Exploring Deep Fully Connected Residual Neural Networks for Initial Energy Estimations of Compton Camera Based Prompt Gamma Imaging Data for Proton Radiotherapy

Michelle Ramsahoye

Mentors: Carlos A. Barajas, Matthias K. Gobbert, and Jerimy C. Polf

Spring 2022

Abstract

Proton radiotherapy uses a beam of protons to irradiate cancer tissue. It has been suggested that real-time imaging can be used to help optimize treatment delivery via prompt gamma ray image data collected with a Compton camera. When a prompt gamma deposits energy twice, it is called a "double" and physicists assume that it has deposited all of its energy after the second collision, thereby being absorbed and the total of the energy depositions is considered to be the initial energy of the prompt gamma ray. This initial energy is used during reconstruction to help determine the origin of the prompt gamma and leads to a well formed reconstruction but the assumption that a double deposits all of its energy is not always true leading to improper origins based on incorrect initial energies. Here, we present a deep residual fully connected regression neural network which can make estimations of the initial energy of double events on data generated by a Monte Carlo simulation using Compton camera detector effects. We conduct a hyperparameter search to explore different models. We then present and discuss the results of the currently best performing regression neural network model. We suggest further improvements that can help further reduce loss and improve model estimations for reconstructed images.

1 Introduction

Proton beams' primary advantage in cancer treatment as compared to other forms of radiation therapy, such as x-rays, is their finite range. The radiation delivered by the beam reaches its maximum, known as the Bragg peak, at the very end of the beam's range. Little to no radiation is delivered beyond this point. By exploiting the properties of the Bragg

peak, it is possible to only irradiate cancerous tissues, avoiding any damage to the healthy surrounding tissues [8]. However, without some way to image proton beams in real time, limitations exist in our ability to take full advantage of the dose delivery properties of the proton Bragg peak. This is due to uncertainties in the beam's position in the body relative to important organs that should not be irradiated.

The Compton camera is one method for real-time imaging, which works by detecting prompt gamma rays emitted along the path of the beam. By analyzing how prompt gamma rays scatter through the camera, it is possible to reconstruct their origin. However, the raw data that the Compton camera outputs does not explicitly record the sequential order of the interaction data, which represents scatterings of a single prompt gamma ray. Previous work done in [4] addresses approaches to reordering of scatter events and removal of false prompt gamma couplings. Compton camera outputs likewise cannot determine the initial energy for some types of multi-scatter events. Notably, the true doubles events must be estimated. In comparison to single and true triple events which follow straight-forward calculations described in Section 3.3, true doubles events include some uncertainty that cannot be accounted for.

We approach the determination of the true doubles initial energy problem by leveraging deep learning techniques. We use neural networks, which, in general, represent data transformations. Our network is trained by passing data through it, then updating it systematically so as to reduce the loss of its output. Doing this properly can create a model that exploits subtleties in the data which traditional models are unable to use [2]. We show how this can be done in the following sections. Additional discussion about the impact of the approach on the application area can be found in [7].

Our neural network output contains estimations of the initial energies for true double events. We are then able to take these estimations and use a reconstruction algorithm to create an image of the beam. By comparing the reconstructions of the neural network data to that of the original data, we are then able to compare image quality. We also discuss possible improvements to the network and to the data that may improve the quality of the neural network estimations.

The remainder of the paper is organized as follows. Section 2 discusses the biomedical application for this research. Section 3 describes a Compton camera and how we use it to collect data for our neural network. The network is then described in Section 4, where we discuss parts of the network's architecture. The model is tested on data in Section 5, where the resulting estimations are used for reconstruction. Section 6 makes general conclusions and proposes future directions for the research.

2 Proton Beam Therapy

To a first order approximation, the radiation dosage emitted by a proton beam is inversely proportional to the kinetic energy of the particles within the beam. Because the beam's particles lose kinetic energy as they traverse the patient, the amount of radiation delivered by the beam is low at its entry point, gradually rising until the beam nears the end of

its range, at which point the delivered dosage rapidly reaches its maximum. This point of maximum dosage is called the Bragg peak. Little to no radiation is delivered beyond the Bragg peak. These characteristics of proton beam therapy give it a distinct advantage over x-rays. Exploiting its finite range, medical practitioners can confine the radiation of the beam to solely areas affected by cancerous tumors. Vital organs beyond the tumor can be spared [8].

While the characteristics of proton beam therapy explained above would in principle greatly reduce the negative effects of radiation therapy, there are still practical limitations. In current practice, the patient's body is imaged before undergoing treatment in order to map the position of the tumor. Because proton beam therapy consists of multiple sessions over a period of one to five weeks, the relative size and position of the tumor within the patient's body may change as surrounding tissues swell, shrink, and shift as a response to radiation. Therefore, whenever using proton beams, a safety margin must be added to the position of the Bragg peak in order to fully irradiate the tumor. This rules out certain beam trajectories that would otherwise minimize damage to healthy tissue [8].

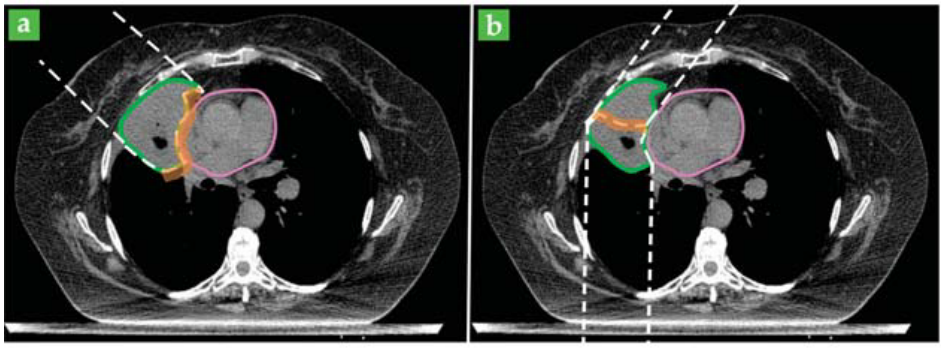


Figure 2.1: (a) Optimal proton beam trajectory. (b) Suboptimal trajectory necessary to protect heart.

Figure 2.1 compares two possible beam trajectories through a cross-section of the chest [8]. In this case, the heart, outlined in purple is positioned at the top-center of the figure and a tumor, outlined in green is located next to it. The optimal trajectory, shown in the left image, uses a single beam, which is represented as the space between the dashed white lines, to fully irradiate the tumor while stopping before reaching the heart. However, due to uncertainty in the exact location that the Bragg peak occurs (and the beam stops), a safety margin is added to the end of the optimal beam path to ensure the tumor always receives the prescribed dose even in the presence of day-to-day changes in patient setup and patient internal anatomy. This safety margin is represented in the figure as an orange strip at the end of the beam. This margin partially overlaps with the heart, which would mean a portion of the heart would receive the full treatment radiation dose. Since the heart is highly sensitive to radiation damage, it is very important that the radiation dose received by the heart is kept to a minimum. Therefore, in practice, the optimal trajectory is not used for treatment and instead the trajectory in the right image with two separate beams is used.

Using two beams reduces the dose to the heart, but is considered suboptimal as these beams result in more radiation being delivered to a larger amount of healthy lung tissue. However, although suboptimal, it is considered preferable for treatment, since the beams avoid the heart [8].

If we were able to provide real-time information on the proton beam as it passes through the patient during live treatment, then we could ensure it is covering the tumor as intended, and safety margins (used to ensure tumor coverage) could be smaller and, thus, the more optimal treatments could be used. For instance, with proper real-time monitoring of the proton beam delivery, the optimal single beam treatment shown in Figure 2.1 (a) could be used without delivering high radiation doses to the heart while minimizing radiation to the lungs.

3 Compton Camera Imaging

3.1 Introduction to the Compton Camera

In order to exploit the full advantages of proton therapy, many researchers are investigating methods to image the beam in real time as it passes through the patient's body [8]. One proposed method for real-time imaging is by detecting prompt gamma rays that are emitted along the path of the beam using a Compton camera.

As the proton beam enters the body, protons in the beam interact with atoms in the body, emitting prompt gamma rays. These prompt gamma rays exit the body, some of which enter the Compton camera. Modules within the Compton camera record interactions with energy levels above some trigger-threshold. These modules have a non-zero time-resolution during which all interactions are recorded as occurring simultaneously. For each interaction (that is, Compton scatter) an (x, y, z) location and the energy deposited are recorded. The collection of all interaction data that a camera module collects during a single readout cycle is referred to as an event [6].

In principle, it is possible to use the data that the Compton camera outputs (paired with a suitable reconstruction algorithm) to image the proton beam, however, this has been shown to only be feasible at low energy levels. At the higher energy levels more typical of proton beam therapy, reconstructions of the beam are far too noisy to be helpful. This is a result of two main limitations in how the Compton camera records events.

At the higher energy levels typically used in treatment, proton beams emit a larger number of prompt gamma rays per unit time, increasing the likelihood of false events. Also, prompt gamma rays are more likely to scatter at higher energy levels, leading to more multi-scatter events which will be unordered. These two effects greatly diminish the accuracy of Compton camera reconstructions at high energy levels, making them unusable as explained further in [6].

There are several prompt gamma image reconstructions which can be used in conjunction with Compton camera data, such as the shifted histogram method seen in [5], but they produce bad results and thus cannot be effectively used with raw Compton camera data.

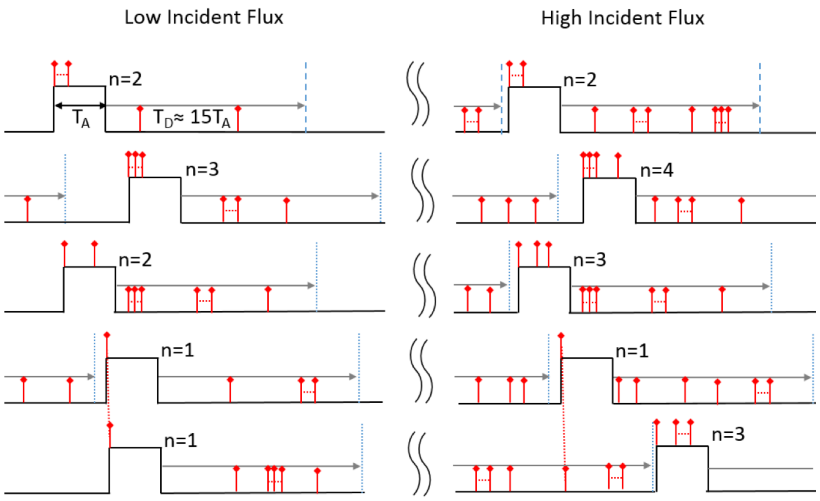


Figure 3.1: An illustration of events.

The algorithms have a base assumption that the data has no false events and no misorderings. The order of interactions is directly connected to the path the prompt gamma took from the origin point. If the order of the interactions is shuffled, then the origin point of the prompt gamma will change and no longer be representative of the proton beam. These algorithms assume that the data they are working on is mostly perfect and all events within are suitable for use. If the Compton camera data could be denoised in an accurate, fast, and systematic way, then the Compton camera would be a viable method for prompt gamma image reconstruction.

3.2 The Representation of Events

Multi-scatter events can be classified into five categories: False Triples, False Doubles, Double to Triple, True Triples, and True Doubles. A False Triple event consists of three interactions which all originate from separate prompt gamma rays that happened to enter the same module of the camera at the same time. These should be removed from the data before reconstruction. Similarly, False Double events contain two interactions originating from separate prompt gamma rays. These too should be removed. A Double to Triple event contains two interactions corresponding to the same prompt gamma ray, and one interaction from a different prompt gamma ray. The non-corresponding interaction should be removed before reconstruction. The two remaining categories of events are True Double and True Triple events, which, once properly ordered, can be used for reconstruction. Detecting the correct ordering for the interactions of true events is, by itself, a non-trivial task. The studies in this paper focus solely on True and False Double events. However, the general neural network architecture used for the Double events in this paper is also used on the other multi-scatter event categories [1].

Figure 3.1 shows a schematic of the Compton camera as it records events. The left side shows events produced at low energy levels and the right shows higher energy levels. Each

row represents an independent module of the camera. The red arrows represent scatters, with those originating from the same prompt gamma ray being connected by a dotted line. A raised pulse represents a single readout cycle within a module of length T_A. The value n is how many interactions occur during the readout cycle. Looking at just the left side, the first two rows show a True Double and True Triple event, respectively. The third row shows a False Double event consisting of two scatters originating from different prompt gamma rays. The fourth and fifth rows show two True Single events that consist of separate scatters by the same prompt gamma ray. The right side representing higher energy levels shows a far greater proportion of false events.

The raw data output but the Compton camera for each interaction is of the form (e_i, x_i, y_i, z_i), where for $i = 1, 2, 3$, e_i is the energy level, and x_i, y_i, z_i are the x-, y-, and z-coordinates respectively. It is important to note that for any event i could be 1, 2, or 3 as each event can have up to 3 interactions included. For events with less than 3 interactions, for the values of i that are not included in the event, the corresponding (e_i, x_i, y_i, z_i) would be left empty or as NaN.

3.3 Initial Energy for Multi-scatter Events

As demonstrated in Figure 3.2, the Compton camera can record deposited energies from individual interactions but cannot determine the origin of the gamma ray.

The determination of the initial energy for a multi-scatter event depends on the event type. For single scatter events, the initial energy calculation is straight-forward: the recorded energy listed is the initial energy for that event. For true triple scatter events, we have three interactions (e_i, x_i, y_i, z_i) for $i = 1, 2, 3$. By using the deposited energy from each interaction as well as the scattering angles, we are able to calculate the initial energy,

$$E_0 = \Delta E_1 + \frac{1}{2}\left(\Delta E_2 + \sqrt{\Delta E_2^2 + \frac{4\,\Delta E_2\, m_e c^2}{1 - \cos\theta_2}}\right). \tag{3.1}$$

For true double scatter events, it is assumed that the prompt gamma rays deposit all their energy and are absorbed after the second collision, but this is not always true. The energy could also be carried away by the gamma ray as it escapes the camera. This prompts the need for a way of estimating an accurate initial energy for the true double events so that the prompt gamma image reconstruction is as accurate as possible.

4 Deep Learning

In the context of our problem, there is no known equation that determines initial energy and also accounts for the energy that may by carried away by the prompt gamma ray within a true double scatter event. The benefit of deep learning and specifically, using a neural network, is its ability to recognize hidden patterns in raw data and continuously learn and improve in its assigned task.

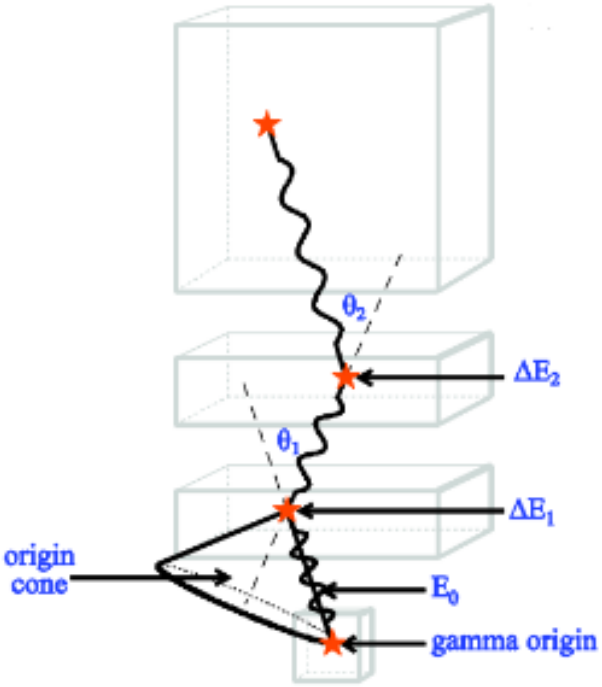


Figure 3.2: The origin cone detected by a three-stage CC. The scattering angles, θ_1 and θ_2, as well as the initial gamma energy, E_0, can be derived from the interaction positions and the two energies deposited, E_1 and E_2. The CC cannot determine the origin of the gamma ray, but because scattering angle θ_1 is known, the origin of the gamma ray is restricted to the surface of the origin cone, as described in [5].

In this work, we train a neural network to make an estimation of initial energy for true doubles events. The use of these estimations help with determining the origin of the prompt gamma and contribute to an accurate image reconstruction. This provides the ability to better visualize the path of the proton beam within the patient, with the ultimate goal to make real-time adjustments to patient treatment plans.

The network contains three main components: an input layer which accepts the data, hidden layers which each perform some transformation on the data, and an output layer which returns the transformed data in some prescribed format [2]. We would like to train the neural network to transform the provided data in some useful way. In the case of the simulated data output by the Compton camera, we would like the neural network to make an accurate prediction of the initial energy for True Doubles scatter events.

To improve the network's performance, it is typical to train the network on all available data multiple times. One pass through all the *training data*, or data isolated specifically for training the model, is referred to as an epoch. Often, the network will be trained for hundreds or thousands of epochs. It is standard practice to set aside some data with which to evaluate the network after each epoch. These data are called the *validation data*. By evaluating

the network at the end of each epoch, it is possible to plot how the network's performance improves over the training process, giving insight into whether or not the network has been fully trained. After the network has finished training, a final data set separate from the training data and validation data is used to test the network. This data set is referred to as the *test data*.

4.1 Fully Connected Residual Blocks

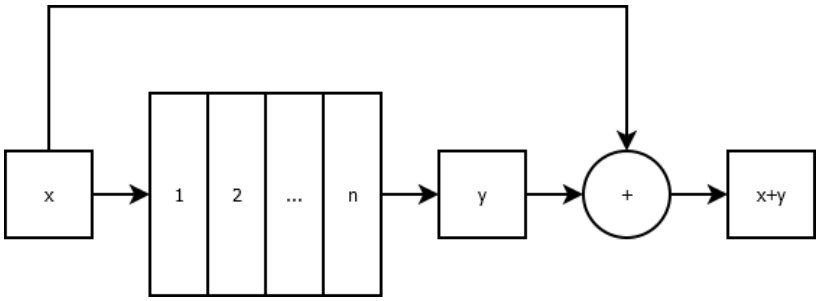


Figure 4.1: A fully connected residual block takes an input and passes it through n layers eventually adding it to the output of the n layers.

The network used in Section 5 is a deep fully connected neural network. Neural networks, especially fully connected ones, break down once they start becoming notably deep and complex. One of the first problems is that the values start to become very small during the forward propagation process. This leads to zeros and like-zero values becoming more prominent as you go deeper and deeper. A fix to this forward propagation issue is to opt to use Leaky ReLU over the traditional ReLU. However, the second problem occurs during back propagation. During back propagation we start to see the gradient becoming like-zero causing little to no update to existing weights which causes learning stagnation. This phenomenon is discussed more intimately in [3] where they detail these effects. The major breakthrough to this problem inspired by ideas in [3] where they create ResNet, a network built from "residual blocks". A visual representation of a residual block can be seen in Figure 4.1. Consider some record x. We pass it as an input to a small group of n layers with their own activators. The result of the layer digestion we can call y. Finally, we concatenate x and y. The concatenation in our case, and the case of the original ResNet, is addition. This addition operation helps push non-zero values through the forward propagation process which helps keep input data to each block fresh and non-zero. This also helps prevent vanishing gradients during the back propagation process. Residual blocks were originally used with convolutional layers whereas we are using fully connected layers.

4.2 Network Design

We used Tensorflow 2.5 with the bundled Keras module for our neural network backbone. The network design is built on the fully connected residual blocks described in Section 4.1.

The architecture starts with an input layer with 256 neurons. The hidden layers are comprised of 128 fully connected residual blocks. Each block is made up of 2 layers, each consisting of 256 neurons with Leaky ReLU serving as the inter-layer activation function for the hidden layers. This gives us 256 hidden layers in total. As this is a regression network, there is no activation function in the output layer to allow the network to freely make an estimation that can be any number. The network training process used Keras' provided Adam optimizer with Mean Squared Error (MSE) loss,

$$\frac{1}{n}\sum_{i=1}^{n}(Y_i - \tilde{Y}_i)^2. \tag{4.1}$$

MSE assesses the average of the squared difference between the ground truth values (Y_i) and model prediction values ($\tilde{Y}_i$). It is generally the most used error function with regression neural networks, and has the quality of weighing outlier predictions heavily.

5 Results

The effectiveness of a neural network is dependent both on the architecture of the network as well as the data provided to it. After training the neural network, we want to test how accurate the model is with estimating initial energy for true doubles events and how these estimations can influence the reconstruction process of the proton beam.

5.1 Data and Hardware Configuration

The training and testing of the neural network alongside the reconstruction process use the distributed-memory cluster taki in the UMBC High Performance Computing Facility (hpcf.umbc.edu). In particular, the networks were trained and tested on a hybrid CPU/GPU node with two NVIDIA K20 GPUs (2496 computational cores over 13 SMs, 4 GB onboard memory), two 8-core Intel E5-2650v2 Ivy Bridge CPUs (2.6 GHz clock speed, 20 MB L3 cache, 4 memory channels), and 64 GB of memory.

The neural network is trained on simulated true doubles data where there are no misorderings and there are no false events present. For our studies, we use 150MeV (Mega electron Volt) beams with two different dosage rates: 20kMU (kilo Monitor Unit) and 180kMU. The larger kMU value corresponds to a more intense dosage rate.

5.2 Hyperparameter Space

For this study, the possible hyperparameters that could be altered included the following:

- Dropout rate: 0.45
- Number of neurons per layer: 64, 128, 256, 512
- Number of layers: 64, 128

- Batch size: 1024, 2048, 4096, 8192, 16384, 32768
- Number of epochs: 128, 256, 512, 1024
- Learning Rate: See Section 5.2.1
- Activation function : Leaky ReLU, SeLU
- Batch normalization : false

Generally, experiments that involved changing the number of neurons per layer and number of layers with a value outside those listed above resulted in NaN values for their loss output. This is indicative of exploding gradients, a problem commonly found in neural networks where the gradient that helps with updating the weights becomes so large that it overflows and results in model instability. Multiple experiments also found that parameters like dropout rate and batch normalization did not have significant effects on model performance when altered, so these parameters were kept at a default of 0.45 and false, respectively. Additionally, some combinations above are not possible and so were not tested. An example of this would be SeLU being chosen as an activation function and batch normalization set to true. This is because the SeLU activation function makes the neural network self-normalizing. Overall, a total of 154 models with various combinations of hyperparameters were tested.

5.2.1 Learning Rates

Learning rate is another parameter that has a large influence over performance of a neural network, as it is the amount of which the model changes its weights in response to error. Around eight different learning rates were explored, ranging from static rates or those that vary depending on number of epochs. The learning current rate for presented model uses a non-adaptive non-constant learning rate scheduler implemented in the Keras `LearningRateScheduler` callback which prescribes the learning rate t_i for epoch i with

$$t_i = L(i) = \begin{cases} 5 \cdot 10^{-4} & 1 \leq i \leq \frac{p}{3}, \\ 1 \cdot 10^{-4} & \frac{p}{3} < i \leq \frac{2p}{3}, \\ 1 \cdot 10^{-5} & \frac{2p}{3} < i \leq \frac{5p}{6}, \\ 1 \cdot 10^{-6} & \frac{5p}{6} < i \leq p, \end{cases} \tag{5.1}$$

where p denotes the total number of epochs used for training. In this schedule, the learning rate steadily decreases based on how many epochs the model has trained for. This is a generally followed technique with the rationale that it will increase the chance of convergence with the global minima of the cost function. A learning rate that is too small initially will require many more epochs to reach the global minima, while a learning rate that is too large will likely converge at a suboptimal local minima.

5.2.2 Final Model Hyperparameters

The neural network with the best performance has the lowest training and validation loss values. Out of all models tested, the best performing neural network has the following parameters:

- Dropout rate: 0.45
- Number of neurons per layer: 256
- Number of layers: 64
- Batch size: 4096
- Number of epochs: 512
- Learning Rate: see Equation (5.1)
- Activation function: Leaky ReLU
- Batch normalization: False

5.3 Neural Network Results

The performance results for the current performing neural network can be visualized with a scatter plot showing a direct comparison of the predicted initial energies to that of the true initial energy. The estimations are made on a scale ranging from -1 to 1. A good model produces estimations that are scattered across the diagonal of the graph. This would indicate that the true and estimated initial energy values are both similar values. We don't expect that the scatter plot would be a perfect diagonal line; in fact, if it was, this would indicate that instead of learning from the training data, it has memorized it and would likely underperform when shown new data. This is also known as overfitting. What is expected in this scatter plot is that generally the model's estimations follow along the diagonal.

Figure 5.1 shows the model's results after being tested with the training data. The figure shows dense concentrations along the diagonal and also that for positive true initial energy values within the range of 0.0 to 0.6, some corresponding estimations are a negative value. This indicates that for some values whose true values are higher, the model is predicting a lower value.

Additionally, loss curves generally serve as a good diagnostic tool to help analyze the behavior of a neural network. The loss curve for our network for epochs 300 to 400 can be seen in Figure 5.2. To clarify, validation loss (orange) is a metric of how well the network functions on data it has not seen. Training loss (blue) is a metric of how well the network functions on the data it was trained with. We see validation loss value is steady around a value of 0.0089. Likewise, the training loss has a close value of about 0.0092. These validation and training loss values were the lowest values out of the many models explored in Section 5.2 during the hyperparameter search. As loss is a subjective metric that depends

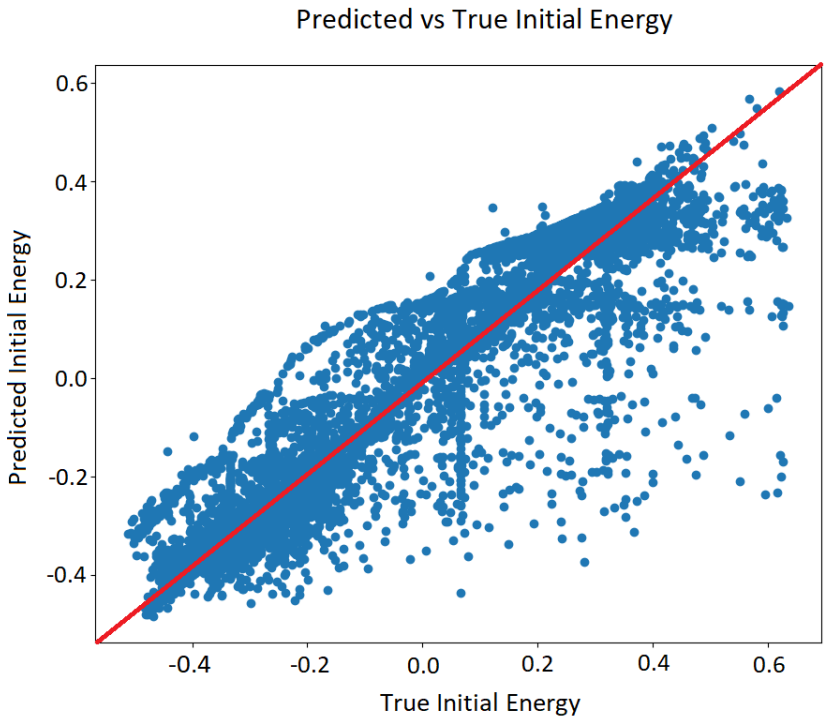


Figure 5.1: Scatter plot of the neural network's performance when tested with the training data. The position of each point on the graph represents the true initial energy value on the x-axis, and the estimated initial energy value on the y-axis.

on the problem and the data, this loss can be considered high in this case as the range of values it is trying to make an estimate for is within -1 to 1. Figure 5.2 does reveal that albeit very subtle, these loss lines are still trending downwards as opposed to reach a plateau, indicating the model is underfitting and could be trained more. The ideal loss for the neural network in this case would be an MSE value for training and validation losses to be around 0.001 as an indicator of optimal performance.

5.4 Reconstruction Results

Based on the estimations of the data from the neural network, we created a new dataset, where there are no misorderings or false events. We reconstructed the proton beam using three versions of the data, as shown in Figure 5.3. Figure 5.3 (a) and (d) shows the reconstruction image from the original test data for each respective dosage. In other words, these images are simply the raw test data being sent through the reconstruction algorithm with no changes. Figure 5.3 (b) and (e) shows the reconstruction image from the neural network estimation data. We took the original test data and sent it through the neural network to generate estimations, and then sent those estimations through the reconstruction algorithm. Figure 5.3 (c) and (f) shows the "perfect" reconstruction. The perfect reconstruction is

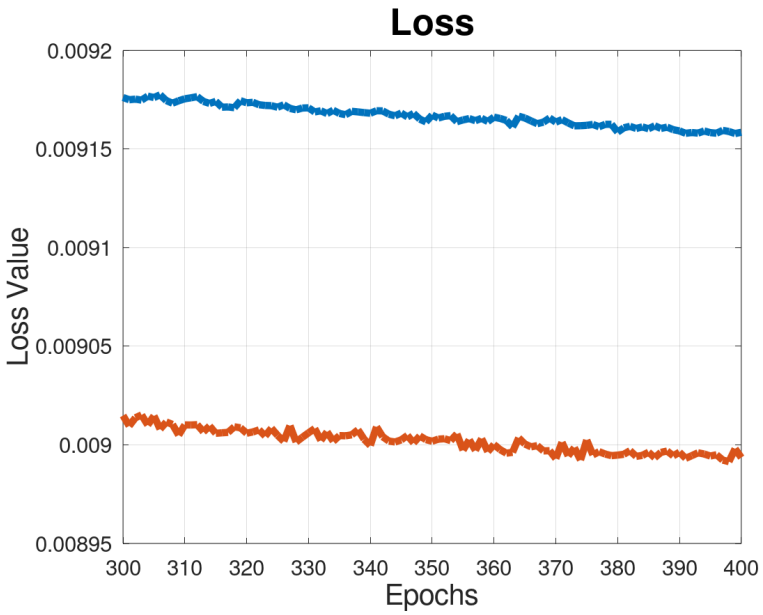
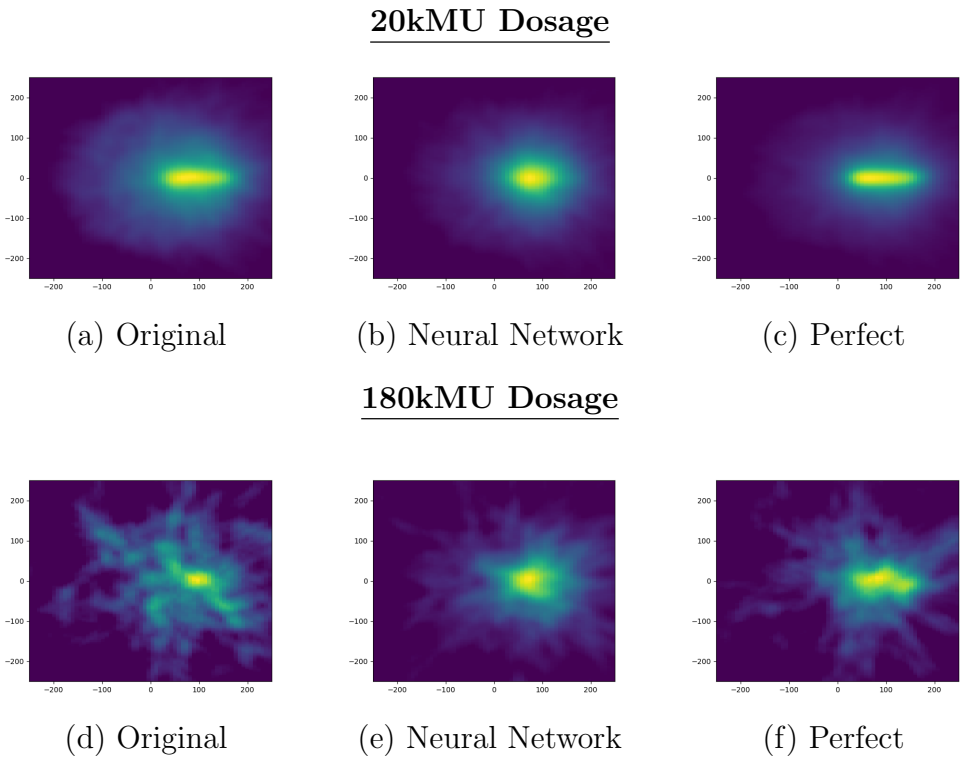


Figure 5.2: Loss curve for the featured neural network from epochs 300 to 400.



(a) Original (b) Neural Network (c) Perfect

(d) Original (e) Neural Network (f) Perfect

Figure 5.3: Comparison of gamma ray images reconstructed with (a/d) original data and (b/e) neural network estimation data to the (c/f) perfect reconstruction.

found using the following adjustments to the original test data:

$$
\begin{aligned}
&e_2 := E_0 - e_1 \\
&\text{if } e_2 < 0, \text{ then } e_2 := E_{2_0}, e_1 := E_0 - e_2 \\
&\text{if } e_1 < 0, \text{ then } e_1 := E_{1_0}
\end{aligned}
$$

We take the known initial energy from the original simulation data E_0 and subtract the first interaction energy e_1 and set this to the second interaction energy e_2. We then check to see if this value is negative. If it is negative, then we revert e_2 to the value it had originally (E_{2_0}), and then use this value to find e_1. Likewise, we determine if e_1 is now found to be negative and if so, revert to the original value (E_{1_0}).

This adjustment takes into account the phenomenon that there are multiple prompt gamma rays that happen to collide at the same point. This can inflate the energy value for an individual interaction and result in negative values, which are not possible in the context of our problem but still can be seen within the original reconstructions. This adjustment corrects the data in this case from this phenomenon, resulting in a "perfect" reconstruction with noticeably less noise present. Overall, the neural network estimation reconstructions should closely resemble the perfect reconstructions more so than the originals. There is visibly less noise present between Figure 5.3 (a) and (b) and also between (d) and (e). However, the Bragg peaks in Figure 5.3 (b) and (e) are more round shaped as opposed to the more linear shape we see in the perfect reconstructions.

This is likely due to a few possible things. It was mentioned earlier that this neural network has a high loss value relative to the range of values it is trying to estimate within. We can see the results of this in the neural network estimation reconstruction's circular Bragg peak, demonstrating that where there are supposed to be higher energy values, it is predicting lower energy values. Additionally, the neural network is also training with the same data that does include the presence of those events where the multiple prompt gamma rays are colliding at the same point. This could be having an effect on its final estimations for areas of high radiation values. Additionally, the nature of reconstructing a proton beam based on scattering data is inherently difficult and not without its own challenges. The nature of reconstruction itself is beyond the scope of this work and can be seen in [8].

6 Conclusions

Section 5.3 describes the best performing neural network model out of various models tested in the hyperparameter space described in Section 5.2. We chose this model based on its performance as determined by its training and validation loss values. In Section 5.4, we can visibly see that there is less noise in the neural network reconstructions than the original data reconstructions. However, the appearance of the Bragg peak is distorted, indicating the neural network still needs improvement. Various directions can be taken with this research to improve the estimations. Firstly, it is possible that this particular neural network can be trained for more epochs to reach an ideal loss value that is closer to 0.001. It is also possible that adaptive learning rates could be utilized. By using information from past training

histories, an adaptive learning rate may produce better results and have more flexibility than the schedules that were hand-designed in this experiment. Using an alternative loss function may also help with better network performance, as MSE has a tendency to weigh outlier estimations heavily.

Additionally, exploring data discretization might also be lucrative. Our regression neural network model currently seeks to estimate some value between the continuous range of -1 to 1. Alternatively, we can instead split that range into some set number of bins, where each bin represents an initial energy estimation. A classification neural network model can then sort the data to one of the bins, and then reconstruct an image based on its classification, which may result in improved reconstructions.

Overall, improvement in true double initial energy value estimations will result in more accurate image reconstructions, which in practice can reduce the amount of uncertainty of the region influenced via the proton beam.

Acknowledgments

Many thanks to my mentors Dr. Matthias K. Gobbert and Dr. Jerimy Polf for their patience and constructive suggestions through this research process. I am deeply grateful to Carlos Barajas for his constructive suggestions and knowledge. Finally, a special thanks to Dr. Jacqueline King and the Meyerhoff program for their unwavering support.

The hardware used in the computational studies is part of the UMBC High Performance Computing Facility (HPCF). The facility is supported by the U.S. National Science Foundation through the MRI program (grant nos. CNS–0821258, CNS–1228778, and OAC–1726023) and the SCREMS program (grant no. DMS–0821311), with additional substantial support from the University of Maryland, Baltimore County (UMBC). See https://hpcf.umbc.edu for more information on HPCF and the projects using its resources.

References

[1] Carlos A. Barajas, Gerson C. Kroiz, Matthias K. Gobbert, and Jerimy C. Polf. Deep learning based classification methods of Compton camera based prompt gamma imaging for proton radiotherapy. Technical Report HPCF–2021–1, UMBC High Performance Computing Facility, University of Maryland, Baltimore County, 2021.

[2] François Chollet. *Deep Learning with Python*. Manning Publications Co., 2018.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[4] Gerson C. Kroiz, Carlos A. Barajas, Matthias K. Gobbert, and Jerimy C. Polf. Exploring deep learning to improve Compton camera based prompt gamma image reconstruction

for proton radiotherapy. In *The 17th International Conference on Data Science (ICDATA'21)*, accepted (2021).

[5] Dennis Mackin, Steve Peterson, Sam Beddar, and Jerimy Polf. Evaluation of a stochastic reconstruction algorithm for use in Compton camera imaging and beam range verification from secondary gamma emission during proton therapy. *Phys. Med. Biol.*, 57(11):3537–3553, 2012.

[6] Paul Maggi, Steve Peterson, Rajesh Panthi, Dennis Mackin, Hao Yang, Zhong He, Sam Beddar, and Jerimy Polf. Computational model for detector timing effects in Compton-camera based prompt-gamma imaging for proton radiotherapy. *Phys. Med. Biol.*, 65(12):125004, 2020.

[7] Jerimy C. Polf, Carlos A. Barajas, Stephen W. Peterson, Dennis S. Mackin, Sam Beddar, Lei Ren, and Matthias K. Gobbert. Applications of machine learning to improve the clinical viability of Compton camera based in vivo range verification in proton radiotherapy. *Front. Phys.*, 10:838273, 2022.

[8] Jerimy C. Polf and Katia Parodi. Imaging particle beams for cancer treatment. *Phys. Today*, 68(10):28–33, 2015.