LA-UR-24-22060
Accepted Manuscript

Source localization for neutron imaging systems using convolutional neural networks

Saavedra, Gary Joseph; Geppert-Kleinrath, Verena; Danly, Christopher Russell; Durocher, Mora; Wilde, Carl Huerstel; Fatherley, Valerie E.; Mendoza, Emily Faith; Tafoya, Landon Reese; Volegov, Petr; Fittinghoff, David; Rubery, Michael; Freeman, Matthew Stouten

To be published in: Review of Scientific Instruments

DOI to publisher's version: 10.1063/5.0205472

Permalink to record:
https://permalink.lanl.gov/object/view?what=info:lanl-repo/lareport/LA-UR-24-22060

Review of
Scientific Instruments

RESEARCH ARTICLE | JUNE 18 2024

Source localization for neutron imaging systems using convolutional neural networks

Gary Saavedra ✉ ; Verena Geppert-Kleinrath ; Chris Danly ; Mora Durocher ; Carl Wilde ;
Valerie Fatherley ; Emily Mendoza; Landon Tafoya ; Petr Volegov ; David Fittinghoff ;
Michael Rubery ; Matthew S. Freeman

Rev. Sci. Instrum. 95, 063503 (2024)
https://doi.org/10.1063/5.0205472

AIP
Publishing

Review of Scientific Instruments ARTICLE pubs.aip.org/aip/rsi

Source localization for neutron imaging systems using convolutional neural networks

Cite as: Rev. Sci. Instrum. 95, 063503 (2024); doi: 10.1063/5.0205472
Submitted: 26 February 2024 • Accepted: 5 June 2024 •
Published Online: 18 June 2024

Gary Saavedra,[1,a)] **Verena Geppert-Kleinrath,**[1] **Chris Danly,**[1] **Mora Durocher,**[1] **Carl Wilde,**[1] **Valerie Fatherley,**[1] **Emily Mendoza,**[1] **Landon Tafoya,**[1] **Petr Volegov,**[2] **David Fittinghoff,**[2] **Michael Rubery,**[2] **and Matthew S. Freeman**[1]

AFFILIATIONS

[1] Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA
[2] Lawrence Livermore National Laboratory, Livermore, California 94550, USA

[a)] **Author to whom correspondence should be addressed:** gsaavedra@lanl.gov

ABSTRACT

The nuclear imaging system at the National Ignition Facility (NIF) is a crucial diagnostic for determining the geometry of inertial confinement fusion implosions. The geometry is reconstructed from a neutron aperture image via a set of reconstruction algorithms using an iterative Bayesian inference approach. An important step in these reconstruction algorithms is finding the fusion source location within the camera field-of-view. Currently, source localization is achieved via an iterative optimization algorithm. In this paper, we introduce a machine learning approach for source localization. Specifically, we train a convolutional neural network to predict source locations given a neutron aperture image. We show that this approach decreases computation time by several orders of magnitude compared to the current optimization-based source localization while achieving similar accuracy on both synthetic data and a collection of recent NIF deuterium–tritium shots.

I. INTRODUCTION

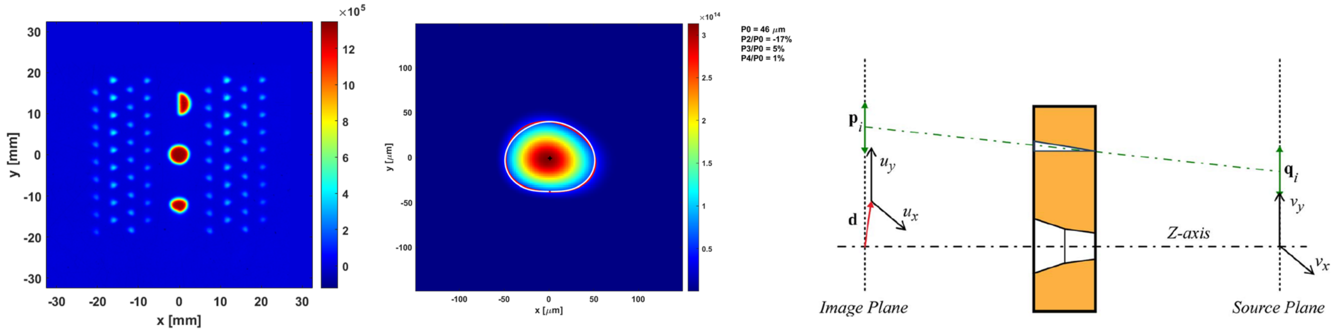
The National Ignition Facility (NIF) at Lawrence Livermore National Laboratory recently achieved thermonuclear ignition.[1] This achievement was aided by a suite of diagnostics. Among these diagnostic techniques, the Nuclear Imaging System (NIS) plays a crucial role in determining the geometry of the fusion reaction.[2] Reconstructed fusion source images give information on the size, symmetry, and shape of the fusion hot spot and surrounding cold fuel and have been vital for inertial confinement fusion (ICF) experimentation, including recent experiments that have achieved ignition. Neutron images, in particular, are used to reconstruct the hot spot geometry.

The NIS team currently utilizes a suite of Bayesian-based inference and optimization algorithms to construct a source image from experimental neutron data. A key component of this algorithmic suite is source localization, i.e., finding the fusion source location within the camera field-of-view. Currently, an optimization routine must be run on a shot-by-shot basis to determine the source location. In this paper, we discuss an approach using machine learning (ML) methods to replace the current source localization algorithm. In particular, we train deep neural networks (DNN) to predict source locations within the camera field-of-view from neutron aperture images.

DNNs have revolutionized the field of image processing with their ability to learn from large datasets.[3] Traditional image processing techniques often rely on handcrafted features, which limits their flexibility and increases the human-time cost of their use. In contrast, DNNs are often more accurate and less time-consuming for human analysts than traditional techniques and have demonstrated major success in imaging tasks such as medical and scientific imaging. The dominance of DNNs in image processing is attributed to their ability to process raw real-world data. In addition, DNNs can often reduce computation time over traditional iterative algorithms.

II. SOURCE LOCALIZATION

During a typical ICF experiment at the NIF, the NIS diagnostic will produce an aperture image [Fig. 1(a)], which is formed via the interaction of neutrons with an aperture array. This aperture



(a) (b) (c)

FIG. 1. (a) Image, *I*, produced by the aperture array for line-of-sight NIS-3. The array consists of three large penumbras in the center vertical column, as well as eight columns of small triangular pinholes. Color intensity represents neutrons per pixel. (b) Reconstructed source distribution *S*. The red outer contour traces the pixels at 17% intensity.[5] The white outer contour is the Legendre polynomial fit to the 17% contour. The Pn represents the modes of the Legendre polynomial fit. (c) Neutron imaging aperture array and coordinate system notation (not to scale). Figure (c) is reproduced from Volegov *et al.*, Rev. Sci. Instrum. **85**, 023508 (2014) and Volegov *et al.*, Rev. Sci. Instrum. **85**, 123506 (2014) with the permission of AIP Publishing LLC.

image is captured via a scintillator-based detection system.[4] A suite of iterative Bayesian inference algorithms uses the aperture image to construct a fusion source distribution [Fig. 1(b)]. This process is carried out over three nearly orthogonal lines of sight, which allow for viewing the source from different angles.

We can analytically define the relationship between the aperture image and the fusion source as follows: Given a source, *S*, and aperture point spread function, *K*, we define the aperture image, *I*, recorded by the detector as

$$I(\mathbf{u}) = \sum_{i=1}^{N_a} \int_V K_i(\mathbf{u} - \mathbf{p}_i + \mathbf{d}, \mathbf{v} - \mathbf{q}_i) S(\mathbf{v}) \partial^2 v, \tag{1}$$

where N_a is the number of apertures, **d** is the center position of the scintillator, and $\mathbf{p}_i$ and $\mathbf{q}_i$ are the coordinates of the central axis of aperture *i* in the detector plane and source plane, respectively. The vectors **u** and **v** define a position in the detector and source plane, respectively [see Fig. 1(c)].

Given an image, *I*, current reconstruction algorithms discretize Eq. (1) and find the source, *S*, by using expectation maximization algorithms to solve for the Maximum Likelihood Estimation (MLE) solution.[6]

The first step in a full source reconstruction is the determination of the source location within the field-of-view. Due to limitations in the mechanical placement of fuel capsules and alignment of the aperture array, the location of the source can vary by several hundred microns for each experiment. Figure 2 shows a collection of source center locations for recent shots on the NIS-3 line-of-sight.

The neutron imaging team currently uses an optimization-based approach to find the source location for each NIF shot. The optimization-based approach, which we will refer to as *SrcLocOpt*, is an iterative algorithm that minimizes the reduced chi-square objective function,

$$\chi^2 = \frac{1}{m} \sum_{k=1}^{m} \left(\frac{I_k - \tilde{I}_k}{\sigma_k} \right)^2, \tag{2}$$

where *m* is the number of pixels in the aperture image, I_k is the *k*th pixel of the recorded image, $\tilde{I}_k$ is the *k*th pixel of the model image produced by Eq. (1), and σ_k is the standard deviation of the *k*th pixel computed according to the noise model described in Volegov *et al.*[7] At each iteration, *SrcLocOpt* minimizes Eq. (2) by adjusting the source location and shape parameters. We use the Nelder–Mead simplex algorithm[8] to optimize, as it has shown the best performance. This optimization problem is computationally expensive and can result in non-optimal solutions.[9] In addition, *SrcLocOpt* often requires multiple runs with different parameter choices made by human operators, thereby increasing the computation and human labor time.

Determination of the source location is crucial as errors can propagate downstream to the final reconstruction, as shown in Fig. 3. In the bottom plot, we compare the effect of source

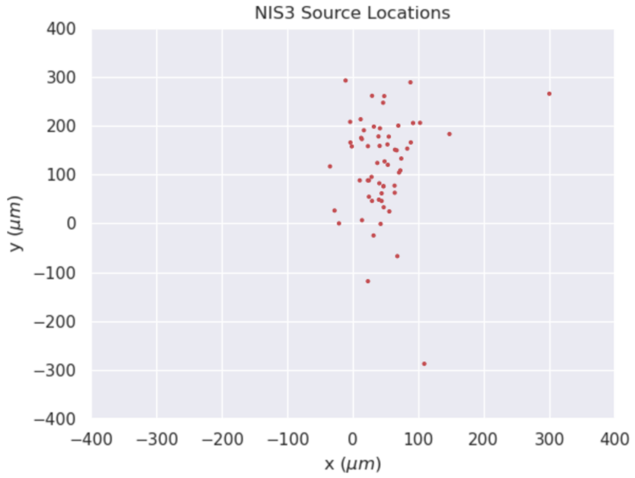


FIG. 2. Locations of source centers for NIF shots in the NIS-3 field-of-view. The origin represents the central axis of the center penumbra. We see that the source location can vary up to several hundred microns from the origin.

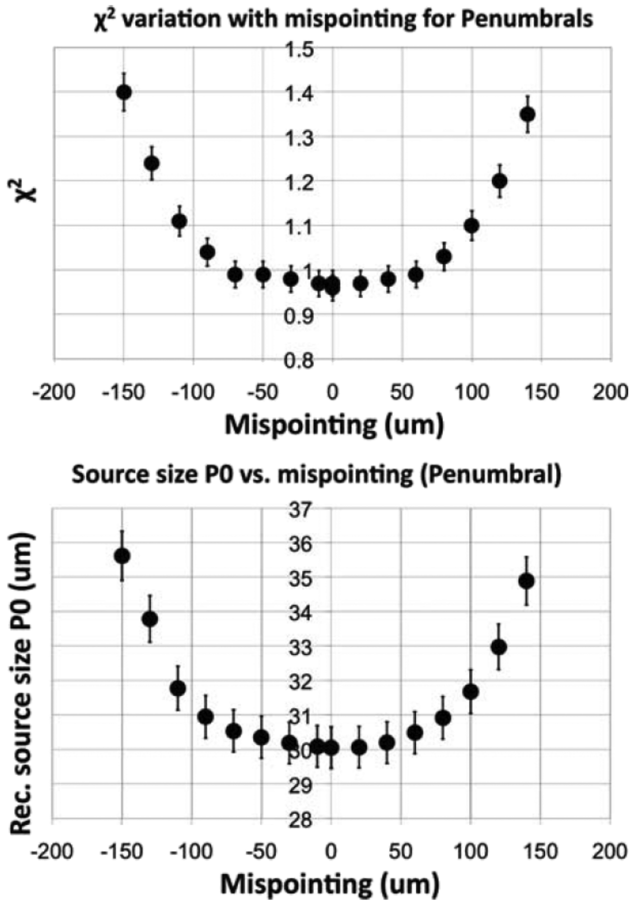


FIG. 3. Reconstruction error of a 30 μm source with varying amounts of source location error (mispointing). The top figure shows the change in reconstruction goodness of fit χ^2 as the source location changes. The bottom figure shows the change in P0 as the source location changes. P0 is the zeroth Legendre polynomial mode describing the 17% outer contour. Figures are reused from Guler *et al.*, Rev. Sci. Instrum. **83**, 10D316 (2012), with the permission of AIP Publishing LLC.

localization on P0 for a circular 30 μm source where P0 is the zeroth Legendre polynomial mode of the 17% outer contour.[2] We choose a 30 μm source, as this size is typical for NIF experiments.[10] We see that a source location prediction error of 50 μm produces an error in P0 of less than 1 μm. However, as we increase beyond 50 μm, the error increases substantially.

To compare the effect of source localization on the overall source reconstruction, we measure the change in the source reconstruction goodness of fit as a function of source location error. These results can be seen in the top plot of Fig. 3. Here, the χ^2 introduced in Eq. (2) is used as a measure of goodness of fit for source reconstruction. Similar to our P0 results before, we see that there is little variation in χ^2 below 50 μm. However, as the source location error increases beyond 50 μm, the χ^2 changes dramatically, indicating larger errors in the overall source reconstruction. Overall, the results of Fig. 3 indicate that we should strive for location prediction methods that stay below 50 μm of error. We will see in Sec. IV that our DNNs meet these criteria. For further discussion on the effects of source localization on reconstruction, see Guler *et al.*[10]

III. OVERVIEW OF NEURAL NETWORKS

In this section, we give a brief overview of two types of neural networks, the feedforward neural network (FNN) and the Convolutional Neural Network (CNN). FNNs are motivated by biological neural networks. Biological neural networks process information using a system of interconnected neurons that transmit electrical and chemical signals. Inspired by this biological model, FNNs simulate these processes using algorithms and data structures to solve complex problems. In FNNs, units called *neurons* are connected in layers and send signals to each other, mimicking the communication in biological networks. Although the functionality of FNNs is simplified compared to their biological counterparts, the foundational principles derive directly from observing how biological neural networks function. For more detail on the biological underpinnings of neural networks and their relationship to FNNs, see Yang and Wang.[11]

FNNs are versatile and have been shown to perform well on a variety of tasks. However, FNNs typically have poor performance on imaging tasks due to the complexity of imaging data and their inability to easily discover spatial relationships through the training process. Conversely, CNNs are designed for tasks involving spatial relationships and work well on variable-size inputs and complex data like images, where traditional FNNs fail. In this work, we adopt CNN, as it has been shown to perform well on a variety of imaging tasks.[12] See the Appendix for more details on the inner workings of FNNs and CNNs.

The parameters of the neural network are iteratively updated through an optimization process called *training*. During a typical training session, a practitioner must choose several hyperparameters that control the optimization process. The choice of these hyperparameters can have a major effect on the final performance of the network. Typical hyperparameters include the number of convolution and feedforward layers, as well as the number of nodes in the feedforward layers. These parameters are crucial, as increased depth and width of the network tend to improve representational capacity.[13] However, increased depth or width comes at the expense of more computation for training. Additionally, the number of output channels for each layer of the CNN, as well as the size of the convolution filter, have an important influence on the final CNN performance. Finally, the choice of optimization algorithm and its associated parameters play an important role in training. In particular, the learning rate determines the weight update size at every iteration and plays a crucial role in gradient-based optimization algorithms that are typically used for training neural networks. This set of hyperparameters tends to have the most impact on the performance of the final model[14], so we focus on tuning them in Sec. IV B. For more information on neural network hyperparameters and hyperparameter tuning, we refer to Yu and Zhu[14] and Yang and Shami.[15]

IV. EXPERIMENTAL RESULTS

A. Dataset and preprocessing

Our goal in this work is to use a CNN to predict source locations given an aperture image. Therefore, to train the CNN, we require a set of data points with aperture images for model input and source locations as output. Ideally, we would like to train the CNN using real-world experiment data. However, a typical neural network requires thousands of data points to train sufficiently, and there are currently only a few hundred total data points from NIF experiments. Therefore, in this work, we use synthetic data generated via a forward model[16] described by Eq. (1). Equation (1) is an idealized imaging equation that does not take into account various imperfections in the imaging system. Nevertheless, this forward model has proven adequate for both source localization and reconstruction with the current NIS suite of algorithms.[6] We will see in Sec. IV F later that these generated data are accurate enough for training the CNN to predict source locations on real NIF data.

Using Eq. (1), we specify the source location and shape and project the source through the aperture array to generate an aperture image. In addition, to reduce the computation time required for generating the synthetic data, we generate aperture images using the penumbral projections only, i.e., the center vertical column in Fig. 1(a). Previous works have shown that the penumbra is often sufficient for determining source location.[6]

A pixel in the aperture image represents the number of neutrons in a $4 \times 4\ \mu\text{m}^2$ region of the field-of-view. The total number of neutrons in the aperture image is determined by the total neutron yield. At the NIF, this neutron yield can be measured accurately using other diagnostics such as neutron time-of-flight.[17] For our synthetic data, we can manually choose a neutron yield and pass it as a parameter to our forward model. For this work, we generate all sources with a yield of 10^{15}, which is a typical yield for many NIF shots.

In order to fully cover the use cases of the model, we require a dataset with a varied number of source locations and shapes. We sample source locations on an evenly spaced grid using 50 μm spacings over a $600 \times 600\ \mu\text{m}^2$ field-of-view, resulting in 29 evenly spaced locations over the field-of-view. For most NIF experiments, the source position is typically within 300 μm of the center of the field-of-view. Therefore, these location samples cover most use cases.

Typical NIF experiment source sizes can be as small as 25 μm or as large as 125 μm. In addition, the source is often not symmetric, i.e., it may be prolate or oblate. Changes in source shape and size affect the neutron aperture image and, therefore, we must include varied source shapes in the training dataset. While a typical NIF source may include higher order asymmetries, we model the asymmetry with an elliptical contour for simplicity. At each of the grid locations, we generate elliptical source shapes with varied radii on the major and minor axes. We choose radii between 25 and 125 μm, inclusive, in increments of 25 μm. Therefore, at each of the 29 locations, we obtain 25 total sources, which include a variety of circular, prolate, and oblate sources. This data generation scheme results in a total of 4225 data points.

We apply several preprocessing steps to the data. First, we reduce the aperture image size from 1000×1000 pixels to 400×400 pixels. This removes empty space from the image and reduces the required computation time for training and inference. Next, we standardize the data using the equation $\frac{I_i - \mu}{\sigma}$, where I_i refers to aperture image i in our dataset and μ and σ refer to the mean and standard deviation of all the collective pixels in the training data. Since the neutron image pixels represent neutron counts, each pixel typically has a value in the thousands. Feeding these images directly into the CNN results in large parameter explosions and unstable training. Standardization ensures that the training dataset distribution as a whole has a mean centered at 0 and a standard deviation of 1. The standardization reduces pixel values to be in a much smaller range while still retaining the pixel distribution of the images, which results in faster and more stable training for the CNN.[18]

After standardization, we center the image on the brightest pixel of the center penumbra. We then produce nine copies of each data point. In each copy, we randomly shift the image by up to 10 μm. Therefore, for each aperture image, we have the original image with centered penumbra as well as nine copies with random shifts. These shifted images assist in training the model by allowing it to make predictions even if the apertures are shifted within the image.

After preprocessing we have a total of 42,250 data points. We use 80% of the data for training and 20% for validation. In addition, we ensure that every data and its shifted variants fall within the same dataset (either training or validation). The shifted variants share the same source location and shape with the original. Therefore, grouping shifted variants together ensures that we do not mix data points between training and validation, i.e., it ensures we do not train on validation data or validate on training data.

We construct a test dataset using real NIF experiment data. The NIF data are used to show the applicability of our CNN to real-world experiments and allow us to test in real-world conditions where our forward model data may not match reality, e.g., additional background and noise. For constructing the NIF data, we collected 56 recent shots on the NIS-3 line-of-sight shown in Table III.

We apply several preprocessing steps to the NIF data as well. As mentioned earlier, we also apply image size reduction, standardization, and centering. We do not make shifted copies, as these images are not used for training. In addition, we must apply yield scaling to the aperture images. Our training data were generated using a yield of 10^{15}; however, NIF experiment images are produced from experiments where the yield can vary drastically. Changes in the yield introduce a proportional change in the number of neutrons per pixel, which will affect the CNN predictions. In order to obtain an equivalent input image representation as used in training, we scale each image according to the equation $\frac{Y_i}{Y_t}$, where Y_i is the yield for the NIF experiment that produced aperture image i and Y_t is the training data yield, which in our case is 10^{15}. This scaling ensures the NIF images have the same proportion of pixel brightness to neutrons per pixel as the training images.

B. Model and training parameters

For our experiments, we performed a grid search over the hyperparameters of a CNN and chose the model with the lowest validation error. In particular, we trained a model with every combination of the parameters in Table I. With this combination of parameters, we train a total of 324 CNN models.

Additionally, for each model, we apply a 2×2 max pooling to the output of each convolution layer. The final hidden layer of the FNN is fed into a layer that consists of two nodes. The first node predicts the x location of the source center, while the second node predicts the y location. The output of each convolution and

TABLE I. Hyperparameters used to train the model. We train models over every combination of these parameters and keep the model that performs best on the validation set. Note that for the convolution output channels, we use the value specified in the table for the first layer and then double the number of channels for each successive convolution layer, i.e., if the first convolution layer has 40 channels, then the second convolution layer has 80 channels, and so on.

Hyperparameter	Hyperparameter value
No. convolution layers	(2, 3, 4)
No. FNN hidden layers	(1, 3)
No. hidden nodes	(128, 256)
Convolution filter size	(3 × 3, 5 × 5, and 7 × 7)
Convolution output channels	(10, 20, 40)
Learning rate	(0.001, 0.01, 0.1)

hidden layer is fed through ReLU activations. We used a training batch size of 8 and trained the model for three epochs with the ADAM optimizer.[19] We use the mean squared error loss function,

$$L = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - \tilde{X}_i\right)^2, \tag{3}$$

where n is the number of training data points, X_i is the source location predicted by the CNN, and $\tilde{X}_i$ is the ground truth source location. We used the PyTorch[20] framework to construct our ML models. In our experiments, we measure error using the Euclidean distance between the ground truth and predicted source locations in the field-of-view.

After a grid search, the best performing CNN has two convolution layers with 40 channels on the first layer and 80 channels on the second layer, a single hidden layer with 256 hidden nodes, a convolution filter size of 5 × 5, and was trained with a learning rate of 0.1. We depict this CNN in Fig. 4. This CNN obtains a validation error of 15.42 μm, i.e., the average error in source location prediction for data not in the training set is 15.42 μm.

C. Performance over field of view

As discussed previously, source locations can vary by hundreds of microns for each NIF shot. Therefore, it is important that our CNN make accurate predictions over a large field-of-view. Figure 5 shows the CNN performance over a 600 × 600 μm^2 field-of-view for two circular sources with radii of 25 and 125 μm. Given a location in the field-of-view, the heatmap color represents the amount of error in the CNN prediction of that particular source location. In addition, we show the training points as red dots. If a dot is missing from the grid structure, it indicates that this location was used in the validation set. We see that the error remains below 30 μm over much of the field-of-view. However, the error tends to increase in areas where we have fewer training data points. For example, in Fig. 5(a), we see the vertical green band near 300 μm rests between two sets of training points. These errors can likely be improved by generating more location data, i.e., by refining the grid of training points.

We also note that the CNN performs poorly in the bottom left corner of the field of view for both source shapes in Figs. 5(a) and 5(b). To analyze this heatmap feature further, we studied the performance of other CNN models in this region. In particular, we plotted the error heatmap of the CNN, which gave the second best validation performance of 16.16 μm in Fig. 6. This CNN has 128 hidden nodes and four convolution layers. All other hyperparameters are identical to our best performing CNN. We see that the regions of strong performance have changed compared to our previous CNN. In particular, this CNN performs strongly in the bottom left corner. However, it performs more poorly in the top left corner. These results indicate that, in addition to dependence on training data location, CNN performance in various regions is highly dependent on the particular CNN. This variance in performance over certain regions occurs regardless of the close agreement in validation error (15.42 vs 16.16 μm).

D. Comparison to *SrcLocOpt*

We now compare the performance of the CNN with previous optimization algorithms used by the NIS team. We generate 50 new

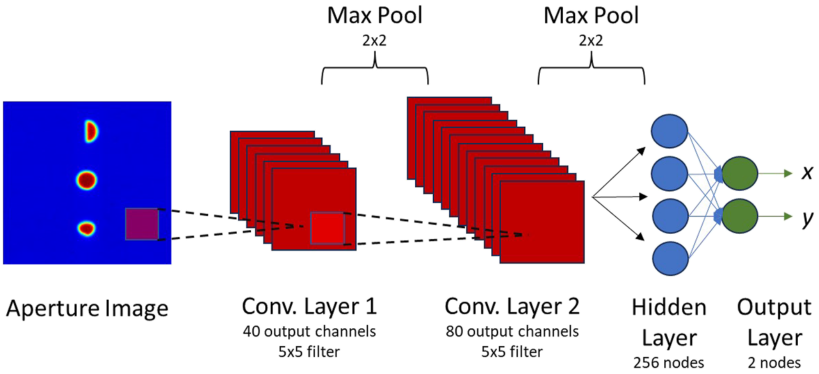
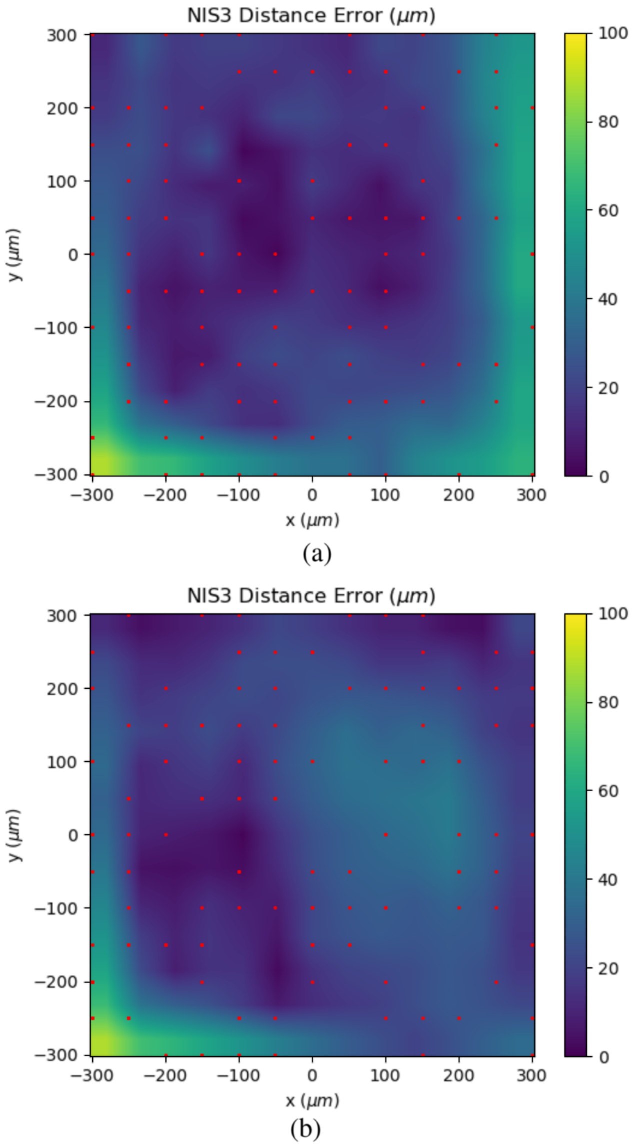


FIG. 4. The best performing CNN after hyperparameter search. This CNN has two convolution layers with 40 output channels on the first layer and 80 output channels on the second layer, a single hidden layer with 256 hidden nodes, a convolution filter size of 5 × 5, and was trained with a learning rate of 0.1. Each convolution layer output is fed through a 2 × 2 max pooling layer. A ReLU activation is applied to every layer except the final output layer. An aperture image with penumbra is the input to the CNN. This CNN obtains a validation error of 15.42 μm.



(a)

(b)

FIG. 5. CNN model prediction error in μm over field-of-view for a circular source radii of (a) 25 μm and (b) 125 μm. Darker colors represent less error. The red points are the locations of the training points. Here, we see that CNN predictions typically have an error of less than 30 μm.

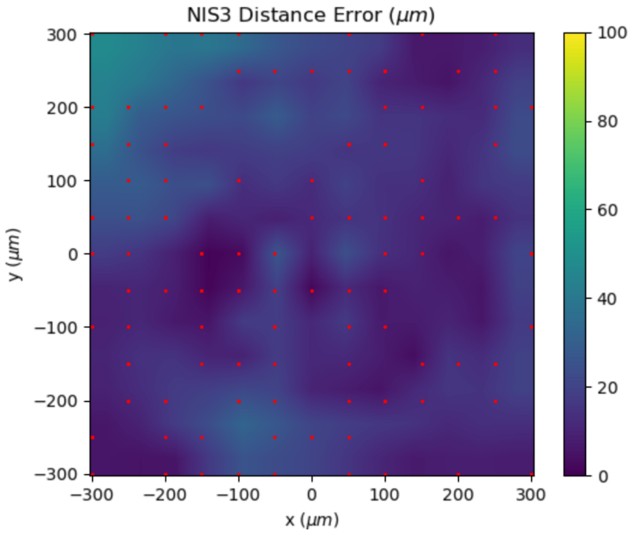


FIG. 6. Prediction error in μm over field-of-view for a source shape of 25 × 25 μm^2. Here, we use our second-best performing CNN. Darker colors represent less error. The red points are the locations of the training points. Note that the features present in the heatmap have changed compared to the best performing CNN used in Fig. 5, indicating that performance for various regions is highly dependent on the CNN parameters.

data points within the 600 × 600 μm^2 field-of-view with a circular source with a radius of 109 μm and predict the source location using the CNN and *SrcLocOpt*. In addition, to facilitate a fair comparison to CNN, we supply the ground truth source shape to *SrcLocOpt* so that it only has to predict locations. The ground truth validation points, as well as the corresponding predictions, are shown in the field-of-view in Fig. 7. The predictions are connected to their ground truth location via a line. We can see that the overall performance is comparable between the two algorithms. The *SrcLocOpt* predictions tend to be more accurate overall. However, in several cases, *SrcLocOpt* makes large inaccurate predictions.

Table II shows the overall error and run time requirements for both algorithms. We see that the CNN outperforms *SrcLocOpt* in mean error. However, the median error is lower for *SrcLocOpt* compared to CNN. The difference in performance can be explained by the large outlier mistakes made by the optimization method. In general, these types of outlier mistakes can be fixed; however,

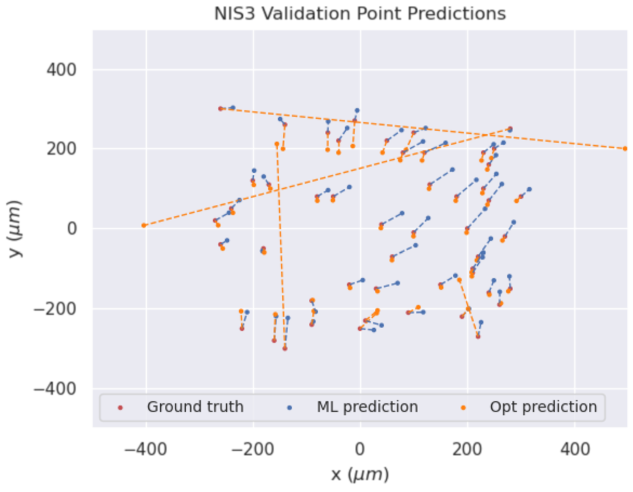
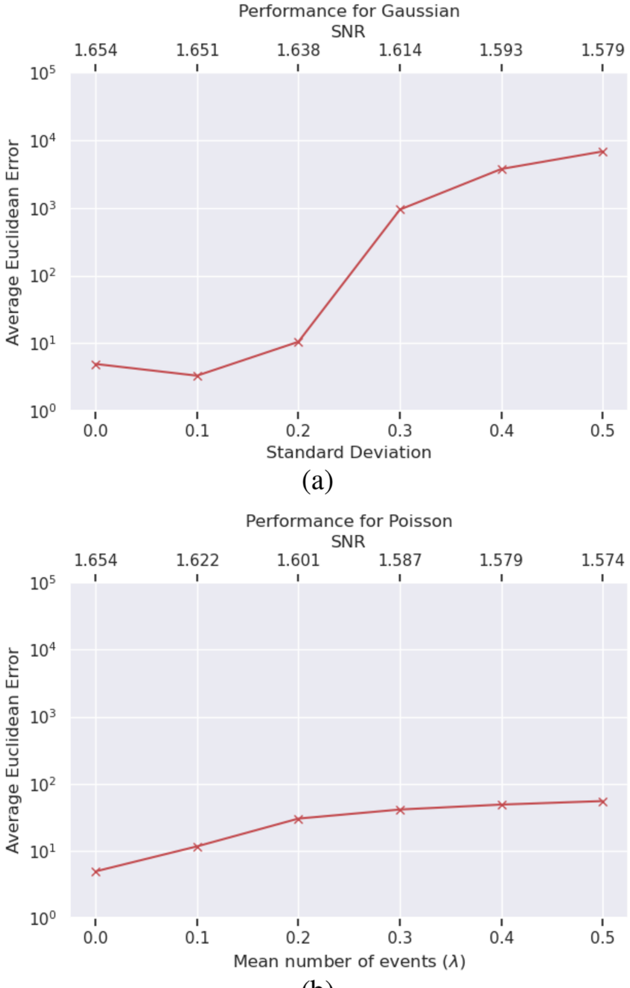


FIG. 7. CNN performance (blue) comparison to *SrcLocOpt* (orange) for 50 data points. Ground truth (red) locations are generated randomly with a circular source of radius 109 μm. Corresponding predictions are connected by a line.

TABLE II. Errors and run times for CNN and *SrcLocOpt*. *SrcLocOpt* is typically more accurate, while CNN requires less computation time.

Method	Mean error (μm)	Median error (μm)	Run time (s)
CNN	26.5	22.8	1.11
SrcLocOpt	60.9	11.4	559.2

they require additional human input and computation time. The table shows that CNN excels at run time, making predictions more than two orders of magnitude faster than *SrcLocOpt*. Therefore, CNN may be preferable in cases where a fast and automated but reasonably accurate prediction is sufficient. As we saw in Sec. II, prediction errors less than 50 μm tend to have little effect on the full source reconstruction. Given that the median error of the CNN prediction is 22.8 μm, we expect that the faster CNN prediction is preferable in most cases.



(a)

(b)

FIG. 8. Change in CNN prediction error (in μm) as a function of noise (bottom x axis) and SNR (top x axis). We randomly sample 50 data points from the validation set. We test for both (a) Gaussian noise as the standard deviation varies and (b) Poisson noise as the mean number of events varies. Note that noise is applied to the standardized images whose pixel values lie heavily in a range between −1 and 1. The CNN performs well for noise levels near typical SNRs (1.636) expected at the NIF.

TABLE III. Data for the NIF shots used in the test set.

Shot no.	Shot	Neutron yield	Shape (μm)	Location (μm)
1	N190318	2.96×10^{15}	(59, 35)	(102, 205)
2	N190707	2.69×10^{15}	(44, 31)	(92, 205)
3	N190721	1.92×10^{16}	(34, 35)	(67, 149)
4	N190730	4.29×10^{15}	(42, 35)	(74, 132)
5	N191007	2.12×10^{15}	(59, 43)	(16, 190)
6	N191013	8.16×10^{14}	(45, 39)	(32, 198)
7	N191021	9.39×10^{13}	(50, 36)	(48, 126)
8	N191105	5.48×10^{15}	(53, 32)	(53, 119)
9	N191110	3.76×10^{15}	(54, 35)	(48, 260)
10	N191126	1.86×10^{14}	(31, 36)	(29, 261)
11	N191202	1.83×10^{14}	(115, 135)	(39, 177)
12	N191229	2.95×10^{14}	(129, 143)	(−1, 157)
13	N200125	2.99×10^{15}	(79, 30)	(54, 177)
14	N200229	4.76×10^{15}	(67, 34)	(46, 246)
15	N200308	2.77×10^{15}	(46, 42)	(69, 200)
16	N200608	5.03×10^{15}	(57, 37)	(−4, 207)
17	N200727	3.35×10^{15}	(56, 44)	(41, 194)
18	N200810	4.62×10^{15}	(46, 75)	(40, 82)
19	N200816	5.91×10^{15}	(49, 29)	(64, 150)
20	N201011	4.80×10^{15}	(44, 44)	(44, 61)
21	N201101	5.74×10^{15}	(38, 54)	(63, 77)
22	N201122	2.04×10^{15}	(44, 36)	(23, −118)
23	N210117	6.40×10^{15}	(55, 55)	(88, 165)
24	N210207	5.99×10^{15}	(44, 40)	(42, −1)
25	N210307	1.67×10^{17}	(46, 38)	(13, 172)
26	N210328	4.86×10^{16}	(56, 34)	(300, 265)
27	N210411	7.52×10^{15}	(40, 32)	(44, 45)
28	N210418	3.42×10^{16}	(36, 50)	(46, 76)
29	N210418	3.42×10^{16}	(36, 50)	(46, 76)
30	N210605	1.33×10^{17}	(38, 38)	(64, 62)
31	N210711	7.56×10^{16}	(44, 46)	(100, 454)
32	N210725	7.39×10^{16}	(55, 39)	(68, −67)
33	N210808	1.37×10^{18}	(59, 55)	(70, 104)
34	N210906	1.93×10^{15}	(56, 53)	(72, 108)
35	N211024	5.24×10^{17}	(36, 48)	(14, 6)
36	N211107	5.58×10^{17}	(52, 52)	(47, 33)
37	N211121	3.34×10^{17}	(50, 51)	(41, 158)
38	N220102	1.39×10^{15}	(50, 50)	(−27, 26)
39	N220109	3.10×10^{17}	(50, 50)	(24, 461)
40	N220115	1.55×10^{14}	(50, 50)	(23, 88)
41	N220124	8.46×10^{15}	(50, 50)	(−21, 0)
42	N220129	1.80×10^{17}	(50, 50)	(52, 161)
43	N220220	9.90×10^{14}	(50, 50)	(31, −24)
44	N220305	4.85×10^{16}	(50, 50)	(39, 48)
45	N220402	1.46×10^{15}	(50, 50)	(24, 54)
46	N220417	1.29×10^{17}	(50, 50)	(−3, 165)
47	N220507	3.78×10^{17}	(50, 50)	(23, 158)
48	N220521	9.65×10^{15}	(50, 50)	(29, 46)
49	N220530	4.53×10^{16}	(50, 50)	(10, 88)
50	N220604	2.16×10^{16}	(50, 50)	(24, 88)
51	N220626	1.59×10^{17}	(45, 47)	(37, 123)
52	N221023	2.37×10^{17}	(47, 37)	(55, 24)
53	N221120	1.87×10^{16}	(37, 43)	(83, 153)
54	N230304	6.57×10^{16}	(46, 10)	(88, 288)
55	N230416	8.39×10^{17}	(54, 47)	(−11, 292)
56	N230508	2.32×10^{17}	(47, 32)	(12, 175)

E. Performance on noisy data

Previous results assumed a noiseless environment. However, noise in NIF experiments can often corrupt the aperture image, which may harm the CNN predictions. Typically, the noise in environments encountered by the neutron imaging system can be described by either a Gaussian or Poisson distribution.[7] In order to confirm the reliability of our models, we require that they perform well under noisy conditions. In this section, we test the performance of the CNN by adding noise to the aperture images and computing the change in prediction error.

Figure 8 shows CNN performance as varying amounts of noise are added to 50 randomly sampled validation data points. In Fig. 8(a), we apply a Gaussian noise distribution with a mean of 0 and vary the standard deviation while measuring the change in the average distance error of the model. In Fig. 8(b), we apply a Poisson noise distribution while varying the mean number of events. We see that under both noise models, the CNN performance degrades as noise is added. However, for low amounts of noise, the CNN prediction error remains near its non-noise performance.

It is important to relate the various noise levels used in Fig. 8 to a typical noise level encountered at the NIF. To make this comparison, we compute a signal-to-noise ratio (SNR) of the validation data for each level of noise added. Additionally, we compute the SNR of the NIF data shown in Table III. By comparing the SNR of the two datasets, we can confirm that the CNN is performing acceptably for expected levels of NIF noise, i.e., the CNN should perform well at the level of SNR computed on the NIF data.

We use the following to compute SNR over the aperture image datasets:

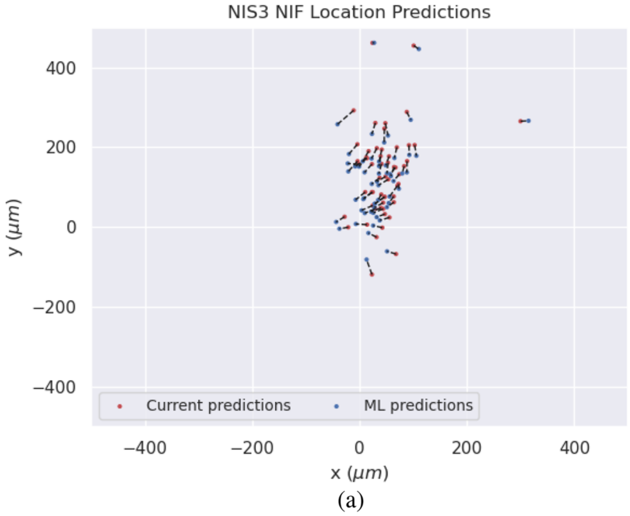
$$SNR_{avg} = \frac{1}{N}\sum_{i=1}^{N}\frac{\mu_i}{\sigma_i}, \tag{4}$$

where N is the number of aperture images in the dataset, μ_i is the mean of aperture image i, and σ_i is the standard deviation of image i. In other words, Eq. (4) calculates the individual SNR[21] for each aperture image and computes an average over the dataset. Using Eq. (4) on the NIF dataset gives an SNR_{avg} of 1.636.

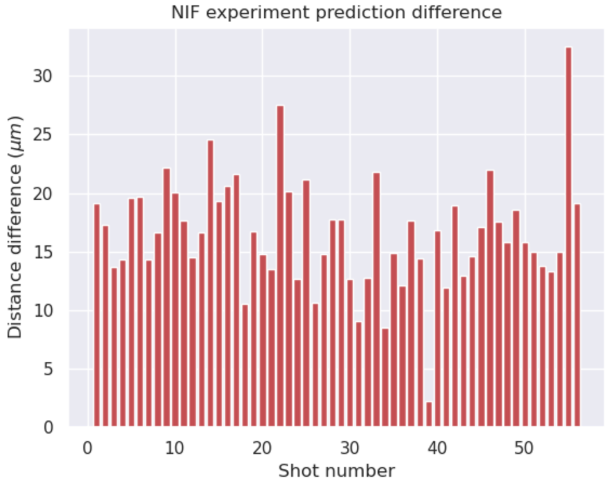
For each noise amount added to the validation data, we calculate SNR_{avg} using Eq. (4). We plot this SNR_{avg} on the top x axis in Fig. 8. We see that the CNN performs well for validation SNR_{avg} greater than or near the NIF SNR_{avg} of 1.636. The noise levels at which the CNN performs poorly correspond to an SNR_{avg} lower than the level typically expected at the NIF. Therefore, we expect the CNN to perform well under NIF noise conditions.

F. Performance on NIF data

We now discuss the performance of CNN on NIF experiment data. We collect the 56 recent shots shown in Table III and test the CNN performance against *SrcLocOpt* on the NIS-3 line-of-sight. Figure 9(a) shows the results over the NIS-3 field-of-view. We see that the CNN predictions are closely aligned with the *SrcLocOpt* predictions. It is important to note that since these are experiment data, no ground truth source locations are known. Therefore, we cannot say for certain which algorithm is more accurate in each of these cases. Regardless, the strong agreement between both algorithms gives us confidence in the performance of CNN. Figure 9(b) shows the Euclidean distance between predictions for the various shots. We see that in most cases, the difference is less than 20 μm. As we saw in Sec. II, this performance is sufficient to ensure an accurate reconstruction. Therefore, given its additional speed advantage over *SrcLocOpt*, CNN is the preferred algorithm for use in real NIF data.



(a)



(b)

FIG. 9. (a) Comparison of predictions for NIS-3 on NIF data for *SrcLocOpt* (red) and the CNN (blue). Corresponding predictions are connected by a line. (b) Euclidean distance for predictions using the two methods. The shot number here corresponds to the shot number in Table III. Most predictions are fewer than 20 μm apart.

If we require additional accuracy, a possible alternative is to combine the two algorithms and receive benefits from both. Currently, *SrcLocOpt* starts optimization using the field-of-view center as the initial location guess. Instead, we can obtain a quick prediction using the CNN and provide this as the *SrcLocOpt* starting state. In doing so, we reduce the number of required iterations and speed up the *SrcLocOpt* optimization process while providing accuracy guarantees.

V. CONCLUSION

Due to the complexity of fusion geometry reconstruction from neutron aperture images, it is critical to determine the source location within the field-of-view. In this paper, we developed a machine learning system based on CNNs to perform source localization. We showed that CNNs can predict source locations two orders of magnitude faster than previous optimization approaches while obtaining comparable accuracy. In addition, we showed that CNNs are robust to noise that may affect the aperture. Finally, we demonstrated the real-world applicability of our approach on the NIS-3 line-of-sight at the NIF.

There are several avenues for future work. First, in this work, we trained the CNN entirely with synthetic data generated by the forward model. We can likely improve performance by generating more data, e.g., either more locations or source shapes. In addition, to improve the model's performance on NIF data, we can utilize transfer learning[22] approaches to train the model on both synthetic and real NIF data. Second, we can improve performance by combining an ensemble of ML models.[23] Ensemble methods can improve robustness and increase confidence in the ML system. Third, our models were trained using non-noisy data. By incorporating noisy data into the training set, we can further improve the robustness of the models in noisy environments.

ACKNOWLEDGMENTS

We would like to acknowledge the dedicated staff at NIF for their contributions and hard work. This work was supported by the U.S. Department of Energy through the Los Alamos National Laboratory (LANL), operated by Triad National Security, LLC, for the National Nuclear Security Administration under Contract No. 89233218CNA000001.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Gary Saavedra: Conceptualization (equal); Data curation (equal); Formal analysis (lead); Investigation (lead); Software (equal); Writing – original draft (lead); Writing – review & editing (lead). **Verena Geppert-Kleinrath**: Data curation (equal); Funding acquisition (lead); Project administration (equal); Software (equal). **Chris Danly**: Conceptualization (equal); Data curation (equal); Software (equal). **Mora Durocher**: Data curation (equal); Software (equal). **Carl Wilde**: Data curation (equal); Software (equal). **Valerie Fatherley**: Data curation (equal). **Emily Mendoza**: Data curation (equal). **Landon Tafoya**: Data curation (equal). **Petr Volegov**: Conceptualization (equal); Data curation (equal); Software (equal). **David Fittinghoff**: Data curation (equal); Project administration (equal); Writing – review & editing (supporting). **Michael Rubery**: Data curation (equal). **Matthew S. Freeman**: Conceptualization (equal); Data curation (equal); Project administration (equal); Software (equal); Writing – review & editing (supporting).

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

APPENDIX: INNER WORKINGS OF NEURAL NETWORKS

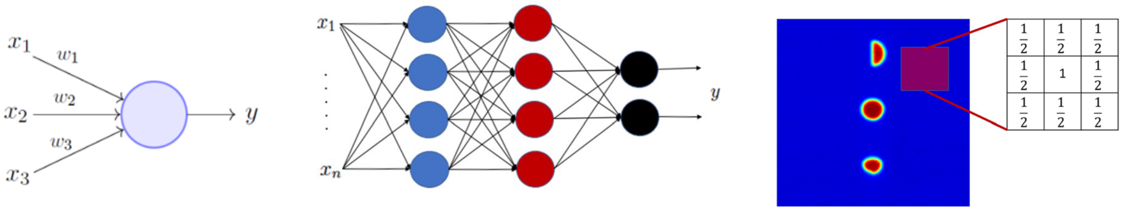
Figure 10(a) shows an example of a neuron used in an FNN. We adopt the common terminology within machine learning and refer to the neuron as a *node* throughout this text. Given an input x, a node outputs variable y according to the following weighted summation:

$$y = \sum_{i=1}^{n} w_i x_i + b, \tag{A1}$$

where x_i is *feature* i of the input variable x, w_i is weight i of the node, and b is an additional bias term. The features that describe the input are often dataset-specific. In imaging applications, a single pixel is often considered a feature. Therefore, a single node may take a weighted summation over every pixel in the image.

Non-linearities are often applied to the output of a neuron to improve the representational ability of the neuron and the network as a whole. A common non-linearity is the Rectified Linear Unit (ReLU),

$$f(y) = \max(0, y). \tag{A2}$$



(a) (b) (c)

FIG. 10. (a) A single neuron with three inputs and one output. Each input is multiplied by the corresponding edge weight. The output is the summation of these weighted inputs. (b) A feedforward neural network constructed with layers of neurons. The output of every neuron in the preceding layer is fed into the input of every node in the next layer. (c) An example of a 3 × 3 convolution kernel over an aperture image. The kernel is applied to every location in the image to produce a convolved image.

We utilize the ReLU in this paper as it has been shown to result in superior convergence properties for neural networks compared to other activation functions.[24]

On their own, single nodes have limited representational ability and, therefore, can only solve very simple problems. Therefore, nodes are often combined to improve the representation power, resulting in a FNN. A FNN is constructed with nodes arranged into a series of layers, as shown in Fig. 10(b). In a FNN, inputs flow through the layers, transforming the input data into a set of output variables. The combination of nodes into layers increases the representational capacity of the network and allows for the solution of more complex problems.

CNNs use image convolutions to find localized patterns within an image. A convolution is a weighted summation similar to Eq. (A1) but applied to small neighborhoods of pixels in the image using a weighted filter, as shown in Fig. 10(c). The convolution is applied iteratively over the whole image to produce various convolved images, often referred to as channels. The final output channels are then input into a FNN for final prediction. For a more in-depth discussion of neural networks, see Goodfellow *et al.*[25]

REFERENCES

[1] H. Abu-Shawareb, R. Acree, P. Adams, J. Adams, B. Addis, R. Aden, P. Adrian, B. Afeyan, M. Aggleton, L. Aghaian *et al.*, "Lawson criterion for ignition exceeded in an inertial fusion experiment," Phys. Rev. Lett. **129**, 075001 (2022).

[2] F. Merrill, D. Bower, R. Buckles, D. Clark, C. Danly, O. Drury, J. Dzenitis, V. Fatherley, D. Fittinghoff, R. Gallegos *et al.*, "The neutron imaging diagnostic at NIF (invited)," Rev. Sci. Instrum. **83**, 10D317 (2012).

[3] J. Valente, J. António, C. Mora, and S. Jardim, "Developments in image processing using deep learning and reinforcement learning," J. Imaging **9**, 207 (2023).

[4] V. Geppert-Kleinrath, M. S. Freeman, C. R. Hurlbut, F. Merrill, J. R. Tinsley, P. Volegov, and C. Wilde, "A liquid VI scintillator cell for fast-gated neutron imaging," Rev. Sci. Instrum. **89**, 10I142 (2018).

[5] C. Danly, K. Christensen, V. E. Fatherley, D. Fittinghoff, G. Grim, R. Hibbard, N. Izumi, D. Jedlovec, F. E. Merrill, D. W. Schmidt *et al.*, "Combined neutron and x-ray imaging at the National Ignition Facility (invited)," Rev. Sci. Instrum. **87**, 11D703 (2016).

[6] P. Volegov, C. Danly, D. Fittinghoff, G. Grim, N. Guler, N. Izumi, T. Ma, F. Merrill, A. Warrick, C. Wilde, and D. C. Wilson, "Neutron source reconstruction from pinhole imaging at national ignition facility," Rev. Sci. Instrum. **85**, 023508 (2014).

[7] P. Volegov, C. Danly, D. Fittinghoff, N. Guler, F. Merrill, and C. Wilde, "Self characterization of a coded aperture array for neutron source imaging," Rev. Sci. Instrum. **85**, 123506 (2014).

[8] J. A. Nelder and R. Mead, "A simplex method for function minimization," Comput. J. **7**, 308–313 (1965).

[9] C. J. Price, I. D. Coope, and D. Byatt, "A convergent variant of the Nelder–Mead algorithm," J. Optim. Theory Appl. **113**, 5–19 (2002).

[10] N. Guler, P. Volegov, C. Danly, G. Grim, F. Merrill, and C. Wilde, "Simultaneous usage of pinhole and penumbral apertures for imaging small scale neutron sources from inertial confinement fusion experiments," Rev. Sci. Instrum. **83**, 10D316 (2012).

[11] G. R. Yang and X.-J. Wang, "Artificial neural networks for neuroscientists: A primer," Neuron **107**, 1048–1070 (2020).

[12] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," IEEE Trans. Neural Networks Learn. Syst. **33**, 6999–7019 (2022).

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016), pp. 770–778.

[14] T. Yu and H. Zhu, "Hyper-parameter optimization: A review of algorithms and applications," arXiv:2003.05689 (2020).

[15] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," Neurocomputing **415**, 295–316 (2020).

[16] D. Wilson, G. Grim, I. Tregillis, M. Wilke, M. Patel, S. Sepke, G. Morgan, R. Hatarik, E. Loomis, C. Wilde *et al.*, "Modeling the national ignition facility neutron imaging system," Rev. Sci. Instrum. **81**, 10D335 (2010).

[17] A. Moore, D. Schlossberg, B. Appelbe, G. Chandler, A. Crilly, M. Eckart, C. Forrest, V. Glebov, G. Grim, E. Hartouni *et al.*, "Neutron time of flight (nToF) detectors for inertial fusion experiments," Rev. Sci. Instrum. **94**, 061102 (2023).

[18] M. Shanker, M. Y. Hu, and M. S. Hung, "Effect of data standardization on neural network training," Omega **24**, 385–397 (1996).

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv:1412.6980 (2014).

[20] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 32 (NeurIps, 2019).

[21] M. Mahesh, "The essential physics of medical imaging, third edition," Med. Phys. **40**, 077301 (2013).

[22] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," Proc. IEEE **109**, 43–76 (2021).

[23] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," Front. Comput. Sci. **14**, 241–258 (2020).

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 25 (NeurIPS, 2012).

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016).