Extrapolation of nuclear structure observables with artificial neural networks

W. G. Jiang ●, G. Hagen, and T. Papenbrock

Department of Physics and Astronomy, University of Tennessee, Knoxville, Tennessee 37996, USA
and Physics Division, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831, USA

Calculations of nuclei are often carried out in finite model spaces. Thus, finite-size corrections enter, and it is necessary to extrapolate the computed observables to infinite model spaces. In this work, we employ extrapolation methods based on artificial neural networks for observables such as the ground-state energy and the point-proton radius. We extrapolate results from no-core shell model and coupled-cluster calculations to very large model spaces and estimate uncertainties. Training the network on different data typically yields extrapolation results that cluster around distinct values. We show that a preprocessing of input data, and the inclusion of correlations among the input data, reduces the problem of multiple solutions and yields more stable extrapolated results and consistent uncertainty estimates. We perform extrapolations for ground-state energies and radii in ^{4}He, ^{6}Li, and ^{16}O, and compare the predictions from neural networks with results from infrared extrapolations.

I. INTRODUCTION

In nuclear physics, *ab initio* methods aim to solve the nuclear many-body problem starting from Hamiltonians with two- and three-nucleon forces using controlled approximations [1–9]. Most of these methods employ finite model spaces, and this makes it necessary to account for finite-size corrections or to extrapolate the results to infinite model spaces. While light nuclei with large separation energies require little or no extrapolations, finite-size effects are non-negligible in weakly bound nuclei or heavy nuclei. Various empirical extrapolation schemes [10–14] have been used. More recently, rigorous extrapolation formulas were derived based on an understanding of the infrared and ultraviolet cutoffs of the harmonic oscillator basis [15–22]. These extrapolation formulas are akin to Lüschers formula [23] derived for the lattice and its extension [24] to many-body systems. Unlike the lattice, however, the harmonic oscillator basis mixes ultraviolet and infrared cutoffs, and this complicates extrapolations. Very recently, Negoita and coworkers [25,26] employed artificial neural networks for extrapolations. They trained a network on no-core shell model (NCSM) results obtained in various model spaces, i.e., for various oscillator spacings $\hbar\omega$ and different numbers $N_{\max}\hbar\omega$ of maximum excitation energies. In practical calculations, $N_{\max} \approx 10 \ldots 20$ in light nuclei. The neural network then predicted extrapolations in very large model spaces of size $N_{\max} \sim 100$. Impressively, the neural network also predicted that the ground-state energies and radii cease to depend on the oscillator spacing as $N_{\max}$ increases. Negoita and coworkers employed about 100 neural networks, each differed by the initial set of parameters (weights) from which the training started. The resulting distributions for observables occasionally exhibited a multimode structure stemming from multiple distinct solutions the neural networks arrived at. In this work, we want to address this challenge and focus on the network robustness and avoidance of multiple solutions.

In recent years, artificial neural networks have been used for various extrapolations in nuclear physics [27–35], and for the solution of the quantum many-body system [36]. Artificial neural networks use sets of nonlinear functions to describe the complex relationships between input and output variables. The universality of using artificial neural networks to solve extrapolation problems is largely guaranteed, because no particular analytical functions are needed. Artificial neural networks are controlled by two hyperparameters, i.e., the number of layers and the number of neurons for each layer.

There are still two major challenges when introducing neural networks in extrapolations of results from *ab initio* computations. First, unlike other applications in which large amounts of training data can be acquired, the inputs provided by the *ab initio* calculations are limited to small data sets. The statistics is clearly not enough to support the network training without overfitting. Secondly, randomness, caused by the nature of basic network algorithms, is an intrinsic quality of the neural network that conflicts with the high-precision requirement for extrapolations.

In this work, we use an artificial neural network and extrapolate observables computed with the NCSM and coupled-cluster (CC) methods. In addition to standard techniques such as regularization, we use interpolation of data to mitigate the overfitting problem and also take into account the correlations in the resulting data set. The random initialization of the network parameters provides us with a "forest" of artificial neural networks. This allows us to gain insights into uncertainties of the extrapolated observables, under the precondition that the distribution of extrapolation results has a single peak.

We note here that the extrapolation problem we are concerned with is special in the sense that a well-defined asymptotic value exists for the observable of interest (i.e., an energy or a radius), that there is a simple pattern in the learning data, and that the learning data is already close to this asymptotic value. We will see below that this makes an artificial neural network a useful tool for this kind of extrapolation. Needless to say, for a general problem there is no tool to extrapolate: We cannot extrapolate from available data to next week's stock market value or next month's weather. We refer the reader to the literature for attempts to use deep learning in extrapolations [37], and for a counterexample [38].

This paper is organized as follows. In the next section we introduce the theoretical framework and artificial neural networks and present a detailed account of how we construct, train, and use neural networks. We then present and discuss the extrapolation results for ^{4}He, ^{6}Li, and ^{16}O. Finally, we summarize our work.

II. THEORETICAL FRAMEWORK

A. Artificial neural network architecture

An artificial neural network is a computing system that consists of a number of interconnected blocks which process the input information and yield an output signal. Modeled loosely after the human brain, the neural network is typically organized by similar blocks called "layers," and each layer contains a certain number of parallel "neurons." The numbers of layers and neurons define the depth and the width of the neural network, respectively.

Figure 1 shows the schematic structure of a simple feed-forward neural network. The algorithm basically consists of two parts. First, the input signal x is propagated to the output layer y by a series of transformations. The whole network can be seen as a complex function between the input and output variables. In the simple case with one hidden layer, the
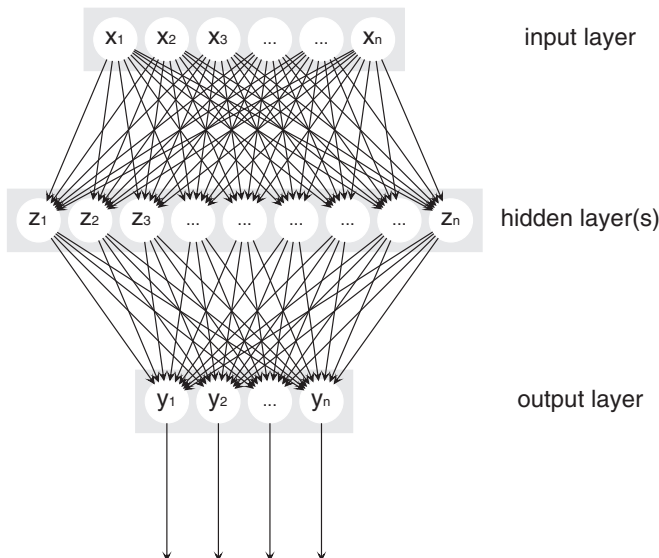

FIG. 1. Schematic structure of a typical feed-forward neural network.

function can be written as follows,

$$z_j = \sum_i x_i w_{ij} + b_j, \tag{1}$$

with σ as the activation function,

$$x'_j = \sigma(z_j), \tag{2}$$

$$y_k = \sum_j x'_j w'_{jk} + b'_k. \tag{3}$$

Here, x_i are the input variables, and y_k are the output variables. The weights w (w') and bias b (b') are free parameters of the neural network. There exist a few choices one can make for the activation function σ, such as the sigmoid, tanh, and Rectified Linear units (ReLu). These are nonlinear functions which enable the neural network to capture complex nonlinear relationships between variables. For the extrapolation we follow Ref. [26] and use a smooth activation function that only acts on the hidden layer. Back-propagation is the second part of the algorithm [39]. This is the central mechanism that allows neural network methods to "learn." The error signals, often referred to as the "loss," which measure the deviation between the predicted output y_{pre} and the training target y_{true}, are propagated backwards to all the parameters of the network and allow the optimizer to update the network status accordingly. Note that, in practice, the neural network always processes the data in batches, which makes the input (output) signals x (y) matrices and the network functions become matrix operations.

To construct the artificial neural network aiming to solve the extrapolation problem, we first need to determine its topological structure. There are a lot of variants for neural networks, such as recurrent neural network (RNN), long short-term memory (LSTM), and convolutional neural network (CNN), which are designed for various assignments. One should choose the appropriate type of network according to the organizational structure of the data set and the goal that one wants to achieve. In the case of extrapolation, the data for training is assigned to a structure consisting of three members, namely $\hbar\omega$, N_{max}, and the corresponding target observables, i.e., the ground-state energy and the point-proton radius. On the other hand, the main purpose of the neural network is to provide reasonable predictions for the observables at any values of $\hbar\omega$ and N_{max}. In this paper we use the feed-forward neural network, which takes the $\hbar\omega$ and N_{max} as two inputs (x) and the target observables as output (y_{true}). One could as well apply the RNN structure to achieve the same goal. The only difference between the two choices is that the data structure needs to be reorganized in terms of sequential observable values with increasing N_{max} under the same $\hbar\omega$.

Once the basic structure is decided, the next task is to control the complexity of the network. The network's ability to describe complex features is determined by the numbers of hidden layers and neurons in each layer. In other words, the depth and the width of the neural network control the upper limit of the neural network description. Ideally, to lower the loss of the training data set, adding more layers and neurons is always helpful to increase its accuracy. However, as the neural network becomes more complex it becomes harder to train.
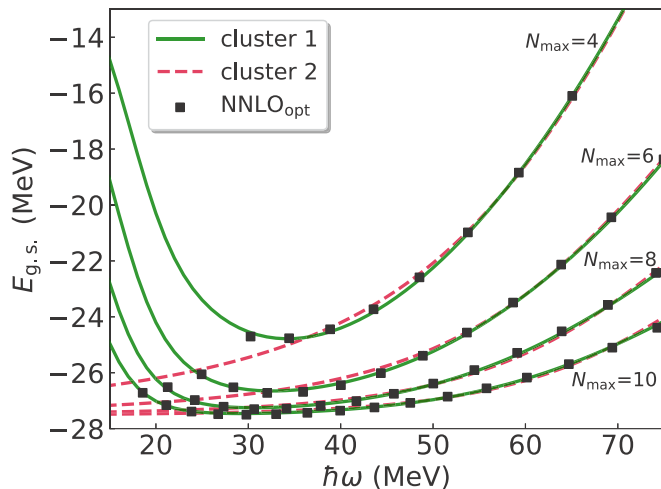
FIG. 2. Ground-state energies from NCSM computations of ^{4}He based on the NNLO$_{opt}$ potential (black data points). The green full line and the red dashed line show two different neural network solutions for learning the ground-state energy of ^{4}He in finite model spaces.

Given the same amount of training data, a deeper and wider network requires more time to get converged results, and one risks overfitting of the network's parameters. In extreme cases, for instance, when the network is so complex that it has many more parameters than the number of input data, it can easily get 100% of accuracy on the training set, but still perform poorly on testing samples. Instead of learning the pattern, the network simply memorizes the training data and exhibits no predictive power.

Even though there is no exact answer for how to configure the numbers of layers and neurons in the neural network, there are still some guiding principles to follow. For a start, we consider a network with one hidden layer. Based on the universal approximation theorem [40–42] any continuous function can be realized by a network with one hidden layer. Of course, a deep neural network (with multiple hidden layers) will have certain advantages over the shallow one (with few hidden layers). For example, the deep neural network can reach the same accuracy of a shallow one with many fewer parameters [43–45]. However, to prevent problems such as vanishing gradients and overfitting, the architecture of the deep neural network needs careful construction including, but not limited to initialization of the network parameters [46], design of the activation function [47], using the proper optimizer [48], and improving the training procedure [49]. For our task of extrapolation, a deep neural network would be overkill. As for the numbers of neurons, there are several empirical rules [50] and techniques, such as pruning [51], that can be applied. In the present work, we start with a simple structure and then increase the numbers of neurons and layers until we arrive at a sufficiently small loss for the training data set. For the results shown below, we arrived at neural networks with a single hidden layer, consisting of eight and 16 nodes for the extrapolation of energies and radii, respectively.

Figure 2 shows some of the data we used in extrapolations of the ground-state energy of ^{4}He. The black points, taken

from Ref. [22], denote results from NCSM computations based on the NNLO$_{opt}$ potential [52]. The ground-state energies are shown as a function of the oscillator frequency and labeled by the number N_{max} of employed oscillator excitations.

Figure 2 also shows that the data exhibit a simple pattern, namely U-shaped curves that get wider and move closer together as N_{max} increases (see also Fig. 8 for another example.) To capture this behavior with an artificial neural network, we choose a sigmoid as the activation function, i.e., $\sigma(x) = (1 + e^{-x})^{-1}$. It is then clear that asymptotic values of large N_{max} either map to zero or to one in the activation function, and this explains why—by design—an asymptotically flat function results in the extrapolation. Indeed, using a ReLu function as the activation function [i.e., $\sigma(x) = \max(0, x)$] leads to noisy extrapolation results.

B. Data interpolation and correlated loss

Despite the fact that we can easily design a neural network that gives satisfactory accuracy on training data, a good performance on making predictions is not guaranteed for the extrapolation problem. More often than not the loss of the testing data will be much larger than the loss of the training data, which is a clear sign of overfitting. Overfitting is a major issue for neural network applications, which is usually caused by the conflict between having insufficient information from a limited data set, and the network flexibility to approximate complex nonlinear functions. This is exactly the case for the *ab initio* extrapolation task at hand. The *ab initio* calculations are restricted to a not-too-large value of N_{max}, and for a given N_{max} only a few oscillator spacings $\hbar\omega$ are available. In the case of ^{4}He, for instance, we only have 144 data points from NCSM calculations, and this is is inadequate for training even a very simple neural network, thus overfitting seems inevitable.

There are a few strategies that can be introduced to avoid overfitting in neural networks, including regularizations [53], dropout [54], and early stopping [55]. Such methods can be used together or separately to increase the network robustness and reduce generalization errors. The price to pay is that one will have to deal with more hyperparameters and determine the best combination of them. In addition to these methods, one of the best ways to reduce overfitting is to enlarge the data set. In our case, however, the commonly used practice of data augmentation [56] and addition of random noise to the data set will not be helpful, because extrapolation is a quantitative problem that requires high accuracy and input data with a clear physical foundation.

To enlarge the data set, we note that the *ab initio* calculations for a given N_{max} should give a continuous smooth curve for the target observable values as a function of $\hbar\omega$. The limited input data is merely restricted by the computation cost but not by the method itself. Thus, performing interpolation on existing data is an economical way to obtain more information. In this work, we employ a quadratic spline for interpolation in $\hbar\omega$ at fixed N_{max}. This procedure increases the robustness of the neural network even with the basic single-hidden-layer architecture and avoids overfitting.

As a large portion of the training data is generated by interpolation, the standard "χ^2" loss function (valid for independent data) might not be appropriate. As the generation of n points via interpolation yields n correlated samples, we introduce the correlated loss function,

$$L = \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} R_i R_j. \tag{4}$$

Here W_{ij} are the elements of a correlation matrix, and $R_i(R_j)$ are the residuals of the y_{pre} and the target y_{true}. In this work, we will either consider the absence of correlations (i.e., $W_{ij} = \delta_{ij}$) or include correlations as described in what follows. The elements W_{ij} form a block matrix, because only points interpolated at fixed $N_{\max}$ are correlated by the spline. For fixed $N_{\max}$ the block matrix is taken to be tridiagonal with all nonzero matrix elements equal to one. This indicates that the correlation is only between neighboring data points. We note that the loss function (4) is usually not a built-in function for much of the mainstream neural network development environments. Thus, we employ a customized loss function, and the position i or j of each data point is needed as an additional input for the network to generate the correlation matrix with elements W_{ij}.

Training a neural network starts with a random initialization of the network parameters (weights and biases). During training the loss function is minimized using the training data set as input. It is clear that the random starting points will lead to different trained networks, because optimizers can generally not find the global minimum of the loss function. The existence of many local minima with an acceptable loss will thus lead to different network predictions.

Inspired by the random forest algorithm [57], in which the decision forest always gives better performance than a single decision tree, we introduce multiple neural networks with the same structure but with different initialized parameters to address the uncertainty problem. The outputs of all the networks are being integrated to obtain a range of predictions and uncertainty estimates. This approach is going to help us to reveal some insights into neural networks, and guide us in selecting favorable neural network solution.

Figure 3 demonstrates the impact of including correlations into the loss function. The left panels shows the predictions of 100 neural networks for the ground-state energy of ^{4}He. The input data consist of NCSM data for model spaces with a maximum value of $N_{\max}$ as indicated, and the correlation matrix W of Eq. (4) is taken to be diagonal, i.e., no correlations are included. The displayed ground-state energies are the neural network predictions for $N_{\max} = 100$, and there is virtually no $\hbar\omega$ dependence. The shown distribution function results from kernel density estimations (KDE), i.e., by replacing the delta function corresponding to each individual data point with a Gaussian. The distribution becomes narrower as the input data include increasing values of $N_{\max}$. We note that the distributions are bi-modal.

The inclusion of correlations, shown in the right panel of Fig. 3, somewhat reduces the importance of the smaller peak. The main peaks, which include most of the network results, exhibit a smaller average loss and therefore are believed to
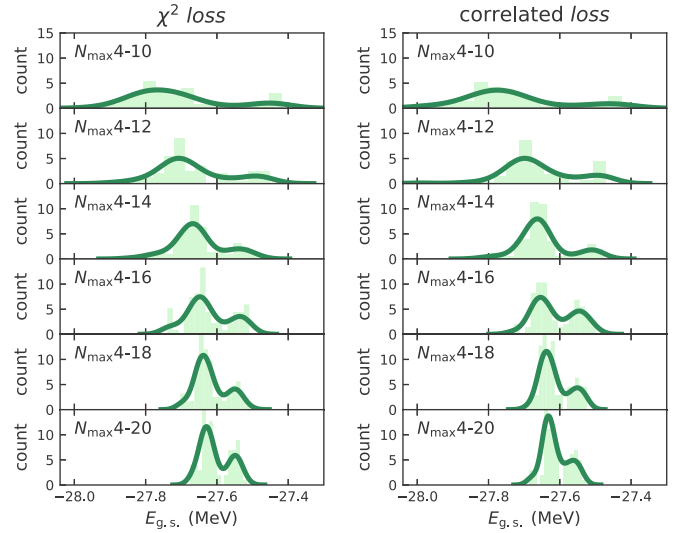

FIG. 3. Distributions of multiple neural network trained with different max($N_{\max}$) data sets for ground-state energy of ^{4}He using χ^2 loss function (left panel) and correlated loss (right panel).

be the better solution. Their central values are likely to be the best predictions for these networks. However, for uncorrelated and correlated loss functions, the second peak does not appear by accident and cannot be neglected. Its persistence against different optimizers and hyperparameter adjustments shows that it is a stable local minimum and not too narrow. From this point of view, both peaks can be treated as the solutions of the multiple neural networks. As the maximum $N_{\max}$ of the input data is increased, the two peaks are getting closer to each other but remain distinguishable. Thus, a significant uncertainty remains.

C. Multiple neural network and data preprocessing

We want to understand the bi-modal structure of the distribution functions. For this purpose, we focus on the correlated loss function. Figure 4 presents results from 100 neural networks for the correlated loss versus the ^{4}He ground-state energy $E_{\text{g.s.}}$. Each cross in Fig. 4 represents one fully trained neural network and has already reached convergence (i.e., the loss shift is within a required accuracy). As before, the shown distribution function results from KDE. Each individual data point (crosses) and contour lines are also shown. The top and right panels show the integrated distributions for the ground-state energy and the loss, respectively.

We understand the double-peak structure as follows. The cluster of networks under the dominant peak predict a U shape for the curves $E_{\text{g.s.}}(\hbar\omega, N_{\max})$ at fixed $N_{\max}$. However, they deviate in "higher-order" terms that define the precise shape. The smaller cluster of networks under the small peak predict curves $E_{\text{g.s.}}(\hbar\omega, N_{\max})$ that increase monotonically as a function of $\hbar\omega$. They have a higher loss. This interpretation is based on the results shown in Fig. 2. Here, the black squares are the input data of ground-state energies for given $\hbar\omega$ and $N_{\max}$. The green full lines show predictions from the first cluster of networks under the dominant peak of Fig. 4. In
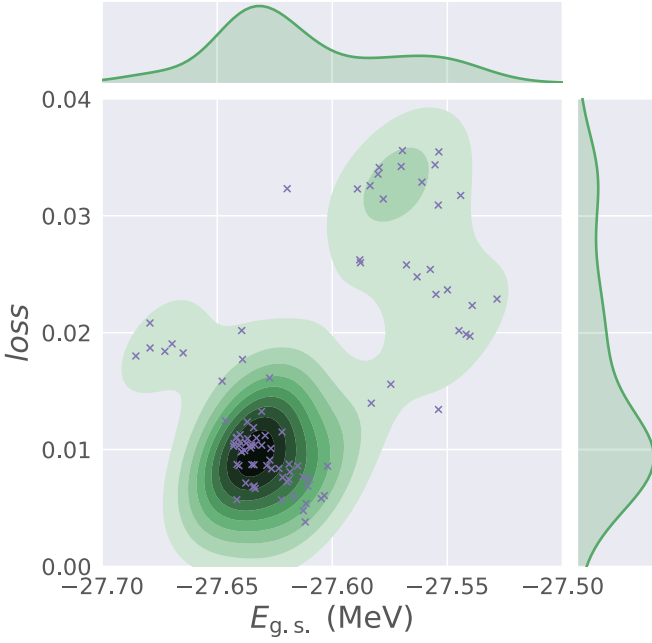
FIG. 4. Multiple neural network predictions for extrapolating ground-state energy of ^{4}He with NCSM calculated data set $\max(N_{\max}) = 20$ as input. Kernel density estimations for $E_{\text{g.s.}}$ and loss are also given along side. The calculation contains 100 independent random initialized neural network.

FIG. 5. Distributions of multiple neural network for ground-state energy of ^{4}He, with the origin data sets (left panel) and with the preprocessed data sets (right panel).

contrast, the red dashed lines are predictions from the second cluster of networks under the smaller peak in Fig. 4. It is evident that the networks of cluster 1 learned the pattern of all data while those of cluster 2 failed to predict the trend of the data points at smaller $\hbar\omega$. How did the neural networks of cluster 2 make this mistake?

Inspection showed that the imbalanced data set is the root of the problem. Our data set includes many points at relatively large $\hbar\omega$ values (as we used such ultraviolet converged points for infrared extrapolations in Ref. [22]), and the corresponding ground-state energies are also much above the variational minimum and the infinite-space result. In contrast, the data set contains a smaller number of data points at relatively small values of $\hbar\omega$, and the corresponding ground-state energies are much closer to the infinite-space result. Thus, the failure to correctly learn about these "minority" data points yields a relatively small increase of the loss function. With random parameter initialization, once the network reaches a local minimum, the imbalanced data set will, to a large extent, prevent the optimizer from pulling the network out of it. Furthermore, with the imbalanced training data, the effort of emphasizing the minority data directly conflicts with the idea of reducing overfitting. Some of the common neural network strategies, such as adding a regularization term, will make things worse. In contrast, removing data points at too large values of $\hbar\omega$ from the training data set, or a stronger weighting of data closer to the variational minimum (at fixed $N_{\max}$) in the loss function, reduces the number of trained networks that would fall into cluster 2.

In the *ab initio* calculation, when the $\hbar\omega$ of the harmonic oscillator basis is too large or too small [i.e., it deviates from
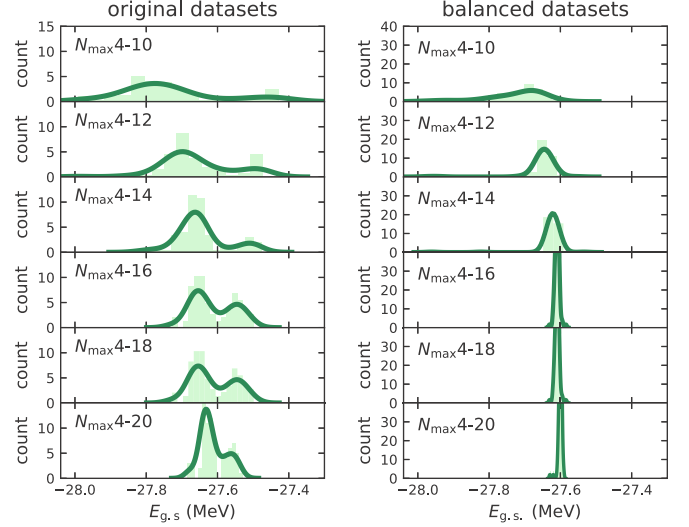
the "optimal" value $\hbar\omega \approx \hbar^2 \Lambda / (mR)$, where Λ and R are the scales set by the cutoff of the potential and the radius of the computed nucleus [58]], the convergence with respect to the increasing $N_{\max}$ is slow, because the employed basis is not efficient to capture ultraviolet and infrared aspects of the problem. The data points that we are most interested in are close to the variational minimum at fixed $N_{\max}$. To overcome the problem of the imbalanced data set, we apply Gaussian weights on the input data, using the values of the minima for the centroids and a standard deviation of about 8.5 MeV. The networks are trained using these weights and a correlated loss function. Figure 5 shows the comparison of multiple neural network results with and without sample weights. We note that the two panels have different ranges for the y axis to better display the distribution of the ground-state energy. Training with the original data sets (left panel) yields the bi-modal distribution. Introducing balanced data sets via Gaussian weights (right panel) suppresses the second peak and leaves us with one solution for the extrapolation problem. At the same time, this improves the precision of the predicted observable and thus yields a smaller uncertainty for the neural network extrapolation.

We note here that the increased weighting of points close to the variational minima is akin to employing a prior in Bayesian statistics. Such techniques could also be used for a quantification of uncertainties [35,59–61]. In this work, we limit ourselves to uncertainty estimates.

III. RESULTS AND DISCUSSIONS

We now present the results of the neural networks' predictions for ground-state energies and radii, and compare with other extrapolation methods. We start with the nucleus ^{4}He. The networks are trained separately for the ground-state energy and radius. The data sets are generated by NCSM calculations using the NNLO$_{\text{opt}}$ nucleon-nucleon interaction.
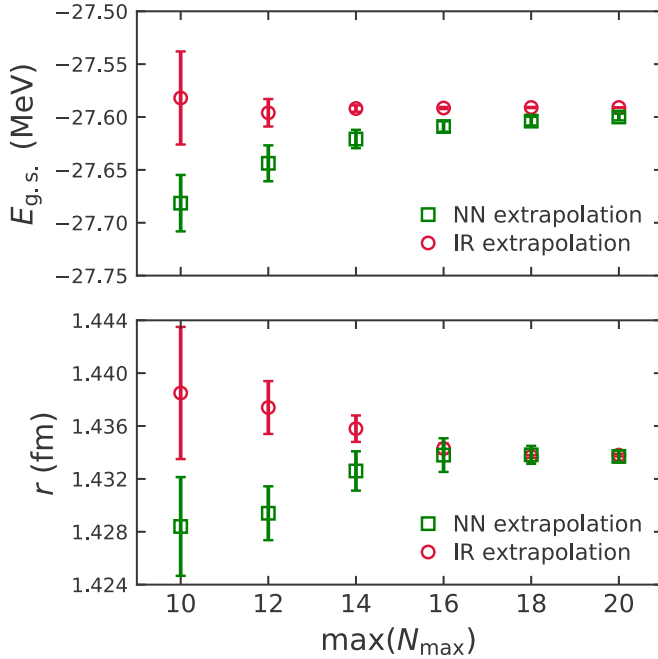
FIG. 6. Extrapolated results for ^{4}He ground-state energy (upper panel) and point-proton radius (lower panel) with NCSM data sets from $\max(N_{\max}) = 10$ to $\max(N_{\max}) = 20$ employing neural network (squares) and IR (circles) extrapolation. Error bars represent the uncertainties of the extrapolations that are from changes in the initial point in the training process.

FIG. 7. Extrapolated results for ^{6}Li ground-state energy (upper panel) and point-proton radius (lower panel) with NCSM data sets from $\max(N_{\max}) = 12$ to $\max(N_{\max}) = 22$ employing neural network (squares) and IR (circles) extrapolation. Error bars represent the uncertainties of the extrapolations that are from changes in the initial point in the training process.

Because the four-nucleon bound state of ^{4}He is already well converged with the maximum model space that NCSM calculation can reach, it is a good case to perform a benchmark and study the performance of the neural network extrapolations. The networks are trained with different data sets which contain the NCSM results from $N_{\max} = 4$ to the given $\max(N_{\max})$. For ^{4}He, six data sets with $\max(N_{\max}) = 10$ to $\max(N_{\max}) = 20$ are given, providing the neural network with a sequence of mounting information. The extrapolation result for the single neural network is given by the prediction of $N_{\max} = 100$ when the observable value is virtually constant in the interval $10\,\mathrm{MeV} < \hbar\omega < 60\,\mathrm{MeV}$. With each data set, the multiple neural network (containing 100 networks) is trained with randomly initialized network values. The distribution of the multiple neural network results is then fitted by the Gaussian function. Finally, the recommended values of the multiple neural networks are set to be the mean value μ and the uncertainties are defined as the standard deviation σ of the Gaussian.

Figure 6 shows the predictions and corresponding uncertainties for the neural network approach compared with the values obtained from the infrared (IR) extrapolations of Ref. [22]. The error bars reflect the variations that are from changes in the initial point in the training process. As we can see, the uncertainty of the neural network predictions decreases with increasing $\max(N_{\max})$. This indicates that the network is learning the pattern as the data set is enlarged. The neural networks reach convergence after $\max(N_{\max}) = 16$ and their predictions agree with the IR extrapolations for both the
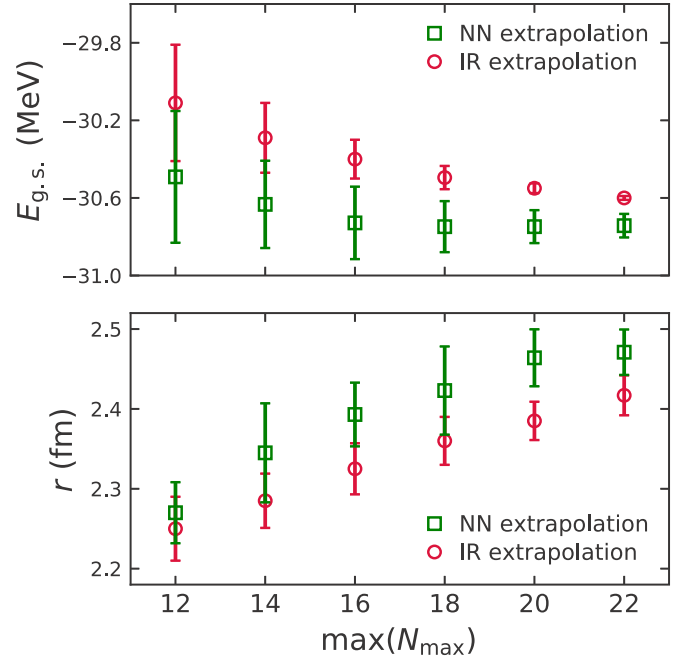
ground-state energy and point-proton radius. We note that the two extrapolation methods exhibit different behaviors while reaching identical converged values.

^{6}Li is a more challenging task for both *ab initio* calculations and extrapolations. This is a weakly bound nucleus where a weakly bound deuteron orbits the ^{4}He core. Thus, the radius is relatively large, and the calculated observables converge slowly as the model space increases. This nucleus is a good challenge for extrapolation methods. The results for neural network extrapolations are shown in Fig. 7. For the ground-state energy, the neural network gives $E_{\mathrm{g.s.}} = -30.743 \pm 0.061\,\mathrm{MeV}$ with the largest data set $\max(N_{\max}) = 22$ and the results start to converge when $\max(N_{\max})$ reaches 16. As a long-range operator the radius converges even slower than the energy, which makes it more difficult for the extrapolation method to obtain a reliable prediction. With the largest data set, the neural network extrapolated result is $r_{\mathrm{p}} = 2.471 \pm 0.028$ fm and the predictions start to converge at $\max(N_{\max}) = 20$. The error bars reflect the variations that are from changes in the initial point in the training process.

So far, we have only studied the uncertainties from the random starting point when training the network. To study the robustness of the trained neural networks, we proceed as follows. Once a network is trained, i.e., once its weights and biases w are determined, we take a random vector (with components drawn at random from a Gaussian distribution with zero mean) Δw in the space of weights and biases and adjust its length such that the loss function fulfills $L(w + \Delta w) = cL(w)$, with $c = 2$ or $c = 10$. These values are motivated

TABLE I. Uncertainty analysis of *NN* extrapolated results for ^{6}Li with weights $w + \Delta w$. The random vector Δw of weights and biases is adjusted to double the loss function, i.e., $L(w + \Delta w) = 2L(w)$. The quantities $\sigma_{E_{\text{g.s.}}}$ and σ_r are the standard deviation of the new predictions for ground-state energy (in MeV) and point-proton radius (in fm), respectively. $\max(\Delta E_{\text{g.s.}})$ (in MeV) and $\max(\Delta r)$ (in fm) show the maximal deviation between the new predictions and the origin results. $|\Delta w|/|w|$ are the ratio between norms of the weight deviation and the origin weights.

	max($N_{\max}$)					
	12	14	16	18	20	22
$\sigma_{E_{\text{g.s.}}}$	0.013	0.010	0.009	0.008	0.009	0.006
$\max(\Delta E_{\text{g.s.}})$	0.068	0.049	0.037	0.031	0.032	0.024
$|\Delta w|/|w|$	0.0009	0.0008	0.0008	0.0008	0.0008	0.0008
σ_r	0.0034	0.0038	0.0042	0.0049	0.0054	0.0061
$\max(\Delta r)$	0.0152	0.0176	0.0200	0.0212	0.0233	0.0269
$|\Delta w|/|w|$	0.0050	0.0043	0.0037	0.0042	0.0043	0.0030

as follows. For a chi-square distribution with uncorrelated degrees of freedom, $c = 2$ would map out the region of one standard deviation. However, our networks are not that simple and network parameters are correlated. For this reason we also consider the case $c = 10$. We note that this approach yields uncertainty estimates but not quantified uncertainties. We then use the new network parameters $w + \Delta w$ to predict the observable of interest. Taking 100 random vectors Δw for each single network, we compute the variance in the observable of interest, and also record the maximum deviation. The results are shown in Tables I and II for $c = 2$ and $c = 10$, respectively. We see that the network is approximately parabolic at its optimal training point (as variances and maximal deviations increase by about a factor $\sqrt{5}$ as we go from $c = 2$ to $c = 10$. For energies and radii, the networks are robust. For $c = 2$ and $c = 10$, the network parameters $|\Delta w|/|w|$ change by about 1/1000 and 1%, respectively. Allowing for a twofold increase of the loss function, the uncertainty from the training of the network does not exceed the uncertainties from the random initial starting points. However, allowing weights and biases to change such that the loss function is increased by a factor of 10, yields larger uncertainties. In this case, the maximum uncertainties from the neural network (when added to the error bars shown in Fig. 7), would lead the error bars from the neural network extrapolation to overlap with those

TABLE II. Same as as Table I but for random vectors Δw of weights and biases that yield a 10-fold increase of the loss function.

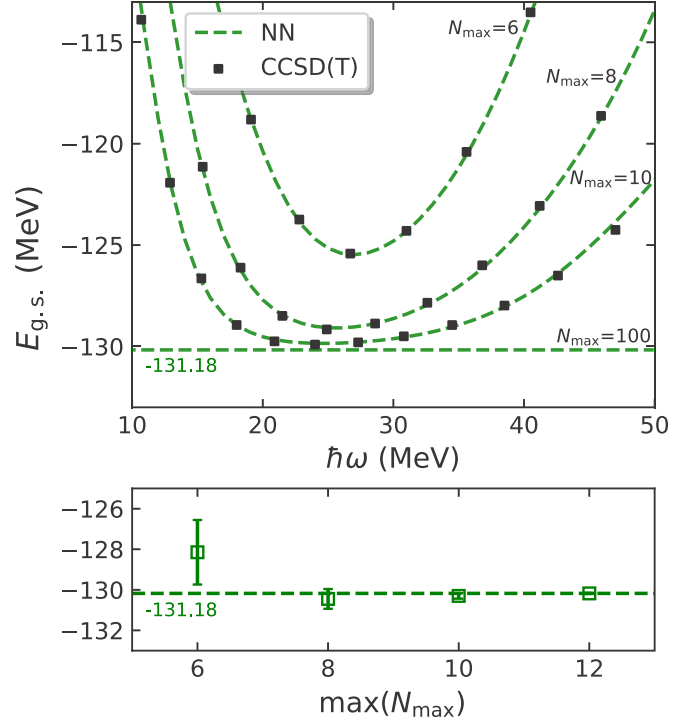|	max($N_{\max}$)					
	12	14	16	18	20	22
$\sigma_{E_{\text{g.s.}}}$	0.037	0.025	0.023	0.020	0.035	0.016
$\max(\Delta E_{\text{g.s.}})$	0.183	0.121	0.098	0.084	0.153	0.066
$|\Delta w|/|w|$	0.0025	0.0023	0.0021	0.0021	0.0023	0.0021
σ_r	0.0093	0.0093	0.0115	0.0139	0.0154	0.0165
$\max(\Delta r)$	0.0415	0.0393	0.0525	0.0635	0.0684	0.0723
$|\Delta w|/|w|$	0.0122	0.0096	0.0105	0.0105	0.0114	0.0090

FIG. 8. Neural network predictions (upper panel) based on the CCSD(T) data set with max($N_{\max}$) = 12 and multiple neural network extrapolated results (lower panel) with data sets form max($N_{\max}$) = 6 to max($N_{\max}$) = 12.

from the IR extrapolation. We note finally that the single-layer neural networks we employ are not resilient with regard to dropout. Removing a single node after training of the network on average changes the predictions for energies and radii by almost 20%.

To illustrate the universality of neural network extrapolation, we apply the multiple neural network approach on the ground-state energy of ^{16}O, computed with the coupled-cluster method [22]. The upper panel of Fig. 8 shows the neural network performance with the largest data sets [max($N_{\max}$) = 12]. As we can see in the lower panel of the figure, the neural network extrapolation results start to converge at max($N_{\max}$) = 8. Note that, by then, the neural network is trained with only three sets of $N_{\max}$ data and is still able to capture the correct pattern. This is because of the quick convergence of the coupled-cluster method itself and the relatively flat curve around the minimum of the energy as a function of $\hbar\omega$, which are both favorable for the neural network extrapolation approach.

IV. SUMMARY

In this paper, we presented a neural network extrapolation method to estimate the ground-state energies and point-proton radii from NCSM and the coupled-cluster calculations. To counter the overfitting problem which is caused by the limited set of *ab initio* results, we enlarged the data set by interpolating between different data points, and used a loss function that accounts for the correlations between the data points.

Because of the random nature of the neural network algorithm, we employed multiple neural network approaches to obtain recommended results and uncertainties of the extrapolations. We applied balanced sample weights as data pre-processing to eliminate the influences of the persistent local minima, and to obtain a more pronounced single solution for the multiple neural network predictions.

We presented neural-network-extrapolated energies and radii of ^{4}He, ^{6}Li for NCSM and compared them with IR extrapolated results from Ref. [22]. The neural network extrapolations gave reliable predictions for both observables with reasonable uncertainties. The extrapolations for the ground-state energy of ^{16}O from coupled-cluster calculations also yielded accurate results. The strong pattern learning ability of the neural network allowed us to apply the same network architecture for NCSM and CC extrapolation without em-

ploying any particular functions. In conclusion, the neural networks studied in this work are useful tools for extrapolating results from *ab initio* calculations performed in finite model spaces.

[1] W. H. Dickhoff and C. Barbieri, Self-consistent Green's function method for nuclei and nuclear matter, Prog. Part. Nucl. Phys. **52**, 377 (2004).

[2] P. Navrátil, S. Quaglioni, I. Stetcu, and B. R. Barrett, Recent developments in no-core shell-model calculations, J. Phys. G: Nucl. Part. Phys. **36**, 083101 (2009).

[3] D. Lee, Lattice simulations for few- and many-body systems, Prog. Part. Nucl. Phys. **63**, 117 (2009).

[4] B. R. Barrett, P. Navrátil, and J. P. Vary, *Ab initio* no core shell model, Prog. Part. Nucl. Phys. **69**, 131 (2013).

[5] R. Roth, S. Binder, K. Vobig, A. Calci, J. Langhammer, and P. Navrátil, Medium-Mass Nuclei with Normal-Ordered Chiral $NN+3N$ Interactions, Phys. Rev. Lett. **109**, 052501 (2012).

[6] T. Dytrych, K. D. Launey, J. P. Draayer, P. Maris, J. P. Vary, E. Saule, U. Catalyurek, M. Sosonkina, D. Langr, and M. A. Caprio, Collective Modes in Light Nuclei from First Principles, Phys. Rev. Lett. **111**, 252501 (2013).

[7] G. Hagen, T. Papenbrock, M. Hjorth-Jensen, and D. J. Dean, Coupled-cluster computations of atomic nuclei, Rep. Prog. Phys. **77**, 096302 (2014).

[8] J. Carlson, S. Gandolfi, F. Pederiva, S. C. Pieper, R. Schiavilla, K. E. Schmidt, and R. B. Wiringa, Quantum Monte Carlo methods for nuclear physics, Rev. Mod. Phys. **87**, 1067 (2015).

[9] H. Hergert, S. K. Bogner, T. D. Morris, A. Schwenk, and K. Tsukiyama, The in-medium similarity renormalization group: A novel *ab initio* method for nuclei, Phys. Rep. **621**, 165 (2016).

[10] M. Horoi, A. Volya, and V. Zelevinsky, Chaotic wave Functions and Exponential Convergence of Low-Lying Energy Eigenvalues, Phys. Rev. Lett. **82**, 2064 (1999).

[11] H. Zhan, A. Nogga, B. R. Barrett, J. P. Vary, and P. Navrátil, Extrapolation method for the no-core shell model, Phys. Rev. C **69**, 034302 (2004).

[12] G. Hagen, D. J. Dean, M. Hjorth-Jensen, T. Papenbrock, and A. Schwenk, Benchmark calculations for ^{3}H, ^{4}He, ^{16}O, and ^{40}Ca with *ab initio* coupled-cluster theory, Phys. Rev. C **76**, 044305 (2007).

[13] C. Forssén, J. P. Vary, E. Caurier, and P. Navrátil, Converging sequences in the *ab initio* no-core shell model, Phys. Rev. C **77**, 024301 (2008).

[14] S. K. Bogner, R. J. Furnstahl, P. Maris, R. J. Perry, A. Schwenk, and J. P. Vary, Convergence in the no-core shell model with

low-momentum two-nucleon interactions, Nucl. Phys. A **801**, 21 (2008).

[15] R. J. Furnstahl, G. Hagen, and T. Papenbrock, Corrections to nuclear energies and radii in finite oscillator spaces, Phys. Rev. C **86**, 031301(R) (2012).

[16] S. A. Coon, M. I. Avetian, M. K. G. Kruse, U. van Kolck, P. Maris, and J. P. Vary, Convergence properties of *ab initio* calculations of light nuclei in a harmonic oscillator basis, Phys. Rev. C **86**, 054002 (2012).

[17] S. N. More, A. Ekström, R. J. Furnstahl, G. Hagen, and T. Papenbrock, Universal properties of infrared oscillator basis extrapolations, Phys. Rev. C **87**, 044326 (2013).

[18] R. J. Furnstahl, S. N. More, and T. Papenbrock, Systematic expansion for infrared oscillator basis extrapolations, Phys. Rev. C **89**, 044301 (2014).

[19] S. König, S. K. Bogner, R. J. Furnstahl, S. N. More, and T. Papenbrock, Ultraviolet extrapolations in finite oscillator bases, Phys. Rev. C **90**, 064007 (2014).

[20] K. A. Wendt, C. Forssén, T. Papenbrock, and D. Sääf, Infrared length scale and extrapolations for the no-core shell model, Phys. Rev. C **91**, 061301(R) (2015).

[21] D. Odell, T. Papenbrock, and L. Platter, Infrared extrapolations of quadrupole moments and transitions, Phys. Rev. C **93**, 044331 (2016).

[22] C. Forssén, B. D. Carlsson, H. T. Johansson, D. Sääf, A. Bansal, G. Hagen, and T. Papenbrock, Large-scale exact diagonalizations reveal low-momentum scales of nuclei, Phys. Rev. C **97**, 034328 (2018).

[23] M. Lüscher, Volume Dependence of the Energy Spectrum in Massive Quantum Field Theories. 1. Stable Particle States, Commun. Math. Phys. **104**, 177 (1986).

[24] S. König and D. Lee, Volume dependence of n-body bound states, Phys. Lett. B **779**, 9 (2018).

[25] G. Alina Negoita, G. R. Luecke, J. P. Vary, P. Maris, A. M. Shirokov, I. J. Shin, Y. Kim, E. G. Ng, and C. Yang, Deep learning: A tool for computational nuclear physics, in *the Proceedings of the Ninth International Conference on Computational Logics, Algebras, Programming, Tools, and Benchmarking COMPUTATION TOOLS 2018, Barcelona* (IARIA, Spain, 2018), arXiv:1803.03215.

[26] G. A. Negoita, J. P. Vary, G. R. Luecke, P. Maris, A. M. Shirokov, I. J. Shin, Y. Kim, E. G. Ng, C. Yang, M. Lockner, and

G. M. Prabhu, Deep learning: Extrapolation tool for *ab initio* nuclear theory, Phys. Rev. C **99**, 054308 (2019).

[27] J. W. Clark, E. Mavrommatis, S. Athanassopoulos, A. Dakos, and K. Gernoth, Statistical modeling of nuclear systematics, in *Fission Dynamics of Atomic Clusters and Nuclei*, edited by J. d. Providência, D. M. Brink, F. Karpechine, and F. B. Malik (World Scientific, Singapore, 2001), pp. 76–85.

[28] S. Athanassopoulos, E. Mavrommatis, K. A. Gernoth, and J. W. Clark, Nuclear mass systematics using neural networks, Nucl. Phys. A **743**, 222 (2004).

[29] N. J. Costiris, E. Mavrommatis, K. A. Gernoth, and J. W. Clark, Decoding β-decay systematics: A global statistical model for β^- half-lives, Phys. Rev. C **80**, 044332 (2009).

[30] S. Akkoyun, T. Bayram, S. O. Kara, and A. Sinan, An artificial neural network application on nuclear charge radii, J. Phys. G: Nucl. Part. Phys. **40**, 055106 (2013).

[31] R. Utama, J. Piekarewicz, and H. B. Prosper, Nuclear mass predictions for the crustal composition of neutron stars: A Bayesian neural network approach, Phys. Rev. C **93**, 014311 (2016).

[32] R. Utama, W.-C. Chen, and J. Piekarewicz, Nuclear charge radii: Density functional theory meets Bayesian neural networks, J. Phys. G: Nucl. Part. Phys. **43**, 114002 (2016).

[33] R. Utama and J. Piekarewicz, Refining mass formulas for astrophysical applications: A Bayesian neural network approach, Phys. Rev. C **96**, 044308 (2017).

[34] R. Utama and J. Piekarewicz, Validating neural-network refinements of nuclear mass models, Phys. Rev. C **97**, 014306 (2018).

[35] L. Neufcourt, Y. Cao, W. Nazarewicz, and F. Viens, Bayesian approach to model-based extrapolation of nuclear observables, Phys. Rev. C **98**, 034318 (2018).

[36] G. Carleo and M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, 602 (2017).

[37] G. Martius and C. H. Lampert, Extrapolation and learning equations, arXiv:1610.2995.

[38] P. J. Haley and D. Soloway, Extrapolation limitations of multi-layer feedforward neural networks, in *Proceedings 1992 IJCNN International Joint Conference on Neural Networks* (IEEE, Piscataway, 1992), Vol. 4, pp. 25–30.

[39] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, Nature (London) **323**, 533 (1986).

[40] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. **2**, 303 (1989).

[41] K.-I. Funahashi, On the approximate realization of continuous mappings by neural networks, Neural networks **2**, 183 (1989).

[42] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural networks **4**, 251 (1991).

[43] Y. Bengio *et al.*, Learning deep architectures for AI, Foundations and trends® in Machine Learning **2**, 1 (2009).

[44] F. Seide, G. Li, and D. Yu, Conversational speech transcription using context-dependent deep neural networks, in *Twelfth Annual Conference of the International Speech Communication Association* (ISCA, Baixas, 2011).

[45] H. Mhaskar, Q. Liao, and T. Poggio, Learning functions: When is deep better than shallow, arXiv:1603.0988.

[46] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (Morgan Kaufmann, San Francisco, 2010), pp. 249–256.

[47] X. Glorot, A. Bordes, and Y. Bengio, Deep sparse rectifier neural networks, in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics* (Morgan Kaufmann, San Francisco, 2011), pp. 315–323.

[48] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.

[49] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, Greedy layer-wise training of deep networks, in *Advances in Neural Information Processing Systems* (MIT Press, Cambridge, 2007), pp. 153–160.

[50] D. Stathakis, How many hidden layers and nodes? Int. J. Remote Sens. **30**, 2133 (2009).

[51] J. Heaton, *Introduction to Neural Networks with Java* (Heaton Research, St. Louis, 2008).

[52] A. Ekström, G. Baardsen, C. Forssén, G. Hagen, M. Hjorth-Jensen, G. R. Jansen, R. Machleidt, W. Nazarewicz, T. Papenbrock, J. Sarich, and S. M. Wild, Optimized Chiral Nucleon-Nucleon Interaction at Next-to-Next-to-Leading Order, Phys. Rev. Lett. **110**, 192502 (2013).

[53] H. Zou and T. Hastie, Regularization and variable selection via the elastic net, J. R. Stat. Soc: Ser. B (Statistical Methodology) **67**, 301 (2005).

[54] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. **15**, 1929 (2014).

[55] L. Prechelt, Automatic early stopping using cross validation: Quantifying the criteria, Neural Networks **11**, 761 (1998).

[56] M. A. Tanner and W. H. Wong, The calculation of posterior distributions by data augmentation, J. Am. Stat. Assoc. **82**, 528 (1987).

[57] L. Breiman, Random forests, Machine learning **45**, 5 (2001).

[58] G. Hagen, T. Papenbrock, D. J. Dean, and M. Hjorth-Jensen, *Ab initio* coupled-cluster approach to nuclear structure with modern nucleon-nucleon interactions, Phys. Rev. C **82**, 034330 (2010).

[59] M. R. Schindler and D. R. Phillips, Bayesian methods for parameter estimation in effective field theories, Ann. Phys. **324**, 682 (2009).

[60] R. J. Furnstahl, D. R. Phillips, and S. Wesolowski, A recipe for EFT uncertainty quantification in nuclear physics, J. Phys. G: Nucl. Part. Phys. **42**, 034028 (2015).

[61] B. D. Carlsson, A. Ekström, C. Forssén, D. F. Strömberg, G. R. Jansen, O. Lilja, M. Lindby, B. A. Mattsson, and K. A. Wendt, Uncertainty Analysis and Order-By-Order Optimization of Chiral Nuclear Interactions, Phys. Rev. X **6**, 011019 (2016).