Chapter 4

Deep Neural Networks for Photon and Neutron Transport

Team Members
Le Nyguyen and Jeffrey Keithley

Mentors
Derek Armstrong (XCP-8), Eric Nelson (XCP-8), and Garry Maskaly (XCP-8)

Abstract

Los Alamos National Laboratory (LANL) simulates low altitude ($<$ 10 - 15 km) and ground based EMP (Electromagnetic Pulse) models (E1 and E2 phases) with MCNP and FDTD codes. These models are extremely computationally expensive, taking on the order of weeks to run. The majority of the run-time is in MCNP calculating the energy deposition rate and photocurrent density to feed into FDTD which calculates the EMP's electric field. In this project, we used a multi-layer recurrent neural network to fit existing MCNP energy deposition rate data and then for prediction. Once trained, the recurrent neural network successfully reproduced MCNP results within 1-2% error in a fraction of the time.

4.1 Introduction

MCNP is a Monte Carlo based particle transport simulation code [MCNP-Team (2003)]. A modified version of MCNP provides reliable results for simulating the energy deposition rate and photocurrent density from low altitude ($<$ 10-15 km) photon and neutron sources in air. The photocurrent density and energy deposition rate are then fed into FDTD, a Maxwell solver, that computes the evolution of an EMP's electric field with time. This is very computationally expensive, and a MCNP and FDTD run generally takes on the order of a few weeks to a month to complete, with most of the time taken by MCNP. The purpose of this project is to develop a machine learning based method to model the energy deposition rate and photocurrent density due to photon and neutron sources in significantly less time. An EMP is driven by the photon and neutron weapon output where the gamma rays, being prompt, scattered, or induced by neutrons, ionize the atmosphere via Compton scattering. This model would be used as a surrogate to MCNP and feed the same results into FDTD, making the entire EMP calculation much faster [Yee (1966)]. Because of the time constraints of the project, we were only able to develop a model for the energy deposition rate.

4.2 Background

4.2.1 Electromagnetic Pulse

MCNP and FDTD simulate the EMP effects of a nuclear device detonated in air at low altitudes ($<$ 10-15 km) and on the ground. The detonation in air versus detonation on the ground determines the geometry of the EMP propagation, but fundamentally it is generated in the same way for all nuclear based EMPs. This paper will briefly go over the course grained model developed by Karzas and Latter (WJ Karzas & Richard Latter, 1962; WJ Karzas & R Latter, 1962) and Longmire (Chavin et al., 1979; H. Longley & Longmire, 1972; H. J. Longley & Longmire, 1973; C. Longmire, 1978; Longmire, 1974; C. L. Longmire, 1978; Longmire, 1986, 1987, 1995, 2004; Longmire & Gilbert, 1980; Longmire et al., 1987; Longmire & Longley, 1973), that is widely accepted by the relevant scientific community [Rivera and et al. (2016)].

The behavior of an EMP is broken up into three phases: E1, E2, and E3. E1 spans the first microsecond, E2 spans from one microsecond out to a second, and E3 spans from one second to 10s of seconds. MCNP and FDTD are only used to model E1 and E2 of an EMP and do not have the computational capability to model E3. E1 is created from prompt gamma rays propagating directly from the blast. The gamma rays Compton scatter electrons off atoms in the air that, in mass, continue outward creating this phase of the EMP. E2 is very similar to E1. It is created by scattered and neutron induced gamma rays Compton scattering electrons off atoms in the air. The scattered gammas are from interactions that have already occurred in the atmosphere, while the neutron induced gamma rays are produced by neutrons from the initial blast interacting with nuclei in the air. These intermediate steps in the gamma ray production are what makes the E2 EMP lag behind E1.

There are a few key factors that determine the effect of an EMP, such as the geometry of the electric field. If the electric field is perfectly symmetric, there would be no net current to produce an EMP outside of the source region. Asymmetry is key to producing an EMP and in a low altitude air burst EMP the asymmetry is created by variations in atmospheric density as well as asymmetry because of the air-ground interface. The height of burst of the EMP is another determining factor in its effect. This is partly due to geometric spreading, which is when the intensity of the EMP drops off rapidly as it travels away from the source.

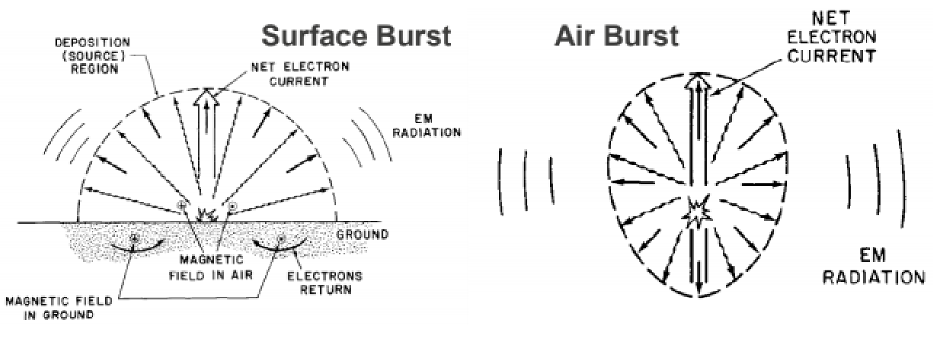


Figure 4.1: Low Altitude EMP, Glasstone and Dolan (1977)

4.2.2 Monte Carlo N-Particle Code (MCNP)

The Monte Carlo N-Particle code developed at LANL is utilized to simulate particle transport behavior relevant for a nuclear burst EMP, calculating energy deposition rate and photocurrent density. The nature of the method used by MCNP is to aggregate many instances of particle behavior instead of deterministically solving the appropriate transport equations [MCNP-Team (2003)]. This results in a long run time due to the computationally intensive process of simulating many particle instances. Despite this, the Monte Carlo method used works well with problems that are difficult to solve deterministically. This method generates the energy deposition rate $(MeV/cm^3/sh)$, which is the energy added to a cell due to particle interactions. MCNP gets the energy deposition per interaction with a look-up table. The energy deposition rate is primarily related to free electrons interacting with air, slowing down, and heating up the air. The energy deposition is fed into FDTD, which is a Maxwell solver that computes the ionization rate, solves for air chemistry, and computes the time-varying electric field. FDTD takes the energy deposition rate and computes an ionization rate by simply assuming an ion pair is created for each 34 eV of deposited energy. MCNP has the ability to model various types of particle interactions, including Compton scattering, the photoelectric effect, and pair production for photons. For neutrons, it has the capability to model interactions such as elastic and inelastic scattering, as well as neutron capture [Reily and et al. (1991)].

Variance reduction in MCNP is used to increase the precision of results from the simulation and avoid consuming additional run time to achieve a smaller error. The prominent methods of variance reduction used by the MCNP calculations in this work are truncation, rouletting, and splitting. The method of truncation operates by eliminating certain data points based on a set of criteria. The main criteria is energy and time values that have little impact on the tallies of interest. Rouletting eliminates particles with a certain probability when they cross a particular boundary, which varies depending on the relevance of the region in the simulation. The last method, splitting, happens when the particle crosses into a region of higher importance. Splitting, as the name implies, is when one particle is replaced by serial particles (split) with the same properties, but reduced weights, to get higher statistics in a region.

4.2.3 Recurrent Neural Networks (RNN)

Neural networks have been at the center of this project since its conception, since they are able to learn the behavior of simulated energy deposition rates. Due to the time series nature of these MCNP results, a recurrent neural network (RNN) is a viable solution [Britz (2015)]. The cornerstone of RNN architecture is the history vector h_t, which is how previous data is remembered. The construction of a node is relatively simple, where the input x_t is concatenated with the history vector h_t, then fed through an activation function to scale the values between -1 and 1. A major drawback of these networks is that previous states are forgotten within a few iterations, therefore the context stored farther back in the sequence is forgotten.

4.2.4 Long Short-Term Memory Networks (LSTM)

The most significant difference between an RNN and LSTM architecture is the complexity of the node. While an RNN is barely a jump from a feed forward network, an LSTM incorporates multiple gates in order to systematically determine whether information is relevant [Olah (2015)]. The underlying feature of the LSTM is the cell state C_t, which consists of what information should be forgotten and what information is relevant. The forget gate feeds the concatenation of the history vector and the input vector, $[h_t{-}1, x_t]$, into a sigmoid function and combines it with the previous cell state C_{t-1}. This information is then combined with a scaled vector representing what values should replace the forgotten ones from the previous step. This is now considered the new cell state C_t and the final step is updating the history vector using a combination of the previous history vector, the input vector, and the newly updated cell state. This cell state and history vector are passed into the next node along with the next input vector.

4.3 Methods

4.3.1 Network Architecture

The RNN architecture used was built in Tensorflow using the Keras API. The specifications of how the network was built are given in table 4.1. As we discussed in the background for recurrent neural networks, a drawback to this type of node is that previous states are soon forgotten. The solution to this problem was solved by utilizing the Long Short-Term Memory architecture, which is better suited for longer term memory.

LSTM Layers	4
LSTM Nodes	128
Dense Layers	1
Dense Nodes	1
Activation	$tanh$
Recurrent	$sigmoid$
Batch Size	64
Learning Rate	Exponential Decay

Table 4.1: Network Architecture

4.4 Data Processing

4.4.1 Feature Engineering

In order to get a better fit to our MCNP data, features were added to give the network more data to fit. The features were the following: average energy was used instead of maximum and minimum energy, column density (amount of air between source and tally location), time squared, and the log of the radial distance to the source of the burst. A rigorous exploration of how these features, or what added features in particular improved the fit was not done, but fit improvement was seen after the features were given to the neural network.

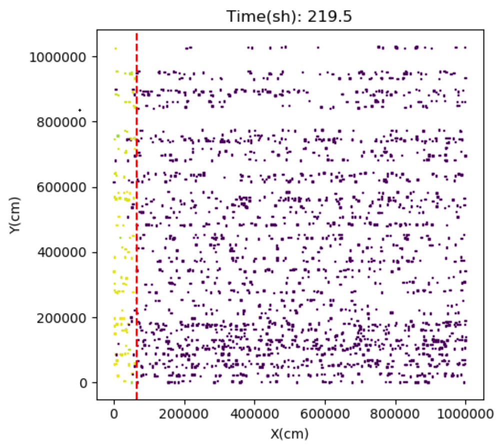
Some of these features are physical in nature, others are simply there to provide more data to fit. The average energy was used in place of the upper and lower bound on the energy range because it is an easier feature to fit than the bounds. The column density is a physical feature; it is a measure of the amount of air from source to tally location. It correlates to the probability of particles making it to a particular location with a given energy, so it is a useful feature for the RNN to use. The log of the radial distance is also another useful physical feature, because the radiative intensity from the source location falls off proportionally to radial distance squared. The log is used in the radial distance due to the energy deposition rate being fit in log-space (i.e., the log of the energy deposition rate is what the RNN is fit against). It will help the RNN predict how many particles should be at a given radius away from the source. The time squared on the other hand is not a physical feature, it does not represent anything in the system, but gives more information the RNN can fit.

Original Features	Original Prediction	Added Features	Final Features	Final Predictions
Height of Burst, Energy Upper Bound, Energy Lower Bound, X Position, Y Position, Time	Energy Deposition Rate	Average Energy, Column Density, t^2, $\log(R)$	Height of Burst, X Position, Y Position, Time, Average Energy, Column Density, t^2, $\log(R)$	$\log$ (Energy Deposition Rate) Zeros Removed

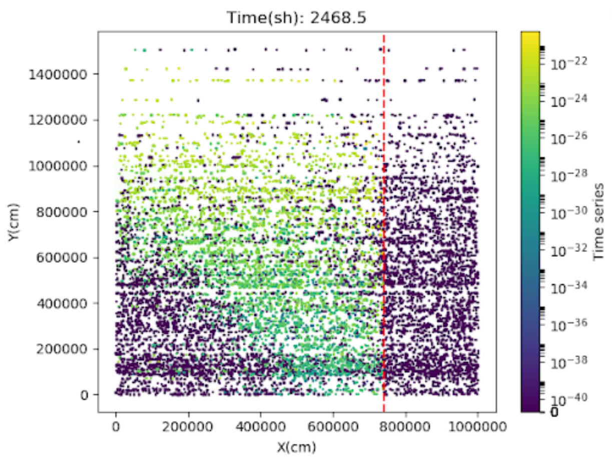
Table 4.2: Original and Processed Features

4.4.2 Data Smoothing

As well as adding features, the log of the energy deposition rate (the prediction) was taken as well. The zeroes in the prediction were also masked out. The energy deposition rate spanned 20 orders of magnitude and the network struggled to fit this, with around 70% of the values being zero, due to the time it took for the particles to populate the tally grid. For fitting to neutron data, the energy deposition rate had to be smoothed in order to fit. The average of every 12 points on each side was taken to achieve this.



(a) Grid at 219.5 sh



(b) Grid at 2468.5 sh

Figure 4.2: Visualization of Features (x, y, time, energy deposition)

4.5 Results

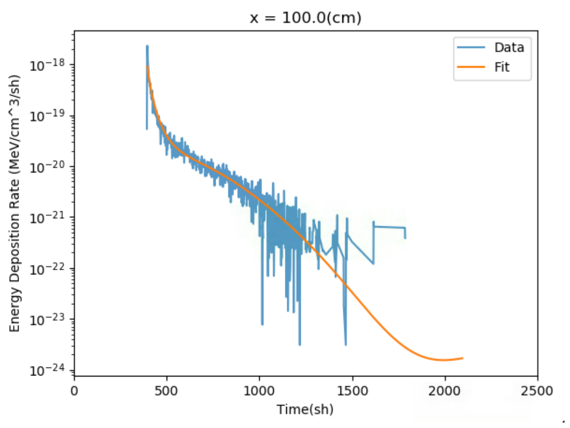
4.5.1 Model Fit

Photon Fit

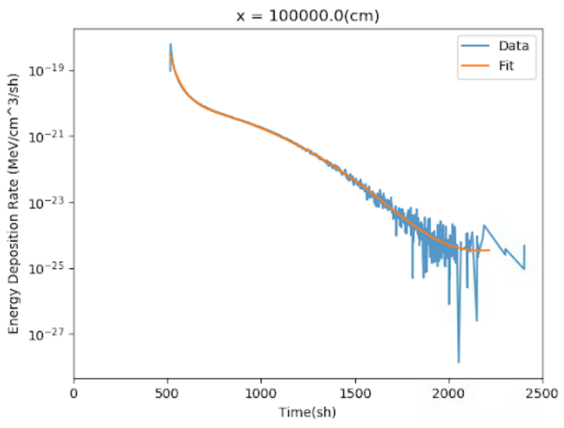
Figure 4.3 shows the RNN fit to the energy deposition rate of our photon data. The photon data was sampled from 50 MCNP runs with a photon energy within the 2-3 MeV range at an unfixed height of burst. The model was set to fit around 2000 shakes at each X location, each with the same Y location.

Height of Burst	X Values	Y value	Energy
$6,135$ m	1 m, 1000 m	$4,950$ m	2.55 MeV

Table 4.3: Photon Model Input Values



(a) 100 Epochs



(b) 100 Epochs

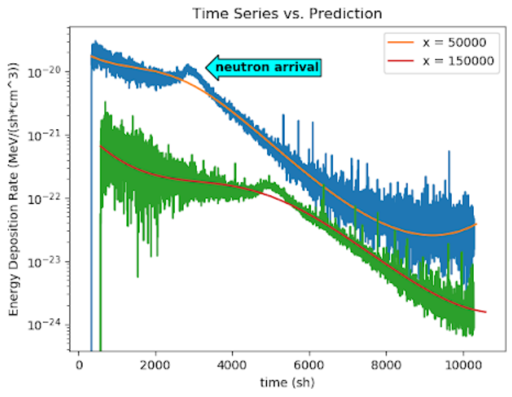
Figure 4.3: Photon Fit Plots

Neutron Fit

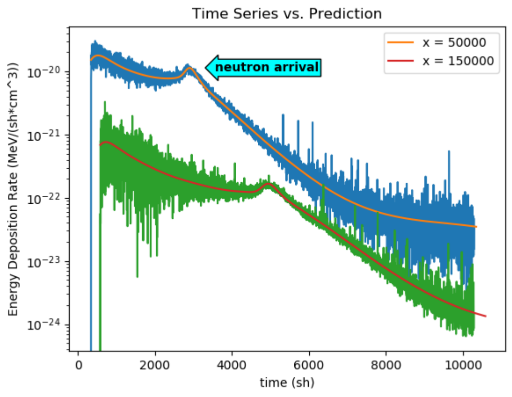
As figure 4.4 clearly shows, the model fits the energy deposition rate for the parameters we specified, except for the neutron arrival times between 3000 and 5000 shakes. This was solved by using smoothed data, a method previously discussed in the data processing section. Even when we train the model in less epochs, the smoothed data still performs drastically better than the original model. The drawback of using smoothed data is that the model takes almost 12 hours to train, as opposed to the model using unsmoothed data, which takes less than 2 hours to train.

Height of Burst	X Values	Y value	Energy
$1,000$ m	500 m, 1500 m	100 m	7.05 MeV

Table 4.4: Neutron Model Input Values



(a) 100 Epochs

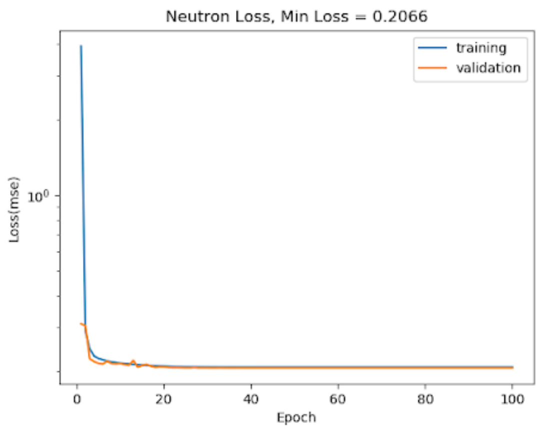


(b) 60 Epochs (Smooth)

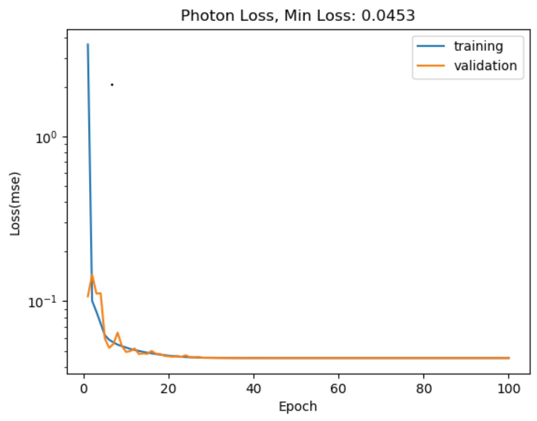
Figure 4.4: Neutron Fit Plots

4.5.2 Training Loss

Figure 4.5 illustrates the loss per epoch during the training time for our models, which show both photon and neutron predictions quickly achieve a low loss (MSE) and converge to a good fit. The loss for the photon produces more noise initially, but settles to a better fit overall by an order of magnitude. The neutron model converges faster, but maintains a higher loss than the photon model. This is due to the differences in complexity of the photon and neutron data. With the neutron data being harder to fit, we would expect a higher loss.



(a) Neutron Training Loss



(b) Photon Training Loss

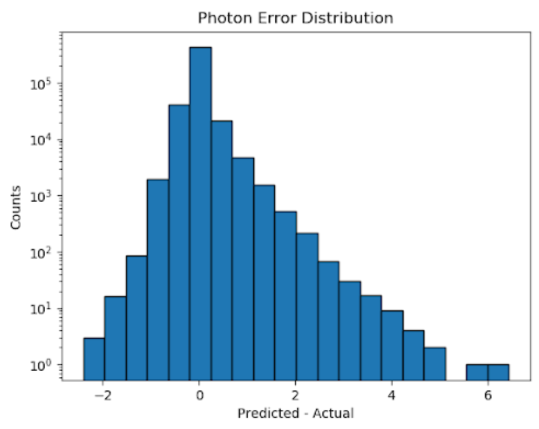
Figure 4.5: Training Loss Plots

4.5.3 Error Analysis

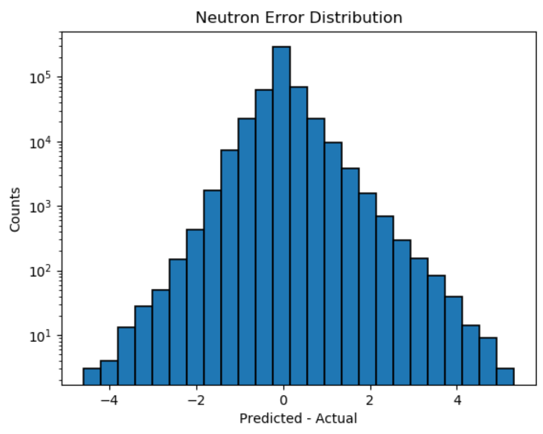
Our neural network fits well to the data with the error being around 1-2%. This error is misleading though, as it is fit to noisy data. Much of the error in the fit is the model ignoring the statistical noise in the MCNP data and does not reflect the accuracy of the fit. As we can see from figure 4.6, the distribution for neutron prediction error $\hat{y} - y$ is symmetric and roughly normal. While we have some outliers, some of these may be attributed to the noise in the actual simulation due to its stochastic nature. From figure 4.6 it may be concluded that the behavior of the outliers is more extreme in the photon prediction model, but values are more centered about 0, so it performs better than the predictor for neutrons in terms of overall fit.

	Photon Fit	Neutron Fit	Neutron (Smooth Data) Fit
MSE (log Space)	0.0452	0.2131	0.1351
RMSE (log Space)	0.2128	0.4616	0.3675
RMSRE	0.00727	0.01811	0.01462

Table 4.5: Fitting Errors



(a) Errors for Photon Fit



(b) Errors for Neutron Fit

Figure 4.6: Error Distribution

4.6 Discussion

It is important to state the neural network we developed is not a replacement for MCNP since it fits to data but cannot generate original EMP results. Our neural network is an extrapolation on current MCNP data, meaning that it can be used to estimate the energy deposition rate from a set of parameters MCNP has not been run on yet based on the sets of parameters data MCNP already has results for. Suppose we want the energy deposition rate over all time and space for a height of burst MCNP had not been initialized at before: We can have our neural network estimate its value in a matter of days instead of performing an entire month long MCNP run.

A back of the envelope calculation was done to estimate how the run time of our RNN compared to the run time of MCNP. The calculation was simple; we took the run-time for our RNN to model a single position and energy at a certain height of burst over all time and multiplied it over all positions and energies. The photon RNN could make a calculation in .08 seconds. There are 2,000 X and Y tallies in a full MCNP grid and 100 energies. This calculation results in the RNN taking about 9,000 CPU hours to run vs. MCNP which takes about 150,000 CPU hours. From this rough calculation we can see that our RNN is about 16.6x faster than MCNP.

4.7 Future Work

Many aspects of this study could be researched more thoroughly, particularly the generalizability of the model. The analysis presented in this paper was specific to a limited energy range for both photon and neutron data with a fixed height of burst. Training models on the entire energy spectrum would be both informative and beneficial for a possible future implementation, as well as smoothing data to fit the neutron arrival time better. The fit of the RNN could have been further optimized with a more thorough hyper parameter search. Parameters such as layers, nodes, learning rate, and batch size were optimized by hand because of time constraints. A grid search or controlled random search should be done to find the parameters that allow the best hyper parameters. Finally, the development of a user-friendly interface that could utilize the trained models without requiring extensive knowledge of the intricacies involved with the method would make the code more usable.

4.8 Conclusion

MCNP along with FDTD simulates low altitude E1 and E2 EMPs. The calculations needed for this take weeks to run with the majority of the run-time being MCNP calculating the energy deposition rate and photocurrent density. To cut down on run-time, a multi-layer recurrent neural network was developed as a way to predict MCNP results of energy deposition rate to feed into FDTD. This was done successfully with RNNs being able to fit photon data and smoothed neutron data. Run time was greatly reduced with the RNN being able to calculate the energy deposition rate for all time, space, and energies in 9,000 CPU hours compared to a recent MCNP run which took about 150,000 CPU hours. The model also fit the data well, with an error of 1-2% mostly due to statistical noise in the MCNP results. The fitting error was mostly Gaussian suggesting that the models were unbiased, but there was a slight skew in the photon error showing it tended to overestimate values. Multi-layer RNNs have shown very promising results as a surrogate to MCNP for FDTD. There is still work to be done as the RNNs are still limited in scope and can be further optimized. Hopefully this project has laid the groundwork for the further exploration of neural networks and other machine learning techniques as solutions to EMP simulation problems.

4.9 Acknowledgments

We would like to thank LANL and the XCP Summer Workshop coordinators for giving us the opportunity to do this project, especially in these unprecedented times (COVID-19 Pandemic). They have gone the extra mile of making this experience totally remote with online collaboration and lectures. We would also like to thank our mentors, Dr. Derek Armstrong, Dr. Eric Nelson, and Dr. Garry Maskaly for the work they did. They guided us through this project and put in the extra work to give us the tools we needed to succeed in spite of not being at the lab.

Le Nguyen *is a senior at Michigan State University majoring in Physics and minoring in Data Science. He is currently looking at graduate programs in Data Science and is interested in computational modeling and algorithm development.*

Jeffrey Keithley *received his bachelor's degree in mathematics from New Mexico Tech and hopes to perform research at LANL for 1-2 years before pursuing a PhD. He is interested in using computational methods and machine learning to improve simulation and solve problems in physics and computer science.*

Glossary

- **Batch:** The set of data passed through the network
- **Column Density:** The amount of air from source to tally location
- **Energy Deposition Rate:** The rate energy is added to a cell due to particle interactions

- **Epoch:** When the entire data set is passed through a network once
- **Iteration:** When one batch is passed through the network
- **Mean Squared Error (RMSE):** $= \frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_i - Y_i\right)^2$
- **Root Mean Squared Error (RMSE):** $= \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_i - Y_i\right)^2}$
- **Root Mean Squared Relative Error (RMSRE):** $= \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{\hat{Y}_i - Y_i}{Y_i}\right)^2}$
- **Variance Reduction:** A method of increasing the precision of a Monte Carlo method without using more time to run

Bibliography

D. Britz. Recurrent neural networks tutorial. *WILDML*, 2015.

S. Glasstone and P. Dolan. The effects of nuclear weapons. *United States Department of Defense*, 1977.

X-5 MCNP-Team. MCNP — A general monte carlo n-particle transport code. *Los Alamos National Laboratory*, 2003.

C. Olah. Understanding LSTM networks. *Colah's Blog*, 2015.

D. Reily and et al. Passive nondestructive assay of nuclear materials. *Los Alamos National Laboratory*, 1991.

M.K. Rivera and et al. EMP/GMD phase 0 report, A review of emp hazard environments and impacts. *Los Alamos National Laboratory*, 2016.

K.S. Yee. Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media. *IEEE*, 1966.