
3000
3500
4000
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-9}

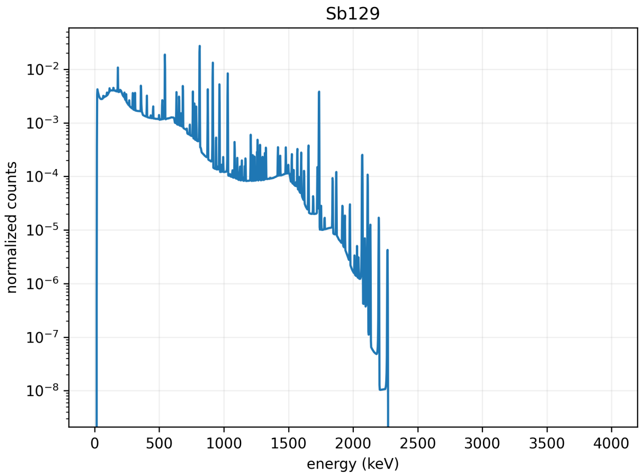

Nd149
normalized counts
10^−2
10^−4
10^−6
10^−8
10^−10
0
500
1000
1500
2000
2500
3000
3500
4000
energy (keV)

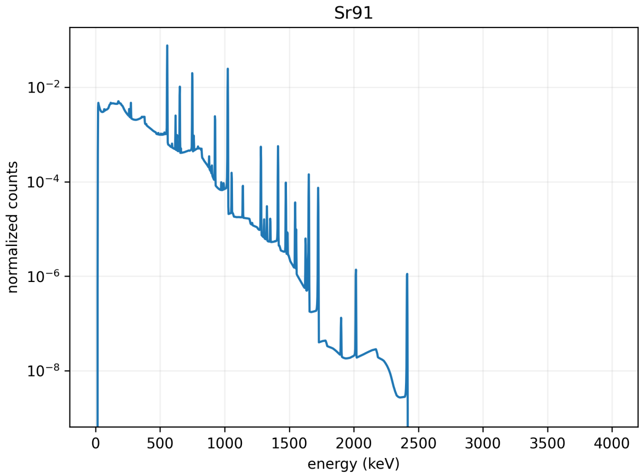

Pm150
normalized counts
10^−2
10^−4
10^−6
10^−8
0
500
1000
1500
2000
2500
3000
3500
4000
energy (keV)

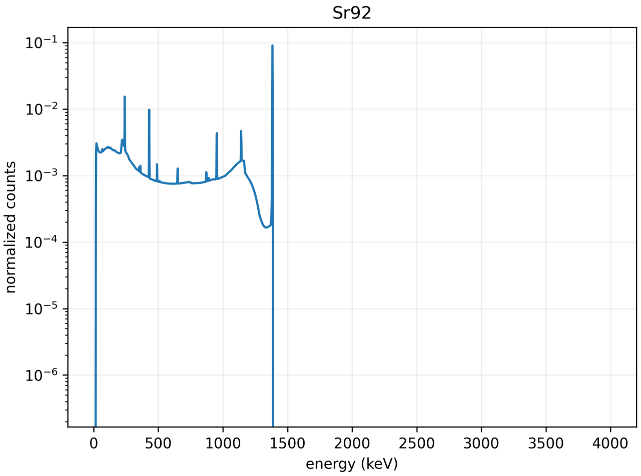

Rh105
normalized counts
10^−1
10^−2
10^−3
10^−4
10^−5
10^−6
0
500
1000
1500
2000
2500
3000
3500
4000
energy (keV)

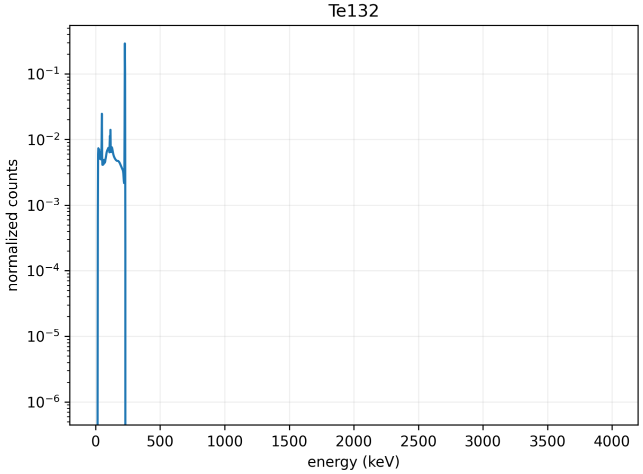

Ru105
normalized counts
10^−2
10^−4
10^−6
10^−8
0
500
1000
1500
2000
2500
3000
3500
4000
energy (keV)

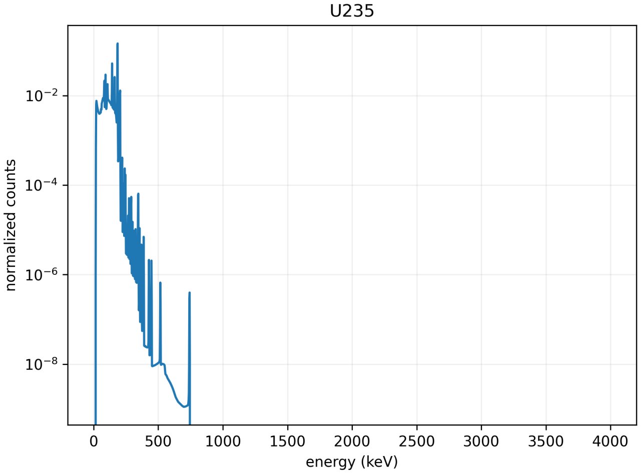

Sb115
normalized counts
10^−2
10^−4
10^−6
10^−8
0
500
1000
1500
2000
2500
3000
3500
4000
energy (keV)

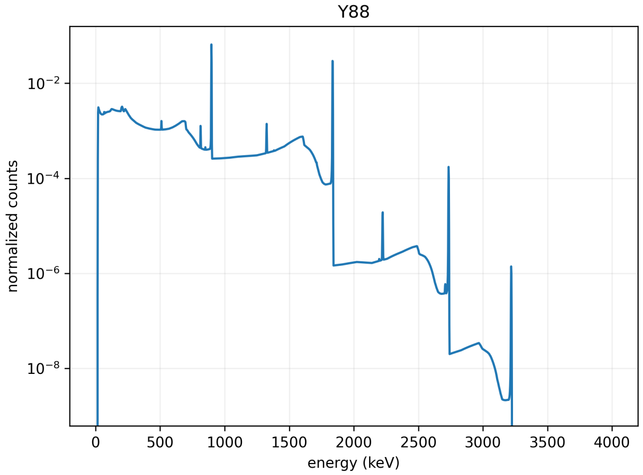

Sb129
normalized counts
10^−2
10^−3
10^−4
10^−5
10^−6
10^−7
10^−8
0
500
1000
1500
2000
2500
3000
3500
4000
energy (keV)

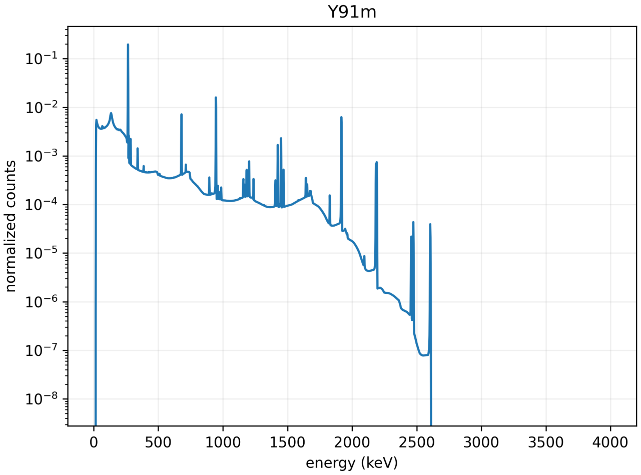

Sr91
normalized counts
energy (keV)
10^{-2}
10^{-4}
10^{-6}
10^{-8}
0
500
1000
1500
2000
2500
3000
3500
4000

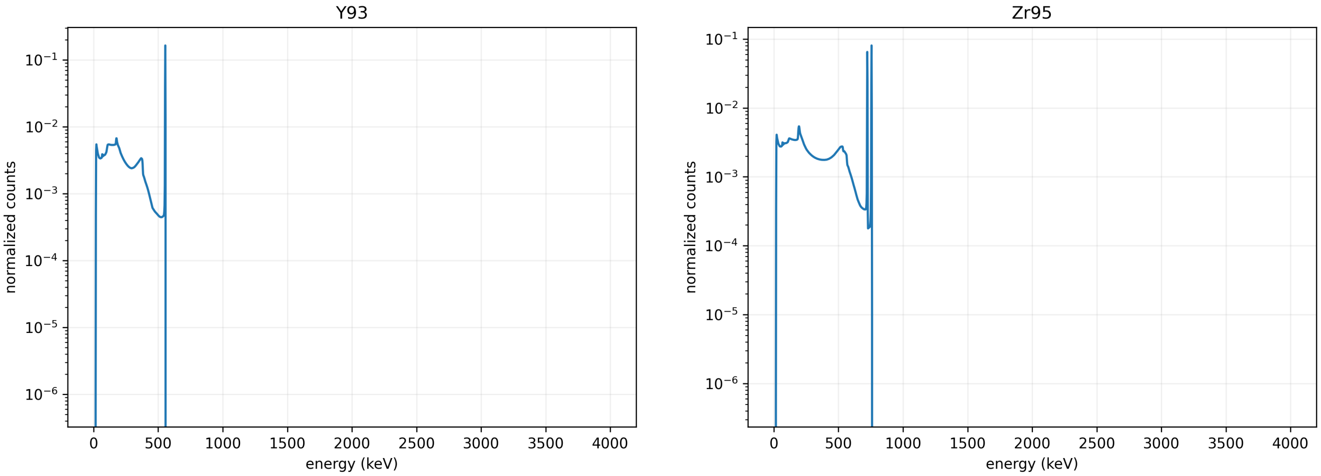

Sr92
normalized counts
energy (keV)
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
0
500
1000
1500
2000
2500
3000
3500
4000


Te132
normalized counts
energy (keV)
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
0
500
1000
1500
2000
2500
3000
3500
4000


U235
normalized counts
energy (keV)
10^{-2}
10^{-4}
10^{-6}
10^{-8}
0
500
1000
1500
2000
2500
3000
3500
4000

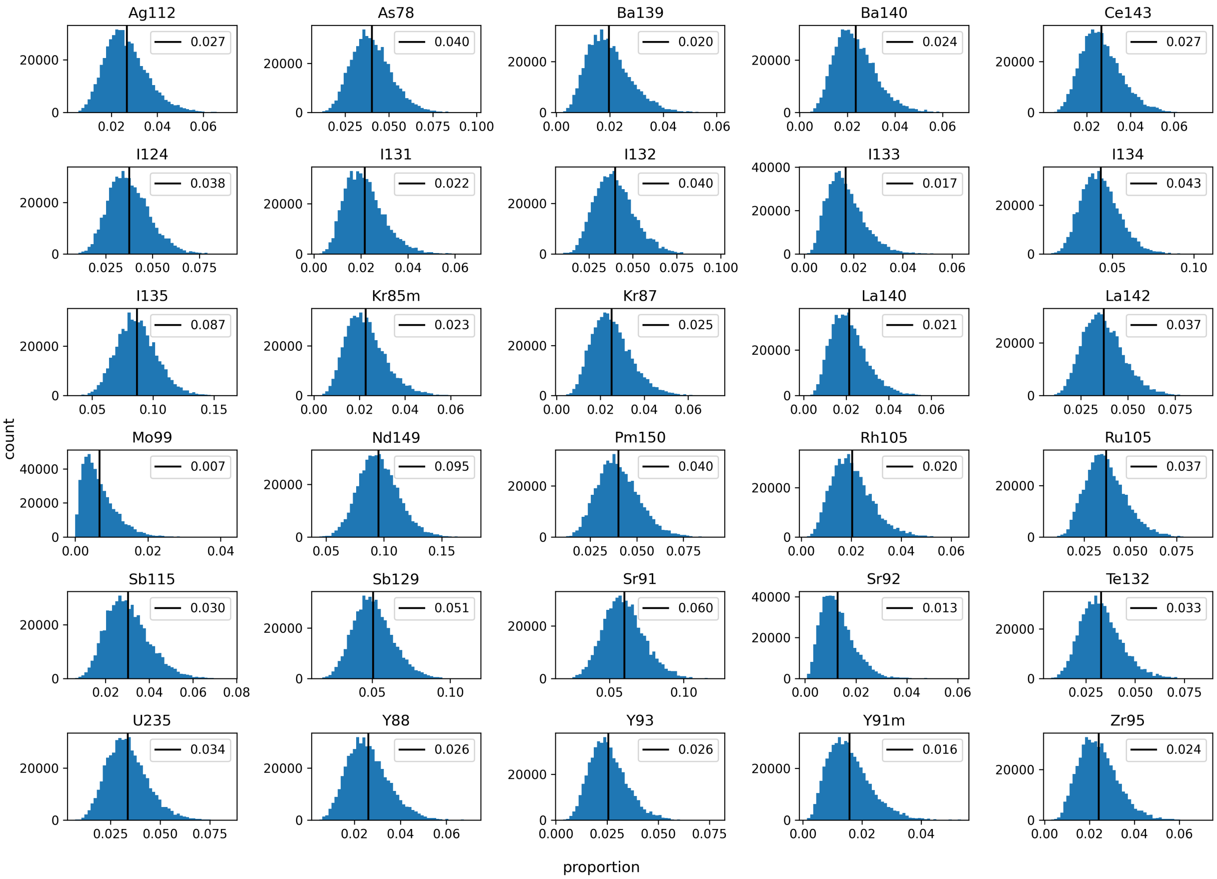

Y88
normalized counts
energy (keV)
10^{-2}
10^{-4}
10^{-6}
10^{-8}
0
500
1000
1500
2000
2500
3000
3500
4000


Y91m
normalized counts
energy (keV)
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
0
500
1000
1500
2000
2500
3000
3500
4000



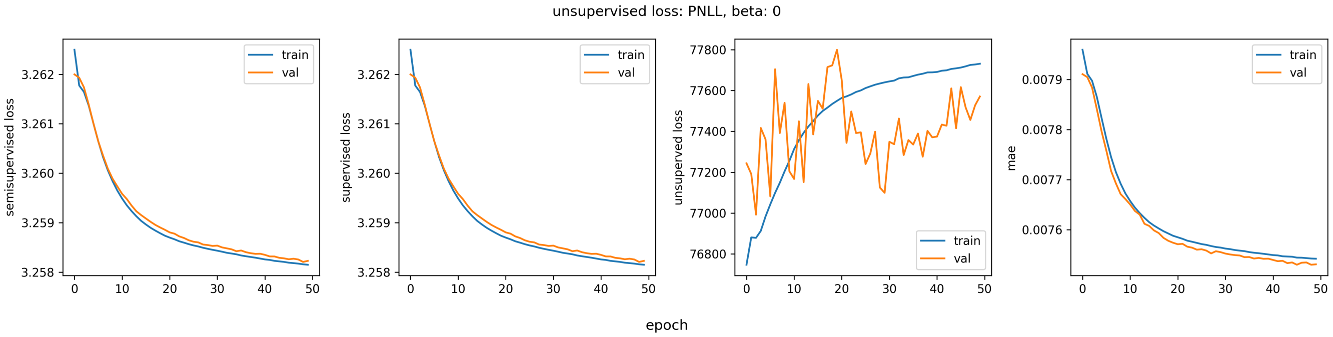
Appendix B Distribution of Mixture Proportions for Sources

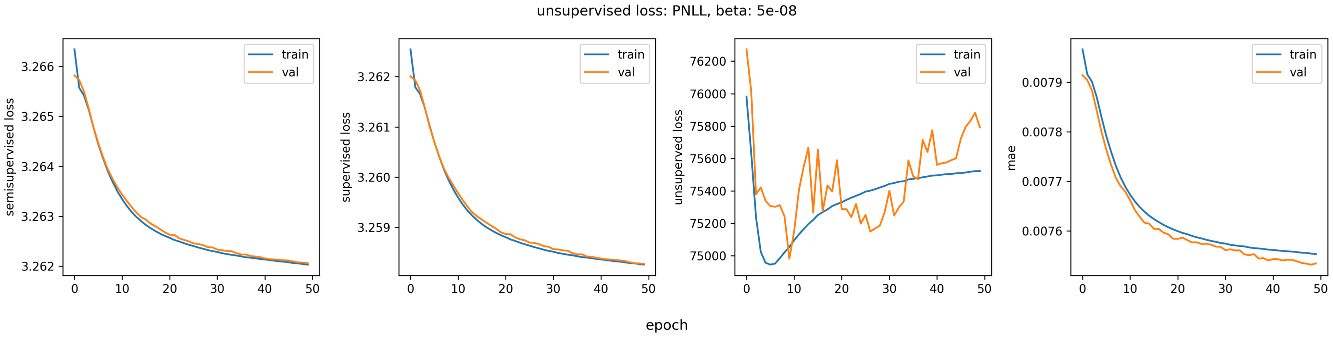
This appendix contains a histogram of the mixture proportions for each individual source in the training dataset.

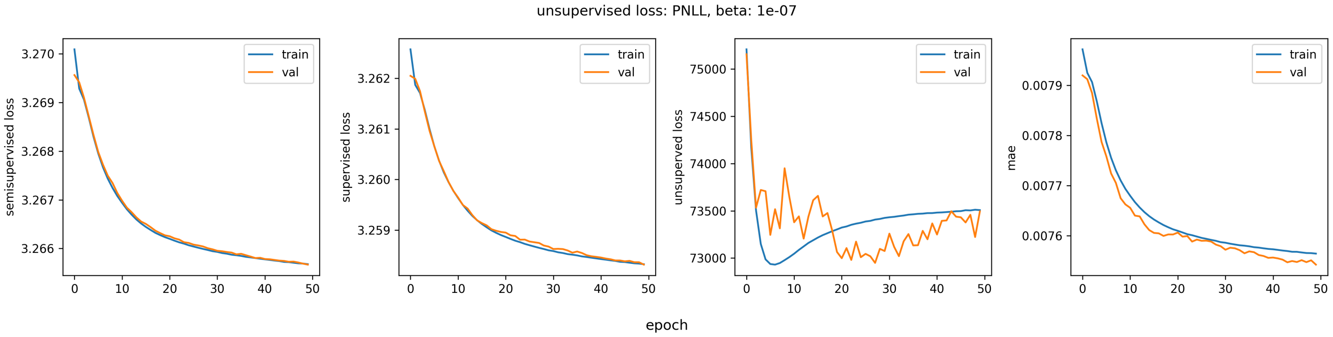


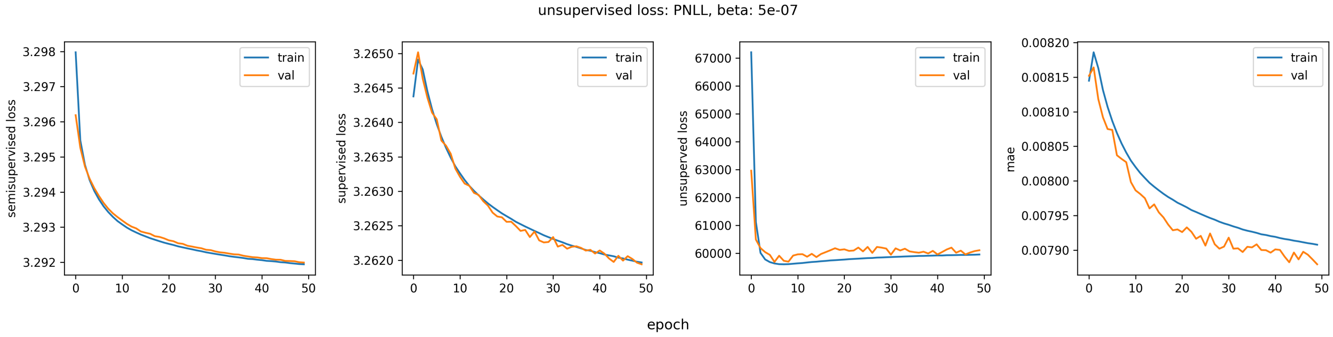
Appendix C Training Curves

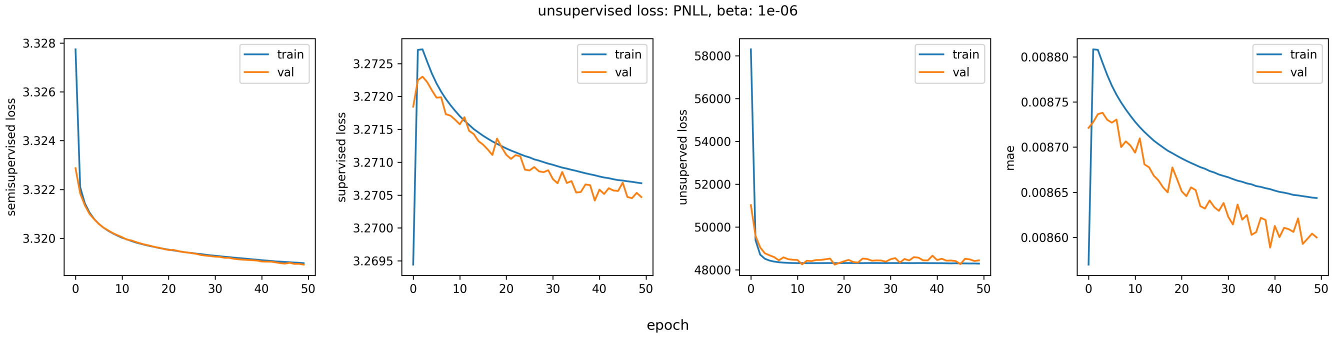
This appendix contains the training/validation curves for each of the 30 models trained and with various betas and unsupervised losses.

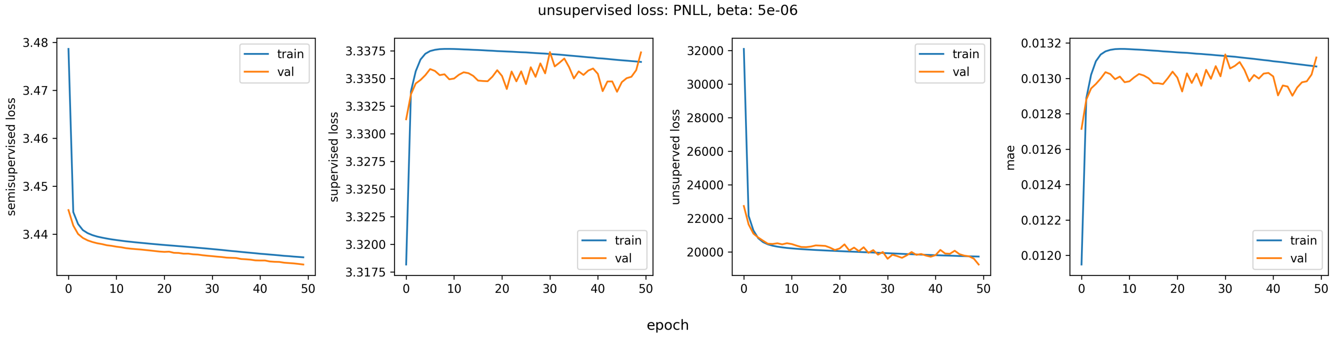


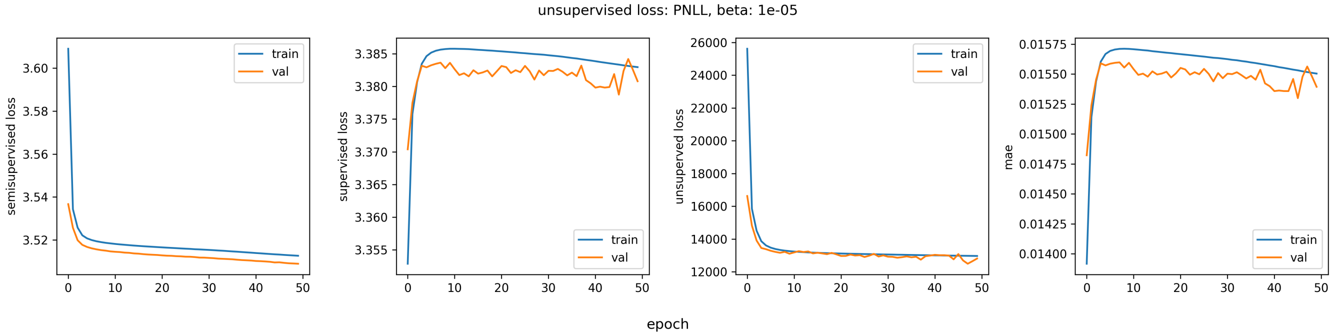


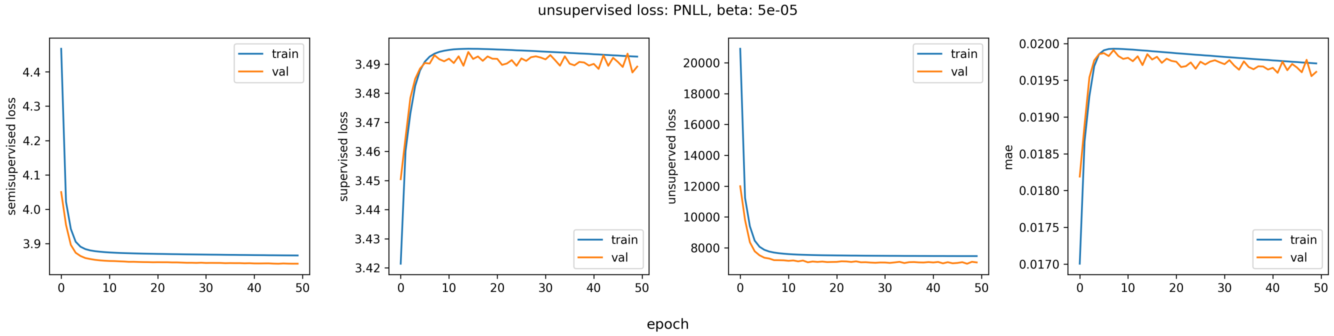


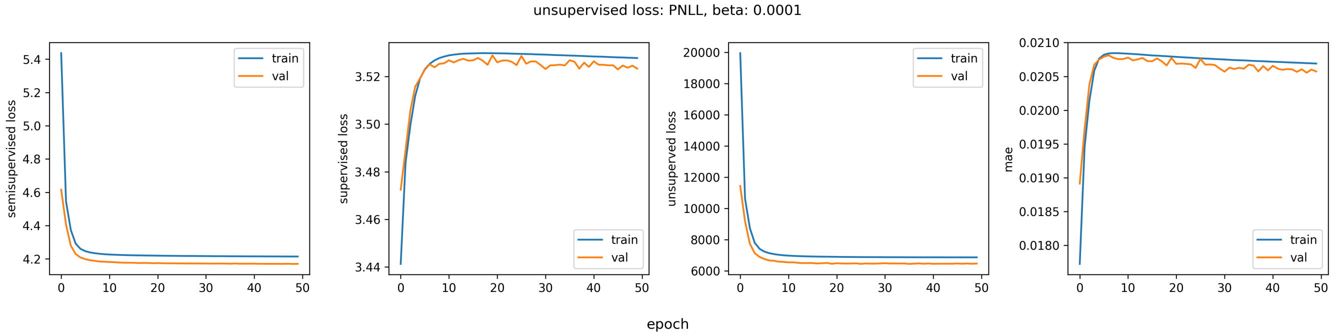


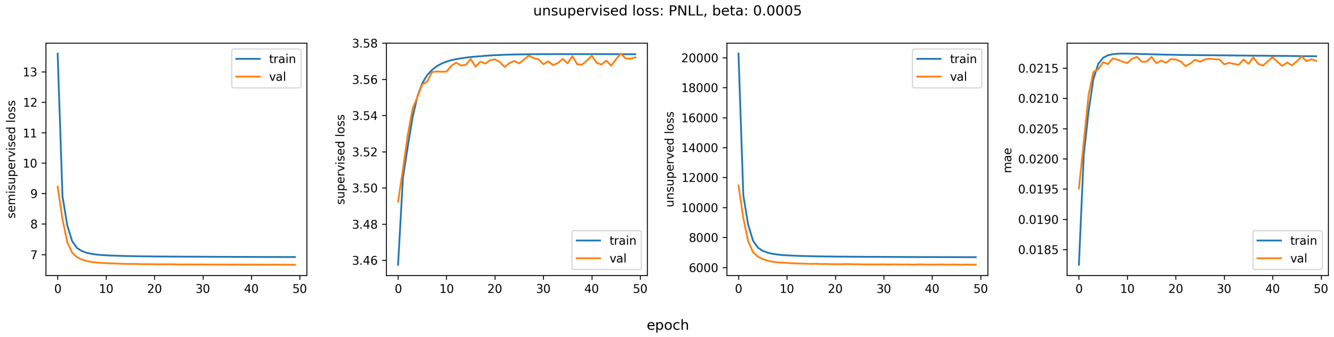

unsupervised loss: PNLL, beta: 1e-06
semisupervised loss
3.328
3.326
3.324
3.322
3.320
supervised loss
3.2725
3.2720
3.2715
3.2710
3.2705
3.2700
3.2695
unsuperved loss
58000
56000
54000
52000
50000
48000
mae
0.00880
0.00875
0.00870
0.00865
0.00860
0
10
20
30
40
50
train
val
epoch

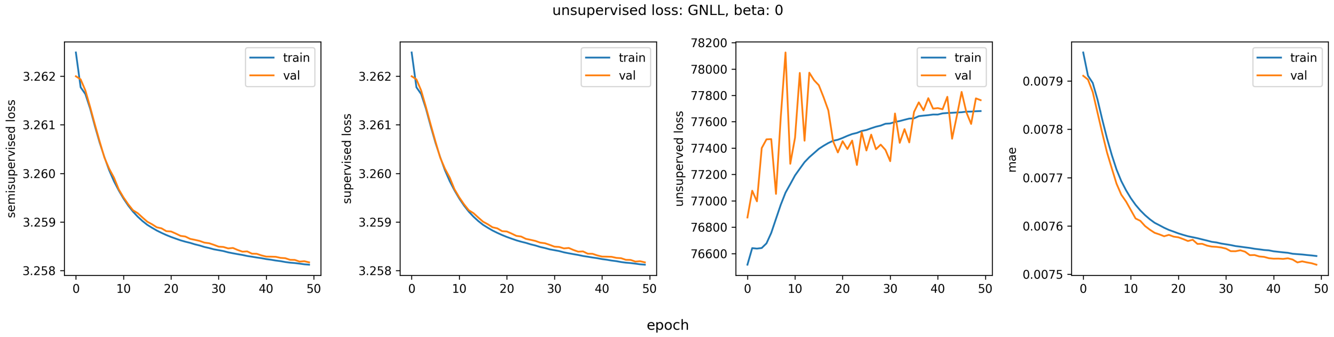

unsupervised loss: PNLL, beta: 5e-06
semisupervised loss
3.48
3.47
3.46
3.45
3.44
supervised loss
3.3375
3.3350
3.3325
3.3300
3.3275
3.3250
3.3225
3.3200
3.3175
unsuperved loss
32000
30000
28000
26000
24000
22000
20000
mae
0.0132
0.0130
0.0128
0.0126
0.0124
0.0122
0.0120
0
10
20
30
40
50
train
val
epoch

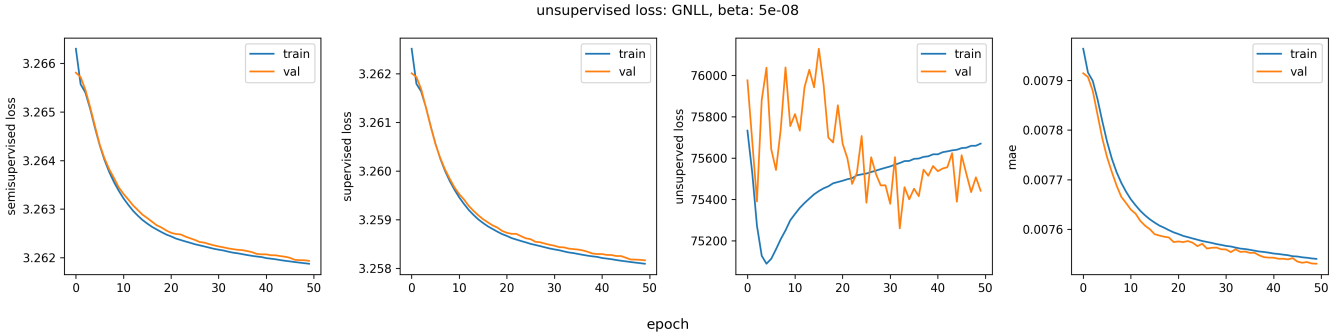

unsupervised loss: PNLL, beta: 1e-05
semisupervised loss
3.60
3.58
3.56
3.54
3.52
supervised loss
3.385
3.380
3.375
3.370
3.365
3.360
3.355
unsuperved loss
26000
24000
22000
20000
18000
16000
14000
12000
mae
0.01575
0.01550
0.01525
0.01500
0.01475
0.01450
0.01425
0.01400
0
10
20
30
40
50
train
val
epoch

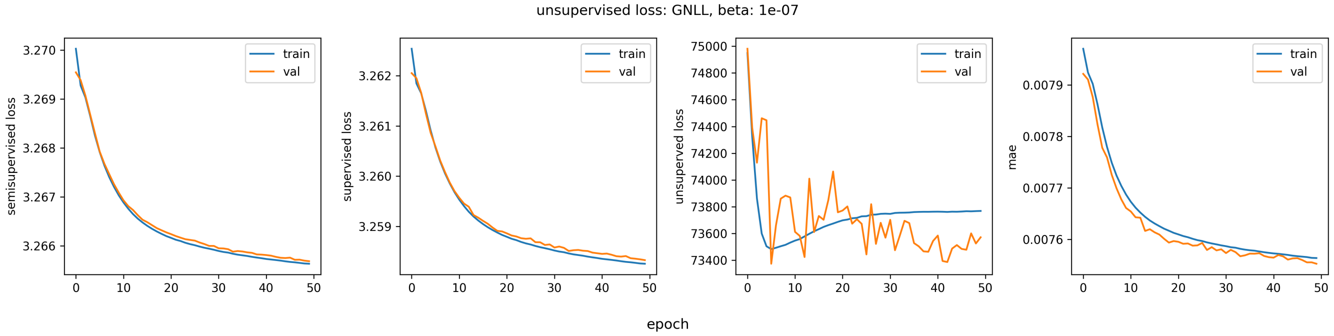

unsupervised loss: PNLL, beta: 5e-05
semisupervised loss
4.4
4.3
4.2
4.1
4.0
3.9
supervised loss
3.49
3.48
3.47
3.46
3.45
3.44
3.43
3.42
unsuperved loss
20000
18000
16000
14000
12000
10000
8000
mae
0.0200
0.0195
0.0190
0.0185
0.0180
0.0175
0.0170
0
10
20
30
40
50
train
val
epoch

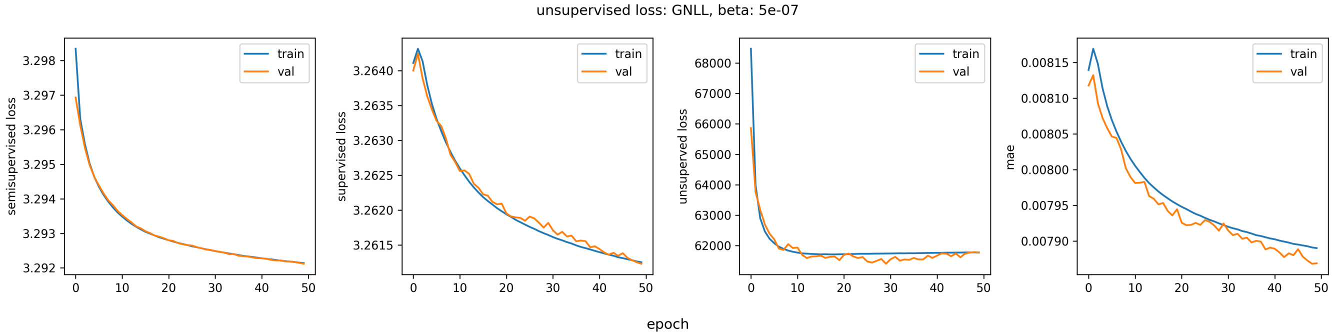

unsupervised loss: PNLL, beta: 0.0001
semisupervised loss
5.4
5.2
5.0
4.8
4.6
4.4
4.2
supervised loss
3.52
3.50
3.48
3.46
3.44
unsuperved loss
20000
18000
16000
14000
12000
10000
8000
6000
mae
0.0210
0.0205
0.0200
0.0195
0.0190
0.0185
0.0180
0
10
20
30
40
50
train
val
epoch

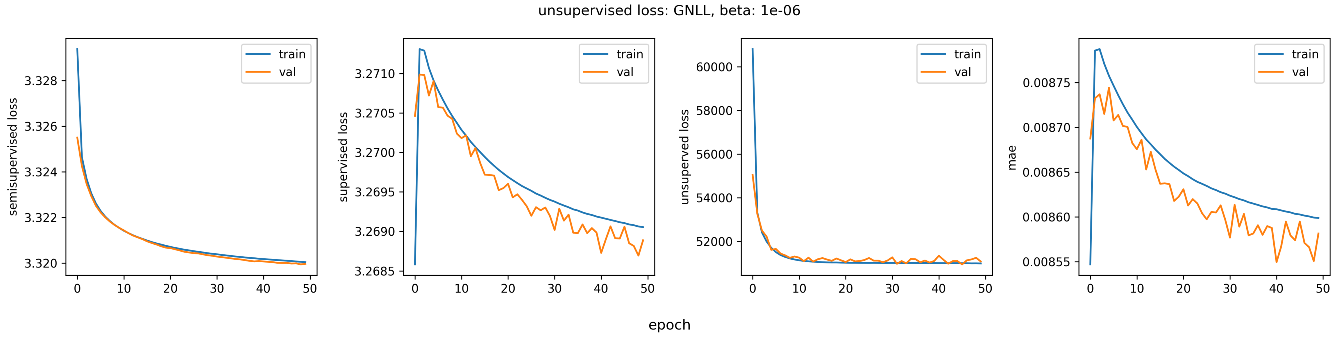

unsupervised loss: PNLL, beta: 0.0005
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch

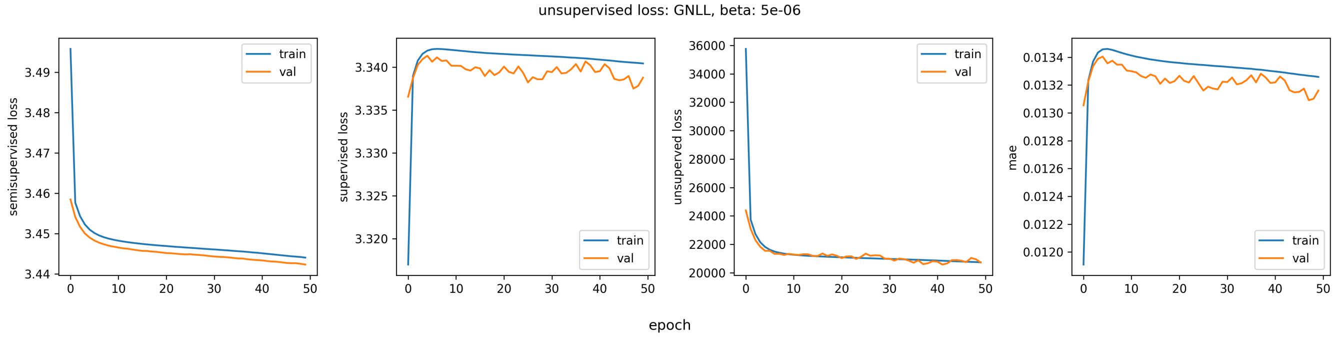

unsupervised loss: GNLL, beta: 0
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch

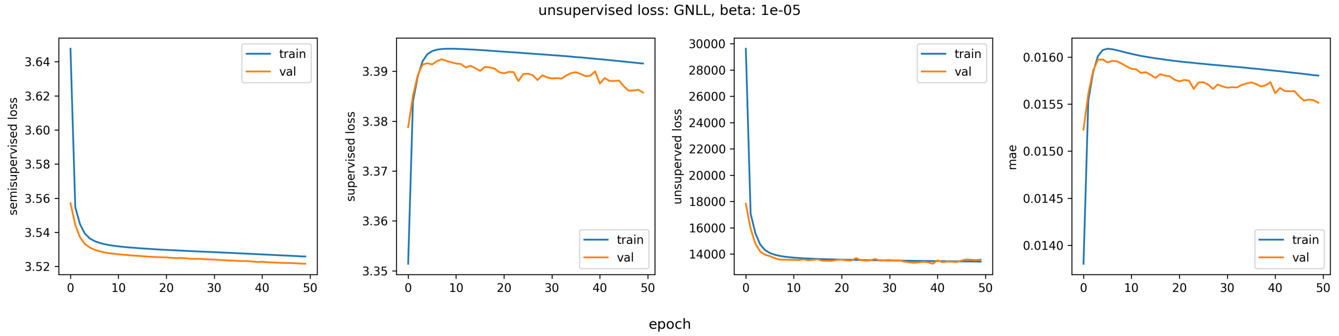

unsupervised loss: GNLL, beta: 5e-08
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch

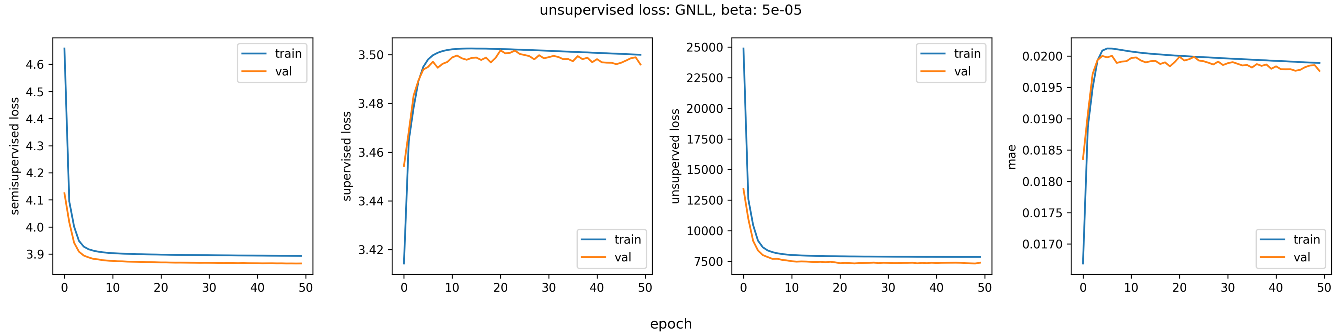

unsupervised loss: GNLL, beta: 1e-07
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch

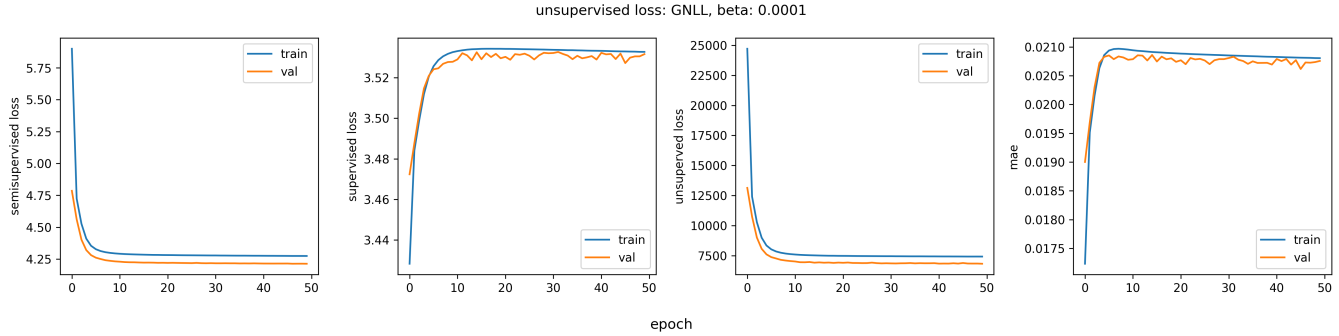

unsupervised loss: GNLL, beta: 5e-07
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch

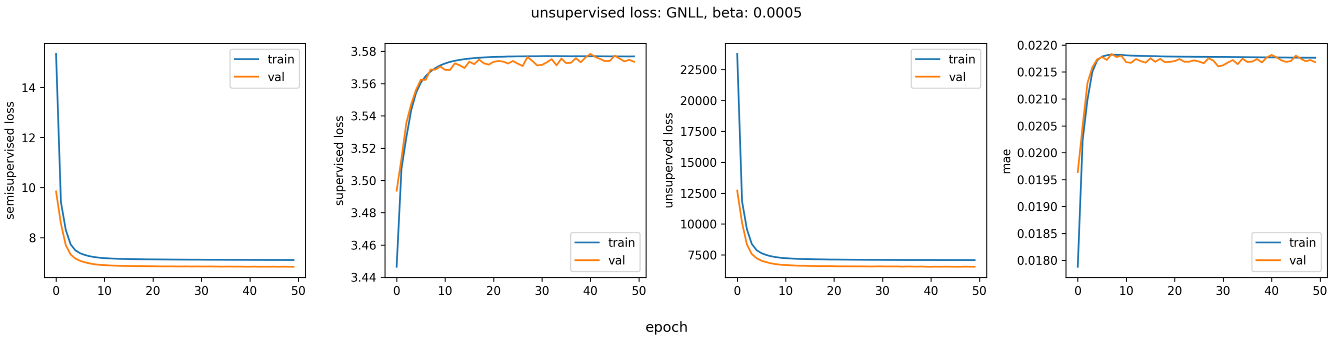

unsupervised loss: GNLL, beta: 1e-06
semisupervised loss
3.328
3.326
3.324
3.322
3.320
supervised loss
3.2710
3.2705
3.2700
3.2695
3.2690
3.2685
unsuperved loss
60000
58000
56000
54000
52000
mae
0.00875
0.00870
0.00865
0.00860
0.00855
train
val
0
10
20
30
40
50
epoch

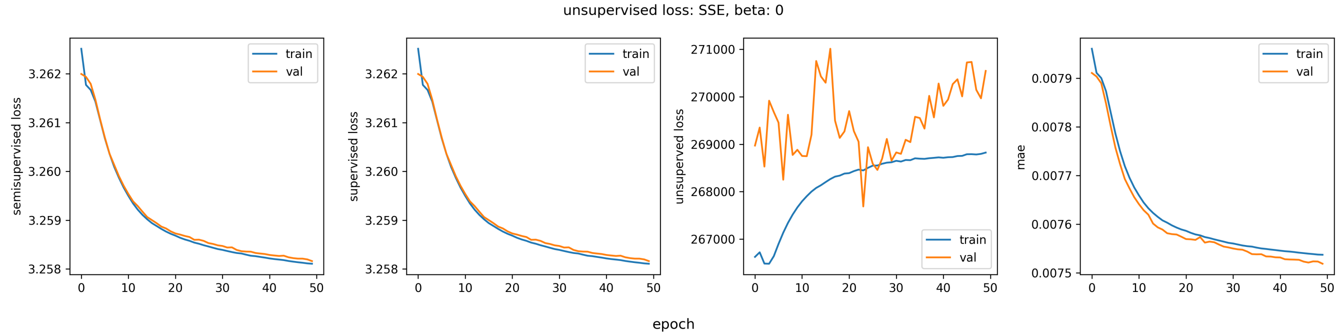

unsupervised loss: GNLL, beta: 5e-06
semisupervised loss
3.49
3.48
3.47
3.46
3.45
3.44
supervised loss
3.340
3.335
3.330
3.325
3.320
unsuperved loss
36000
34000
32000
30000
28000
26000
24000
22000
20000
mae
0.0134
0.0132
0.0130
0.0128
0.0126
0.0124
0.0122
0.0120
train
val
0
10
20
30
40
50
epoch

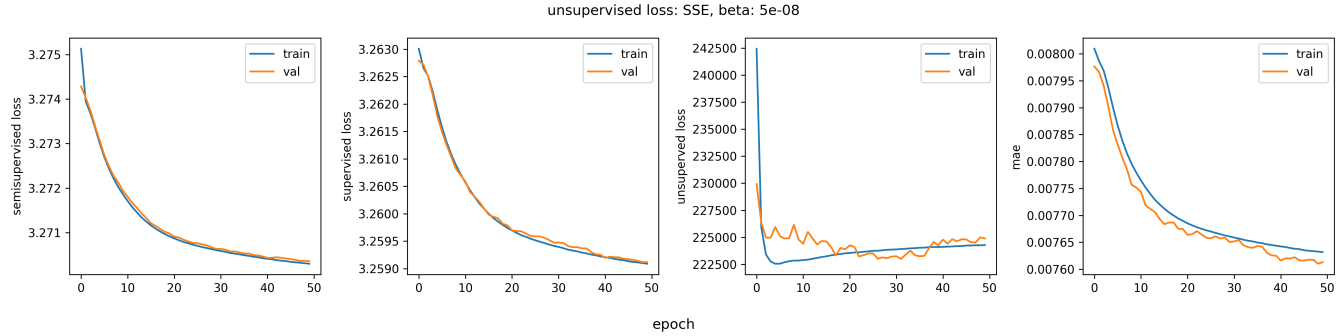

unsupervised loss: GNLL, beta: 1e-05
semisupervised loss
3.64
3.62
3.60
3.58
3.56
3.54
3.52
supervised loss
3.39
3.38
3.37
3.36
3.35
unsuperved loss
30000
28000
26000
24000
22000
20000
18000
16000
14000
mae
0.0160
0.0155
0.0150
0.0145
0.0140
train
val
0
10
20
30
40
50
epoch

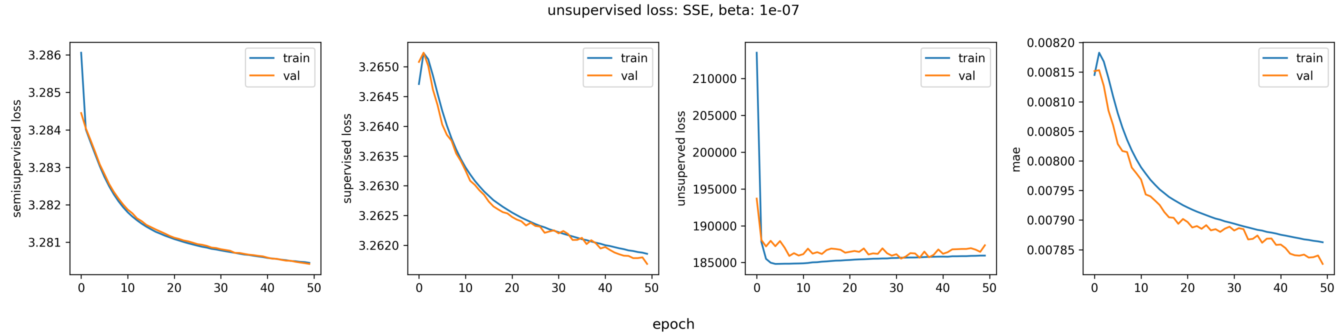

unsupervised loss: GNLL, beta: 5e-05
semisupervised loss
4.6
4.5
4.4
4.3
4.2
4.1
4.0
3.9
supervised loss
3.50
3.48
3.46
3.44
3.42
unsuperved loss
25000
22500
20000
17500
15000
12500
10000
7500
mae
0.0200
0.0195
0.0190
0.0185
0.0180
0.0175
0.0170
train
val
0
10
20
30
40
50
epoch

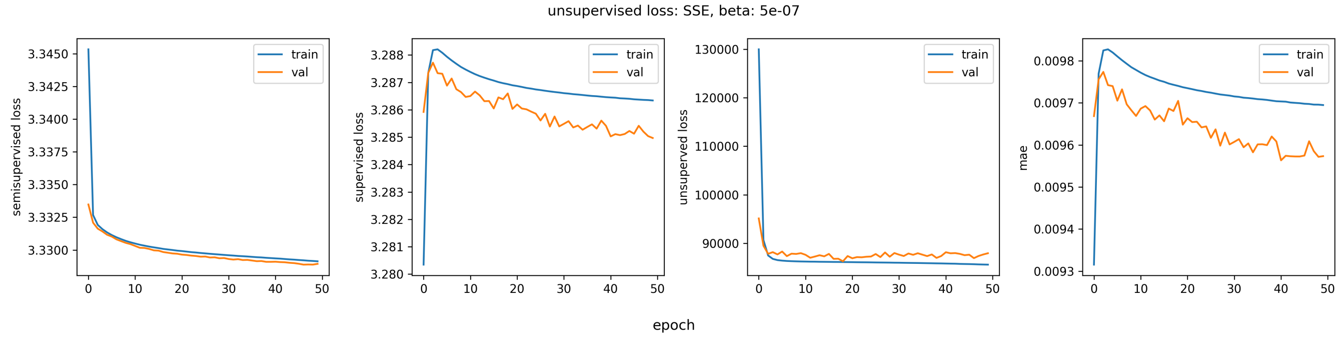

unsupervised loss: GNLL, beta: 0.0001
semisupervised loss
5.75
5.50
5.25
5.00
4.75
4.50
4.25
supervised loss
3.52
3.50
3.48
3.46
3.44
unsuperved loss
25000
22500
20000
17500
15000
12500
10000
7500
mae
0.0210
0.0205
0.0200
0.0195
0.0190
0.0185
0.0180
0.0175
train
val
0
10
20
30
40
50
epoch

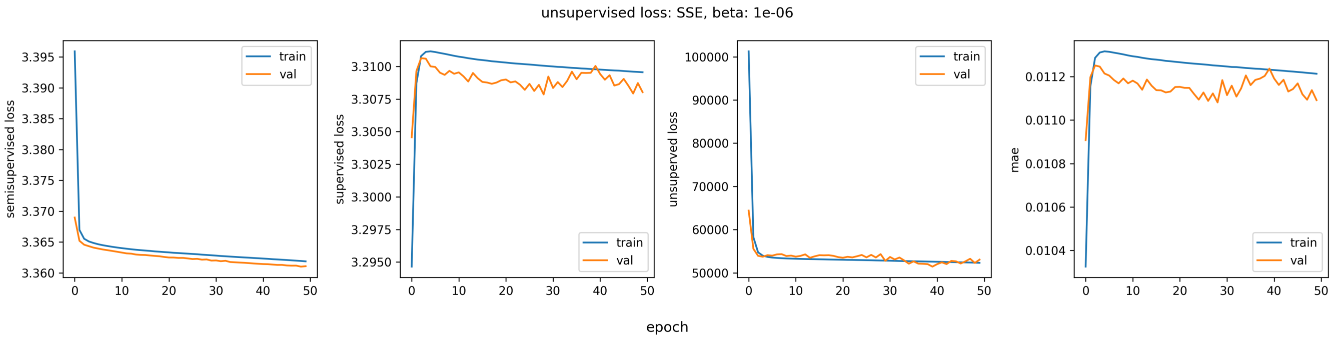

unsupervised loss: GNLL, beta: 0.0005
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch
0
10
20
30
40
50
14
12
10
8
3.58
3.56
3.54
3.52
3.50
3.48
3.46
3.44
22500
20000
17500
15000
12500
10000
7500
0.0220
0.0215
0.0210
0.0205
0.0200
0.0195
0.0190
0.0185
0.0180

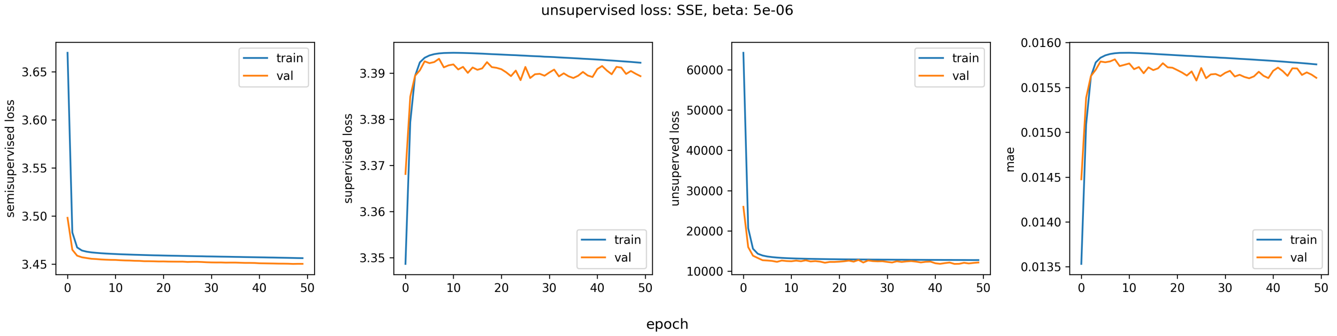

unsupervised loss: SSE, beta: 0
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch
0
10
20
30
40
50
3.262
3.261
3.260
3.259
3.258
271000
270000
269000
268000
267000
0.0079
0.0078
0.0077
0.0076
0.0075

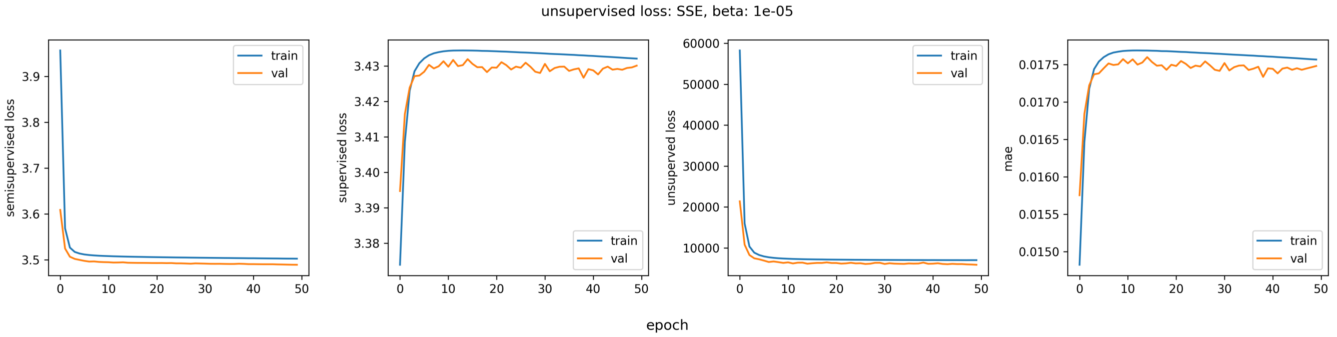

unsupervised loss: SSE, beta: 5e-08
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch
0
10
20
30
40
50
3.275
3.274
3.273
3.272
3.271
3.2630
3.2625
3.2620
3.2615
3.2610
3.2605
3.2600
3.2595
3.2590
242500
240000
237500
235000
232500
230000
227500
225000
222500
0.00800
0.00795
0.00790
0.00785
0.00780
0.00775
0.00770
0.00765
0.00760

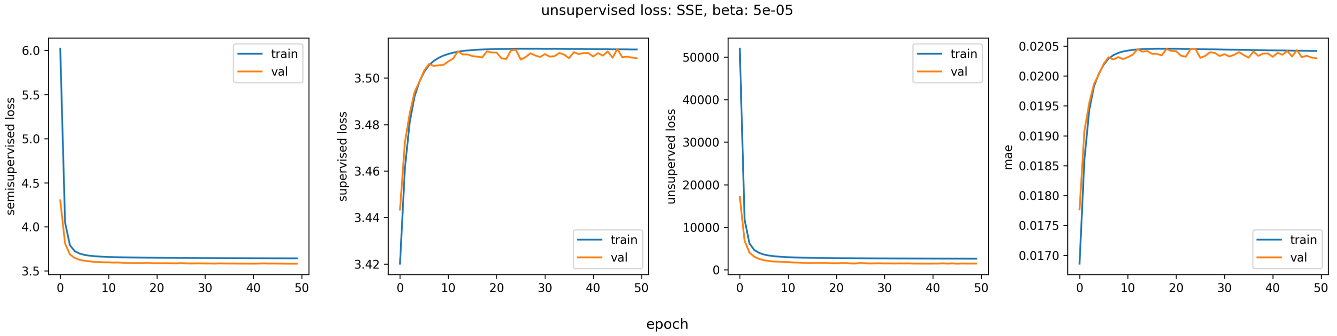

unsupervised loss: SSE, beta: 1e-07
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch
0
10
20
30
40
50
3.286
3.285
3.284
3.283
3.282
3.281
3.2650
3.2645
3.2640
3.2635
3.2630
3.2625
3.2620
210000
205000
200000
195000
190000
185000
0.00820
0.00815
0.00810
0.00805
0.00800
0.00795
0.00790
0.00785

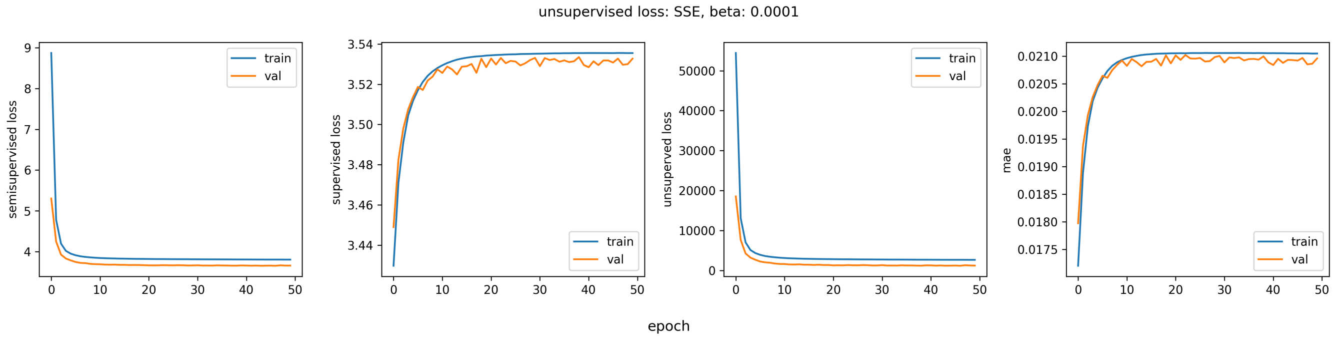

unsupervised loss: SSE, beta: 5e-07
semisupervised loss
supervised loss
unsuperved loss
mae
train
val
epoch
0
10
20
30
40
50
3.3450
3.3425
3.3400
3.3375
3.3350
3.3325
3.3300
3.288
3.287
3.286
3.285
3.284
3.283
3.282
3.281
3.280
130000
120000
110000
100000
90000
0.0098
0.0097
0.0096
0.0095
0.0094
0.0093

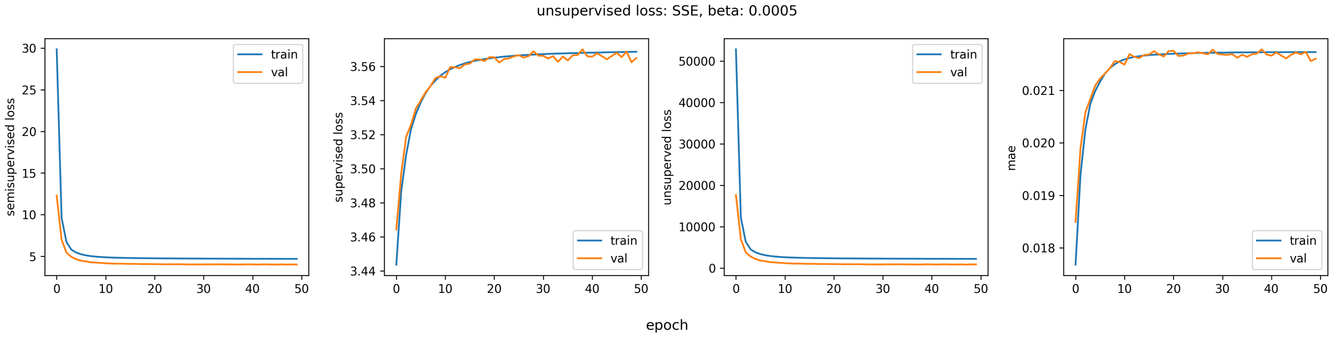

unsupervised loss: SSE, beta: 1e-06
semisupervised loss
3.395
3.390
3.385
3.380
3.375
3.370
3.365
3.360
supervised loss
3.3100
3.3075
3.3050
3.3025
3.3000
3.2975
3.2950
unsupervised loss
100000
90000
80000
70000
60000
50000
mae
0.0112
0.0110
0.0108
0.0106
0.0104
train
val
0
10
20
30
40
50
epoch


unsupervised loss: SSE, beta: 5e-06
semisupervised loss
3.65
3.60
3.55
3.50
3.45
supervised loss
3.39
3.38
3.37
3.36
3.35
unsupervised loss
60000
50000
40000
30000
20000
10000
mae
0.0160
0.0155
0.0150
0.0145
0.0140
0.0135
train
val
0
10
20
30
40
50
epoch


unsupervised loss: SSE, beta: 1e-05
semisupervised loss
3.9
3.8
3.7
3.6
3.5
supervised loss
3.43
3.42
3.41
3.40
3.39
3.38
unsupervised loss
60000
50000
40000
30000
20000
10000
mae
0.0175
0.0170
0.0165
0.0160
0.0155
0.0150
train
val
0
10
20
30
40
50
epoch

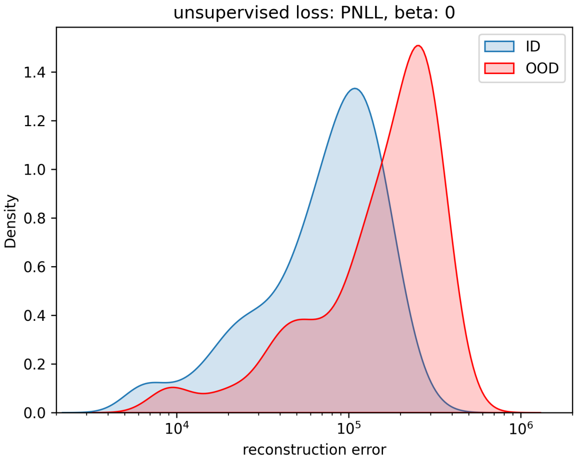

unsupervised loss: SSE, beta: 5e-05
semisupervised loss
6.0
5.5
5.0
4.5
4.0
3.5
supervised loss
3.50
3.48
3.46
3.44
3.42
unsupervised loss
50000
40000
30000
20000
10000
0
mae
0.0205
0.0200
0.0195
0.0190
0.0185
0.0180
0.0175
0.0170
train
val
0
10
20
30
40
50
epoch

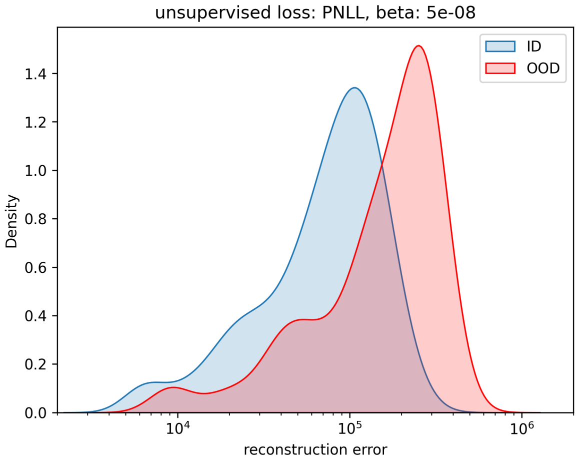

unsupervised loss: SSE, beta: 0.0001
semisupervised loss
9
8
7
6
5
4
supervised loss
3.54
3.52
3.50
3.48
3.46
3.44
unsupervised loss
50000
40000
30000
20000
10000
0
mae
0.0210
0.0205
0.0200
0.0195
0.0190
0.0185
0.0180
0.0175
train
val
0
10
20
30
40
50
epoch

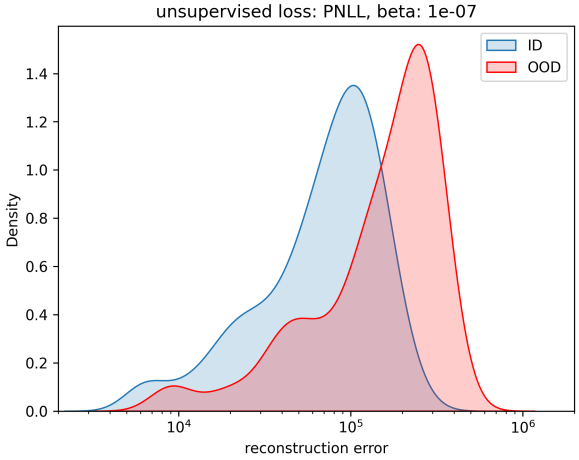


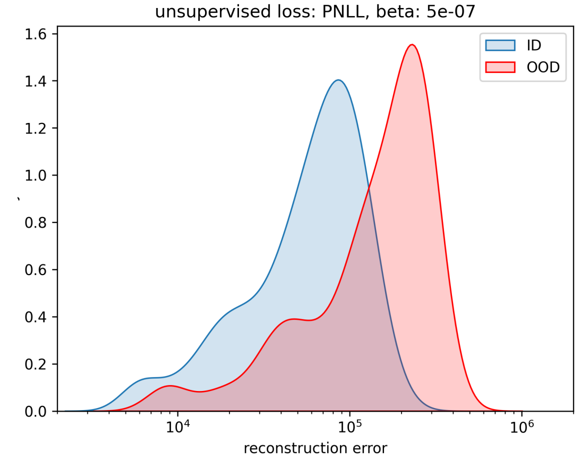
Appendix D KDE Plots for ID and OOD Test Data

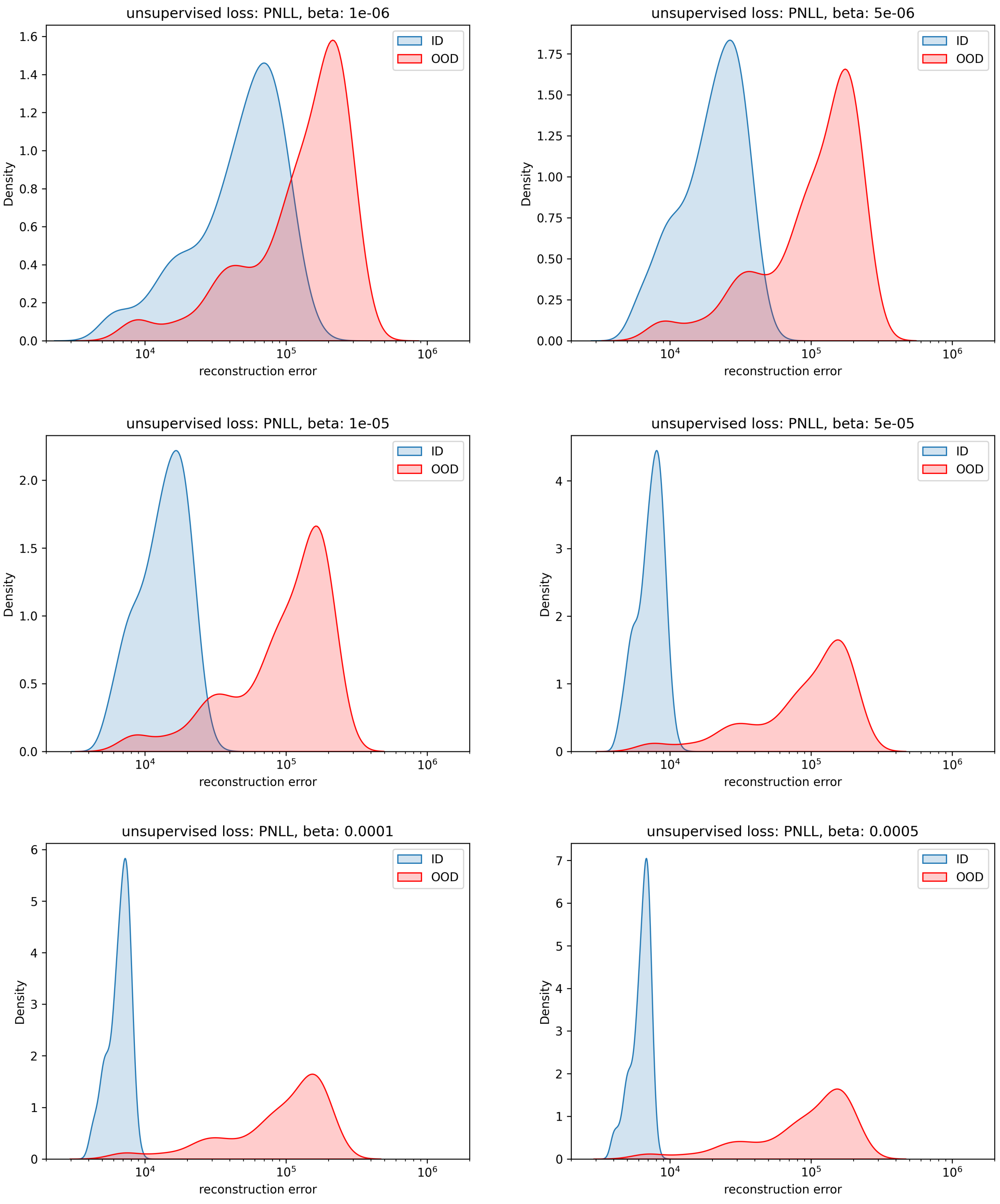
This appendix contains the plots with the KDE distributions for both the ID and OOD data (10% anomaly contribution) for each of the 30 models that were trained.

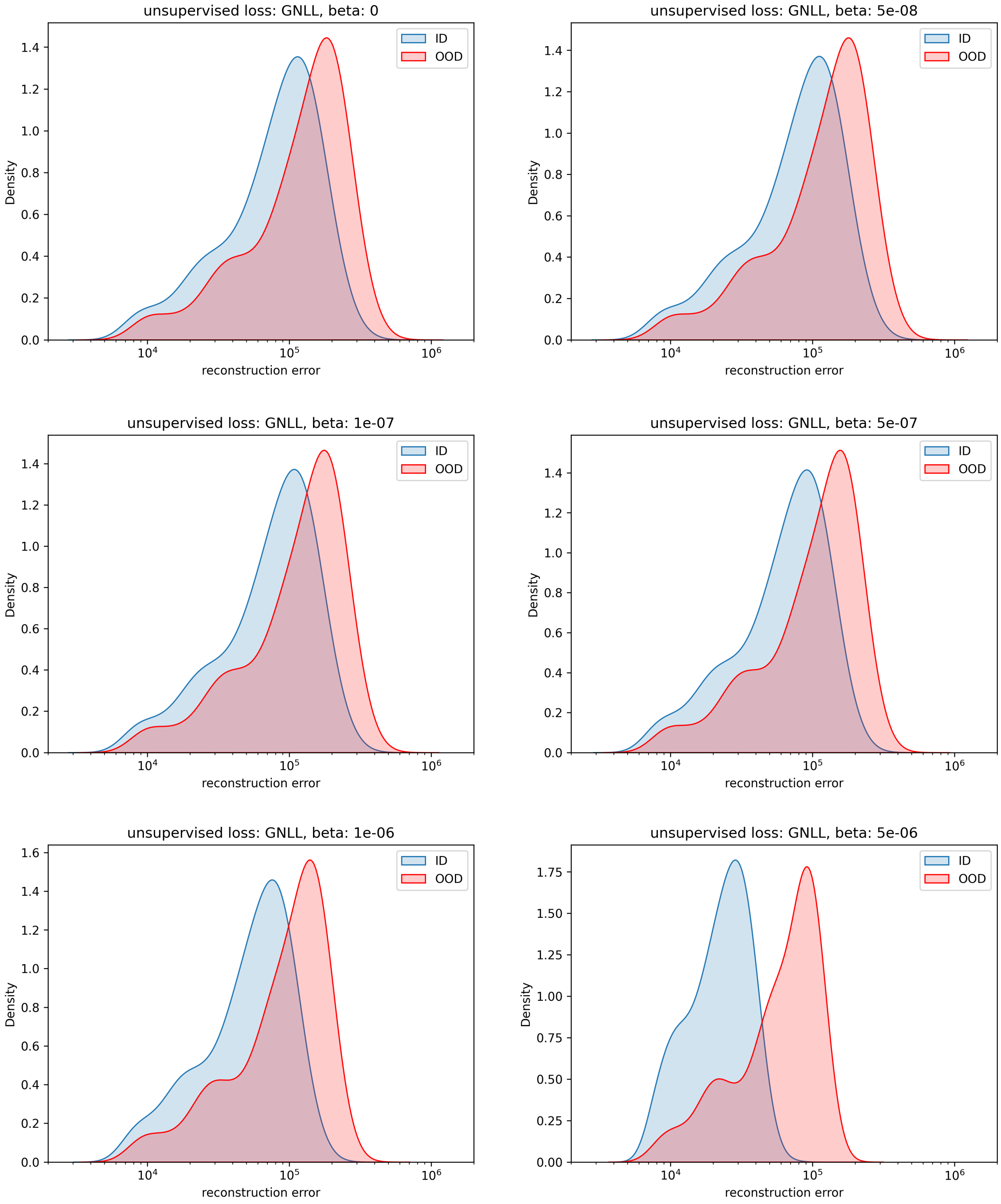


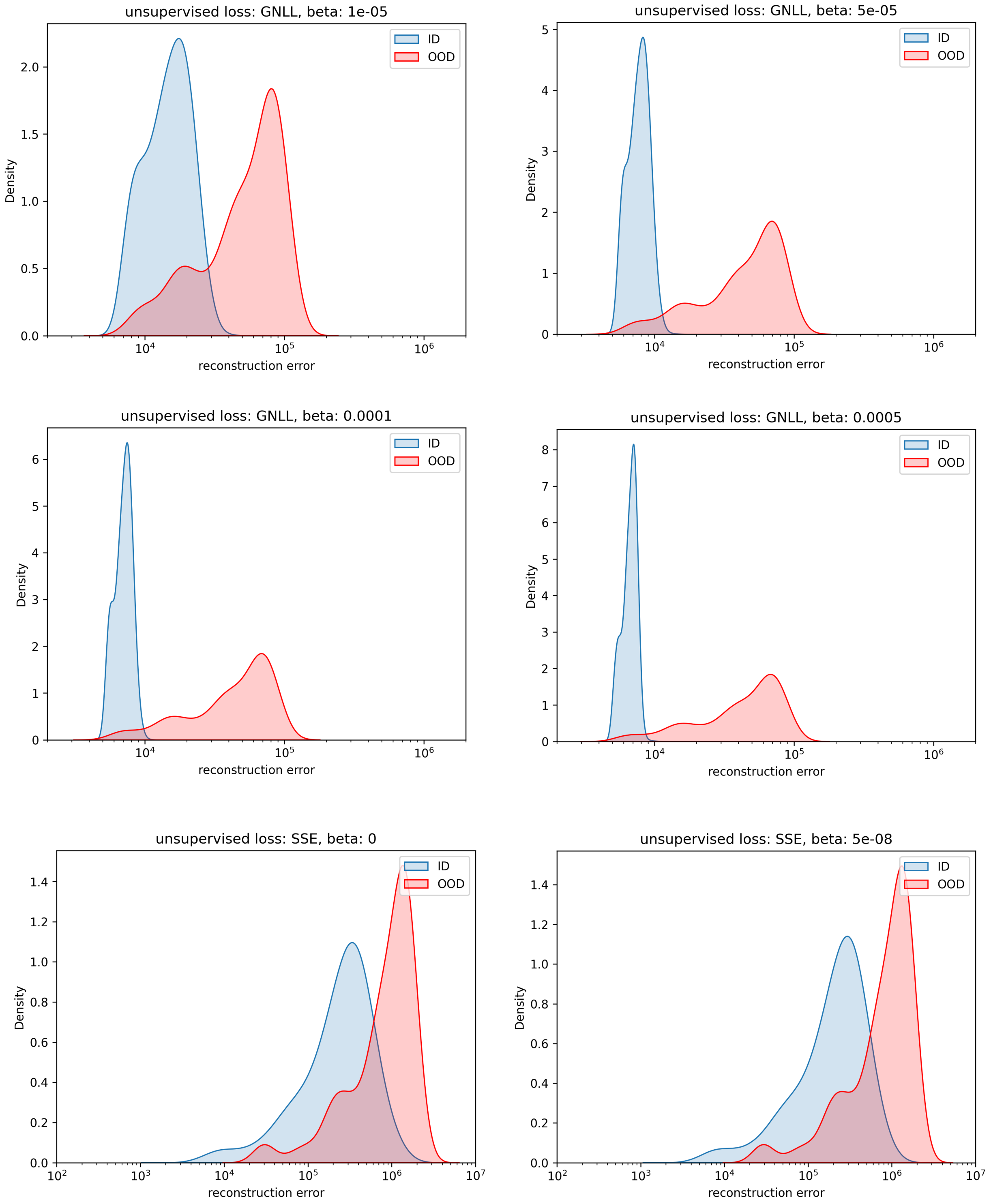




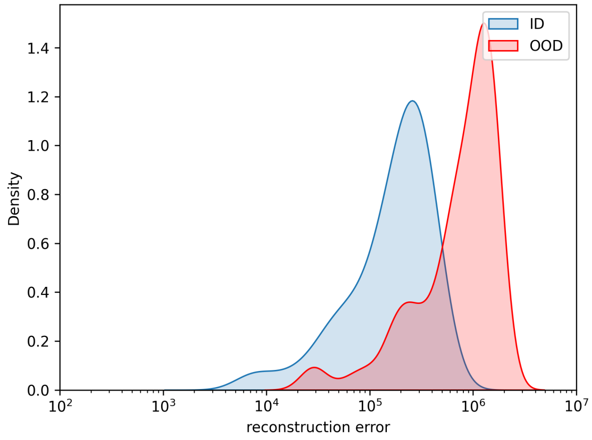



unsupervised loss: PNLL, beta: 1e-06
unsupervised loss: PNLL, beta: 5e-06
unsupervised loss: PNLL, beta: 1e-05
unsupervised loss: PNLL, beta: 5e-05
unsupervised loss: PNLL, beta: 0.0001
unsupervised loss: PNLL, beta: 0.0005
ID
OOD
Density
reconstruction error
10^4
10^5
10^6
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
0.5
1.5
2.0
0
1
2
3
4
5
6
7

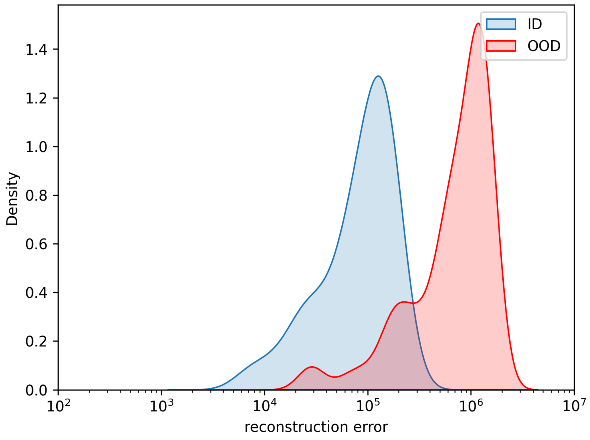

unsupervised loss: GNLL, beta: 0
unsupervised loss: GNLL, beta: 5e-08
unsupervised loss: GNLL, beta: 1e-07
unsupervised loss: GNLL, beta: 5e-07
unsupervised loss: GNLL, beta: 1e-06
unsupervised loss: GNLL, beta: 5e-06
ID
OOD
Density
reconstruction error
10^4
10^5
10^6
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75


unsupervised loss: GNLL, beta: 1e-05
Density
0.0
0.5
1.0
1.5
2.0
10^4
10^5
10^6
reconstruction error
ID
OOD
unsupervised loss: GNLL, beta: 5e-05
Density
0
1
2
3
4
5
10^4
10^5
10^6
reconstruction error
ID
OOD
unsupervised loss: GNLL, beta: 0.0001
Density
0
1
2
3
4
5
6
10^4
10^5
10^6
reconstruction error
ID
OOD
unsupervised loss: GNLL, beta: 0.0005
Density
0
1
2
3
4
5
6
7
8
10^4
10^5
10^6
reconstruction error
ID
OOD
unsupervised loss: SSE, beta: 0
Density
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
10^2
10^3
10^4
10^5
10^6
10^7
reconstruction error
ID
OOD
unsupervised loss: SSE, beta: 5e-08
Density
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
10^2
10^3
10^4
10^5
10^6
10^7
reconstruction error
ID
OOD

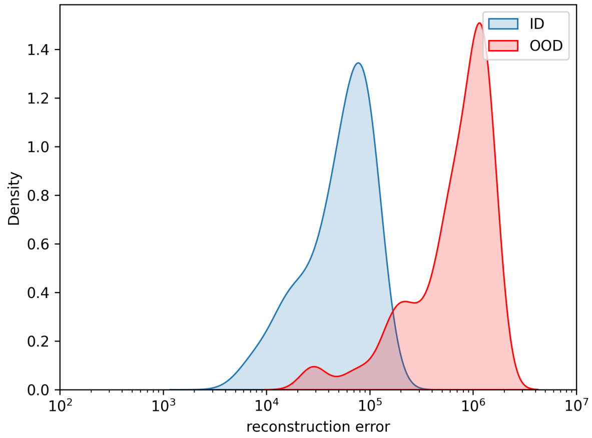
unsupervised loss: SSE, beta: 1e-07



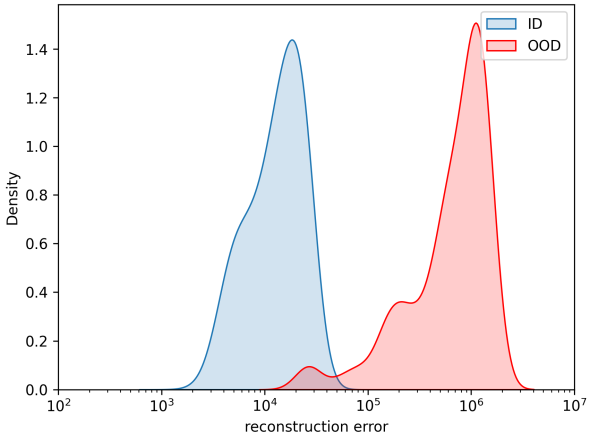
unsupervised loss: SSE, beta: 5e-07



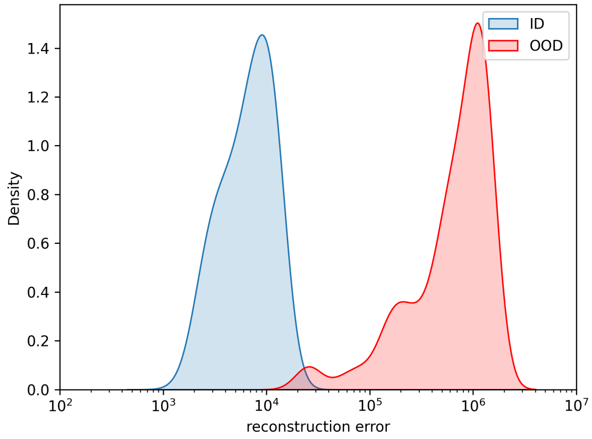
unsupervised loss: SSE, beta: 1e-06



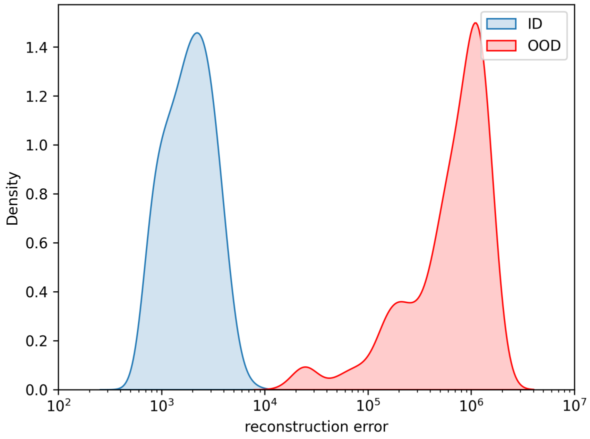
unsupervised loss: SSE, beta: 5e-06

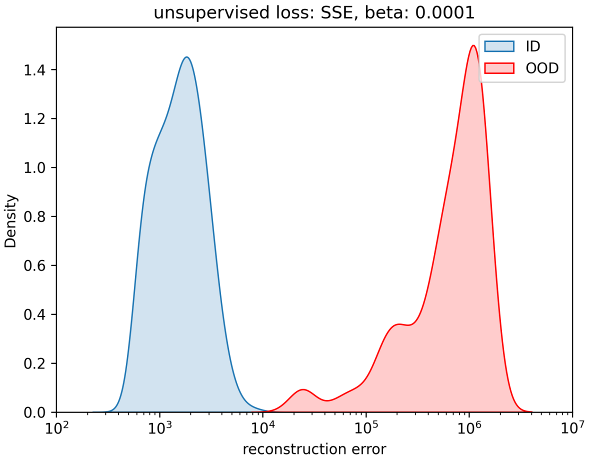


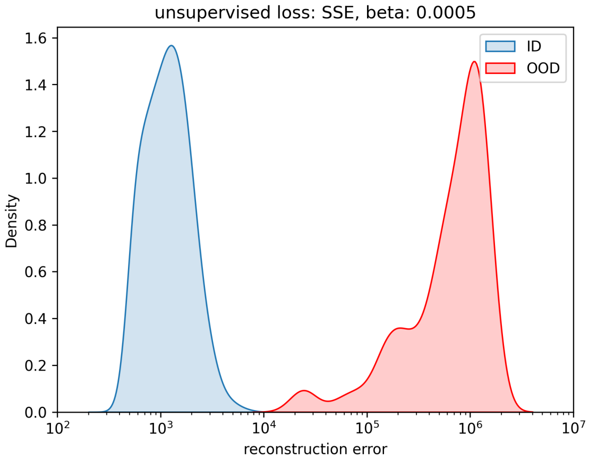
unsupervised loss: SSE, beta: 1e-05



unsupervised loss: SSE, beta: 5e-05

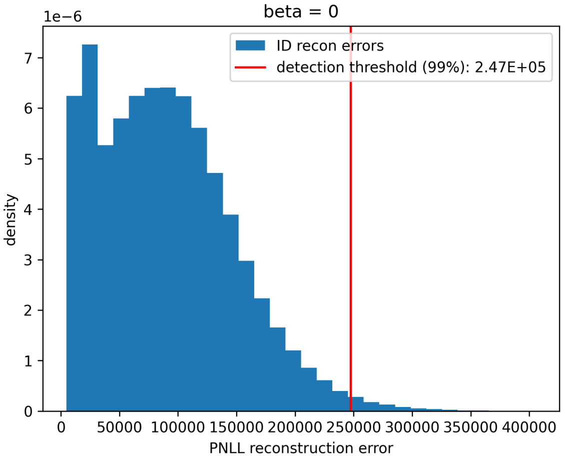


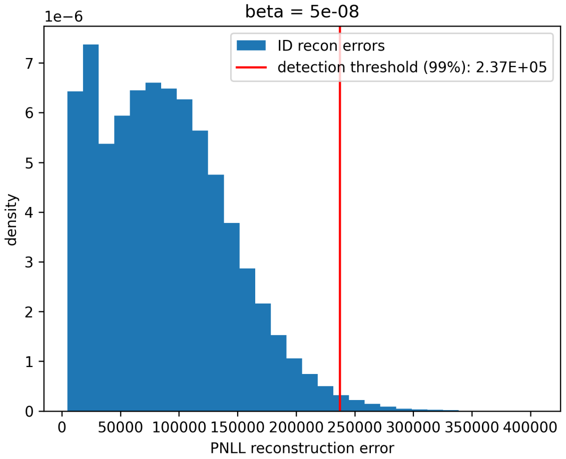


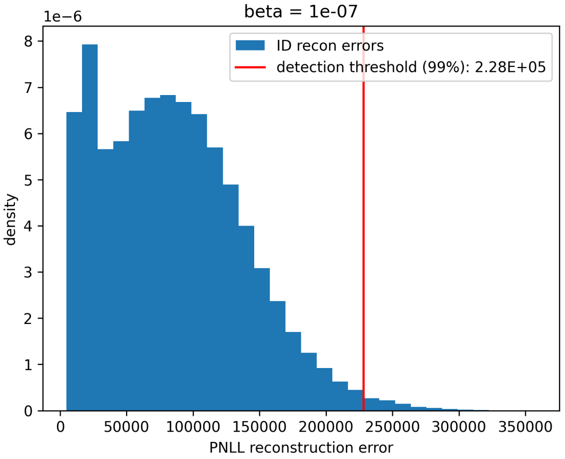


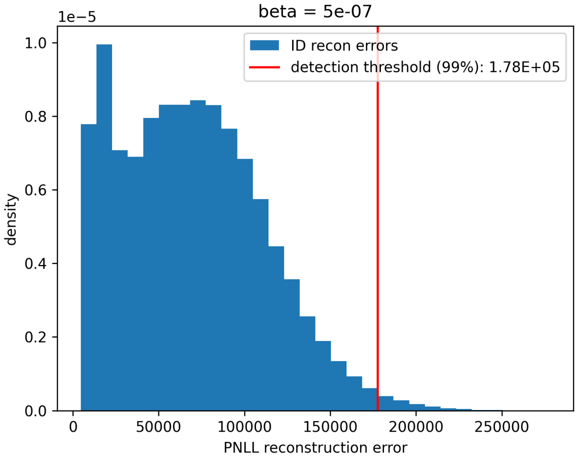
Appendix E In-Distribution Distribution Plots with OOD Detection Threshold

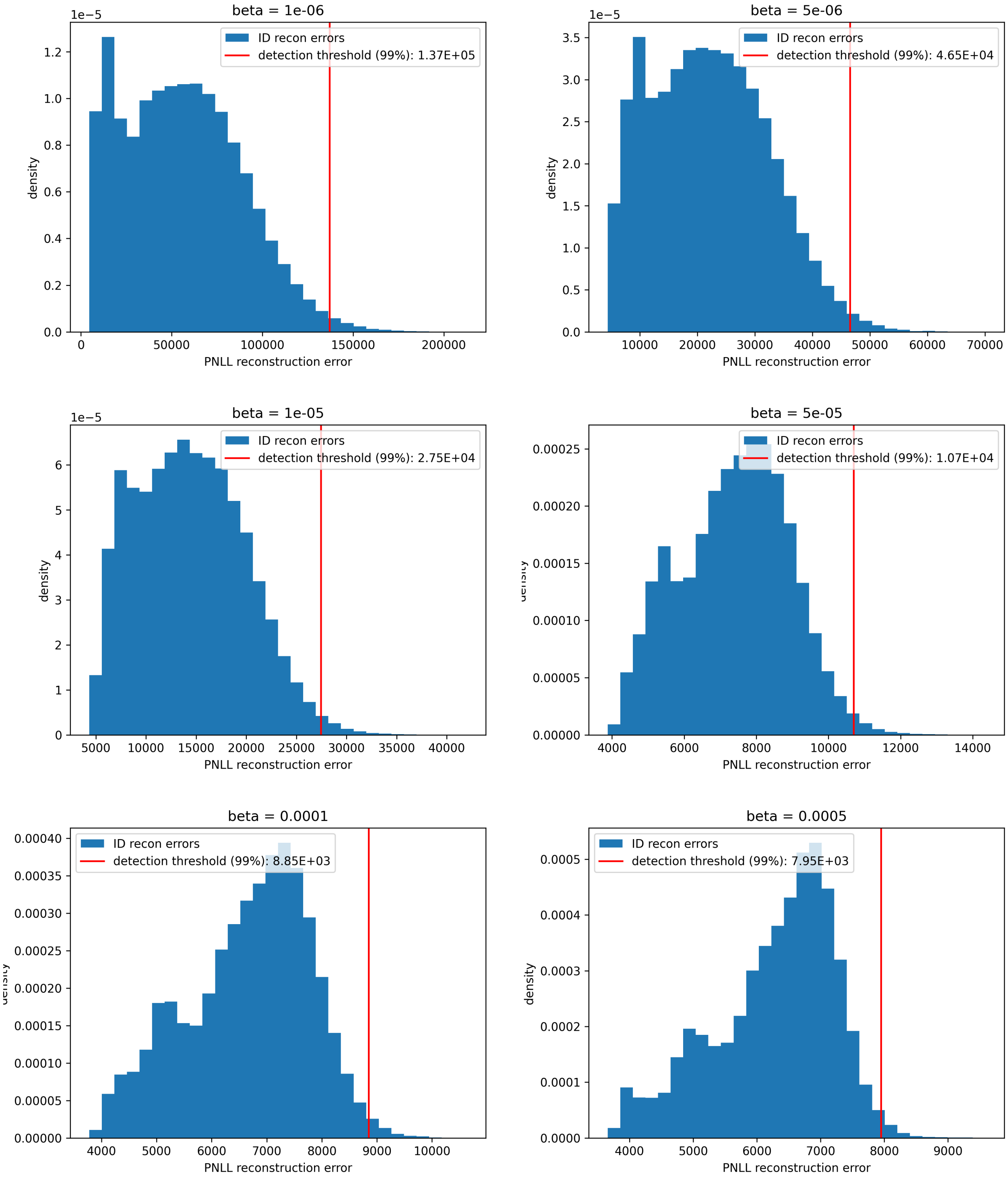
This appendix contains a histogram showing the ID plots for each model, along with the fitted Gaussian distribution and OOD detection threshold, corresponding to the 99th percentile.

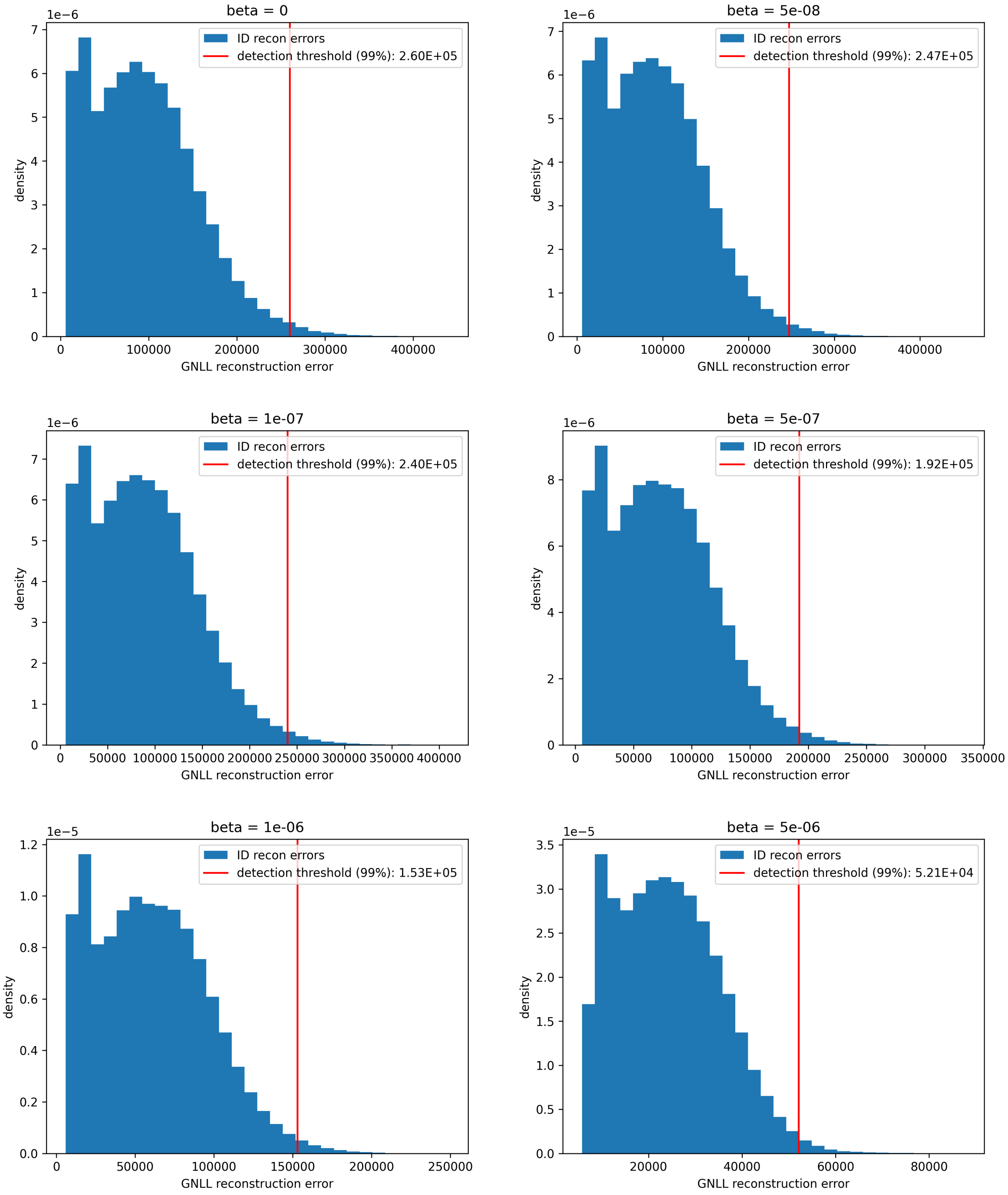


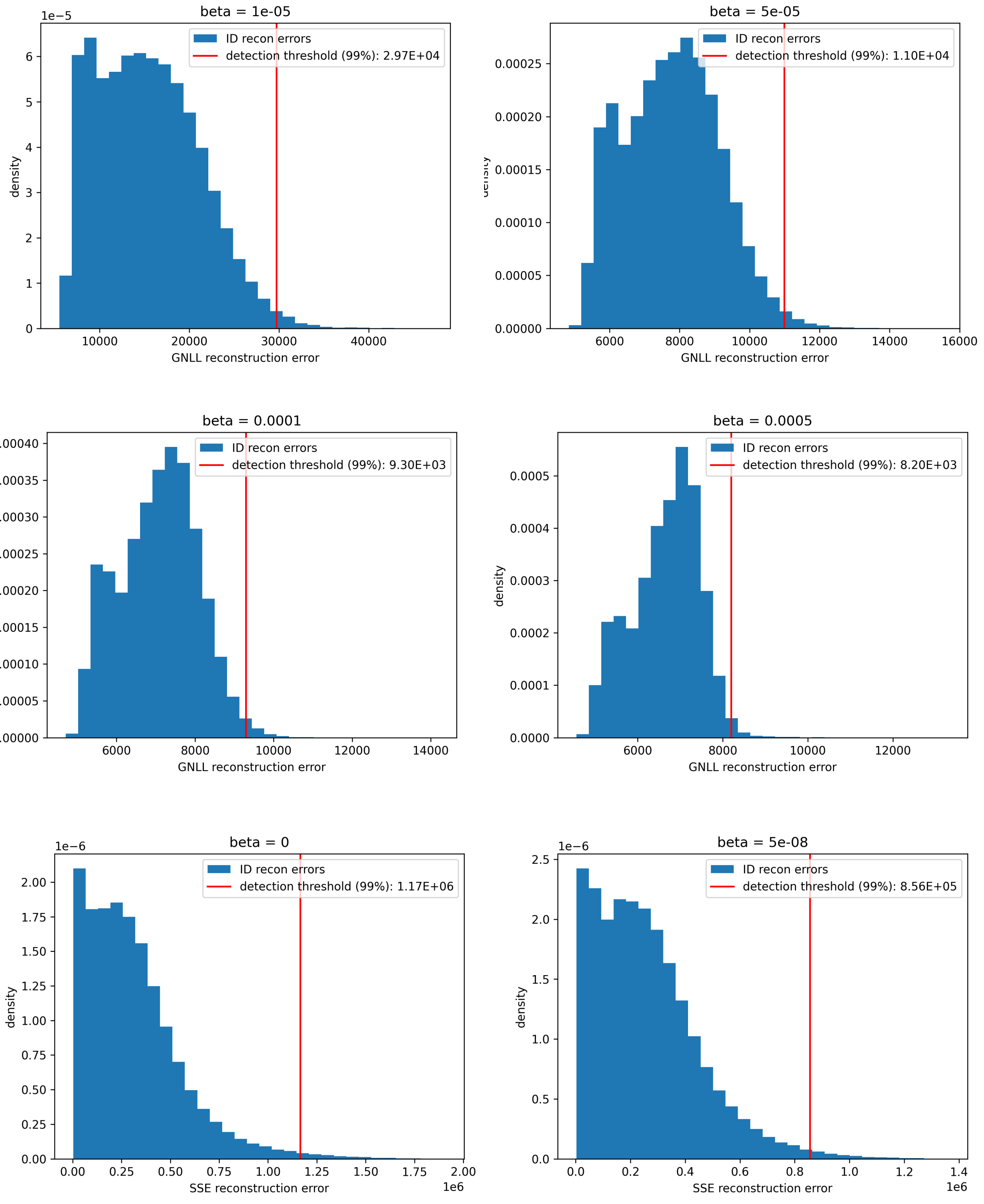


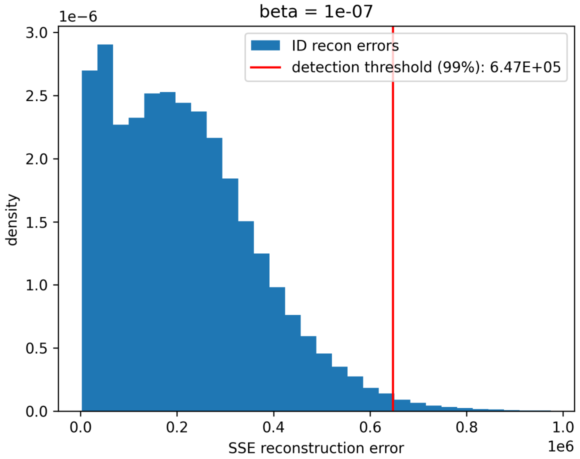


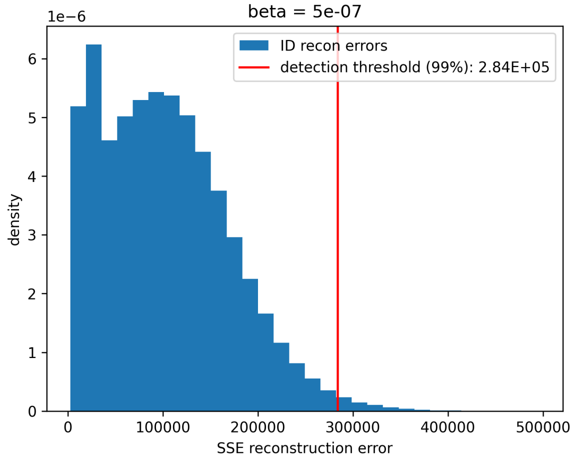


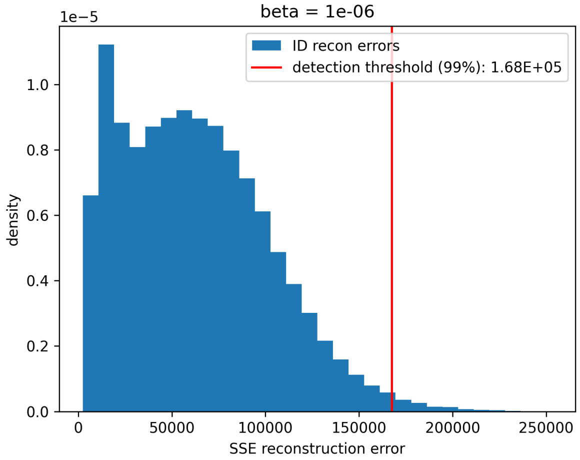

beta = 1e-06
1e−5
ID recon errors
detection threshold (99%): 1.37E+05
density
0.0
0.2
0.4
0.6
0.8
1.0
1.2
0
50000
100000
150000
200000
PNLL reconstruction error
beta = 5e-06
1e−5
ID recon errors
detection threshold (99%): 4.65E+04
density
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
10000
20000
30000
40000
50000
60000
70000
PNLL reconstruction error
beta = 1e-05
1e−5
ID recon errors
detection threshold (99%): 2.75E+04
density
0
1
2
3
4
5
6
5000
10000
15000
20000
25000
30000
35000
40000
PNLL reconstruction error
beta = 5e-05
ID recon errors
detection threshold (99%): 1.07E+04
0.00000
0.00005
0.00010
0.00015
0.00020
0.00025
4000
6000
8000
10000
12000
14000
PNLL reconstruction error
beta = 0.0001
ID recon errors
detection threshold (99%): 8.85E+03
0.00000
0.00005
0.00010
0.00015
0.00020
0.00025
0.00030
0.00035
0.00040
4000
5000
6000
7000
8000
9000
10000
PNLL reconstruction error
beta = 0.0005
ID recon errors
detection threshold (99%): 7.95E+03
density
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
4000
5000
6000
7000
8000
9000
PNLL reconstruction error

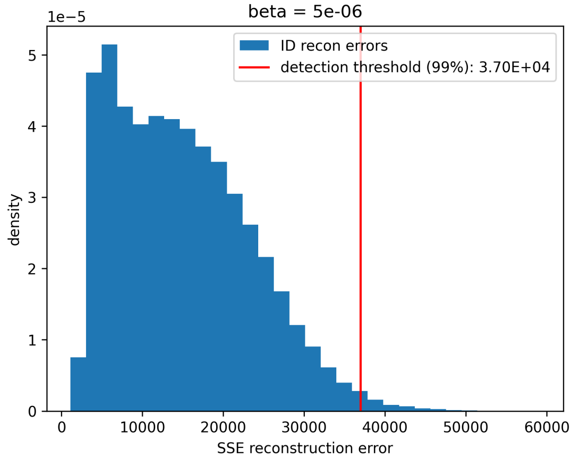

beta = 0
1e−6
ID recon errors
detection threshold (99%): 2.60E+05
density
GNLL reconstruction error
0
1
2
3
4
5
6
7
0
100000
200000
300000
400000
beta = 5e-08
1e−6
ID recon errors
detection threshold (99%): 2.47E+05
density
GNLL reconstruction error
0
100000
200000
300000
400000
beta = 1e-07
1e−6
ID recon errors
detection threshold (99%): 2.40E+05
density
GNLL reconstruction error
0
50000
100000
150000
200000
250000
300000
350000
400000
beta = 5e-07
1e−6
ID recon errors
detection threshold (99%): 1.92E+05
density
GNLL reconstruction error
0
2
4
6
8
0
50000
100000
150000
200000
250000
300000
350000
beta = 1e-06
1e−5
ID recon errors
detection threshold (99%): 1.53E+05
density
GNLL reconstruction error
0.0
0.2
0.4
0.6
0.8
1.0
1.2
0
50000
100000
150000
200000
250000
beta = 5e-06
1e−5
ID recon errors
detection threshold (99%): 5.21E+04
density
GNLL reconstruction error
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
20000
40000
60000
80000

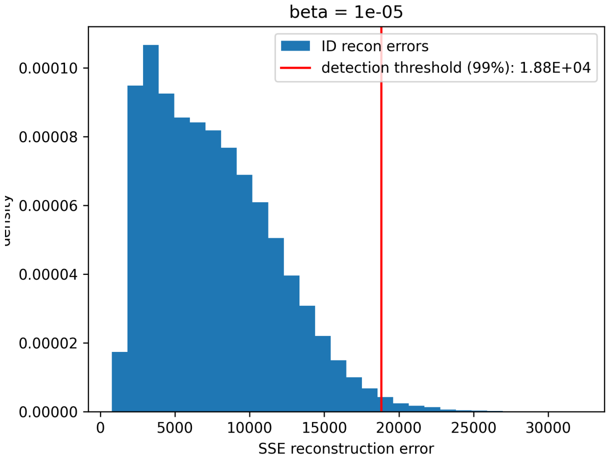

beta = 1e-05
1e−5
density
ID recon errors
detection threshold (99%): 2.97E+04
0
1
2
3
4
5
6
10000
20000
30000
40000
GNLL reconstruction error
beta = 5e-05
ID recon errors
detection threshold (99%): 1.10E+04
0.00000
0.00005
0.00010
0.00015
0.00020
0.00025
6000
8000
10000
12000
14000
16000
GNLL reconstruction error
beta = 0.0001
ID recon errors
detection threshold (99%): 9.30E+03
0.00000
0.00005
0.00010
0.00015
0.00020
0.00025
0.00030
0.00035
0.00040
6000
8000
10000
12000
14000
GNLL reconstruction error
beta = 0.0005
density
ID recon errors
detection threshold (99%): 8.20E+03
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
6000
8000
10000
12000
GNLL reconstruction error
beta = 0
1e−6
density
ID recon errors
detection threshold (99%): 1.17E+06
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
1e6
SSE reconstruction error
beta = 5e-08
1e−6
density
ID recon errors
detection threshold (99%): 8.56E+05
0.0
0.5
1.0
1.5
2.0
2.5
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1e6
SSE reconstruction error

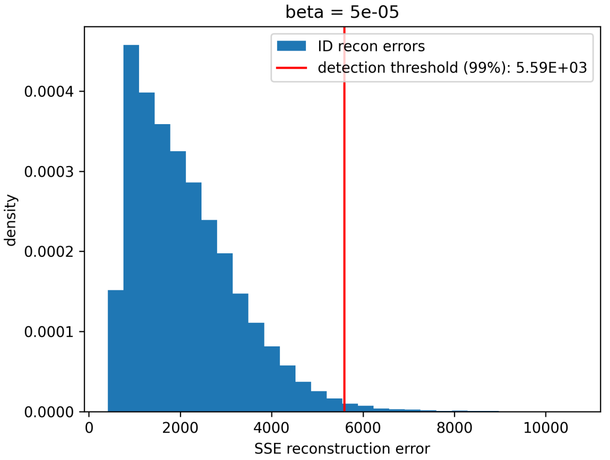

beta = 1e-07
1e−6
3.0
2.5
2.0
1.5
1.0
0.5
0.0
density
ID recon errors
detection threshold (99%): 6.47E+05
0.0
0.2
0.4
0.6
0.8
1.0
1e6
SSE reconstruction error

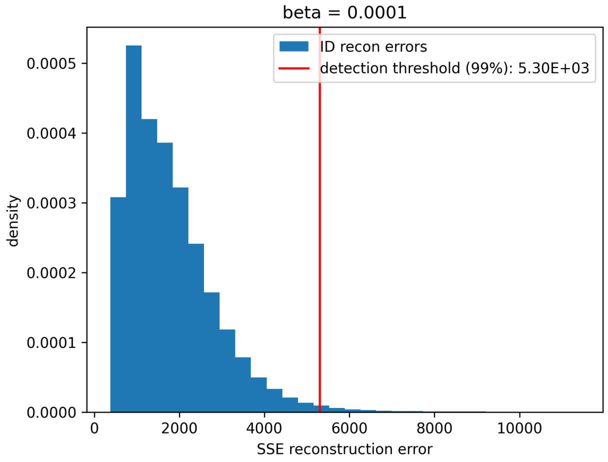

beta = 5e-07
1e−6
6
5
4
3
2
1
0
density
ID recon errors
detection threshold (99%): 2.84E+05
0
100000
200000
300000
400000
500000
SSE reconstruction error

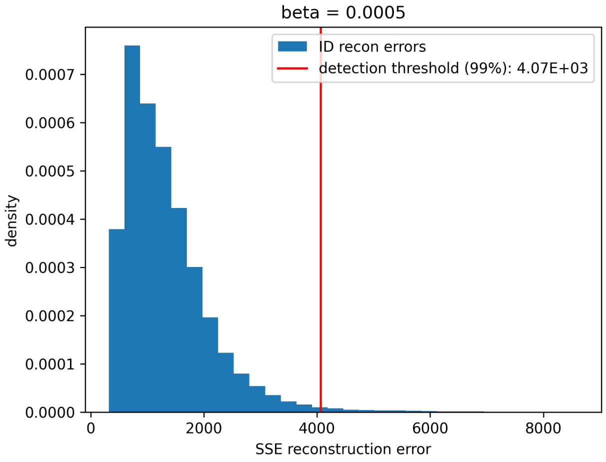

beta = 1e-06
1e−5
1.0
0.8
0.6
0.4
0.2
0.0
density
ID recon errors
detection threshold (99%): 1.68E+05
0
50000
100000
150000
200000
250000
SSE reconstruction error


beta = 5e-06
1e−5
5
4
3
2
1
0
density
ID recon errors
detection threshold (99%): 3.70E+04
0
10000
20000
30000
40000
50000
60000
SSE reconstruction error


beta = 1e-05
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
density
ID recon errors
detection threshold (99%): 1.88E+04
0
5000
10000
15000
20000
25000
30000
SSE reconstruction error


beta = 5e-05
0.0004
0.0003
0.0002
0.0001
0.0000
density
ID recon errors
detection threshold (99%): 5.59E+03
0
2000
4000
6000
8000
10000
SSE reconstruction error


beta = 0.0001
ID recon errors
detection threshold (99%): 5.30E+03
density
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
2000
4000
6000
8000
10000
SSE reconstruction error


beta = 0.0005
ID recon errors
detection threshold (99%): 4.07E+03
density
0.0007
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
2000
4000
6000
8000
SSE reconstruction error

Bibliography

[1] G. F. Knoll, Radiation detection and measurement, John WIley and Sons, 2010.

[2] E. C. Finn, L. A. Metz, R. F. Payne, J. I. Friese, L. R. Greenwood, J. D. Kephart, B. D. Pierson and T. A. Ellis, "Methods to collect, compile, and analyze observed short-lived fission product gamma data," Pacific Northwest National Lab. (PNNL), Richland, WA (United States), 2011.

[3] M. Rawool-Sullivan, J. Bounds, S. Brumby, L. Prasad and J. Sullivan, "Steps Toward Automated Gamma Ray Spectroscopy," Los Alamos National Laboratory, 2010.

[4] P. Olmos, J. Diaz, J. Perez, P. Gomez, V. Rodellar, P. Aguayo, A. Bru, G. Garcia-Belmonte and J. De Pablos, "A new approach to automatic radiation spectrum analysis," *IEEE Transactions on Nuclear Science,* vol. 38, no. 4, pp. 971-975, 1991.

[5] R. Abdel-Aal and M. Al-Haddad, "Determination of radioisotopes in gamma-ray spectroscropy using abductive machine learning," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment,* vol. 391, no. 2, pp. 275-288, 1997.

[6] L. Chen and Y.-X. Wei, "Nuclide identification algorithm based on K--L transform and neural networks," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment,* vol. 598, no. 2, pp. 450-453, 2009.

[7] C. Bobin, O. Bichler, V. Lourenço, C. Thiam and M. Thévenin, "Real-time radionuclide identification in gamma-emitter mixtures based on spiking neural network," *Applied Radiation and Isotopes,* vol. 109, pp. 405-409, 2016.

[8] M. Kamuda, J. Stinnett and C. Sullivan, "Automated isotope identification algorithm using artificial neural networks," *IEEE Transactions on Nuclear Science,* vol. 64, no. 7, pp. 1858-1864, 2017.

[9] J. Kim, K. Park and G. Cho, "Multi-radioisotope identification algorithm using an artificial neural network for plastic gamma spectra," *Applied Radiation and Isotopes,* vol. 147, pp. 83-90, 2019.

[10] S. J. Murray, J. Schmitz, S. Balkır and M. W. Hoffman, "A low complexity radioisotope identification system using an integrated multichannel analyzer and embedded neural network," in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2019.

[11] G. Daniel, F. Ceraudo, O. Limousin, D. Maier and A. Meuris, "Automatic and real-time identification of radionuclides in gamma-ray spectra: a new method based on convolutional neural network trained with synthetic data set," *IEEE Transactions on Nuclear Science,* vol. 67, no. 4, pp. 644-653, 2020.

[12] S. Qi, W. Zhao, Y. Chen, W. Chen, J. Li, H. Zhao, W. Xiao, X. Ai, K. Zhang and S. Wang, "Comparison of machine learning approaches for radioisotope identification using NaI (TI) gamma-ray spectrum," *Applied Radiation and Isotopes,* vol. 186, p. 110212, 2022.

[13] A. Khatiwada, M. Klasky, M. Lombardi, J. Matheny and A. Mohan, "Machine Learning technique for isotopic determination of radioisotopes using HPGe γ-ray spectra," *arXiv preprint arXiv:2301.01415,* 2023.

[14] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trasactions on Image processing,* vol. 15, no. 12, pp. 3736-3745, 2006.

[15] W. Meiniel, J.-C. Olivo-Marin and E. D. Angelini, "Denoising of microscopy images: a review of the state-of-the-art, and a new sparsity-based method," *IEEE Transactions on Image Processing,* vol. 27, no. 8, pp. 3842-3856, 2018.

[16] K.-K. Huang, D.-Q. Dai, C.-X. Ren and Z.-R. Lai, "Learning Kernel Extended Dictionary for Face Recognition," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 28, no. 5, pp. 1082-1094, 2017.

[17] X.-Y. Jing, F. Wu, X. Zhu, X. Dong, F. Ma and Z. Li, "Multi-spectral low-rank structured dictionary learning for face recognition," *Pattern Recognition,* vol. 59, pp. 14-25, 2016.

[18] A. Nguyen, J. Yosinksi and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[19] J. T. Zhou, K. Di, J. Du, X. Peng, H. Yang, S. Pan, I. Tsang, Y. Liu, Z. Qin and R. S. M. Goh, "Sc2net: Sparse lstms for sparse coding," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 32, no. 1, 2018.

[20] W. Luo, W. Liu, D. Lian, J. Tang, L. Duan, X. Peng and S. Gao, "Video anomaly detection with sparse coding inspired deep neural networks," *IEEE transactions on pattern analysis and machine intelligence,* vol. 43, no. 3, pp. 1070-1084, 2019.

[21] V. Papyan, Y. Romano and M. Elad, "Convolutional neural networks analyzed via convolutional sparse coding," *The Journal of Machine Learning Research,* vol. 18, no. 1, pp. 2887-2938, 2017.

[22] S. Arora, R. Ge, T. Ma and A. Moitra, "Simple, efficient, and neural algorithms for sparse coding," in *Conference on learning theory*, 2015.

[23] X. Sun, N. M. Nasrabadi and T. D. Tran, "Supervised deep sparse coding networks," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018.

[24] J. Mairal, F. Bach and J. Ponce, "Task-driven dictionary learning," *IEEE transactions on pattern analysis and machine intelligence,* vol. 34, no. 4, pp. 791-804, 2011.

[25] J. Yang, K. Zhou, Y. Li and Z. Liu, "Generalized out-of-distribution detection: A survey," *arXiv preprint arXiv:2110.11334,* 2021.

[26] J. Katz-Samuels, J. B. Nakhleh, R. Nowak and Y. Li, "Training ood detectors in their natural habitats," *International Conference on Machine Learning,* pp. 10848-10865, 2022.

[27] M. Salehi, H. Mirzaei, D. Hendrycks, Y. Li, M. H. Rohban and M. Sabokrou, "A unified survey on anomaly, novelty, open-set, and out-of-distribution detection: Solutions and future challenges," *arXiv preprint arXiv:2110.14051,* 2021.

[28] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *arXiv preprint arXiv:1610.02136,* 2016.

[29] S. Liang, Y. Li and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," *arXiv preprint arXiv:1706.02690,* 2017.

[30] M. Hein, M. Andriushchenko and J. Bitterwolf, "Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem," in

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019.

[31] J. Bitterwolf, A. Meinke and M. Hein, "Certifiably adversarially robust detection of out-of-distribution data," *Advances in Neural Information Processing Systems,* vol. 33, pp. 16085-16095, 2020.

[32] D. Hendrycks, M. Mazeika and T. Dietterich, "Deep anomaly detection with outlier exposure," *arXiv preprint arXiv:1812.04606,* 2018.

[33] C. a. L. A. Qiu, M. Kloft, M. Rudolph and S. Mandt, "Latent outlier exposure for anomaly detection with contaminated data," *International Conference on Machine Learning,* pp. 18153-18167, 2022.

[34] K. Lee, K. Lee, H. Lee and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," *Advances in neural information processing systems,* vol. 31, 2018.

[35] J. Ren, S. Fort, J. Liu, A. G. Roy, S. Padhy and B. Lakshminarayanan, "A simple fix to mahalanobis distance for improving near-ood detection," *arXiv preprint arXiv:2106.09022,* 2021.

[36] E. Techapanurak, M. Suganuma and T. Okatani, "Hyperparameter-free out-of-distribution detection using cosine similarity," in *Proceedings of the Asian conference on computer vision*, 2020.

[37] J. Van Amersfoort, L. Smith, Y. W. Teh and Y. Gal, "Uncertainty estimation using a single deep deterministic neural network," *International conference on machine learning,* pp. 9690-9700, 2020.

[38] H. Huang, Z. Li, L. Wang, S. Chen, B. Dong and X. Zhou, "Feature space singularity for out-of-distribution detection," *arXiv preprint arXiv:2011.14654,* 2020.

[39] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," *International conference on learning representations,* 2018.

[40] D. Abati, A. Porrello, S. Calderara and R. Cucchiara, "Latent space autoregression for novelty detection," *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition,* pp. 481-490, 2019.

[41] T. Denouden, R. Salay, K. Czarnecki, V. Abdelzad, B. Phan and S. Vernekar, "Improving reconstruction autoencoder out-of-distribution detection with mahalanobis distance," *arXiv preprint arXiv:1812.02765,* 2018.

[42] Y. Zhou, "Rethinking reconstruction autoencoder-based out-of-distribution detection," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,* pp. 7379-7387, 2022.

[43] Z. Xiao, Q. Yan and Y. Amit, "Likelihood regret: An out-of-distribution detection score for variational auto-encoder," *Advances in neural information processing systems,* vol. 33, pp. 20685-20696, 2020.

[44] J. M. Ghawaly Jr, A. D. Nicholson, D. E. Archer, M. J. Willis, I. Garishvili, B. Longmire, A. J. Rowe, I. R. Stewart and M. T. Cook, "Characterization of the Autoencoder Radiation Anomaly Detection (ARAD) model," *Engineering Applications of Artificial Intelligence,* vol. 111, p. 104761, 2022.

[45] A. Martins and R. Astudillo, "From softmax to sparsemax: A sparse model of attention and multi-label classification," in *International conference on machine learning*, 2016.

[46] W. Feller, An introduction to probability theory and its applications, John Wiley & Sons, 1967.

[47] J. Lass, M. E. Boggild, P. Hedegard and K. Lefmann, "Multinomial, Poisson and Gaussian statistics in count data analysis," *Journal of Neutron Research,* vol. 23, no. 1, pp. 69-92, 2021.

[48] D. J. Mitchell, H. Lee, G. G. Thoreson and S. M. Horne, "GADRAS Detector Response Function.," *Sandia National Lab. (SNL-NM), Albuquerque, NM (United States),* 2014.

[49] T. Morrow, N. Price and T. McGuire, "PyRIID v.2.0.0," 2021. [Online]. Available: https://www.osti.gov//servlets/purl/1894123.

[50] A. A. P. B. E. B. Z. C. C. C. G. S. C. A. D. J. D. M. D. S. G. I. G. A. H. G. I. M. I. R. J. Y. J. L. Martín Abadi, *TensorFlow: Large-scale machine learning on heterogeneous systems,* https://www.tensorflow.org, 2015.

[51] I. Malkiel and L. Wolf, "Mtadam: Automatic balancing of multiple training loss terms," *arXiv preprint arXiv:2006.14683,* 2020.

[52] S. Sharma, C. Bellinger, N. Japkowicz, R. Berg and K. Ungar, "Anomaly detection in gamma ray spectra: A machine learning perspective," in *2012 IEEE symposium on computational intelligence for security and defence applications*, 2012.

[53] M. Blondel, A. F. Martins and V. Niculae, "Learning with fenchel-young losses," *The Journal of Machine Learning Research,* vol. 21, no. 1, pp. 1314-1382, 2020.