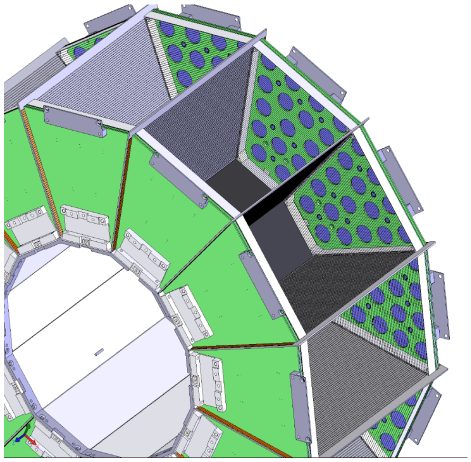
Machine Learning Event Reconstruction for Liquid Xenon Positron Emission Tomography Time Projection Chambers: Capstone Project Report



Brendan Posehn, Daniel Backhouse, Jonah Gourlay
Project Sponsors: Dr. Doug Bryman (UBC) and Dr. Wojciech Fedorko (TRIUMF)

ENPH 459
The University of British Columbia

Apr 8, 2020

Project Number: 2007

Contents

Executive Summary

This project has focused on the development of a machine learning (ML) model to reconstruct the position of particle interactions inside a Liquid Xenon (LXe) Positron Emission Tomography (PET) Scanner. The model uses scintillation light data produced by particle interactions (detected on the walls of the scanner modules) for position reconstruction. LXe scanners are desirable over traditional PET scanners as they can be built at a lower cost and resolve finer images. Our model is intended to be used for prototyping a physical LXe scanner by helping to evaluate the performance of different designs. This work will be carried forward by the sponsorship team for this project, a group of researchers at TRIUMF and UBC.

Our model must take simulated particle interaction data (provided by the team at TRIUMF, validated by this team) as it's input, then predict a bounding box (BB)—within the volume of a single scanner module —that contains all particle interactions. The following approaches were tested:

- Model predicts corners of bounding box.
- Model predicts centroid values (center values, lengths in x, y, z) of bounding box.
- The module itself was discretized into many cubes, model predicts which cubes interactions occurred in.

Each approach was ran on differing ML architectures:

- Multi-layer Perceptron
- Engineered Features Network
- Convolutional Neural Network
- Residual Neural Network

It was found that the Discretized Volume approach trained on a ResNet implementation was the most effective solution. Our model was able to contain 81.5% of interactions using input data from all 6 module walls. Reducing the number of active walls is necessary as practical modules only have data from 2 walls used as input. We found that with 4 and 2 active walls our model was able to achieve 79.3% and 50.0% interactions contained. Unfortunately we were not able to exhaust our list of ideas to improve performance of the model further, this document will include modifications which we believe are worth pursuing.

System Level Diagram

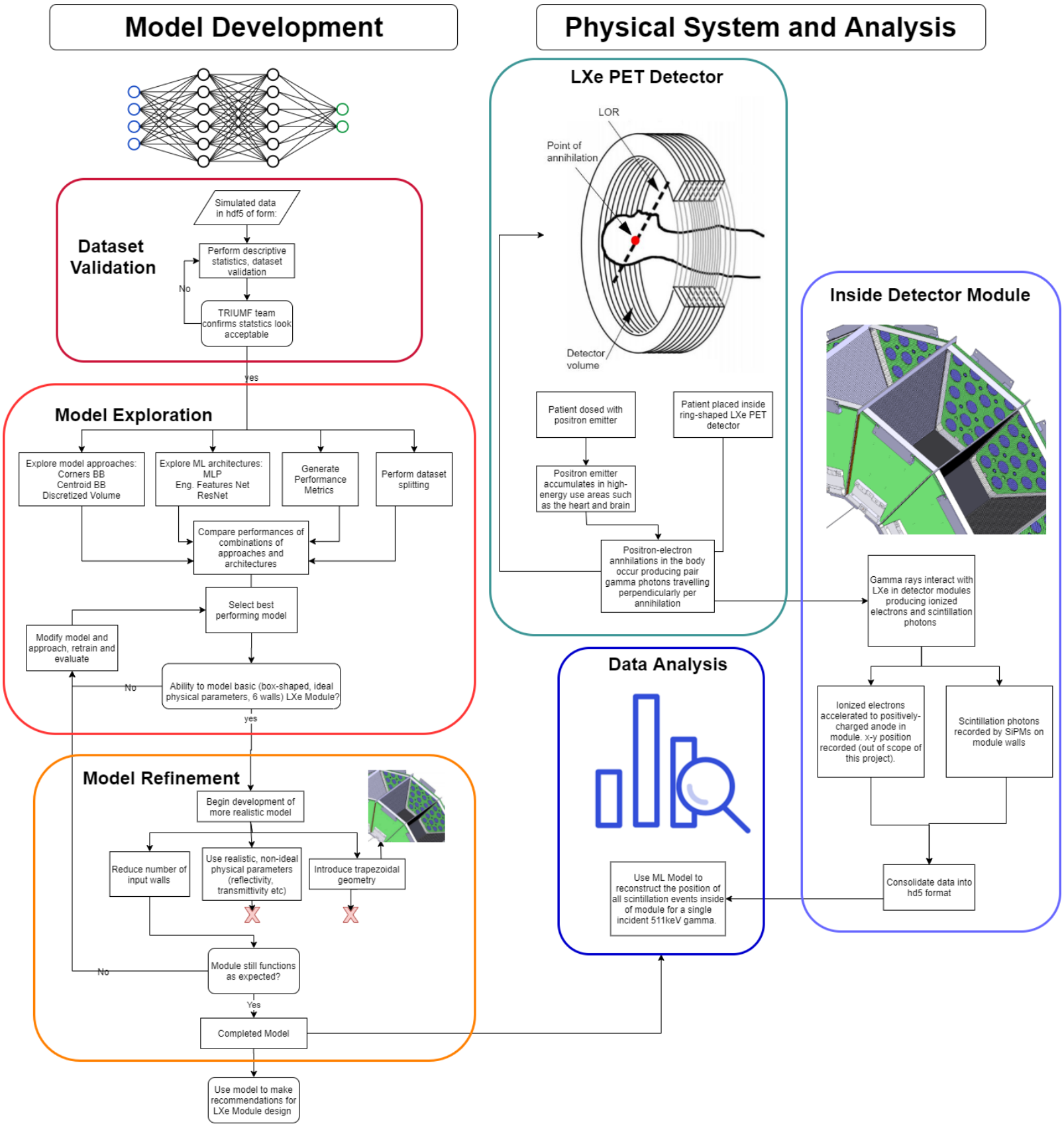


Figure 0.1: System Level Diagram including physical workings of a LXe scanner and how it would integrate with our ML model (right side), as well as the development process of the model (left side).

1 Introduction

1.1 Background

This project was overseen by Dr. Doug Bryman (UBC), Dr. Wojciech Fedorko, Dr. Aleksey Sher, and Dr. Patrick de Perio (TRIUMF). These individuals are part of various groups at TRIUMF, some giving support to our project specifically, and others following the project in order to continue with it once terminated.

PET Background
Positron Emission Tomography (PET) scanners allow medical practitioners to image areas of the body with high chemical activity, and are used in cancer, heart disease, and brain disease diagnoses [5].

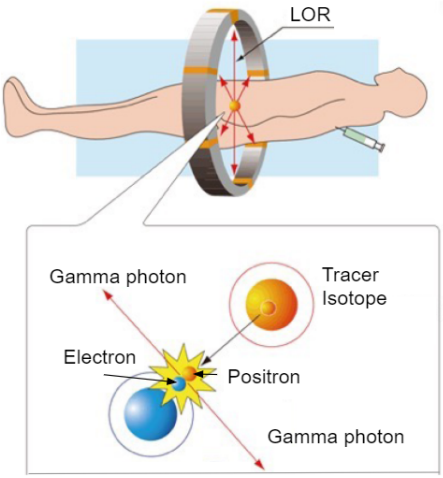


Figure 1.1: PET scan diagram
Image: https://content.sciendo.com/view/journals/hkbrd/17/1/article-p12.xml

To perform a PET scan, a patient is injected with a radioactive tracer isotope linked to a metabolically active molecule such as glucose, and then placed inside of a PET scanner. The tracer molecule accumulates in metabolically active regions of the body, such as the brain. When the tracer decays it releases a positron which travels a few millimetres before annihilating with an electron, producing a pair of gamma photons travelling in opposite directions. These photons will then interact with scintillation material inside of the PET scanner ring, undergoing light-producing interactions that can be recorded. A scintillator is a material that, when struck by a particle of sufficient energy, will absorb this energy momentarily before re-emitting it in the form of photons. The types of interactions that may occur in the LXe scintillator include photo-electric interactions in which the photon's total energy is deposited, and Compton scattering in which a fraction of the photon's energy is deposited and a lower energy photon is emitted [5]. These photon interactions are then position-resolved and matched with the interaction corresponding to the other photon of the pair generated during annihilation. From the location of the first interaction a Line of Response (LOR) to the position in the patient's body where the initial annihilation occurred can then be determined [5].

LXe PET scanners do not currently exist physically. Currently all PET scanners use inorganic crystals, not LXe, in their detector volume. Using LXe instead of inorganic crystals has many benefits, as shown in Table 1.1 [2].

	Crystals	LXe
Spatial Resolution	4mm	1mm
Time Resolution	6ns	1ns
Cost per volume	\$50 per cc	\$3 per cc

Table 1.1: Performance differences between inorganic crystals currently used in commercially available PET scanners versus LXe. Adapted from [5].

1.2 Problem and Objectives

The objective of this project was to design an ML model to reconstruct photon emission events within an LXe PET scanner module. For the scope of this project, the model does not predict where the initial annihilation occurred inside of the patient, and focuses only on a single module of the PET scanner.

To model the device, it was necessary to use simulated data representing the system. The team at TRIUMF provided us with large truth datasets with which we could train our model. The model is trained on a subset of this large set of data, and evaluated on another subset of the data to test its performance. Model training is based off of simulated light collected on each active wall from particle interactions inside the volume. In a physical LXe scanner, the light intensity would be recorded by silicon photomultipliers (SiPM) sensors, which would be arranged in pixel-like arrays on the walls of the module.

Note that the module design used in this project is a simplified version of the design used in an actual scanner. Our simplified module is a cube where each wall is covered with SiPMs, assuming 100% detection rate.

The simulation includes the following data for a single event:

- The true number and energy deposited of interactions.
- The true location of each interaction (x, y, z values), with which a true BB can be calculated (truth data).
- The SiPM measured wall intensity arrays (input data).

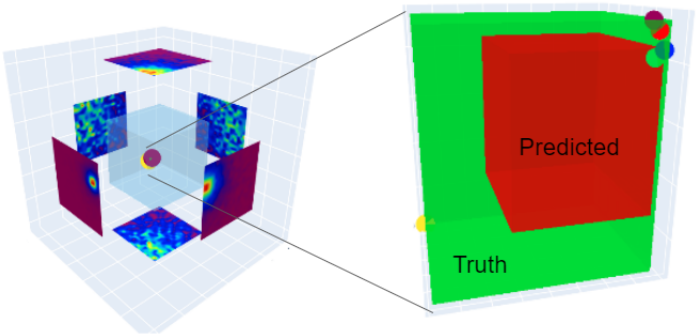


Figure 1.2: Layout of the simulation data with an example of predicted data with the simulated module. In the left side of the image we see a translucent blue box that represents the LXe module. The 6 heatmaps offset from the box represent recordings from the SiPM arrays on each wall. A hotter colour corresponds to a higher intensity of recorded photons by that SiPM. The dots in the module correspond to all of the interactions that occurred for a single event. The right side of the image shows an exploded view of the interactions. The green truth box is the BB that contains each interaction (5 for this event), while the red box is predicted by the model. Note that an event corresponds to a single gamma ray entering the LXe module.

Initially it was planned to model this simplistic setup of an LXe module, and once this was performing adequately we would attempt to refine the model for more realistic constraints. An adequately performing model was defined by the sponsors as:

- For any events in the dataset, there is less than 1cm^3 of incorrectly predicted volume.

This proved to be a substantial challenge. Our best performing model was not able to achieve this, predicting greater than 1 cm^3 of incorrect volume for more than 60% of events.

2 Discussion

2.1 Theory: ML

The underlying concept of a neural network is to learn a mapping from some input x to some output y; most often a very complex relationship that is highly non-linear. This is achieved by feeding the input through a series of parameterized functions, and iteratively updating the parameters proportional to the error they induce when the network makes a prediction.

An important ML concept is a loss function. This is a function that compares the output predicted by the network to the corresponding truth value, and is evaluated every time an input is passed through the network during training. Training is the process of minimizing the loss function. An example of a commonly used loss function is:

$$(\vec{x}_{truth} - \vec{x}_{predicted})^2$$

2.1.1 Multi-layer Perceptron (MLP)

An MLP is the simplest implementation of a neural network. They have at least 3 layers: a single input layer, multiple hidden layers, and a single output layer. Each layer is composed of "neurons", each representing a single computation within the network. Each neuron takes as its input a weighted linear combination of other neurons, and feeds this through a non-linear function. It is the aforementioned weights that are changed to facilitate the 'learning' process. The weights are changed with a gradient descent performed on the model's Loss Function.

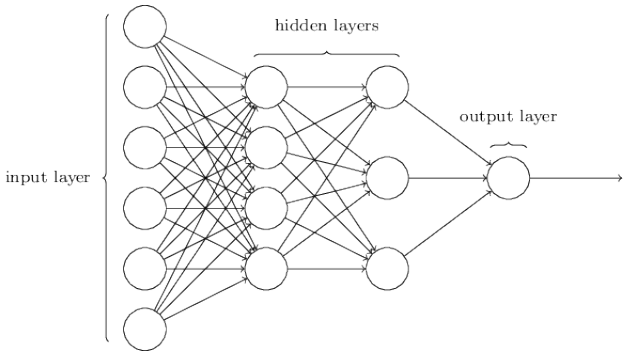


Figure 2.1: Architecture of an MLP.
Image: https://github.com/rcassani/mlp-example

A variant of an MLP is an Engineered Features Network (EFN). For this type of network, the inputs are significantly pre-processed based on features that the programmer believes should lead the model to the truth values. For an example, in trying to determine where a particular interaction occurs based on the light intensity values from the 6 SiPM walls, one could use features such as the coordinates of the maximum light intensity on each wall.

2.1.2 Convolutional Neural Network (CNN)

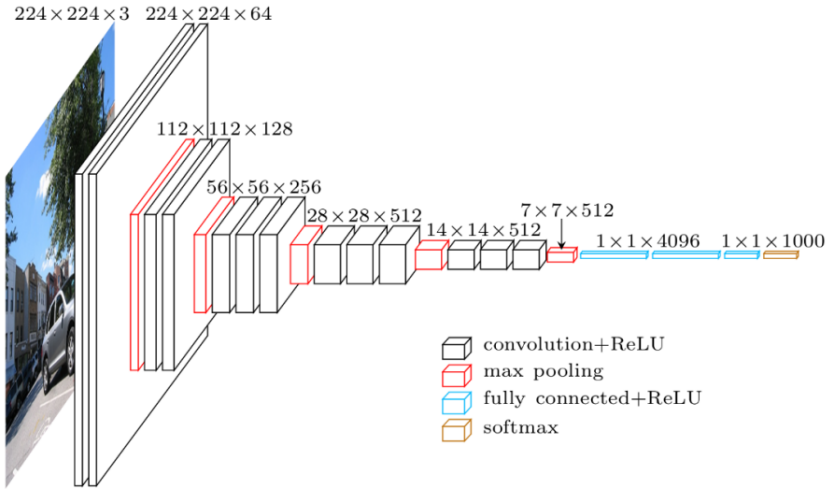


Figure 2.2: This image shows how the dimensionality of an input is reduced as it propagates through a CNN [4]

CNNs are a type of neural network designed to effectively learn representations for image data. In contrast to MLPs, CNNs use image-filter convolutions before applying a non-linearity in each neuron. A convolution refers to an operation across a matrix of input data (often pixels in an image), as shown in Figure 2.3.

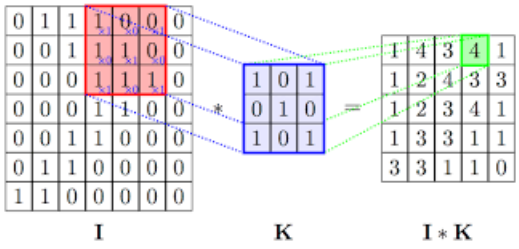


Figure 2.3: Convolution operation in a CNN with filter K and inputted data I. The pixel value at each index is multiplied with an intensity value at the overlapping filter index and summed with all other overlapping indices, before the filter is shifted and this operation is repeated [3]. The filter contains the values which are being updated in each training iteration.

2.1.3 Residual Neural Network (ResNet)

A ResNet is a variant of a CNN. As CNNs were being explored by ML researchers, an issue with increasing the number of layers became apparent: training becomes extremely slow. This is due to the gradient being back-propagated to earlier layers, where repeated multiplication may make the gradient infinitesimally small. This leads to the weights being updated by tiny values, therefore not changing the model performance significantly between passes of data.

To remedy this problem, the idea of a skip connection as shown in Figure 2.4 was introduced. Adding these skip connections has been proven eliminate the vanishing gradients problem while improving rate of training significantly.

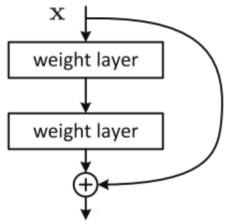


Figure 2.4: Skip connection used in ResNet architecture. We can see that the output of this diagram would be x plus (x after being pushed through two weight layers).
Image: https://towardsdatascience.com/an-overview-of-ResNet-and-its-variants-5281e2f56035

2.2 Dataset Validation

An important step in this project was verifying the simulated data provided by TRIUMF. This meant confirming that the data resembles what would be obtained from the real system very closely. If there are significant discrepancies then the model would not perform well on physical data.

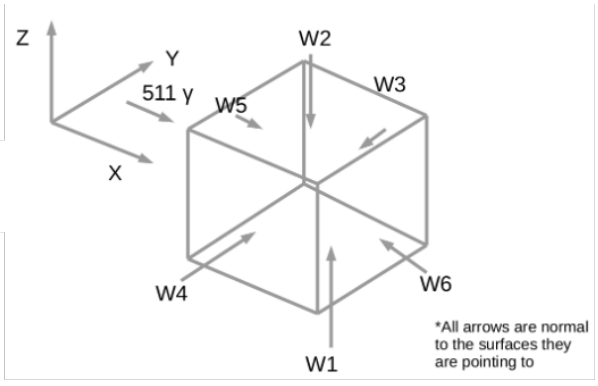


Figure 2.5: Physical layout of module walls. Note that the incident gamma photon is shot from the origin.

The original simulation provided to us consisted of a gamma ray point-source located a distance away from a box with walls covered in SiPM arrays. For each event, there are tuples of x, y, and z coordinates as the truth values for each interaction location, as well as 6 *20x20* arrays that represent the SiPMs intensity on each wall of the box. An important assumption that this simulation makes is that each SiPM can record multiple 'hits', and that each possible 'hit' on an SiPM is recorded with absolute certainty. In reality, an SiPM can only record a single event; once it has been 'hit' it can not record any more information for some time. Also, a real SiPM only registers a hit with a 5% chance. The group at TRIUMF has confirmed with us that each of these assumptions should not result in an incorrect model.

In order to verify the validity of the dataset, we completed a set of descriptive statistics. The largest issue we found with the dataset was that the particle point source was too close (1 cm) to the box. This resulted in a non-uniform (heavily-peaked in the centre) distribution of interactions in the Y-Z plane. This would have resulted in a model which would have poorly predicted interactions near the edges of the box in y and z. The module was moved 35 cm in the x direction to correct for this.

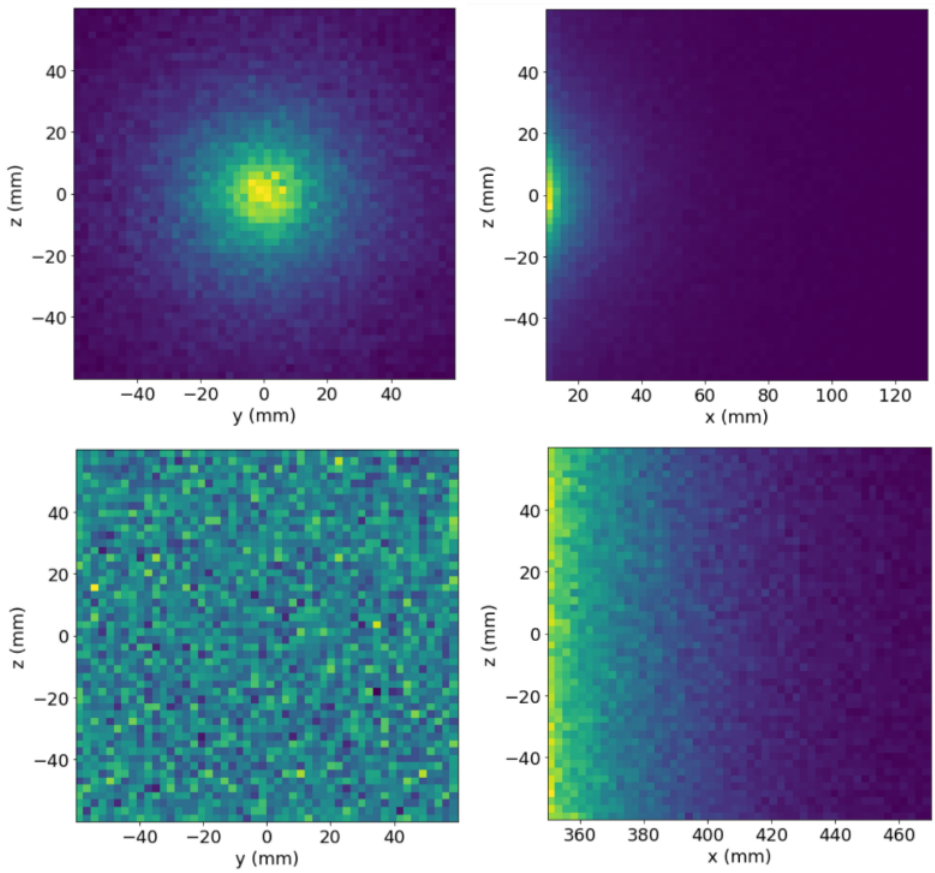


Figure 2.6: Distributions of the first interactions for the datasets before (top 2 plots) and after (bottom) the simulation was changed to move the particle source further away from the box. The x vs y plots are virtually identical to the x vs z and therefore are not shown.

Another issue found with the dataset involved the arrays of SiPM intensities. After looking at different wall heatmaps manually, we observed that the entries were rolled (i.e. the last column of the array should have been the first). This was easily fixed in the simulation and brought with it a small improvement in model performance.

The following plots were important in determining validity of the simulated dataset.

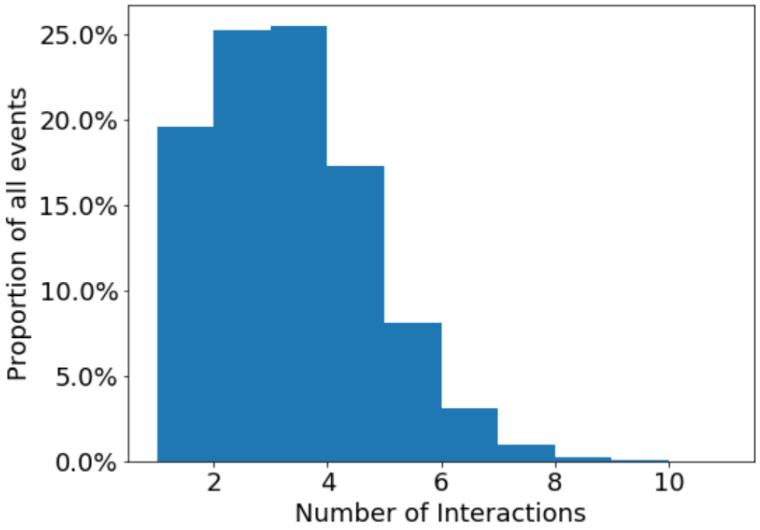


Figure 2.7: Distributions of number of interactions for each event from the simulated data. We see that about 90% of events have less than 5 interactions. The values in this figure deviate slightly from those given in the Wilson thesis, stating that exactly 1, 2, 3, 4, or $\geq$5 interactions occur with 22%, 36%, 19%, 7%, and 6%, respectively [5]. The team at TRIUMF has confirmed that the interaction data we have obtained is acceptable.

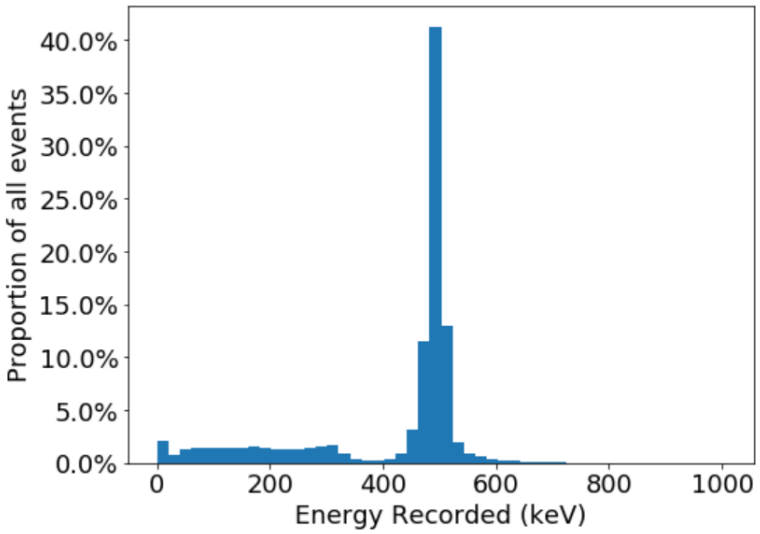


Figure 2.8: Distributions of total energy deposited inside of the module per event. Note that each event corresponds to a single 511 keV photon entering the box, so the large peak around that point makes sense. The lower deposited energy values correspond to events in which particles escape from the box. The events with deposited energy values greater than 511 keV are due to nuclear de-excitation processes that are simulated in Geant4. The events which have greater than 511 keV are due to a second gamma photon always produced in the simulation which shoots in the direction opposite of the module. If scintillation light from the interactions of that photon happen to reach the module then they would be recorded. We have confirmed with the team at TRIUMF that this should not cause issues while training the model.

Figure 2.9 is included to show the differences between SiPM intensity heatmaps for different event types. Notice that we see correlation between intensity loci on adjacent and parallel walls as expected physically.

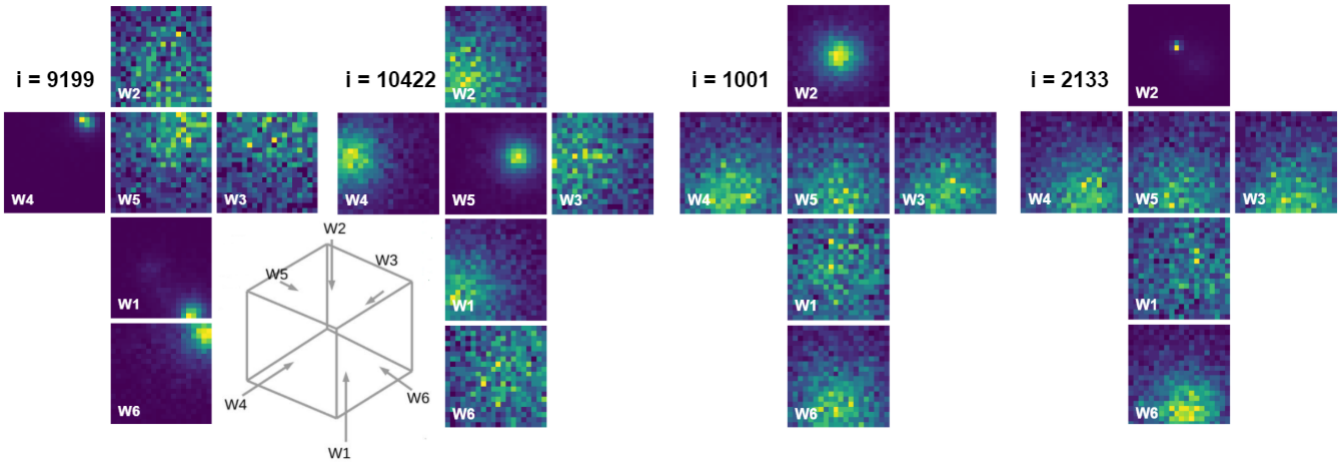


Figure 2.9: SiPM intensity heatmaps for 4 different events. Diagram of wall layout included for readability.
i = 9199: 7 interactions, 261 keV deposited 4mm from wall 6.
i = 10422: 1 interaction where 476 keV is deposited.
i = 1001: 2 interactions, very close to each other, located centrally in the module creating very spread out patterns.
i = 2133: 3 interactions, including 185 keV deposited 2 mm from wall 2, giving the heavily peaked distribution. In this event there was also a 287 keV deposit that occurs more centrally that is shown much less prominently than the lower energy interaction.

2.3 Dataset Filtering

In order to evaluate the performance of the model for different types of events (based on number of interactions, total energy deposited in the volume, etc.) it was necessary to filter the provided datasets. The main dataset consisted of about 1.5 million events. It was determined that the filters in Figure 2.10 should be applied:

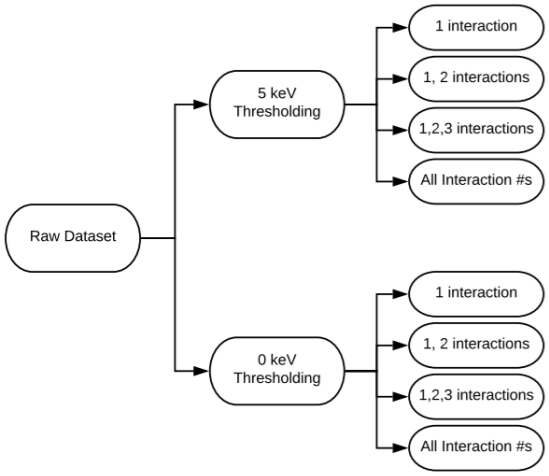


Figure 2.10: Visualization of the filters applied to the provided dataset. Thresholding at 0 keV or 5 keV corresponds to removing interactions which have deposits less than 0 keV or 5 keV.

An important finding from this filtering of the datasets was another bug in the provided dataset. It was found that about 13% of all events included interactions that released exactly 0 keV of energy. This was found to be another bug with the simulation. We are unsure on the the origin of this bug and also its impact on how a model trained with it would perform if these interactions were included in data given to the model. It does not make sense to test the model on these events as they release no photons and are therefore undetectable by the model. Thresholding 0 keV corresponds to removing these interactions with exactly 0 keV. Also note that 5 keV was the

other threshold that was chosen to evaluate based on the performance of SiPM sensors that may be used for the real LXe module. These sensors can only detect particles of energy higher than 5 keV, and therefore it makes sense for the simulation not to detect them either.

Appendix 6.1 shows plots concerning the effects of thresholding at 0 keV or 5 keV to the dataset.

2.4 Metric Design

In order to evaluate the performance of different models it was vital to have different performance metrics. In this section these metrics will be defined.

1. Percentage Energy Contained: For each event, the sum of the energies of each interaction which takes place inside of the predicted BB, all divided by total energy deposited by the event. This is more informative than just using the percentage of interactions contained as higher energy interactions are much more important in position resolution.
2. Intersection over Truth (IoT): For each event, the volume of the intersection of the truth and predicted BBs divided by the total volume of the truth BB.
3. Extra Predicted Volume: The volume of the intersection of truth and predicted BB subtracted from the volume of predicted volume. This is a necessary statistic to ensure that the model does not simply predict a very large BB volume which would capture interactions blindly.
4. Centroid Distance: For each event, the distance between the geometric centers of the truth and predicted BBs.

2.5 Model Implementations

2.5.1 Corner-based Bounding Box Model

For the majority of this project we were using NN architectures that output values corresponding to a bounding-box in 3D space:

```
prediction = [x_max, x_min, y_max, y_min, z_max, z_min]
```

It was expected for the model to perform decently just using a simple MLP. This was not the case as per Table 2.1.

Metric	5 keV Threshold	0 keV Threshold
Avg. Loss	62 mm^2	59 mm^2
Avg. IoT	12.2%	3.56%
Avg. Energy Contained	9.9%	2.2%
Avg. Centroid Distance	8.5 mm	7.7 mm

Table 2.1: MLP performance following 2 months of time spent refining the model and verifying the dataset. Less than 10% of energy contained on average is not acceptable.

Note that Table 2.1 uses data produced from our MLP implementation. At the same stage of the project, a ResNet gave slightly better but still unacceptable performance, and an EFN gave worse performance as described in Appendix 6.3.

The results given from prior versions of the model which did not include small changes in the dataset and other less significant code updates were all worse performing. This team was under the impression that the poor performance was a result of either a bug in the software, a systematic issue with our implementation, or a fundamental issue with the data or simulation. As we had completed a large set of dataset verification tasks, we prioritized looking into the model and its code.

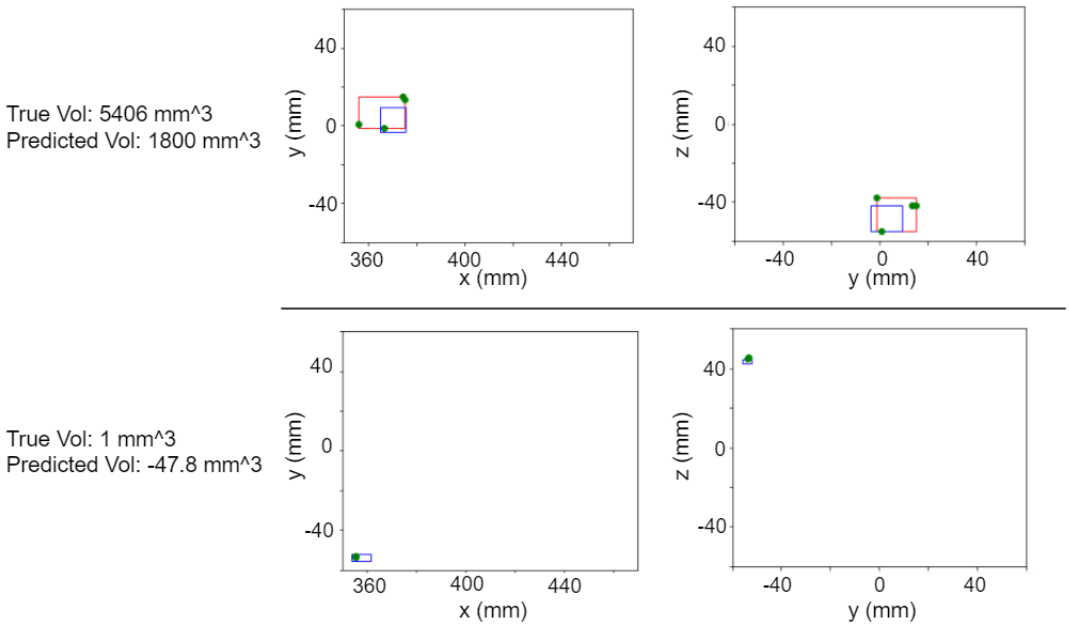


Figure 2.11: Comparison of truth and predicted BBs for two separate events (one on top, one on bottom) on different projections. Note that for the event shown on the bottom two plots, the predicted volume is negative. This was the most significant issue with the corners based model. It was found that 59% of events had negative predicted volumes, and of these, 97.5% are within -100 to 0 mm^3.

The most significant issue found was that the model was predicting BBs with negative volumes (see Figure 2.11). A negative volume for a predicted BB is a result of the outputs of the model being in an incorrect order (i.e `prediction[0]` < `prediction[1]` even though `prediction[0]` should be `x_max`). When this occurs, the loss and most other metrics are meaningless.

Solving this problem was not as easy as post-processing the output and swapping values if `x_max` < `x_min` (same for `y`, `z`). All changes that occur outside of the model training block are not propagated through the system and therefore no learning occurs. If we were to first ensure proper order of outputs before Loss was calculated just by swapping values explicitly, then the iterations loss value may be smaller, however the model would not learn anything related to the orderings of the output entries. Therefore the overall loss would remain high as it does not learn to output `x_max` before `x_min`.

In order to solve this problem, we employed two approaches:

1. A network that outputs centroid-based values (i.e. `[x_center, y_center, z_center, x_length, y_length, z_length]`, where centroid values corresponds to the geometric centroid of the box, and the length values correspond to lengths of each box dimension). This was a quick change to the code, but unfortunately did not result in any significant performance improvement (see Appendix 6.4 for performance), and therefore we focused on the following approach:
2. A network that represents a discretized version of a module, where the prediction corresponds to which voxels (cubes which comprise the discretized module) an interaction occurred in. This will be referred to as the Discretized Volume Model.

2.5.2 Discretized Volume Model

This approach was fundamentally different from previous ones. It had been discussed in planning stages to try this, but this team as well as the sponsors were weary of the performance issues coming from having an output vector with so many unknowns (if each box has 5 mm side lengths then there are a total of 13824 individual voxels). This model's output vector is 13824 elements, where each element corresponds to the model's prediction on if a corresponding voxel contained an interaction or not.

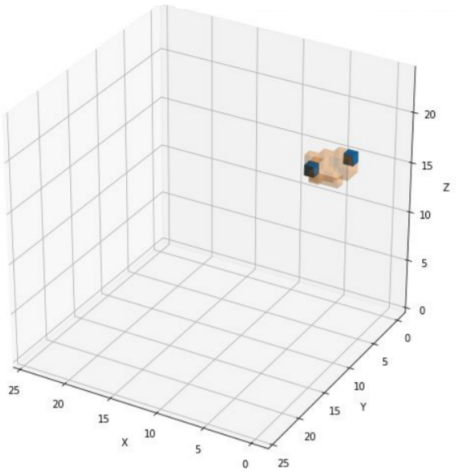


Figure 2.12: Visualization of Discretized Volume output. The blue boxes are the truth values of the interactions that occurred for this event, and the translucent orange is the bounding volume predicted by the network.

Figure 2.12 shows how the Discretized Volume model would predict a bounding volume. Information on the loss function chosen for the Discretized Volume Network can be found in Appendix 6.6.

2.5.3 LXeNET

The final neural network architecture we implemented for this project is based on an 18 layer ResNet *(ResNet-18)*, which we call the *"backbone"* of our model. We call our novel architecture LXeNET, and in this section, will detail implementation specifics as well as provide motivation for certain design decisions.

Network Input

The training data for our task was given as 6 *20x20* images, each representing photon detection counts across an array of SiPM detectors on a wall of the LXe chamber. For clarity, we will continue refer to these images as *heatmaps* in this section.

We observed that as an event or sequence of events got closer to a wall, the spread of detected photons would become more focused around a central point. As these events got farther away, the spread of detected photons becomes more noisy, with no obvious centroid. For two opposing walls, this means that one wall is likely to encode a better in-plane representation of events that are closer to it, as shown in Figure 2.13. For this reason, and due to the fact that opposing walls encode the same spatial information in-plane, we decided to feed the 6 heatmaps into our model as 3 opposing wall SiPM image pairs, where each pair was simply a 2-channel image.

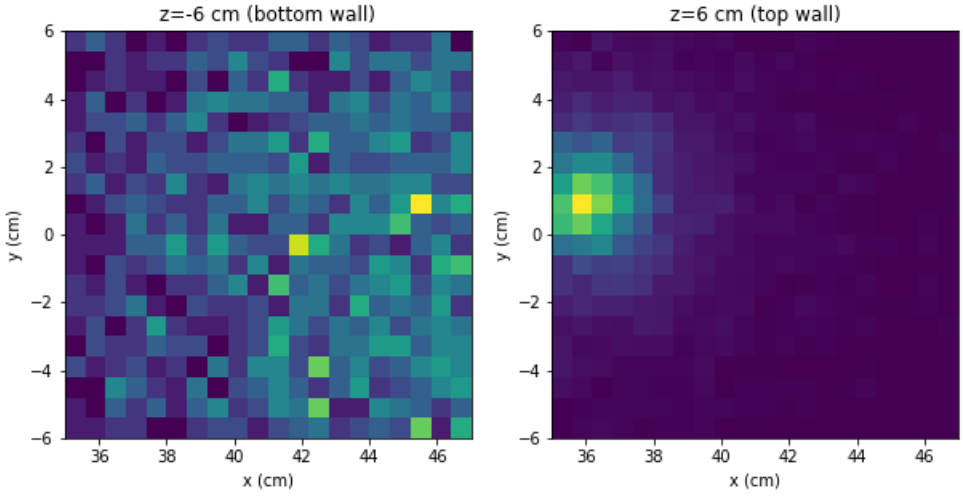


Figure 2.13: heatmaps for a typical opposing wall pair. The events in this wall pair are evidently closer to the wall shown on the right (top wall located at `z=6cm`). The centroid of the events for these heatmaps is located at `(x=36.08 cm, y=0.96 cm, z=4.41 cm)`.

ResNet Backbone

As mentioned above, the ResNet backbone we implemented is based on ResNet-18. The traditional ResNet-18 architecture consists of an input block, 4 residual blocks, and a fully-connected output block, as shown in Figure 2.14.

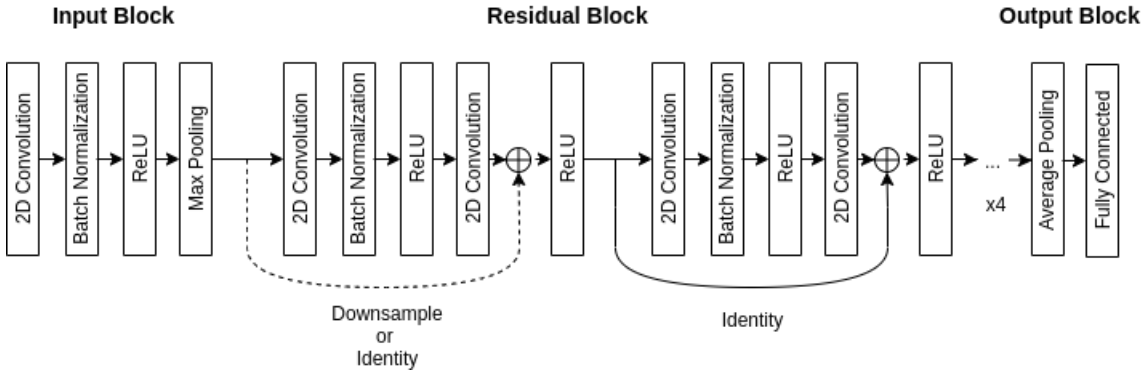


Figure 2.14: ResNet-18 architecture diagram

Since traditional ResNets are meant to operate on image data with resolutions upwards of *200x200* pixels, the input block is meant to quickly reduce dimensionality, while adding representational shift invariance. For our purposes, we would like to eliminate shift invariance and maintain dimensionality for as long as possible, since the input resolution is only *20x20* pixels and we are interested in reconstructing accurate event locations. For these reasons, our ResNet adaptation has the input block removed entirely.

Similarly, the adaptive average pooling layer in the output block of ResNet-18 is included to flatten the output of the final residual block in a way that preserves prominent high-level features (i.e. it also adds shift invariance). We wanted to remove the shift invariant property of this layer, while still flattening the output of the final residual block. So, instead of removing it entirely, we replaced it with a *squeeze MLP*.

The squeeze MLP is intended to reduce a two-dimensional input to a one-dimensional output via a convex combination of its components, in order to retain as much spatial information as possible. This is done across all channels in the residual block output, so that the input to the fully connected layer is a one-dimensional vector, which can be seen in Figure 2.15. The final result of these modifications is the LXeNET backbone, as shown in Figure 2.16.

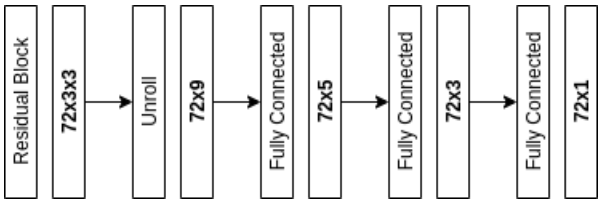


Figure 2.15: Squeeze MLP architecture. For our implementation the output of the final residual block is *72x3x3* (72 feature channels, 3x3 pixels per channel)

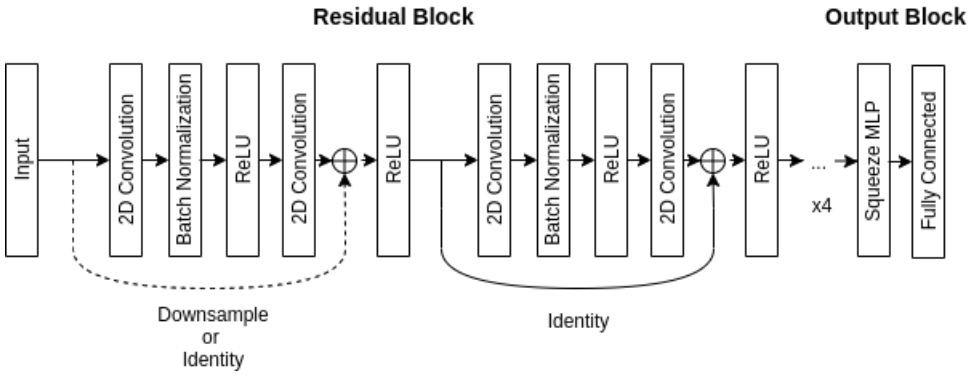


Figure 2.16: LXeNET backbone architecture

Overall Architecture

The inputs to our network can be considered as orthographic projections of events in three-dimensional space onto three two-dimensional spatial planes (`xy,xz,yz`). This means that the weights in our network should be learning a one-dimensional representation that is optimized over all three spatial planes.

For this reason, we feed each pair of opposing heatmaps into the backbone separately and concatenate the outputs. In order to obtain a meaningful prediction, the concatenated backbone outputs must be fed through a final MLP that can be adjusted to the required output dimensions. This gives the LXeNET architecture, as shown in Figure 2.17.

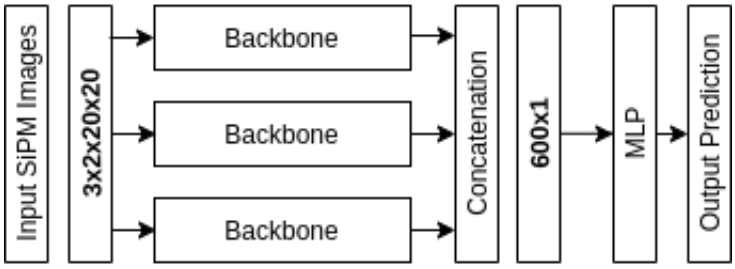


Figure 2.17: LXeNET architecture

2.6 Results

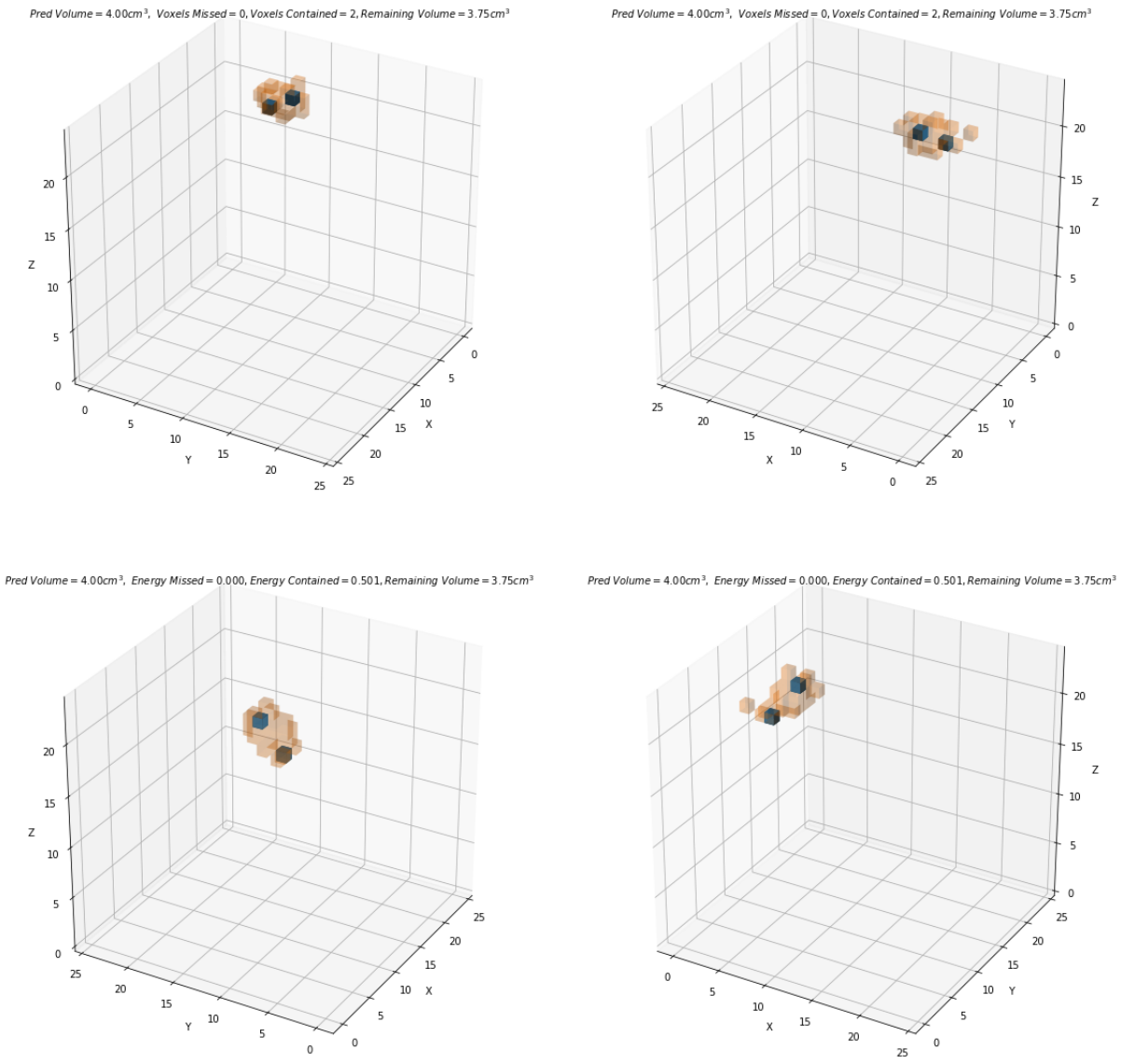


Figure 2.18: Event display showing the predicted and true voxels for a 4 Wall ResNet model. The orange shaded voxels represent the predicted output and blue voxels the true points of interaction. Each voxel in the above images are cubes of side length 5mm.

	ResNet 6 Walls 5keV	ResNet 6 Walls 0keV
Avg. Predicted Volume (cm^3)	1.71	1.86
Avg. Percent Energy Contained (%)	81.5	53.5
Avg. Percent Voxels Contained (%)	79.5	53.0
Avg. X Centroid Residual (mm)	0.58	3.04
Avg. Y Centroid Residual (mm)	0.027	0.74
Avg. Z Centroid Residual (mm)	0.66	0.56
Avg. Number of True Voxels ($frac.\ voxel$)	1.79	1.79
Avg. Voxels Missed ($frac.\ voxel$)	0.504	0.965
Avg. Voxels Contained ($frac.\ voxel$)	1.29	0.824

Table 2.2: The results shown are for the ResNet 6 Walls approach. 5 keV - trained and tested on data where interactions <5keV are elimanted. 0 keV - trained and tested on data with no interaction eliminations.

Table 2.2 shows differences in performance for the ResNet model trained and tested for 5 keV and 0 keV thresholded data. The model trained on 5 keV outperforms the 0 keV model by almost 30% for energy contained. Issues relating to the simulation are likely. The 0 keV bug was recently traced to a problem in data recording where electron energy loss through first step of propagation through LXe is not recorded in truth information. This implies an effective cut of 30 keV on total energy deposited from a compton scatter electron. The bug in the simulation was brought to our attention on the day of report submission and so effects on data splitting and model results are unclear and could not be investigated further. Figure 2.19 shows plots for each model comparing the number of true voxels as a function of the predicted volume. We notice the 2 wall distribution is more spread along the predicted volume than the 4 and 6 wall cases. The histogram did not reveal any clear biases in the models predicted volume, since the marginal distribution of the predicted volume is the same shape for different number of true voxels, only shifted right.

	ResNet 6 Walls 5keV	ResNet 4 Walls 5keV	ResNet 2 Walls 5keV	MLP 6 Walls 5keV
Avg. Predicted Volume (cm^3)	1.71	1.59	3.09	1.45
Avg. Percent Energy Contained (%)	81.53	79.2	50.0	72.9
Avg. Percent Voxels Contained (%)	79.5	77.3	48.3	70.7
Avg. X Centroid Residual (mm)	1.35	1.35	2.32	1.43
Avg. Y Centroid Residual (mm)	2.26	2.19	3.83	2.50
Avg. Z Centroid Residual (mm)	2.60	2.60	7.31	2.62
Avg. Number of True Voxels ($frac.\ voxel$)	1.79	1.79	1.79	1.789
Avg. Voxels Missed ($frac.\ voxel$)	0.504	0.556	1.01	0.699
Avg. Voxels Contained ($frac.\ voxel$)	1.29	1.23	0.780	1.09

Table 2.3: The results from four seperate approaches are tabulated above. The results are from the a test set thresholded at 5 keV.

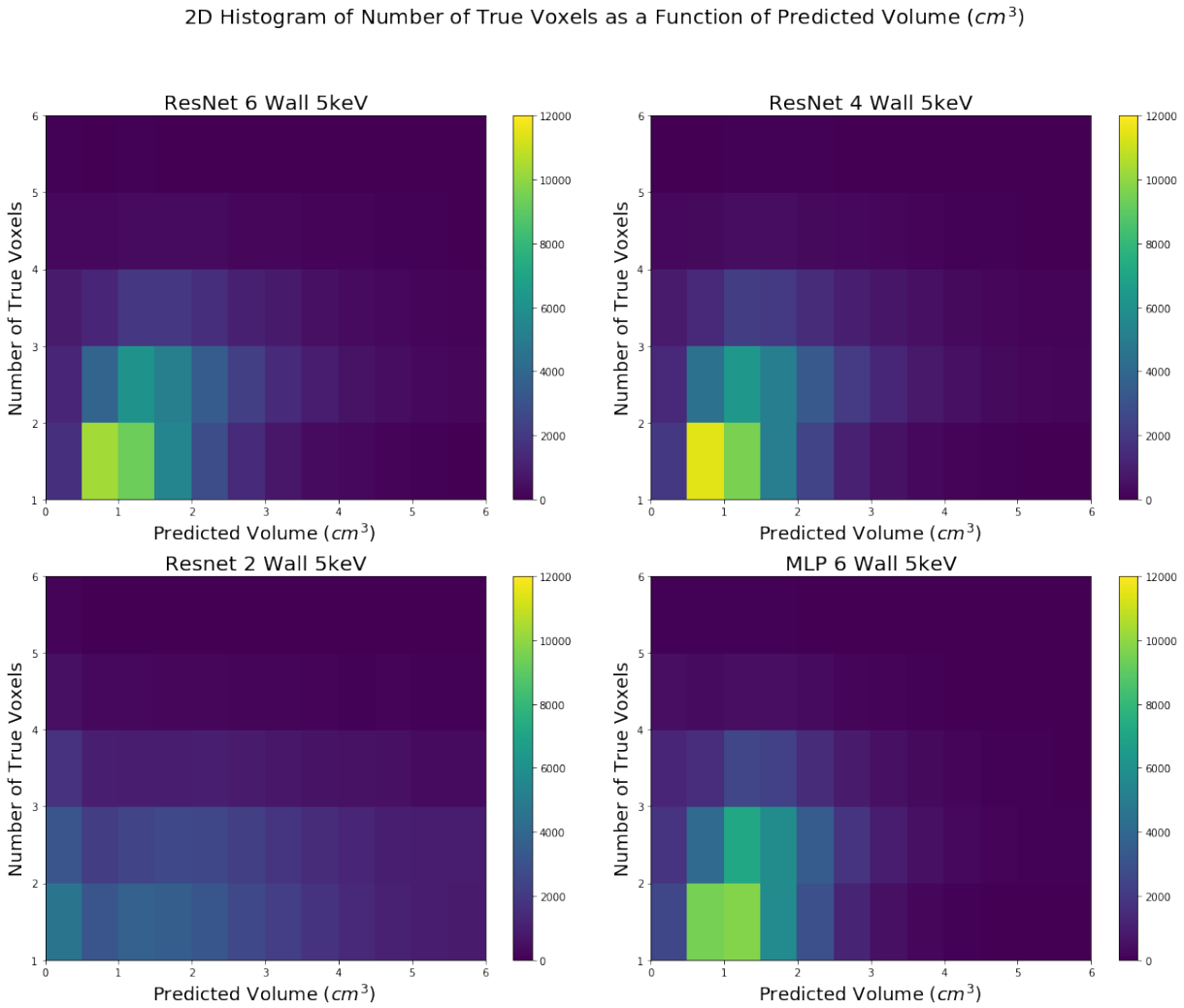


Figure 2.19: Comparison of four different models using 5 kev test set to compare number of ground truth voxels as a function of the predicted volume.

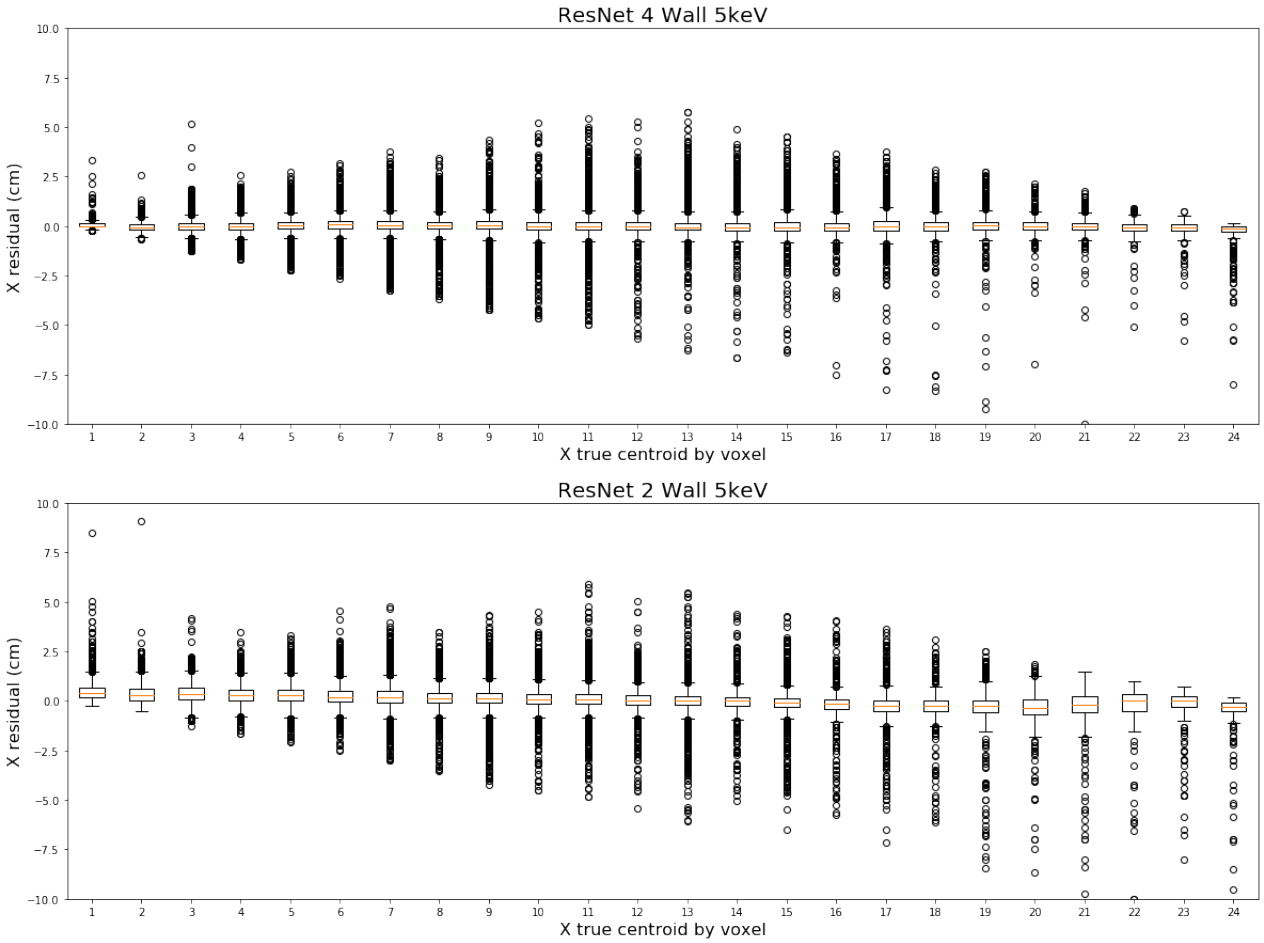


Figure 2.20: Box plot of the residual voxel count between the x centroid of the predicted outputs as a function of the true centroid of the each voxel along x for the 4 and 2 wall ResNet 5 keV

Following the model performance given in Table 2.3, 5 keV was used as the baseline test and train set. Model performance was then compared for the ResNet 6 Walls (ResNet trained on 6 SiPM walls), ResNet 4 Walls (trained on walls 1, 2, 3, 4 as per Figure 2.5), ResNet2 Walls (trained on walls 1, 2) and MLP 6 Walls. There were no significant drops in performance between the 6 and 4 wall case. There is a noticeable difference in performance for the ResNet 2 wall model. We observe a substantial discrepancy in average percent energy and voxels contained. We also note a large average z centroid residual relative to the other models. Since W1 and W2 are orthogonal to the z-axis (and therefore parallel), the model receives limited information on the z location of an interaction point. This uncertainty likely leads to predicted volumes extruded along z, causing a large residual and increasing predicted volume size.

	ResNet 6 Walls 5keV All	ResNet 6 Walls 5keV N1	ResNet 6 Walls 5keV N1-2	ResNet 6 Walls 5keV N1-3
Number of Events (N)	72505	3029	16799	33670
Avg. Predicted Volume (cm^3)	1.71	1.41	1.52	1.69
Avg. Percent Energy Contained (%)	81.53	98.5	94.8	88.5
Avg. Percent Voxels Contained (%)	79.46	98.5	94.0	86.8
Avg. X Centroid Residual (mm)	1.35	0.753	0.87	1.08
Avg. Y Centroid Residual (mm)	2.26	1.59	1.67	1.88
Avg. Z Centroid Residual (mm)	2.60	1.85	1.85	2.03
Avg. Number of Voxels ($frac.\ voxel$)	1.789	1.0	1.24	1.56
Avg. Voxels Missed ($frac.\ voxel$)	0.504	0.015	0.108	0.28
Avg. Voxels Contained ($frac.\ voxel$)	1.285	0.985	1.14	1.28

Table 2.4: The results shown were computed using the ResNet 6 Wall approach. 5 keV All - All test data with 5 keV interactions thresholded. 5 keV N1 - Only events with 1 interaction and with 5 keV interactions thresholded. N1-2 - indicates only events with 1 or 2 interaction, etc.

	ResNet 4 Walls 5keV All	ResNet 4 Walls 5keV N1	ResNet 4 Walls 5keV N1-2	ResNet 4 Walls 5keV N1-3
Number of Events (N)	72505	3029	16799	33670
Avg. Predicted Volume (cm^3)	1.59	1.34	1.45	1.59
Avg. Percent Energy Contained (%)	79.3	98.4	93.0	86.7
Avg. Percent Voxels Contained (%)	77.26	98.4	93.2	85.0
Avg. X Centroid Residual (mm)	1.35	0.771	0.878	1.07
Avg. Y Centroid Residual (mm)	2.19	1.55	1.66	1.85
Avg. Z Centroid Residual (mm)	2.60	1.69	1.71	1.92
Avg. Number of Voxels ($frac.\ voxel$)	1.79	1.00	1.24	1.56
Avg. Voxels Missed ($frac.\ voxel$)	0.556	0.015	0.125	0.319
Avg. Voxels Contained ($frac.\ voxel$)	1.23	0.984	1.12	1.2

Table 2.5: The results shown were computed using the ResNet 4 Wall approach. 5 keV All - All test data with 5 keV interactions thresholded. 5 keV N1 - Only events with 1 interaction and with 5 keV interactions thresholded. N1-2 - indicates only events with 1 or 2 interaction, etc.

	ResNet 2 Walls 5keV All	ResNet 2 Walls 5keV N1	ResNet 2 Walls 5keV N1-2	ResNet 2 Walls 5keV N1-3
Number of Events (N)	72505	3029	16799	33670
Avg. Predicted Volume (cm^3)	3.09	3.56	3.50	3.44
Avg. Percent Energy Contained (%)	50.0	66.8	63.3	58.0
Avg. Percent Voxels Contained (%)	48.3	66.8	62.6	56.4
Avg. X Centroid Residual (mm)	2.32	1.81	1.93	2.14
Avg. Y Centroid Residual (mm)	3.83	3.25	3.28	3.44
Avg. Z Centroid Residual (mm)	7.31	6.59	6.11	5.83
Avg. Number of Voxels ($frac.\ voxel$)	1.79	1.0	1.24	1.56
Avg. Voxels Missed ($frac.\ voxel$)	1.01	0.332	0.494	0.744
Avg. Voxels Contained ($frac.\ voxel$)	0.780	0.668	0.750	0.818

Table 2.6: The results shown were compute using the ResNet 2 Wall approach. 5 keV All - All test data with 5 keV interactions thresholded. 5 keV N1 - Only events with 1 interaction and with 5 keV interactions thresholded. N1-2 - indicates only events with 1 or 2 interaction, etc.

Comparisons were done for each wall variation across four different test splits (note all models were still trained using the 5 kev split). Information on the splitting can be found in Figure 2.10 and Table 6.1 in the appendix. Note the difference in N values is a result of filtering for full energy containment. The tables revealed no unusual biases in performance for events of type 1, 1 to 2, or 1 to 3 interactions. Performance improved as the number of interactions were reduced. An increased number of interactions (especially when distanced further apart) make prediction more challenging for the model, since information deposited on the walls becomes more noisy. Note that the number of interactions does not correspond to the number of true voxels, since two or more interactions could take place within one 5 mm^3 voxel.

	ResNet 4 Walls 5keV All R=0.34	ResNet 4 Walls 5keV All R=0.35	ResNet 4 Walls 5keV All R=0.36	ResNet 4 Walls 5keV All R=0.37
Number of Events (N)	72505	72505	72505	72505
Avg. Predicted Volume (cm^3)	1.91	1.59	1.31	1.06
Avg. Percent Energy Contained (%)	82.2	79.3	75.9	72.3
Avg. Percent Voxels Contained (%)	80.2	77.3	73.9	70.3
Avg. X Centroid Residual(mm)	1.32	1.35	1.36	1.36
Avg. Y Centroid Residual(mm)	2.16	2.19	2.21	2.21
Avg. Z Centroid Residual (mm)	2.24	2.60	2.60	2.24
Avg. Number of Voxels ($frac.\ voxel$)	1.79	1.79	1.79	1.79
Avg. Voxels Missed ($frac.\ voxel$)	0.490	0.556	0.626	0.701
Avg. Voxels Contained ($frac.\ voxel$)	1.30	1.23	1.16	1.09

Table 2.7: The results shown are for ResNet 4 Wall 5keV for four different R parameter values.

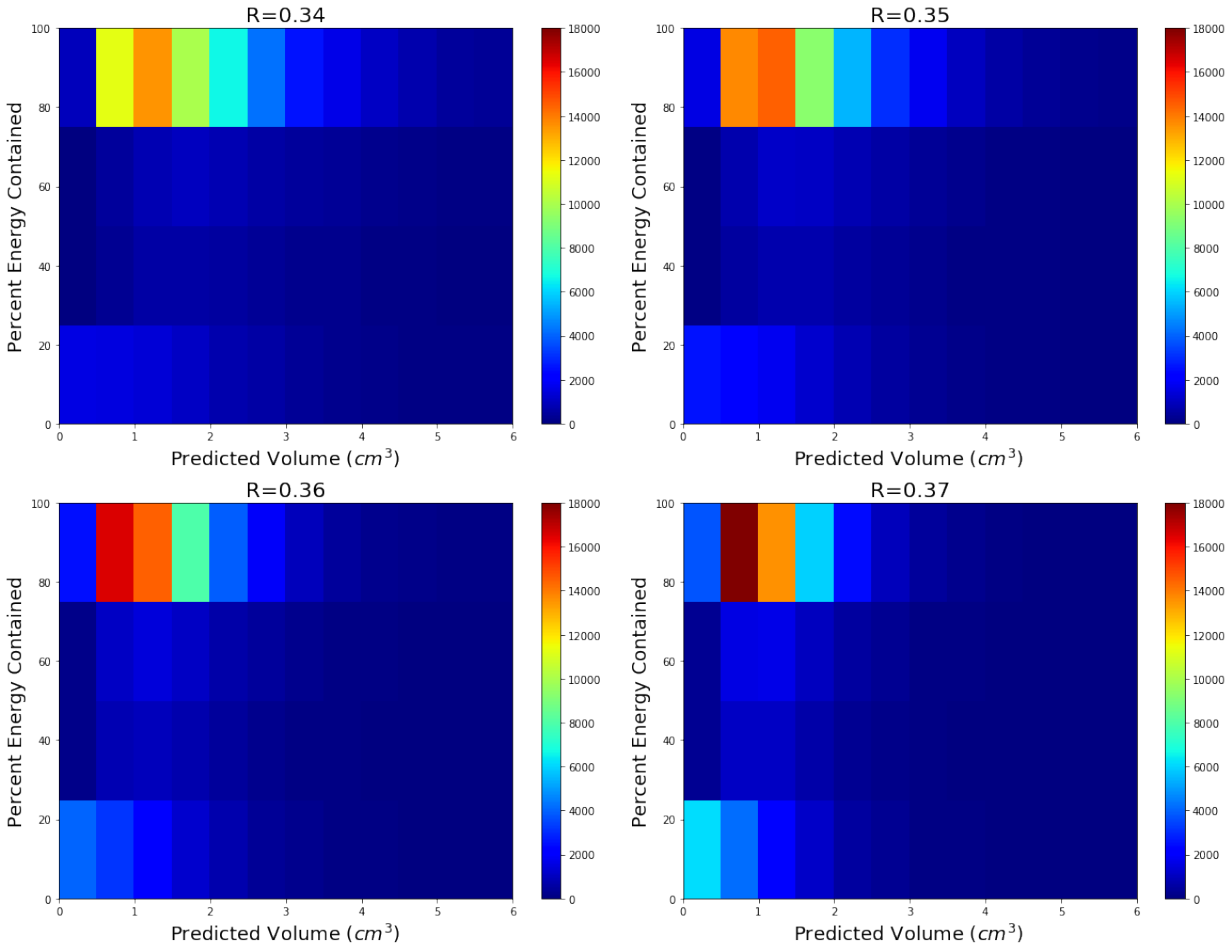


Figure 2.21: Comparison for threshold parameter R using 4 wall 5keV ResNet model. Plots show 2D histograms plotted with percent energy contained as a function of predicted volume to give an idea of ROC.

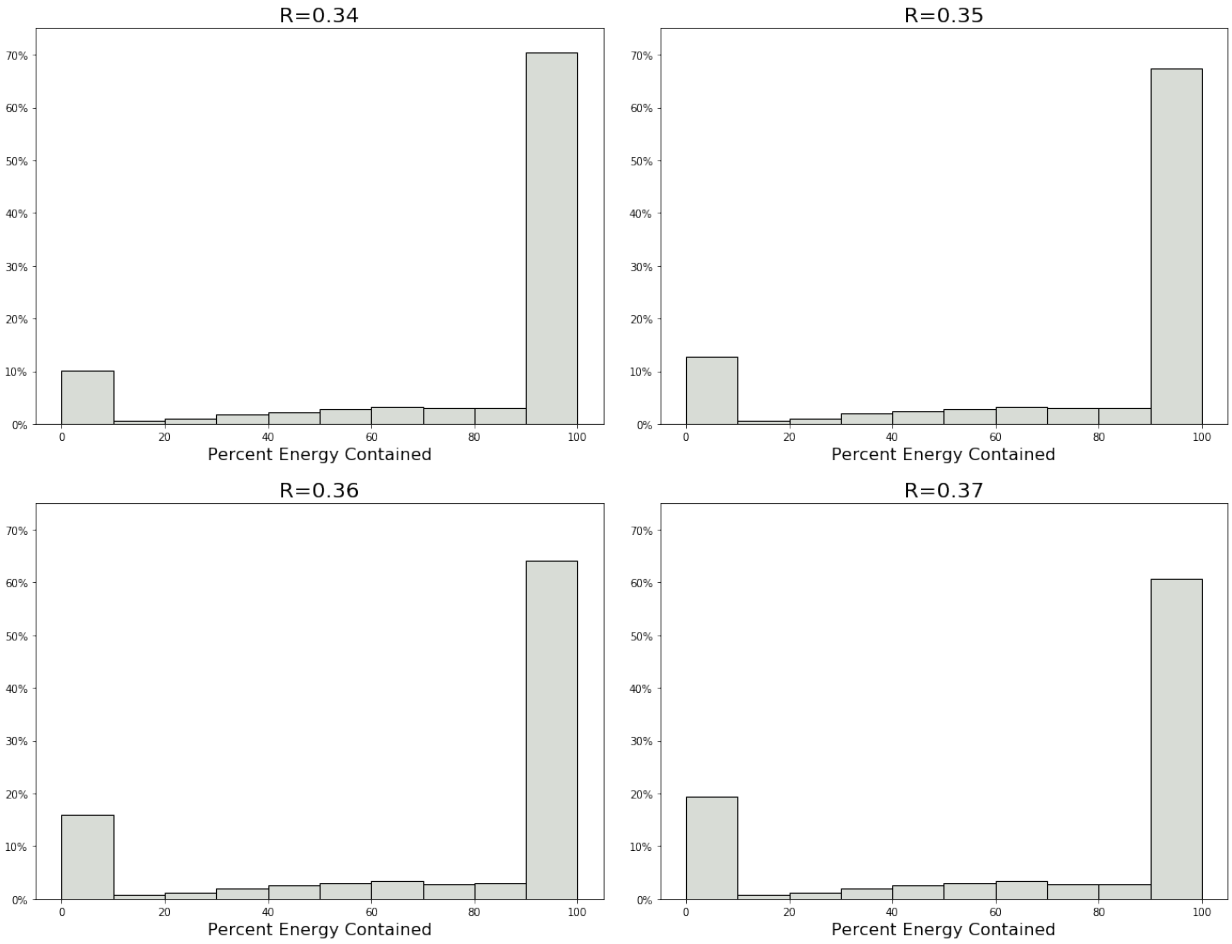


Figure 2.22: Histogram of the energy contained by the predicted voxels for the 4-wall ResNet model on the 5 keV thresholded test set across 4 different R parameter values.

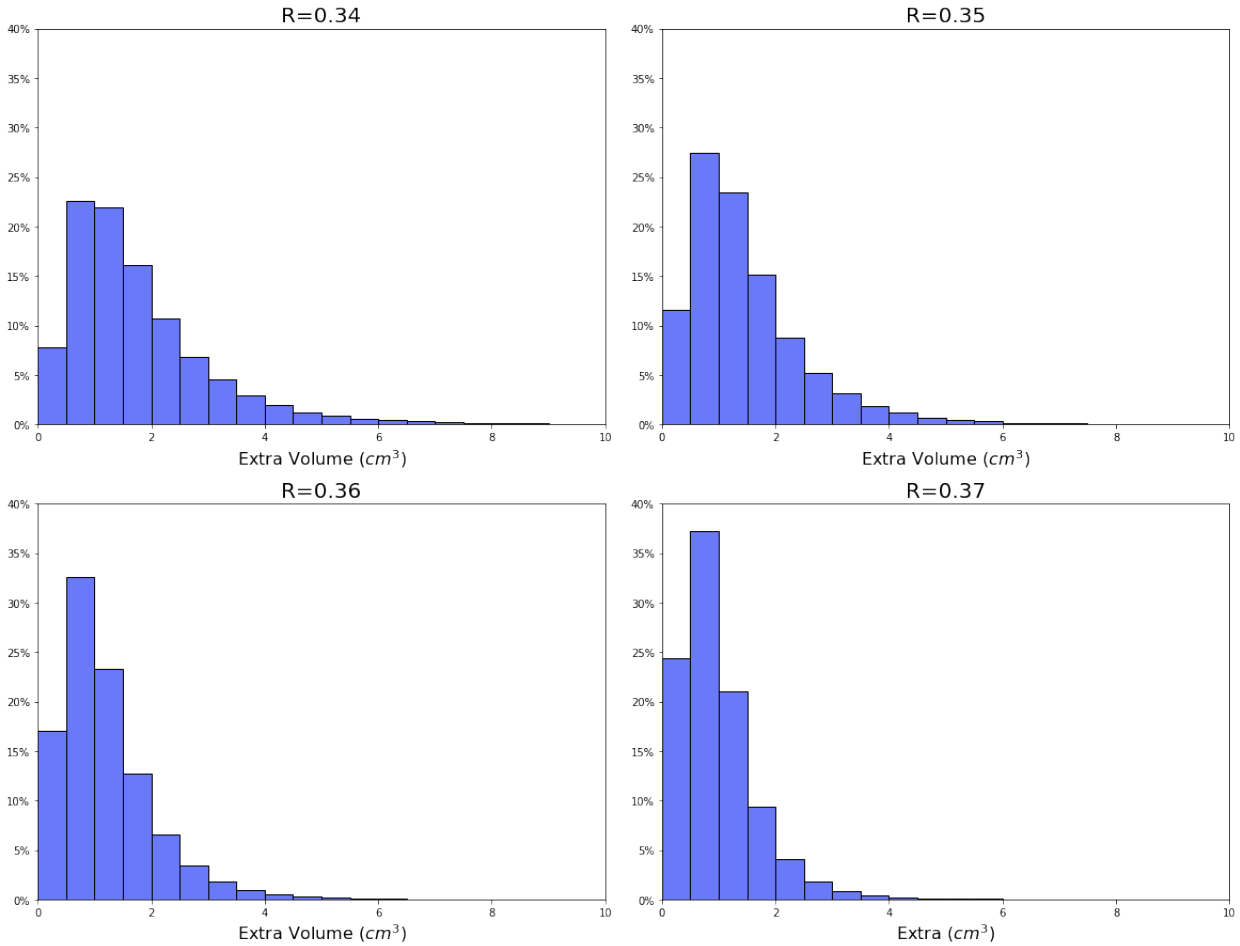


Figure 2.23: Histogram of the total remaining volume of the voxels output from the 4-wall ResNet model on the 5 keV thresholded test set across 4 different R parameter values. The remaining volume is just the predicted volume minus the true voxel volume.

The output of the discretization approach is a voxel-wise probability distribution, where each voxel is independent of the other and assigned a value between 0 and 1 for the probability that the particular voxel contains any interactions. This discrete probability distribution must then be converted into a binary output. This was done be setting a threshold parameter (R) that set each voxel to 0 or 1 depending on if the value was greater or less than the parameter R. R was set to 0.35 as a baseline. The value was determined by comparing the specificity (true positives) and sensitivity (false positives) of the model using Table 2.7 and Figures 2.21, 2.22 and 2.23. The energy contained is a function of the true positive voxels and so represents the specificity of the model whereas the remaining volume is a function of the false positive voxels and so represents the sensitivity of the model. Plots were made for a suite of R values (R=0.34, 0.35, 0.36, 0.37), to determine the optimal value to select based on balancing the specificity (energy contained) with the sensitivty (remaining volume) of the model. R of 0.35 was chosen since it predicted 80% energy contained and $1.5cm3$, which fit best within the 1cm$cm3$ of extra volume objective of the project.

3 Conclusions

LXe PET is a promising new technology that will hopefully offer faster and higher resolution imaging over the current inorganic crystal methods. Our results show that deep learning techniques have a relevant application in the reconstruction of photon scattering events within the LXe chamber; the first steps in being able to reconstruct gamma emissions from a patient's body. We found that intensity maps obtained by placing *20x20* SiPM arrays mounted to the LXe chamber walls could be passed through a neural network in order to reconstruct events to a

relatively high degree of accuracy.

We tested a variety of deep learning techniques, including a simple MLP, an MLP with feature-based input preprocessing, and our own novel ResNet-based LXeNET architecture. These experiments showed that LXeNET could achieve demonstrably higher performance than the other two architectures across all of our data baselines. This was especially true for predictions made over a voxelized detector volume, as opposed to simpler event bounding box predictions.

We found that an average of up to 81% of the energy deposited in the detector volume could be contained within predictions made by our novel LXeNET architecture; when data was used from all six input walls. Our results also show that the centroid of the voxelized volume predicted by the network has a minimum average residual distance of *0.1 cm* from the centroid of the voxelized truth volume. In combination with 28% of events over-predicting by only 1 cm^3, this indicates that the network is also fairly precise.

These metrics demonstrate the strong potential of our novel architecture when applied to the LXe event reconstruction task. Although these are promising initial results, there are areas of improvement we believe could be targeted in future iterations. In particular, we observe that the network over-predicts the volume of around 60% of events by over $1cm^3$ (the target resolution for this project). As well, the ideal situation for a development version of the LXe detector has only 2 detector walls with SiPM arrays—a configuration for which our network performs fairly poorly (57% average energy contained). Considering these points in conjunction with some strange effects including large zero percent energy contained peaks (Figure 2.22), and a potential correlation between event location and centroid residual distance (Figures 2.20), leads us to conclude that there is still a lot to explore and improve upon. Our model is far from perfect, but is a powerful starting point towards developing a model that is capable of reconstructing photon scattering events within an LXe chamber to a high degree of accuracy and precision.

4 Recommendations

Based on our findings, the success of the discrete volume predictions made by LXeNET lead us to believe that this combination should continue to be explored. We believe that clustering methods could potentially be used to more precisely threshold voxel predictions made by the network. This could possibly lead to a significant reduction in the excess volume predicted, and give a clearer picture of the network's centroid residual accuracy. As well, the correlations shown in the figures of Section 6.5 could be further explored to gain a better understanding of the network's behaviour.

Figures 2.20 point to possible reduced accuracy as the events get further away from the detector walls. They may also demonstrate a learned bias that may be reducing the networks accuracy in some edge cases; if so, these effects should be explored and quantified in order to eliminate them. Finally, the outlying peaks present in Figures 2.22 show that for a non-trivial percentage of—possibly similar—events, the network is unable to make any meaningful prediction. These events should be identified, characterized, and analyzed to flatten the 0%-contained peaks.

5 Deliverables

- Gitlab repository containing all code, `.npy` files and many figures from completed tests, and this report being left with TRIUMF team
- Necessary documentation required to continue working with the repository
- Meeting scheduled with TRIUMF team to ensure repository can be used following this team's departure from the project

6 Appendices

6.1 0 keV, 5 keV Thresholding

	0keV: Training	0keV: Testing	5keV: Training	5keV: Testing
Number Events	1319087	72508	1319032	72505
Mean Energy (keV)	426	425	426	425
Mean Number of Interactions	2.88	2.87	2.88	2.87
Median Number of Interactions	3	3	3	3
Average True Voxels	1.79	1.79	1.79	1.79

Table 6.1: Dataset split properties. Training sets are used to train the model on while testing sets are used to evaluate the model's performance once training is completed.

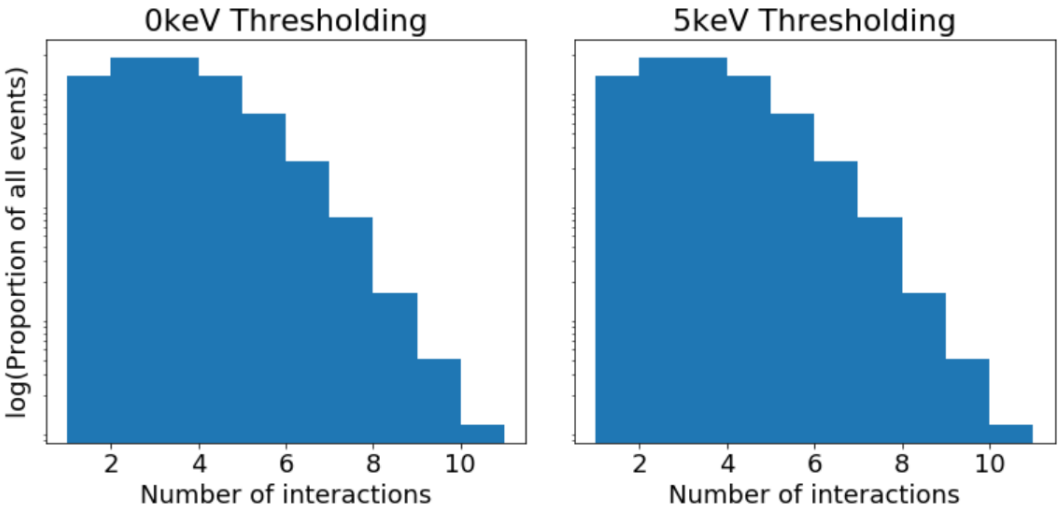


Figure 6.1: Comparison of distribution of the number of interactions between 0 keV and 5 keV thresholding. A log scale is used on the y axis to magnify any differences between the two, however nothing significant is visible.

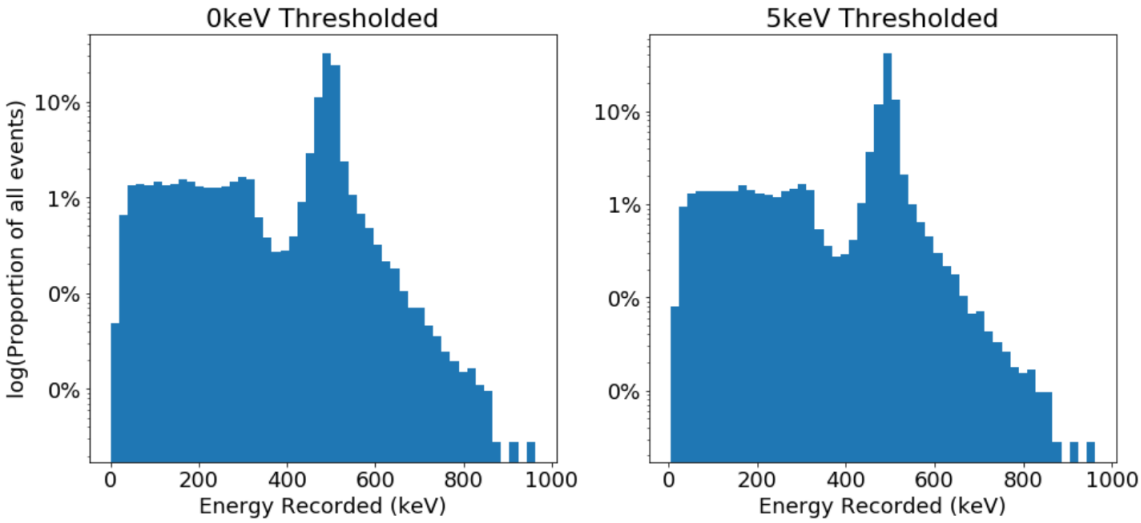


Figure 6.2: Comparison of distributions of total energy deposited inside of the module per event. A log scale is used on the y axis to magnify any differences between the two, however nothing significant is visible.

6.2 Bounding Box Method Results

When developing the architecture for LXeNET, we used the bounding box model (Section 2.5.1) as a baseline for design variants due to its simplicity, and faster training times. The following tables outline the differences in performance when comparing various LXeNET experiments.

Avg. True Volume (cm^3): 2.524	No Pooling Centroid Method	No Pooling Box Method
Avg. Predicted Volume (cm^3)	2.13	2.42
Avg. Percent Energy Contained (%)	12.7	22.4
Avg. Percent Interactions Contained (%)	11.9	21.3
Avg. X Centroid Residual (cm)	0.032	0.030
Avg. Y Centroid Residual (cm)	0.014	0.0050
Avg. Z Centroid Residual (cm)	0.078	0.10
Avg. Percent Intersection Over Truth (%)	18.5	33.4
Avg. Percent Intersection Over Union (%)	7.35	8.78

Table 6.2: Comparison of centroid and bounding box methods. The centroid method was tabled following these findings.

Avg. True Volume (cm^3): 2.524	Pooling No Thresholding	No Input Block No Thresholding	No Pooling No Thresholding
Avg. Predicted Volume (cm^3)	2.16	2.73	2.42
Avg. Percent Energy Contained (%)	10.3	8.74	22.4
Avg. Percent Interactions Contained (%)	10.3	8.72	21.3
Avg. X Centroid Residual (cm)	0.081	0.033	0.030
Avg. Y Centroid Residual (cm)	0.086	0.048	0.0050
Avg. Z Centroid Residual (cm)	0.067	0.12	0.10
Avg. Percent Intersection Over Truth (%)	19.9	18.2	33.4
Avg. Percent Intersection Over Union (%)	6.64	6.16	8.78

Table 6.3: Comparison of different pooling combinations. The "pooling" architecture included both max pooling and adaptive average pooling layers. The "no input block" architecture included an adaptive average pooling layer and had the input block (Section 2.5.3) removed. The "no pooling" architecture had the input block removed, and the adaptive average pooling layer replaced by a squeeze MLP (Section 2.5.3).

	No Pooling No Thresholding	No Pooling 0 keV	No Pooling 5 keV
Avg. Predicted Volume (cm^3)	2.73	2.42	2.20
Avg. True Volume (cm^3)	2.82	2.48	2.48
Avg. Percent Energy Contained (%)	16.2	19.4	25.6
Avg. Percent Interactions Contained (%)	15.8	18.9	24.6
Avg. X Centroid Residual (cm)	0.051	0.068	0.0024
Avg. Y Centroid Residual (cm)	0.0030	0.015	0.052
Avg. Z Centroid Residual (cm)	0.027	0.048	0.026
Avg. Percent Intersection Over Truth (%)	31.7	36.7	40.4
Avg. Percent Intersection Over Union (%)	11.7	12.8	11.8

Table 6.4: Comparison of different dataset splits. The results show that the network performs best when low energy interactions have been removed from the training and testing data. These networks were trained on larger datasets than the networks in the tables above.

6.3 Engineered Features Network

Property	Value
Average Loss (mm^2)	761
IoT (%)	0.036
Interactions Contained (%)	3.4
Extra Volume (mm^3)	0.914

Table 6.5: Performance of Engineered Features network. Note that we decided to stop using this model before we began doing any energy thresholding. We can see that the EFN predicts the overall volume of the BB quite well as per the low extra volume value, but that is it.

6.4 Centroid Based Network

Property	0keV Thresholding	5keV Thresholding
Avg Loss (mm^2)	2777	2635
Avg IoU (%)	0.6111	1.168
Avg IoT (%)	5.742	12.34
Avg Interactions Contained (%)	5.287	10.38
Avg Remaining Predicted Vol (%)	0.9415	0.9168
Avg Energy Contained (%)	5.1344	10.13
Avg Predicted Vol (cm^3)	4.29	7.93
Avg True Vol (cm^3)	2.6	2.6
Avg Extra Vol (cm^3)	0.431	1.28
Avg Centroid Distance (cm)	4.3	4.4

Table 6.6: Performance of Centroid Based network trained on an MLP.

6.5 Discretized Volume Net

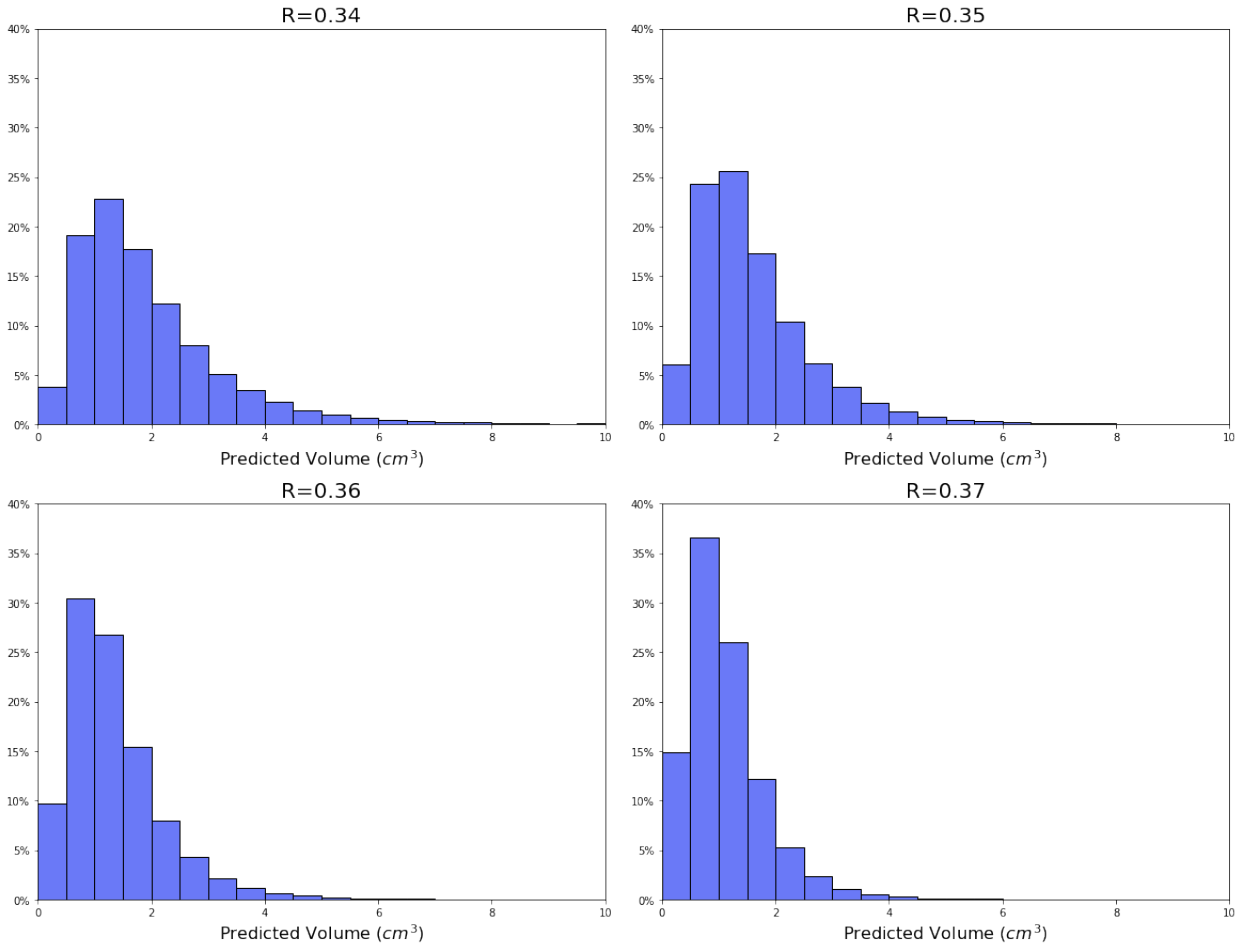


Figure 6.3: Histogram of the total predicted volume of the voxels output from the 4-wall ResNet model on the 5 keV thresholded test set.

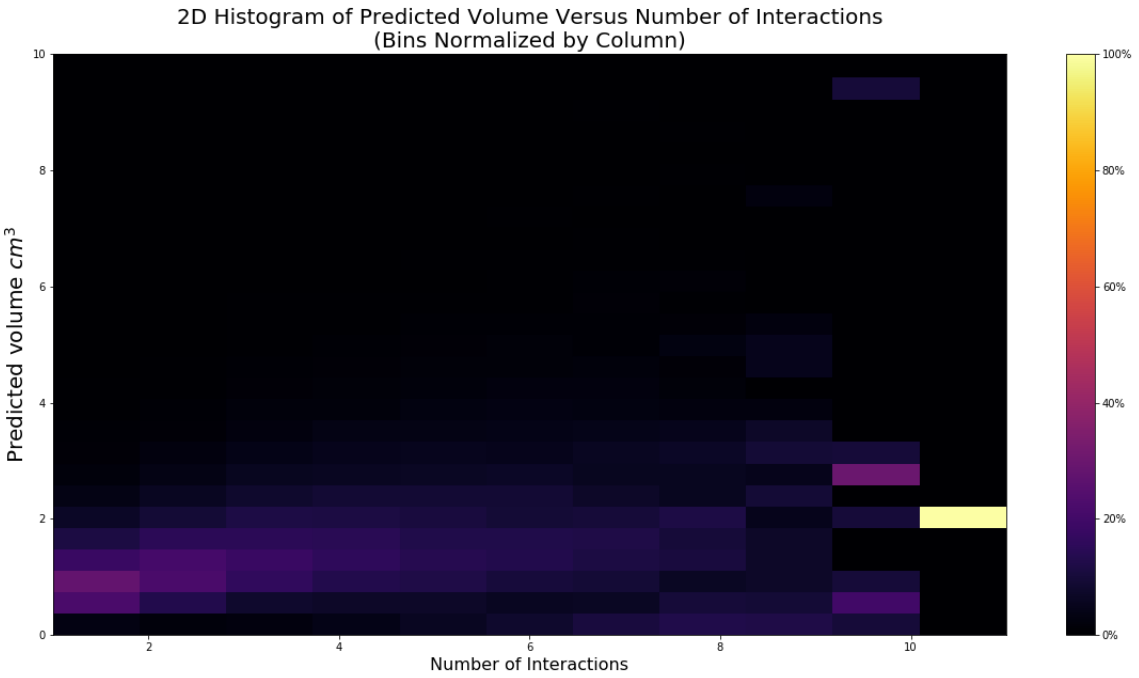


Figure 6.4: 2D histogram of predicted volume versus number of interactions from the 4-wall ResNet model on the 5keV thresholded test set.

6.6 Discretized Volume Net: Loss Function

The output of the discretized volume model is a vector of 13824 voxels, where each value is 1 if it has been predicted to contain an interaction and a 0 otherwise. On average, each event contains 1.79 true voxels of the 13824 total voxels, posing a large class imbalance. This is a ML term describing a problem in which there are many more instances of one type of class than another, in this problem the classes are:

- Voxel containing interaction
- Voxel not containing interaction

A class imbalance can often lead to a poorly performing model due to the fact that each class is penalized the same by traditional loss functions. This would allow the model to still have a low loss function just by predicting the majority class for everything. This is also bad because often the non-majority class is the most important classification task for the problem, as it is for this project. For class imbalance problems 'false positives', events where the majority class is classified as a minority class are less critical than a 'false negative', where a minority class is classified as a majority. The loss function should represent this importance.

Researchers at Facebook's AI division have developed a loss function which has proved extremely robust in class imbalance problems which they call Focal Loss [1] that is based around cross entropy loss, a common loss function used for binary classification tasks. Focal Loss is defined as:

$$FL(p_t) = -\alpha_t(1-p_t)^\gamma log(p_t), \text{ where } p_t = \begin{cases} p & y=1 \\ 1-p & y=0 \end{cases} \text{ and } \alpha_t = \begin{cases} \alpha & y=1 \\ 1-\alpha & y=0 \end{cases}$$

$y \in \{0,1\}$ corresponds to the class, $p \in [0,1]$ is the prediction of the model for a single value before being thresholded, α is a constant chosen to penalize values for class 0 or 1 differently, and γ is a tuning parameter which weights easily-classified examples less as γ is increased.

$FL(p_t)$ not only penalizes different classes differently to account for differences in number of occurrences (due to α_t), but also penalizes examples where there is more uncertainty about classifying it as the more important minority class $y = 1$ due to the $(1-p_t)^\gamma$ factor [1]. Focal loss proved to be very well suited to the discretized volume model.

6.7 Literature Review

Although not substantial, research has already been done into event reconstruction in LXe PET time projection chambers. The most relevant being an undergraduate thesis [5] done in 2008 under supervision of our sponsor Doug Bryman. This student used machine learning to localize the positions and energies of particles in LXe detectors for PET scanners. While not comprehensive, much of the methods used in the thesis are directly applicable to our project, and provides a foundation of information for us to build on.

This thesis used the same simulation software (Geant) to generate data, however used a very simple neural network 6-layer MLP architecture. In 2008 machine learning was not as developed as it currently is, which gives us access to strong research surrounding CNNs and ResNets.

References

[1] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[2] A Miceli, J Glister, A Andreyev, D Bryman, L Kurchaninov, P Lu, A Muennich, F Retiere, and V Sossi. Simulations of a micro-PET system based on liquid xenon. *Physics in Medicine and Biology*, 57(6):1685–1700, mar 2012.

[3] Arun Prakash. Machine learning - convolution for image processing, May 2018.

[4] David Shahid. Convolutional neural network, Feb 2019.

[5] Andrew Gordon Wilson. Position and energy reconstruction from scintillation light in a liquid xenon gamma ray detector designed for pet, 2008.