Channel Islands
CALIFORNIA STATE UNIVERSITY

Neural Networks and Laser Experiments

Andrew Graves • Advisor: Dr. Scott Feister • COMP 499, Section 07

Introduction

When conducting bleeding-edge experiments, physicists are usually given two options to get experimental data. They can opt to run a full-fledged experiment where the entire environment is painstakingly set up to the degree of accuracy that is required to make valuable observations, or they run the experiment in a simulated environment where the physical world is modeled by a computer.

The former situation requires a lot of maintenance time as a crew needs to set up, tear-down or reconfigure their environment after every "fire" of the system. Even the actual experiment can be costly, as these high-powered lasers require a lot of energy and need to go through their own maintenance cycles. The latter option of running a full world simulation seems like it would be a better option, as it eliminates the tedium of having to set up a physical environment. However, recreating the physical environment with all of its quirks is not only extremely computationally intensive, but also is very difficult. As we have yet to explain every phenomenon in physics, there is always the possibility that a detail from experimentation simply will not be produced in a simulation, especially as these experiments are usually operating on the atomic and subatomic scale. However, there is another option that seems to have some promising results.

This other option involves the use of a neural network. The idea here is that we would start with experimental data to train this network, and it would be able to, given input conditions, predict the output values. This is the option that I choose to explore with my capstone. Working with Dr. Feister and some of his fellow researchers at Ohio State University, I attempted to devise not only a method of predicting environmental outputs given environmental inputs, but I also sought to flip that on its head and devise a system to predict input variables given an output. This second part turned out to be challenging, as I will elaborate on later in this poster.

The main purpose of this project was to attempt to reduce the time and energy required to predict the outcomes of novel experiments and possibly the possible inputs required to generate a desired output.

Methodology

When approaching this project, I started with trying to learn as much as I could about the physics aspect of it as I could. I watched lots of content from research students, and universities across the world going over the different types of laser-solid experiments there were and what different methods existed for measuring / predicting their results. I've included some of them in the sources of this poster.

Some of the content that I watched and read mentioned that neural networks were a technology that seemed promising for use in physics, and interest was expressed in some of the cost and time savings the technology may bring.

I followed this up with getting some experience creating my own neural networks with Python and attempting to think of an architecture that would allow me to work backwards as well as forwards.

Throughout the semester, I worked with Dr. Feister to ensure that I had the correct idea of the physical experiment I was attempting to make predictions on. He helped me understand what the most ideal use case for a neural network like this would be and how people in his field would benefit.

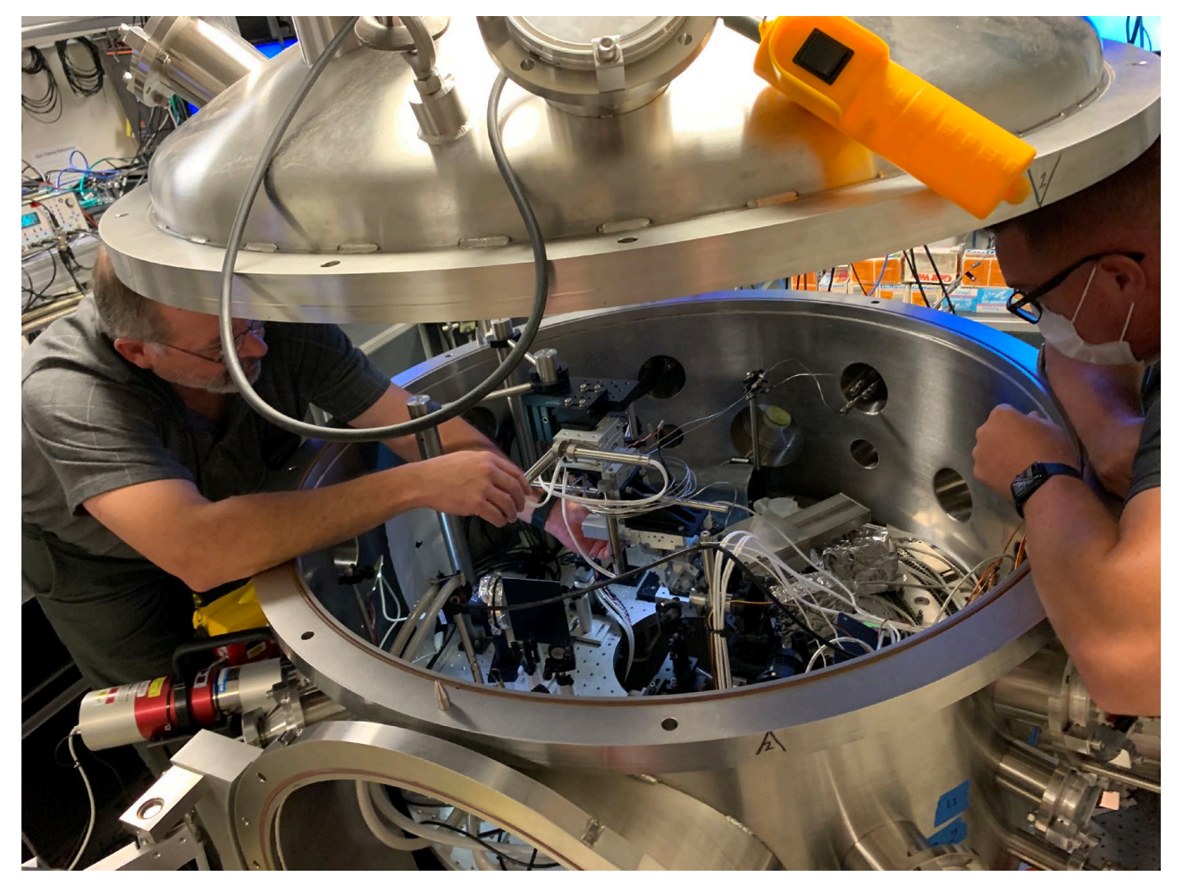
Fig 1. Physicists setting up a laser experiment

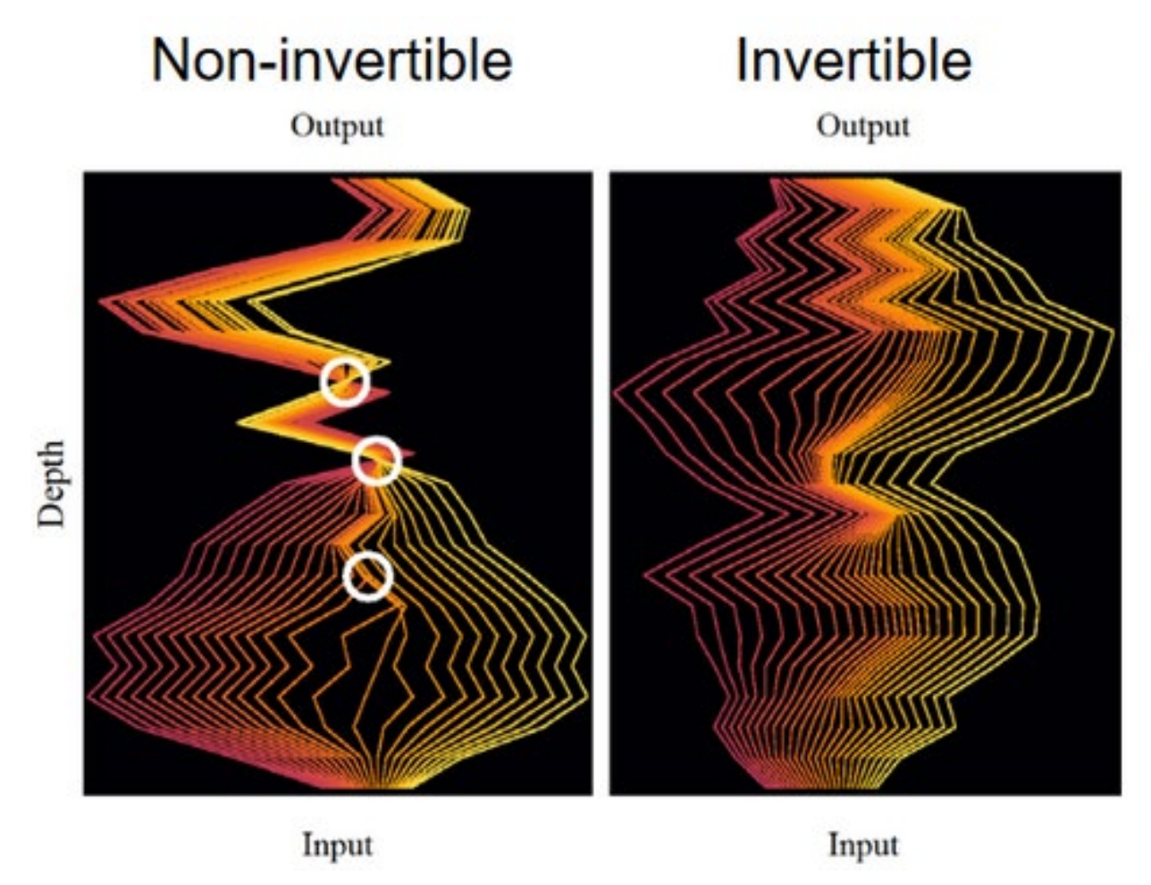

Fig 2. Visual guide to an invertible neural network

Results

The main idea here was to use a neural network because, ultimately, it's just a large collection of functions, weights and biases and can be used to fit a complex multidimensional function complicated enough to represent the experimentation space that the experiment is operating within.

My initial work was a success. I was able to create a neural network that predicted experimental outcomes with a high degree of accuracy (Fig. 3). The network was created in python using PyTorch, which allowed me to quickly turn my ideas into a functioning NN. I trained it using artificial data that was generated together with people from Ohio State University. I've included some images depicting training error and outcome predictions. Currently, I am modeling a Deep Neural Network (DNN) with 2 hidden layers. The input layer takes 4 inputs that include "Laser Intensity (W_cm2)", "Laser Pulse Duration (fs)", "Spot Size (FWHM um)", and "Target Thickness (um)". The following two layers have a size of 64 and 8 nodes respectively, and the last layer is the output with a value labeled "Max Proton Energy (MeV)". The activation functions are all rectified linear unit functions or ReLU, a non-linear activation function commonly used in DNNs (Fig. 5).

However, I also sought to create a model that could suggest inputs to generate a given continuous output. This involved more creative thinking and additional research on my side. I was unable to create the functional system that I desired to create with my initial ideas, but I did brainstorm the structure for a model that, I believe, could work.

Just mirroring the original structure was not an option, as it tends to be really difficult to train a model to predict many variables from the value of a single input variable. As proof, I flipped the model and graphed the training and testing error to show that this is not a viable option (Fig. 4). The other option I had was to create a structure that incorporated some of what I learned about invertible neural networks to allow some backtracking functionality.

Normal neural networks only function in a single direction. Their operations are destructive and do not allow for any information to be derived about the inputs that created an output. Invertible Neural Networks turn this idea on its head by allowing information to be gained in both directions, forward and back. This is possible because their structure is arranged in such a way that there is no loss of data through the layers. Figure 2 demonstrates this visually. Thus, in practice an Invertible neural network is a normal neural network but with special restrictions on how the activation functions, weights and biases can be set up.

My proposed structure would be the blending of Invertible, AutoEncoder (Fig. 6) and linear layers as described below. There would be two networks, one for getting the outputs and one for getting the inputs. The first one would consist of linear layers, with only the last layer having an invertible transition between them. Once the first model was trained, the second model would use the weights and biases from those last layers as its input section. The next layer section would consist of an autoencoder that will attempt to find a mapping between the outputs of that second layer and an output layer of 4 nodes mirroring the original input. I have yet to implement this as an actual model, but it was what I was planning to do if I were to continue working on this project.

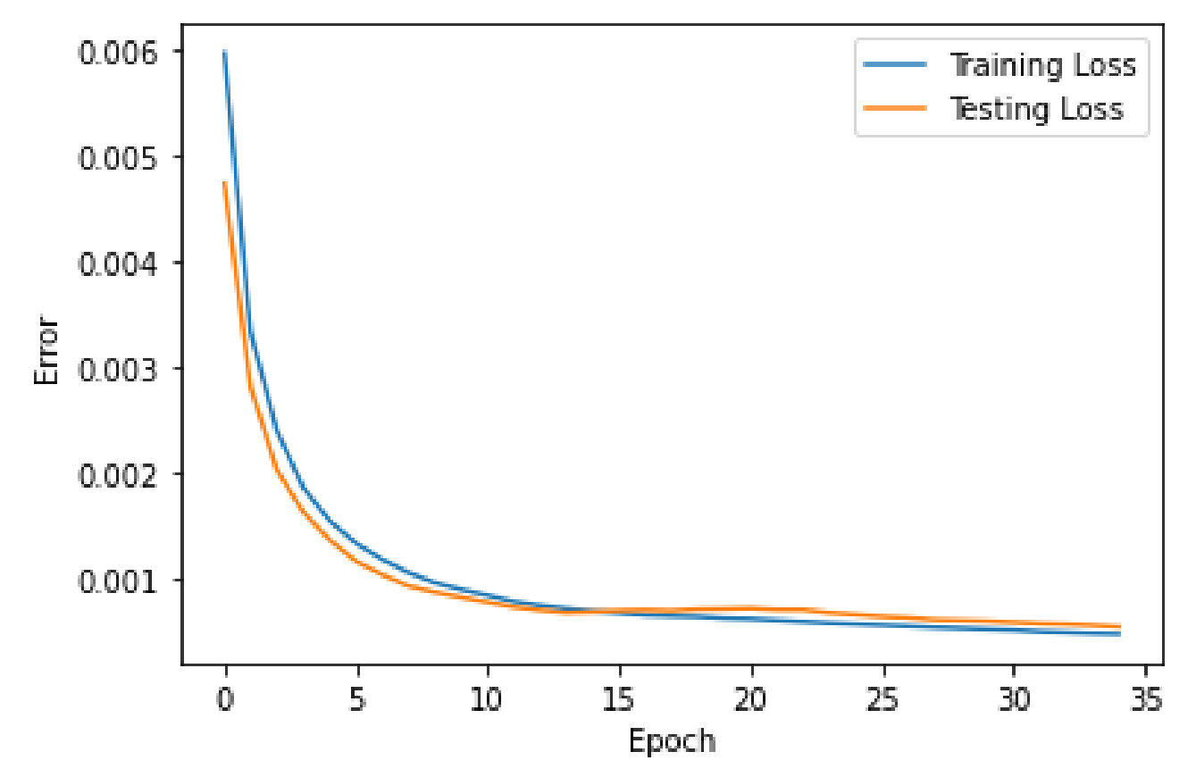

Fig 3. Training and Testing error relative to time of the initial model

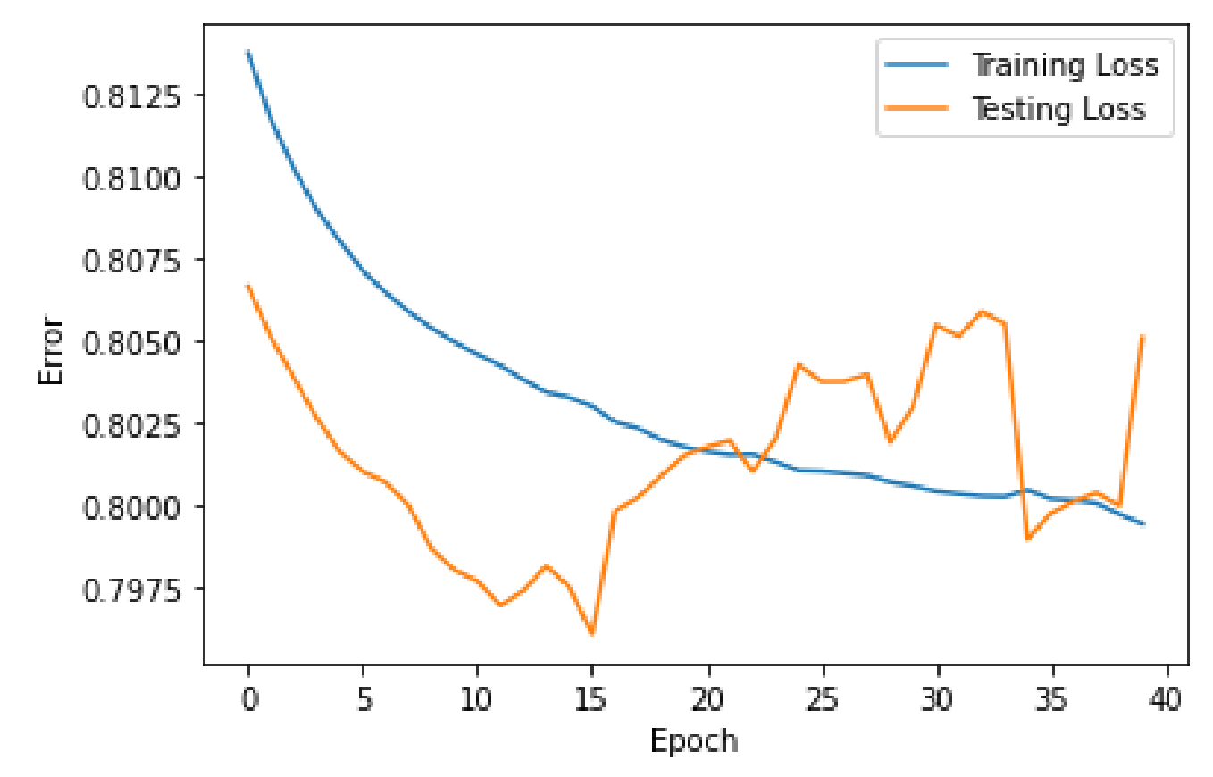

Fig 4. Training and Testing error relative to time of the mirrored model

Conclusions

My main takeaway from this project was how applicable neural networks are to many different types of real-world problems. It was interesting to learn more about specific types and how they modify the traditional structure to gain specific functional attributes. I learned that there were many great resources out there for the main types of neural networks but realized quickly that invertible neural networks were actively being researched and contained much less information and documentation discussing their implementation. I enjoyed working with Dr. Feister and his colleges as I attempted to fit my research and development to meet their needs.

Furthermore, I also got to learn about invertible neural networks, something that I never encountered while studying machine learning in school. They really piqued my interest and motivated me to take a deeper dive into some of the different types of neural networks.

After university, I intend to continue to develop this system, as I am curious to see if it would work as I intended it to.

Resources

LPA Seminars, (2022). Workshop Day 4. [Cognitive Simulation models for inertial confinement fusion]. YouTube. https://www.youtube.com/watch?v=haCVtpoqr70&t=877s
LPA Seminars, (2022). The data-driven future of high-energy-density physics. YouTube. https://www.youtube.com/watch?v=PSZArvFd1UI
LPA Seminars, (2022). Optimisation of high-intensity laser-solid interactions using gaussian process regression. YouTube. https://www.youtube.com/watch?v=SCe1QFLIl68

Further Reading

Badr, W. (2019). "Auto-Encoder: What Is It? And What Is It Used For?" https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726
Jacobsen, J. (2018). "i-REVNET: DEEP INVERTIBLE NETWORKS." https://openreview.net/pdf?id=HJsjkMb0Z

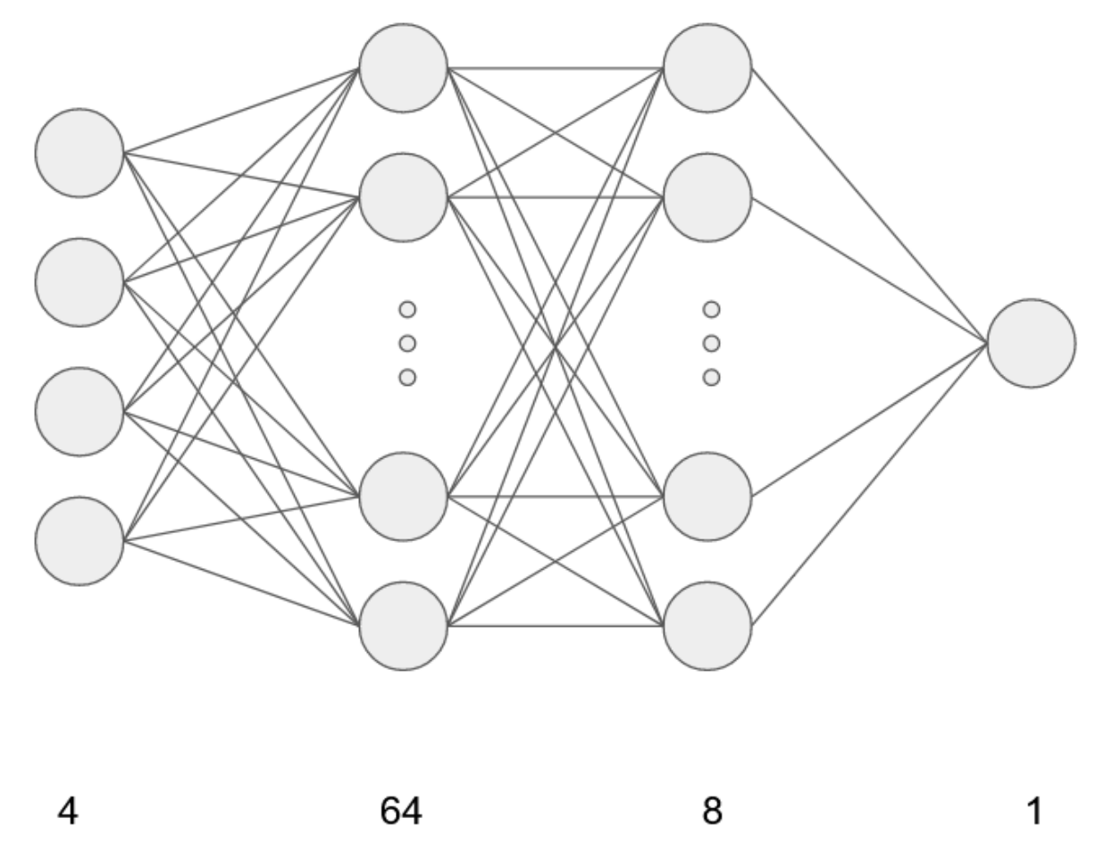

Fig 5. The initial model

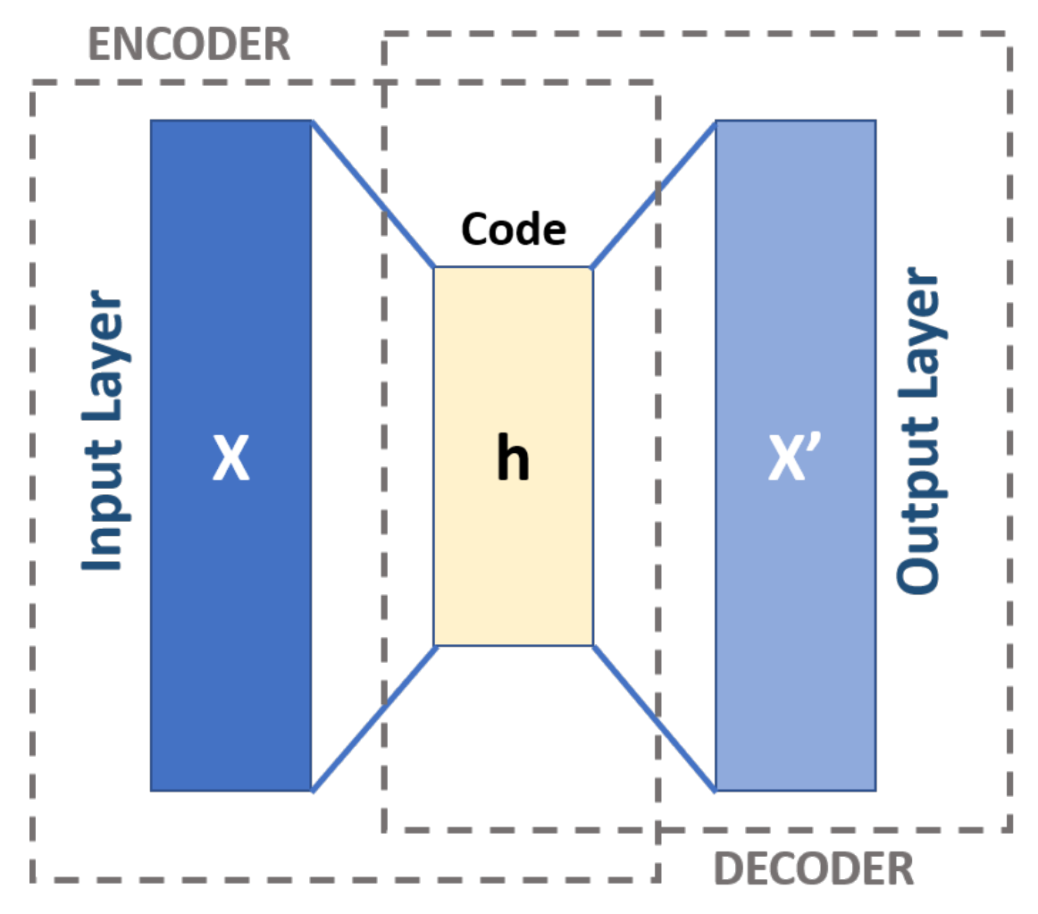

Fig 6. The architecture of an autoencoder

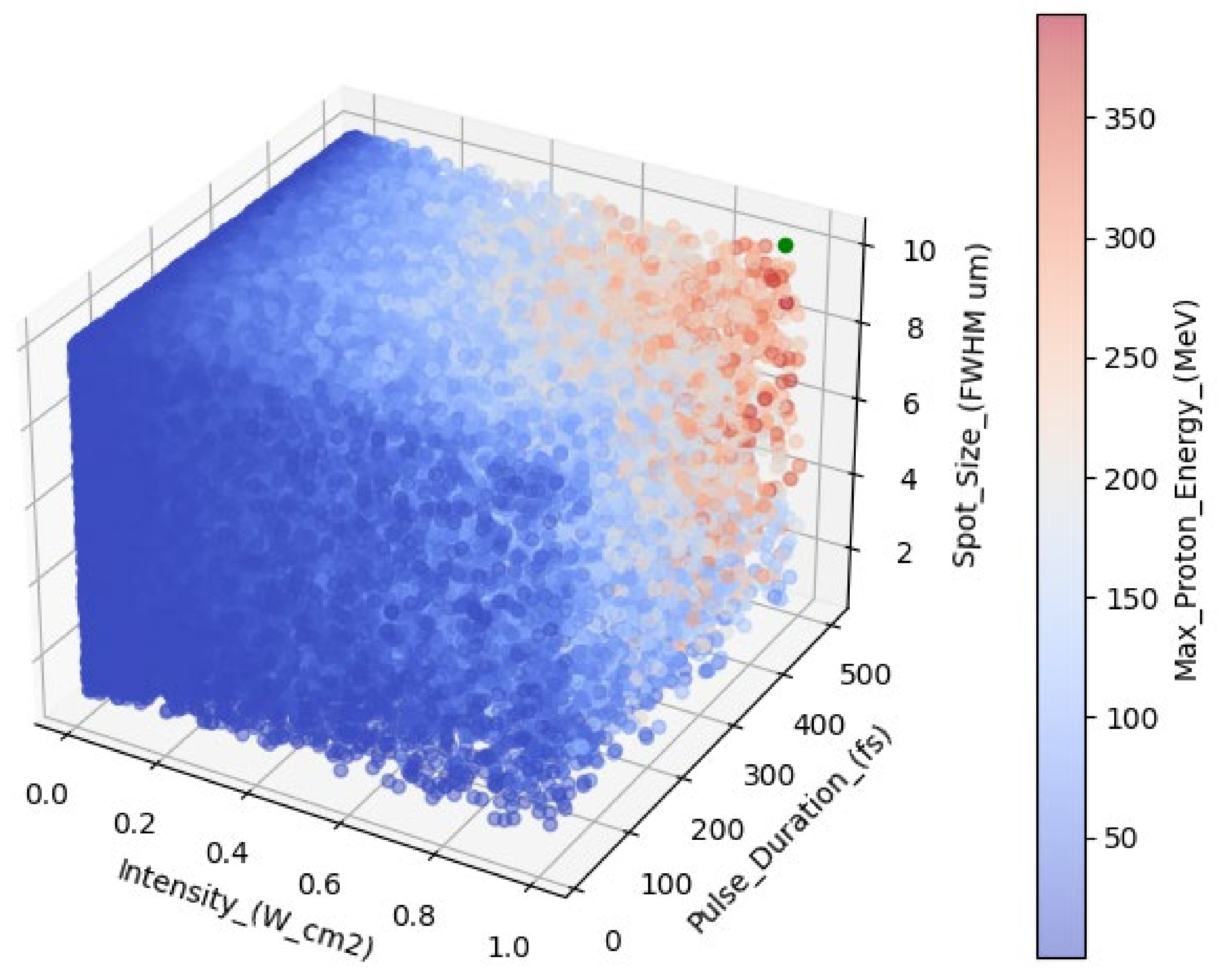

Fig 7. The dataset graphed with 4 of the 5 parameters. The green point is a data point plugged into the prediction model and the predicted value is about what is expected for a value in that region of the graph.