Proceedings of the INMM & ESARDA Joint Virtual Annual Meeting
August 23-26 & August 30-September 1, 2021

COMPARISON OF DIFFERENT SUPERVISED MACHINE LEARNING ALGORITHMS TO PREDICT PWR SPENT FUEL PARAMETERS

Vaibhav Mishra
Department of Physics and Astronomy
Uppsala University, Sweden

Erik Branger
Department of Physics and Astronomy
Uppsala University, Sweden

Zsolt Elter
Department of Physics and Astronomy
Uppsala University, Sweden

Sophie Grape
Department of Physics and Astronomy
Uppsala University, Sweden

Peter Jansson
Department of Physics and Astronomy
Uppsala University, Sweden

ABSTRACT

Nuclear safeguards verification of spent nuclear fuel (SNF) is imperative to ensure peaceful use of nuclear material. To verify the correctness and completeness of operator declarations, non-destructive assay (NDA) measurements of gamma and neutron radiation from the SNF play a central role. Verification of fuel based on such measurements is done routinely by safeguards inspectors and is also expected to be conducted prior to preparation of SNF for final disposal. Traditionally, SNF verification has been carried out by analyzing data from a single NDA instrument at a time. In this study, we compare the performances of different machine learning algorithms in their ability to make predictions of the SNF parameters such as fuel burnup (BU), initial enrichment (IE), and cooling time (CT). Predictions were made based on simulated signatures such as gamma-ray intensities from individual radionuclides, the total Cherenkov light intensity, and the parameterized differential die-away time (τ). In this work, multiple machine learning algorithms have been trained and tested on a set of simulated data containing 596,181 fuel samples providing a broad range of these three parameters to encompass the majority of the spent nuclear fuel worldwide. Additionally, the resilience of the machine learning algorithms on noisy data was evaluated. The results show that the non-linear methods can provide highly reliable predictions of SNF parameters. In nearly all situations assessed in this work, we have found that shallow learning methods have a clear advantage over deep learning models investigated in the present study. We also found that shallow learning methods such as *k*-Nearest Neighbors outperform other tree-based methods as well as neural networks at noise levels above 5%.

Keywords: Nuclear safeguards verification, supervised machine learning, multivariate analysis, deep learning, neural networks, surrogate models.

INTRODUCTION

Ensuring peaceful use of nuclear material is one of the most important aims of nuclear safeguards frameworks and it often relies on non-destructive assay (NDA) measurements of spent fuel [1] to achieve this goal. Such measurements include measuring gamma, neutron, and Cherenkov light radiation emitted by the spent fuel. Information gathered from such measurement campaigns is key to verifying the correctness and completeness of operator declarations of spent nuclear fuel (SNF). It is also well-established that measurements of this nature can be used to deduce fuel parameters such as burnup (BU), initial enrichment (IE), and cooling time (CT) owing to underlying correlations that exist between these fuel parameters and concentrations of certain radionuclides in the SNF. In the past, non-machine learning methods have been used to deduce fuel BU, IE, and CT using measurements of ^{134}Cs, ^{154}Eu, ^{137}Cs intensities [2]. While more recently, it has been demonstrated in [3–5] that machine learning methods can be used to deduce fuel BU, IE, and CT. Therefore, use of machine learning allows for the systematic analysis of different signals to uncover correlations. Past work in the domain conducted in [4, 5] has focused on the implementation and performance assessment of linear machine learning algorithms such as ordinary least squares (OLS), partial least squares (PLS) and principal component regression (PCR) while [3] has shown that the implementation of a non-linear, tree-based model such as random forest regression shows promising returns in the form of lower prediction errors while predicting fuel BU, IE, and CT. Additionally, the ability of deep learning techniques (primarily employing artificial neural networks) to predict fuel properties has been investigated in [6, 7] and these studies have demonstrated that predictions of this nature using deep learning surrogate models would be feasible.

In the present study, we aim to build several such non-linear machine learning models and train them on a data set [8] of simulated irradiated PWR fuel with different values of BU, IE, and CT along with the associated isotopic activities, the Cherenkov light intensity, and the parameterized differential die-away time or τ values. The performances of the resulting models were compared to see which model(s) perform best in predicting fuel BU, IE, and CT. The scope of the current work encompasses the use of four machine learning models; *k*-Nearest Neighbors Regressor, Random

Forest Regressor, and AdaBoost Regressor as well as fully-connected type artificial neural networks (as a deep learning model). Additionally, we have also examined the performance of these algorithms on noise-riddled test data to see the robustness of these methods and their associated prediction errors. The current work is limited to use of simulated data only. However, the results from this study are expected to be useful in further planned work with experimental data.

The following sections will provide information on the data set used in the analysis, the assumptions used and set up of the machine learning problem, the machine learning algorithms employed in the analysis, the methodology used, and the results obtained from the evaluation along with final conclusions and recommendations.

DATA SET OF SIMULATED OBSERVABLES

The data set used in the present study consists of a subset of 596,181 simulated burnup scenarios of irradiated uranium oxide fuel samples from [8]. Each of the samples comprises calculated isotopic concentrations of ten nuclides deemed important from a nuclear safeguards standpoint, along with values of the parameterized differential die-away time (τ), and the total Cherenkov light intensities with contributions from beta-minus decays in the fuel rods at different combinations of fuel BU, IE, and CT values. These isotopes fulfil the following criteria: firstly, their relative abundance compared to other nuclides in nuclear material, secondly, their ease of detectability due to gamma or neutron emission using available detection instrumentation and lastly, a half-life that's long enough so the activity remains detectable. The total Cherenkov light intensity used in the data set has been computed with a dedicated software [9] and more information about the principle behind it can be found in [10]. The data set also includes the differential die-away time for all BU, IE, and CT combinations and has been computed with the help of a parameterization function from [11]. The range of different fuel parameters along with other key features of the data set are summarized in Table 1.

Table 1. Summary of key features of the data set.

Parameter	Total samples	BU (in MWd/kgU)	IE (in wt. % U-235)	CT (in years)	Isotopes + derived quantities
Values	596,181	15–70	1.5–5.5	0–70	^{141}Ce, ^{95}Nb, ^{95}Zr, ^{144}Ce, ^{106}Ru, ^{134}Cs, ^{154}Eu, ^{137}Cs, tau (τ), Cherenkov

Within the data set, the values of IE range from 1.5 to 5.5 wt. % U-235 in steps of 0.5. The BU values range from 5 MWd/kgU to 70 MWd/kgU in steps of 0.5 MWd/U. And lastly, the CT ranges from 0 years to 70 years in steps of 0.25 years between 0 and 10 years, in steps of 0.5 years between 10 and 40 years and in steps of 1 year between 40 and 70 years. This implies that while fuel BU and IE span the input parameter space quite uniformly, CT is unevenly distributed with more samples in the data set below CT of 20 years and increasingly fewer samples at longer cooling times. This was done intentionally since in the first two decades of cooling, a lot of short-lived radionuclides decay and the fuel's composition undergoes significant changes.

The isotope number densities in the data set were used to compute the respective nuclide activities. The selected isotopes and their respective half-lives are listed in Table 2. It is worth noting that only eight out of the 10 total isotopes from the data set were deemed important for the present study since they are major gamma/neutron radiation emitters in SNF.

Table 2. Isotopes used in present study and their respective half-lives (data from [12]).

Nuclide	Half-life	Nuclide	Half-life	Nuclide	Half-life	Nuclide	Half-life
^{95}Nb	34.991 *d*	^{95}Zr	64.032 *d*	^{106}Ru	371.5 *d*	^{134}Cs	2.064 *y*
^{137}Cs	30.05 *y*	^{141}Ce	32.5 *d*	^{144}Ce	284.89 *d*	^{154}Eu	8.601 *y*

Preparation of input features

As described previously, the isotopic number densities for eight isotopes were first converted to radionuclide activities using half-lives from Table 2. Once the activities were calculated, a global activity threshold of 0.1 % of ^{137}Cs activity was imposed on the data set similar to what was done in [3]. This implies that activities for the seven isotopes from Table 2 were set to zero if they were below the activity threshold of 0.1 % of the minimum ^{137}Cs activity in the data set. This was done to simulate the role of a "detectability threshold" in a realistic fuel measurement scenario; if the activity of an isotope in the fuel is low enough, it will become undetectable by the experimental setup in use. The value of this detectability threshold was entirely arbitrary in nature and its value can easily be changed. The threshold is introduced to force models to learn that some activities may not be measurable for some fuel parameter combinations, which does provide some data on the fuel parameters. A heat plot with Pearson's correlation coefficients [13] is given in Figure 1 showing the existing correlations between fuel parameters and radionuclide activities, Cherenkov intensity and τ values. The Pearson's correlation coefficient were computed using Equation 1:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 (y_i - \bar{y})^2}} \tag{1}$$

The correlation coefficient varies between -1 and +1 denoting negative and positive correlation between variables (x and y) respectively. In absence of sufficient knowledge about the ideal choice of prediction input features, one may use

information from this correlation heatmap as a starting point to set up the machine learning problem. This heatmap can also help identify the most important parameters and can help weed out the redundant ones with minimum loss of accuracy.

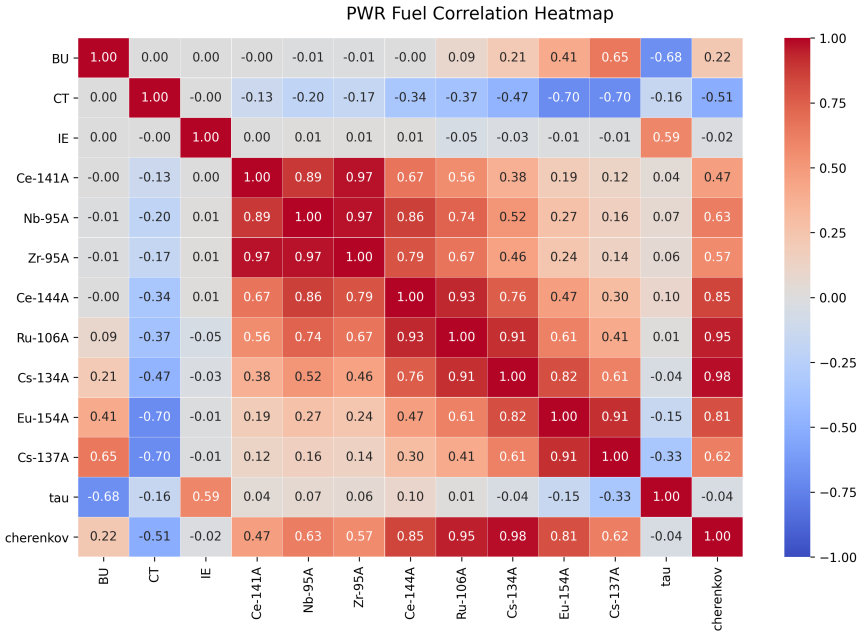


Figure 1. Correlation coefficient heat plot showing statistical correlations between fuel parameters and radionuclide activities, Cherenkov light intensities, and τ.

SELECTION OF MACHINE LEARNING MODELS

In the present study, both shallow machine learning models and deep learning models have been evaluated in their ability to make meaningful and accurate predictions on the data set of simulated variables from [8]. Mentions of all Python classes are typeset with a monospace font throughout this publication. Within the regime of shallow learning, the implementations for k-Nearest Neighbors Regressor, Random Forest Regressor, and AdaBoost Regressor were used from the `scikit-learn` library [14]. For deep learning algorithms, a dense layer neural network or a multi-layer perceptron (MLP) was implemented using the `tensorflow-keras` [15] library. It should be noted that all algorithms used in the present study were used as supervised machine learning algorithms since the output variables (fuel BU, IE, and CT) from the data set were used in the training. In other words, for all the algorithms explored in the present work, fuel BU, IE, and CT served as our output variables or the Y variable(s) while different combinations of individual radionuclide activities and other observables such as Cherenkov light intensity and the τ served as predictors or the X variables and the same has been summarized in Equation 2 (where E is a generalized regression error term).

$$Y = \mathbf{A} * X + E \tag{2}$$

Shallow learning models

The following sections will explain the choice of shallow machine learning algorithms and motivate the choice and approach implemented to tune the algorithms for the problem at hand. The key strengths and weaknesses of each algorithm are discussed in brief.

k-nearest neighbors regressor

K-nearest neighbors regressor (or simply k-NN) is a non-parametric, non-linear machine learning algorithm that is well-suited for both classification and regression type problems. The k-NN algorithm is a good choice where information about the problem or its outcome is limited or completely lacking i.e. the relationship between $X's$ and $Y's$ in the problem is unknown. The algorithm is unique from other machine learning algorithms in the sense that it does not compute a globally applicable function that maps the input features to the output. Rather, it computes local estimates of the output based on the number of neighbors supplied by the user. It does so by computing the distances (Euclidean or other selected by user) between input features plotted in the input hyperspace corresponding to the "k" neighbors specified by the user. It then uses these distances and computes an average of the output variable at these neighbors to provide the estimate of the output. This algorithm is thus sensitive to local fluctuations in the input data. The optimal k value is usually obtained using a cross-validation scheme run with varying values of k while monitoring the root mean squared error (RMSE) metric during training and validation. Where the RMSE is mathematically defined in Equation 3 as:

$$RMSE = \sqrt{\frac{1}{n \times m} \sum_{i=1}^{n} \sum_{j=1}^{m} (Y_{i,j,true} - Y_{i,j,predicted})^2} \tag{3}$$

Here, m and n imply number of predicted output variables and number of data points in the training or validation data sets respectively, Y_{true} is the known value of the output variable in the data set, and $Y_{predicted}$ is the prediction from the model. The RMSE is used as the performance metric for all the algorithms in the present work.

Random forest regressor

Random forest regressors fall under the umbrella of classification and regression trees algorithms which are also non-linear algorithms. The principle is the same as that of decision tree regressors which are algorithms that perform classification or regression based on a greedy approach (that looks only one step ahead) to minimize an entropy or loss function metric. The key differentiating aspect of random forest from decision trees is a feature called "bagging" which is short for bootstrap aggregation. Bootstrap feature enables the random forest algorithm to train on subsets of the data set sampled with replacement and with a limited selection of input features. Trees grown in this manner reduce the risk of overfitting on the data, thereby making the algorithm more robust while keeping the model variance low. Finally, in order to make a prediction, the test data point is run down along the trees to produce an output which is the average of the trees. Random forest algorithm is an ideal choice for many since they are a relatively straight-forward to implement method and perform well with minimal user tweaking of model hyperparameters.

AdaBoost regressor

AdaBoost (short for adaptive boosting) algorithm builds on the random forests' concept and differs from traditional tree based methods in an inherent feature called "boosting". Boosting enables AdaBoost algorithm to sequentially build trees (unlike random forests where trees are built in parallel) while assigning more weight to predictions with large errors while reducing the weights of points that were predicted correctly. Therefore, the algorithm spends more time and effort on "hard to predict" data points, and the successive trees built are centered around correctly predicting such data points. AdaBoost successively builds shallow trees (called "stubs") which, by themselves, are weak learners but the overall effect of combining all weak learners is reduced variance as well as a lower model bias. Although AdaBoost may have significant advantages over k-NN and random forests, they are often sensitive to outliers in the data set as the algorithm can spend way too much time and resources on such data points. So, in theory, AdaBoost is not an ideal choice while working with noisy data.

Deep learning models – Artificial neural networks

Feed-forward artificial neural networks (or ANNs) are a class of machine learning algorithms that employ layers of interconnected nodes of neurons which can map a set of inputs to one or several outputs. Within a neural network, every single node can be interpreted as a mathematical transformation to the input data as it "flows" through the network and the process repeats until the data reaches the output nodes of the network. These transformations rely on the "weights" and "biases" associated with all neurons in every single layer. The process in its mathematical form is shown in Equation 4.

$$y = f(\mathbf{W}^\top x + b) \tag{4}$$

Here, $\mathbf{W}^\top$ is the weight vector, x is a vector with input variables, b are the biases, f is an activation function (to introduce non-linearity) and y is the predicted output. The process of learning is handled by auto-differentiation and backpropagation (shortened to "backprop") wherein the neural network computes the difference between the expected and predicted output (called a loss function) and adjusts the weights and biases of the different neurons throughout the neural network architecture based on the computed loss. At the end of the day, the task takes the form of a minimization/optimization problem where the loss is expected to decrease while training the model over several epochs (an epoch is one complete pass over the training data set) of data. An example of a neural network which would simultaneously predict BU, IE, and CT of the fuel is shown in Figure 2.

Model selection and hyperparameter optimization

This section will provide more information on how the machine learning problem was formulated in the current work and elaborate on the choice of model architectures and hyperparameters used for all the models. Hyperparameters are defined as user-supplied parameters that are set prior to execution of the algorithm. Typical shallow machine learning models use only a few (of the order of ten) hyperparameters to tune the model to get the desired performance, while for ANNs, the hyperparameter grid can often suffer from the curse of dimensionality and may become a machine learning problem to solve on its own. Firstly, as far as the k-NN algorithm is concerned, the only two important hyperparameters are the number of nearest neighbors (k) and the distance metric to be used in the calculation of distances between the input features. Both were obtained by means of cross-validation runs with changing values of k and the distance metric. For random forests, the values of hyperparameter ranges were taken from [3] and confirmed to be applicable in the

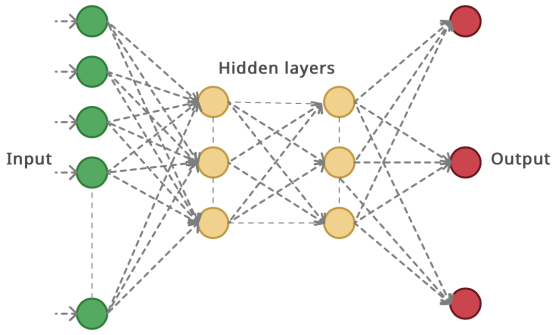


Figure 2. ANN architecture showing input, hidden and output layers and flow of data across the network.

present study as well. Lastly, with regard to the AdaBoost model, the hyperparameters were determined in a manner identical to that of *k*-NN model. The most important hyperparameters were number of estimators, tree depth and to a smaller extent, the learning rate.

The optimization of hyperparameters that apply to neural networks was a more arduous task. Neural network models should be avoided if the problem at hand is relatively easy to solve, and should only be used when other methods do not give satisfactory results due to the sheer computational time that is needed to tune their architectures. This is owing to the fact that neural networks require an extensive set of hyperparameters from the user (ranging from the number and type of layers, activation function, losses to learning rates and optimizers) to perform as expected and give acceptable results. In the absence of optimum hyperparameters, model performance tends to suffer thereby resulting in either under or overfitting. Taking into account the full gravity of this problem, the hyperparameter search for neural networks in the present work was done through the Bayesian Optimization approach available in `keras-tuner` [16] (available as the `BayesianOptimization` class). Bayesian optimization has a clear edge over `GridSearchCV` and `RandomSearchCV` in cases where the hyperparameter grid is highly protracted, while time and computational power is limited. Whereas `GridSearchCV` is extremely brute force and can take a lot of time to cover the whole hyperparameter grid, `RandomSearchCV` has a slight advantage as it samples from the whole hyperparameter grid randomly but still leaves a lot to chance and randomness. After every search iteration, Bayesian Optimization updates the prior knowledge of the hyperparameter space and the corresponding model performance metric and does not waste time and computational resources in regions of hyperparameter space where the model performance is low. A more detailed explanation of the method is provided in [17]. One unfortunate, albeit unavoidable aspect of this method is that the method itself requires a few hyperparameters (i.e. hyper-hyperparameters) so this approach is fruitful only when it is more computationally sensible to search the hyperparameters than to build the surrogate model and solve the machine learning problem itself. For sake of verification, results from `BayesianOptimization` and `RandomSearchCV` were compared in the present study and it was found that they were identical i.e. both approaches arrived at the same set of model hyperparameters, however `RandomSearchCV` took nearly twice as long to do so.

METHODOLOGY

This section describes the steps followed in the analysis including: preparation of input data, selection of appropriate input features for the machine learning problem, searching for the optimum model parameters and hyperparameters and finally execution of the models to make predictions.

Basis for input feature selection

Prior to implementation of the models for prediction of the SNF parameters, the appropriate model input features need to be ascertained. A summary of selected input features corresponding to prediction of each fuel parameter (i.e. BU, IE, or CT) has been shown in Table 3. The primary selection of radionuclides was done based on their half-lives. Thereafter, all radionuclide activities used as input features were subjected to a minimum detectable activity threshold as described previously. As shown in Table 3, the CT space is divided into three sub-regimes namely, CT$<$10 years, CT$<$20 years, and lastly, 20$\leq$CT$\leq$70 years. This was done keeping in mind that CT$<$10, many isotopes have activities that are within the range of detectability of modern gamma spectrometric equipment. While for CT$<$20 most of the short-lived isotopes decay away and only the somewhat long-lived isotopes remain detectable i.e. ^{134}Cs, ^{154}Eu, ^{137}Cs. For 20$\leq$CT$\leq$70 years, additional signatures are needed to reliably predict the fuel parameters and to keep the prediction RMSE as low as possible. Therefore, in addition to the individual isotope activities, the total Cherenkov light intensity, the parameterized differential die-away time τ, and the total gamma activity (denoted as "TOT_A") i.e. sum of gamma activities (above the activity threshold), were used to provide additional data to the machine learning models as in [3]. To emphasize the need for additional prediction features above a CT of 20 years, [3] established a gross improvement in prediction RMSE of up to 50% for IE and CT predictions and of over 100% for BU prediction with random forest regression. Therefore adding the total Cherenkov light intensity, the differential die-away time τ, and the total gamma activity of the eight important nuclides from Table 2 would be a prudent approach in our study for the longest cooled fuels.

Table 3. Summary of selected input features for prediction of SNF parameters.

CT Regime	Case Description		
	Input Features	Algorithms	Targets
CT<10 years	^{141}Ce, ^{95}Nb, ^{95}Zr, ^{144}Ce, ^{106}Ru, ^{134}Cs, ^{154}Eu, ^{137}Cs	*k*-NN, RFR, AdaBoost, Neural Networks	BU, IE, CT
CT<20 years	^{134}Cs, ^{154}Eu, ^{137}Cs		
20≤CT≤70 years	^{134}Cs, ^{154}Eu, ^{137}Cs, Tot_A, tau ^{134}Cs, ^{154}Eu, ^{137}Cs, Cherenkov, tau		

Data preparation and hyperparameter optimization

Firstly, the optimum hyperparameters were determined for all, shallow and deep machine learning models. Once the optimal hyperparameters for each of the shallow learning models were determined, a CT regime was chosen from Table 3 and the data set was split three ways (60%-20%-20%) where the training set had 60% while the validation and test data sets each contained 20% of the samples. Each of the models corresponding to a CT regime from Table 3 was trained and tested on the training and test data sets using the applicable hyperparameters from [18]. All models were run a total of ten times to reduce statistical fluctuations in the results. For neural networks, the ideal hyperparameters were determined using the `BayesianOptimization` module. It should be pointed out that due to limited computational capacity and time constraints, model architecture search was limited to a maximum possible network depth of 20 layers with a maximum ceiling of 512 neurons in each layer. An ideal hardware setup for such a multidimensional optimization problem would require dedicated GPUs (Graphics Processing Unit) or preferably TPUs (Tensor Processing Unit). Finally, after the optimal model was set up, predictions for BU, IE, and CT were made for each CT regime. It is also worth pointing out that an adaptive learning rate scheduler (which reduces learning rate by a pre-defined factor periodically) was used while making fuel parameter predictions with neural networks. Lastly, every neural network model was run ten times with random splits of the data set and with 600 epochs in each run, totalling 6000 passes over the entire data set to minimize statistical fluctuations.

The performance of machine learning models used in the present study was also evaluated on noisy data wherein all shallow and deep learning algorithms were trained on noise-free training data but, their performance was assessed on data with varying levels of Gaussian noise. For this, all models were trained and tested on samples from CT<20 years regime with noise in the test data varying between 0–20% and the results are discussed in later sections.

RESULTS

This section shows the results for hyperparameter optimization for the selected machine learning algorithms. It also presents results for fuel parameter predictions with and without noisy test data.

Results for hyperparameter optimization

The hyperparameters for shallow learning algorithms were established using the approach described previously and have been documented in [18]. Much like the shallow learning algorithms, the neural network parameters were optimized using approach described previously and have been outlined in [18]. It should also be taken into account that for nearly every case of neural network from [18], the network architecture search algorithm returned a rather shallow (seven layer deep) network and in all cases, the total number of neurons in the hidden layers were roughly of the same order. The only one case where a deeper network (with 19 hidden layers) was utilized was the case with CT<20 years where it is suspected that the network needed a deeper structure to learn the underlying associations to produce an acceptable data representation since only three input features were used to train the network. Shallower network architectures were given preference over deeper networks if the improvement in prediction RMSE was only marginal.

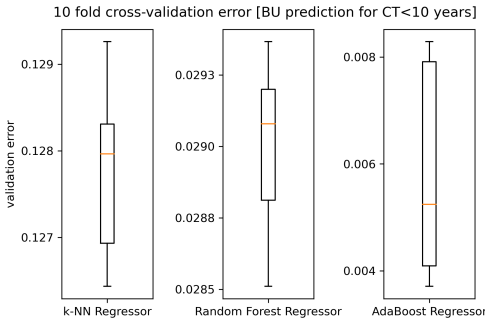
Results for BU-IE-CT estimation

Predictions for BU, IE, and CT were made using the set of hyperparameters from [18]. The prediction RMSE was the sole performance metric for evaluating the model performances and is reported for each algorithm after a ten-fold cross-validation run to reduce the impact of statistical fluctuations for predictions of BU, IE, and CT. For all cases, RMSE values are reported in MWd/kgU for BU, in wt. % U-235 for IE, and in days for CT and are represented by box-plots in the subsequent subsections. The spread of RMSE values is represented by the box in the plot with the outliers depicted as whiskers if they are within $1.5 * interquartile\ range\ (IQR)$, else they are represented by small circles (denoting outliers) on the plot.

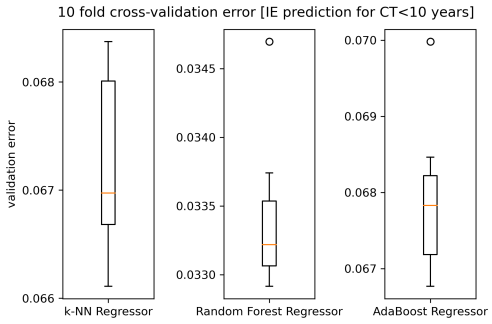
Estimation with shallow learning models

Box-plots showing the results for BU, IE, and CT prediction for fuels with CT<10 years are presented in Figure 3. It is evident that for BU prediction, the RMSE values were by far the lowest for AdaBoost regressor and the associated median RMSE value was 0.005 MWd/kgU. With regard to IE prediction, RMSE values for AdaBoost and *k*-NN

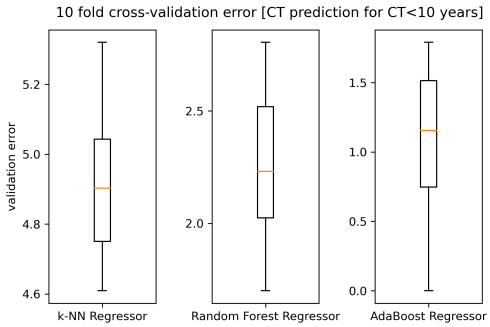
algorithms were comparable with the median value close to 0.067 wt. % U-235, while random forest produced the lowest RMSE with a median value of 0.033 wt. % U-235. For CT prediction, the median RMSE in all three algorithms was below five days and AdaBoost had the lowest median RMSE of approximately one day.



(a) BU prediction RMSE (MWd/kgU).



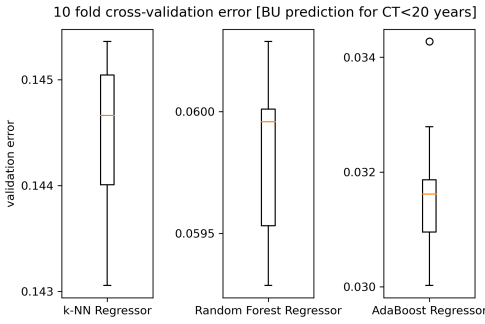
(b) IE prediction RMSE (wt. % U-235).



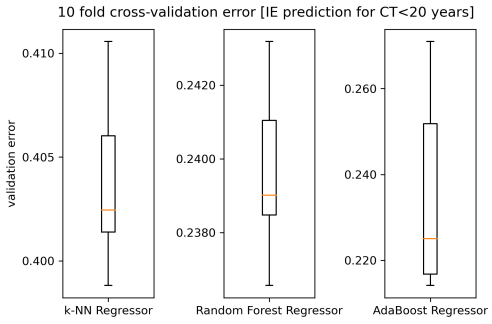
(c) CT prediction RMSE (days).

Figure 3. BU (a) IE (b) CT (c) prediction errors for the selected shallow learning algorithms for CT<10 years.

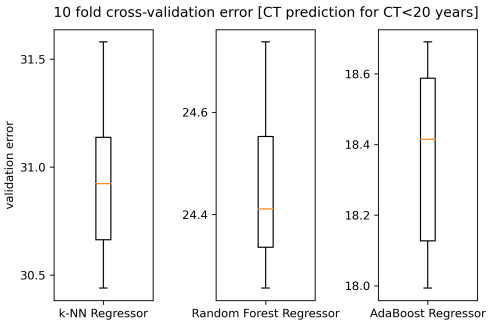
Results for BU, IE, and CT predictions (using the applicable input features from Table 3) for fuels with CT<20 years are presented in Figure 4. Here, the models use only three radionuclides to make predictions. It is seen that for BU prediction, the RMSE values are the lowest for the AdaBoost regressor and the median RMSE value was close to 0.032 MWd/kgU. With regards to the IE prediction, RMSE values for random forest and AdaBoost algorithms were comparable with the median RMSE value close to 0.23 wt. % U-235. For predictions of CT, the median RMSE in all three algorithms was below 50 days and AdaBoost had the lowest median RMSE of 18.4 days.



(a) BU prediction RMSE (MWd/kgU) .



(b) IE prediction RMSE (wt. % U-235) .

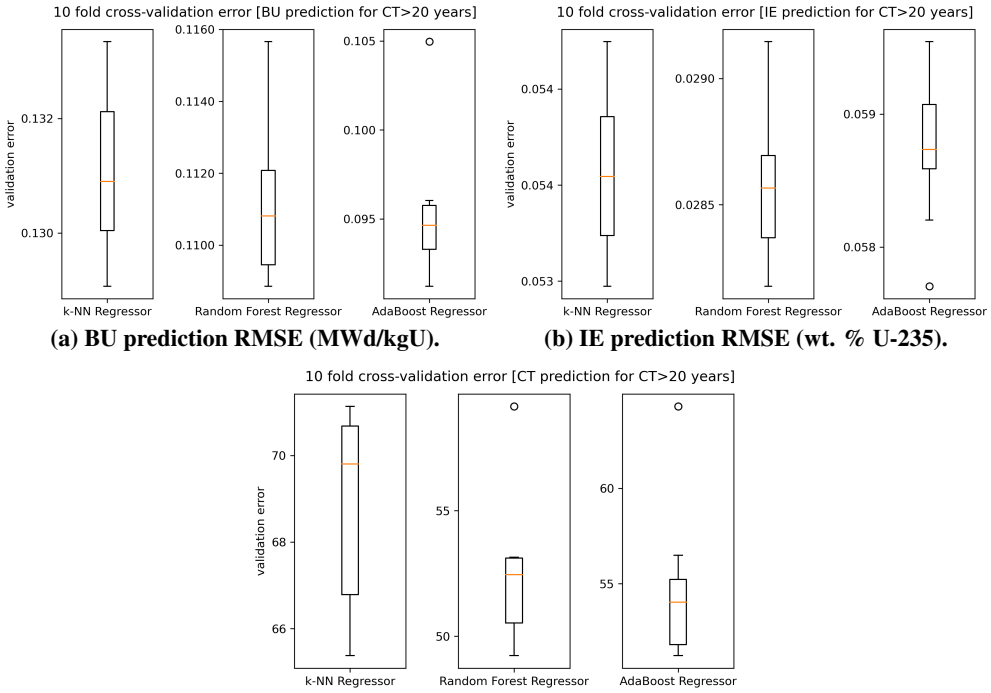


(c) CT prediction RMSE (days).

Figure 4. BU (a) IE (b) CT (c) prediction errors for the shallow learning algorithms for CT<20 years.

Compared to predictions for CT<10 years, the performance of all three models is slightly worse owing to longer cooled fuels present in the data set and fewer prediction input features used in this CT regime.

Lastly, with regards to the results for fuel parameter prediction for the longest cooled fuels, i.e. CT≥20 years are shown in Figure 5.



(a) BU prediction RMSE (MWd/kgU). **(b) IE prediction RMSE (wt. % U-235).**

(c) CT prediction RMSE (days).

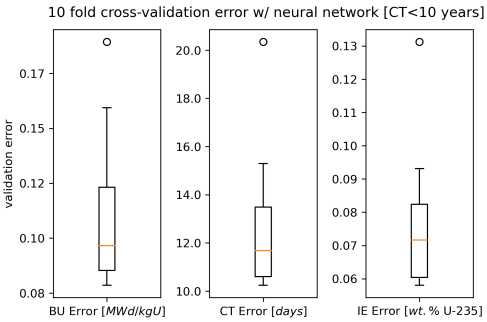
Figure 5. BU (a) IE (b) CT (c) prediction errors for the shallow learning algorithms for CT≥20 years.

It was found that all three algorithms performed equally well with comparable median RMSE values for BU in the neighborhood of 0.1 MWd/kgU. For IE, RFR gave the lowest RMSE of approximately 0.03 wt. % U-235. The results for CT prediction showed that AdaBoost and random forest both had median RMSE close to 55 days. Compared to model predictions for CT<10 years and CT<20 years regimes, the performance of all three models is markedly worse (especially for CT prediction) owing to presence of much longer cooled fuels in the data set. However this worsening is offset by improvement in prediction performance of these models due to inclusion of additional prediction input parameters such as Cherenkov intensity and the parameterized differential die-away time (τ).

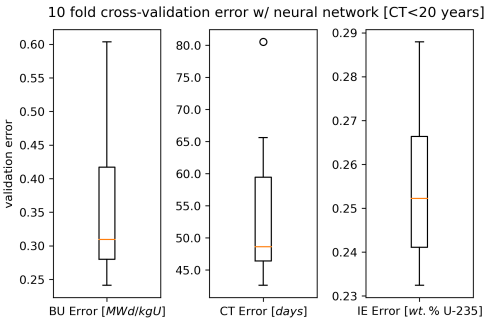
Estimation with neural network models

As far as making predictions using neural networks is concerned, the CT space was again divided into three sub-regimes prior to setting up the models and making predictions. However, one key difference between the shallow learning models and the neural networks is that a single neural network was set up and configured to make BU, IE, and CT predictions simultaneously, while the shallow learning models predicted only a single fuel parameter at a time and each model required a separate set of hyperparameters to make the best possible prediction. In order to set up the model for neural networks, the optimal hyperparameters from [18] were used. The steps for setting up the model are outlined in previous sections.

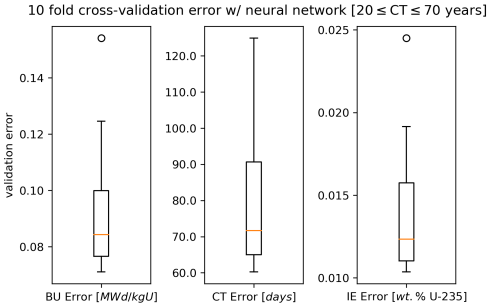
Firstly, for fuels with CT<10 years and CT<20 years, the results are presented in Figure 6. The results show that prediction RMSE for BU, IE, and CT were roughly of the same order as those for shallow learning algorithms from the previous section. For fuels with CT<10 years, median prediction RMSE for BU using neural networks was 0.1 MWd/kgU, about 12 days for CT, and 0.07 wt. % U-235 for IE. For fuels with CT<20 years, median RMSE for BU was roughly 0.3 MWd/kgU, about 50 days for CT, and 0.25 wt. % U-235 for IE. Whereas the RMSE values for fuels with CT<10 years were slightly on the higher end when compared to those from tree-based methods such as random forest and AdaBoost regression, they were comparable to those from the *k*-NN algorithm. For fuels with CT<20 years, the RMSE values for BU and CT were higher but of the same order. For IE, RMSE values were comparable to those obtained with shallow learning methods. For 20≤CT≤70 years, neural networks show lower prediction RMSEs for BU and IE with median RMSE for both cases lower than those from shallow learning algorithms. The median RMSE for CT prediction with neural networks was comparable to *k*-NN estimators.



(a) ANN Prediction RMSE for CT<10 years.



(b) ANN Prediction RMSE for CT<20 years.



(c) ANN Prediction RMSE for 20≤CT≤70 years.

Figure 6. BU (a) IE (b) CT (c) prediction errors using using the selected neural networks.

Model resilience towards noisy test data

Since most experimental data always contains noise, it is of interest to assess the performance of machine learning models trained on noiseless data and tested on noisy data (to simulate an actual measurement). The results for model performance on varying levels of noisy test data are displayed in Table 4. The results in Table 4 correspond to RMSE

Table 4. Impact of noise in test data on prediction RMSE.

Noise (%)	BU RMSE (MWd/kgU)				IE RMSE (wt. % U-235)				CT RMSE (days)			
	k-NN	RFR	AdaBoost	ANN	*k*-NN	RFR	AdaBoost	ANN	*k*-NN	RFR	AdaBoost	ANN
0.00	0.194	0.075	0.055	0.338	0.458	0.295	0.300	0.262	37.16	30.49	27.39	56.91
5.00	2.050	2.562	2.562	3.481	1.370	1.418	1.468	2.407	144.2	143.8	149.2	540.3
10.00	3.813	4.911	4.912	6.226	1.670	1.719	1.733	2.702	224.9	224.9	227.9	1033
20.00	7.092	9.076	9.080	12.26	1.874	1.908	1.961	2.999	408.9	390.9	392.2	1314

median values associated with fuel parameter predictions using the same algorithms as before, and with fuel samples having CT<20 years. The input features and hyperparameters were taken from Table 3 and tables from [18] respectively. The levels of Gaussian noise added to the test data varied from 0–20%. We observed that the prediction RMSE values increased as noise levels were increased for all algorithms. From the results, it is evident that the shallow learning algorithms make better predictions than neural networks when we have noisy test data. Among the shallow learning methods, the AdaBoost regressor performs the worst since it is known to be more sensitive to outliers (due to added noise) in the data. The *k*-NN algorithm performs the best in nearly all cases with increasing noise when compared to other algorithms except for CT prediction where it performs only marginally worse than random forest.

In summation, for situations where anticipated noise levels are low (<5%), the ideal choice would be to select a shallow learning algorithm since according to results from Table 4, prediction RMSE values are lower than those for neural networks. From the results it is not clear which one algorithm would be best-suited in all situations but rather it depends on fuel parameter being predicted and the noise level.

CONCLUSIONS AND OUTLOOK

In this work, we aimed to present a comparative study of different shallow and machine learning algorithms and their abilities to predict SNF properties such as BU, IE, and CT in a reliable manner. For this purpose we trained various algorithms such as *k*-NN, random forest, AdaBoost, and neural networks on a data set of simulated observables from the SNF (such as radionuclide activities, τ and Cherenkov light signatures). We have demonstrated that in most situations, shallow learning algorithms provide reliable predictions and have a clear edge over deep learning methods due to their ease of implementation and advantage of working "out-of-the-box". The one advantage of using neural networks for predicting fuel parameters would be their ability to simultaneously predict all three SNF parameters of interest instead of using standalone surrogate models for each parameter, albeit at the expense of slightly higher prediction RMSE. Neural networks also presented and added advantage of using pre-trained models by storing and reusing model

weights to reduce time to deployment of such models in the field. In our study, we have found that in all CT regimes investigated, shallow learning methods showed lower RMSE vales for prediction of BU (as low as 0.005 MWd/kgU), for prediction of IE (as low as 0.02 wt. % U-235), and for prediction of CT (as low as one day approx.). When compared to previous related work in the field, our prediction RMSE values are comparable to or lower than those reported in [5]. However, it is important to note that the analysis conducted in [5] involved the use of a different data set and different algorithms for prediction of fuel parameters. It should also be noted that the lowering of prediction RMSE could be attributed to the use of non-linear methods in the current study since non-linear algorithms are able to learn associations in the data better than linear algorithms. It may also be said that in our study, we have found that shallow learning methods fare better than deep learning methods in a noisy test data environment. Our choice of noise levels in the test data in this study was motivated by the typical noise levels in spent fuel measurement data. In the future, it is envisioned that one (or perhaps multiple) algorithms used in the present study will be used with SNF measurement data obtained from passive gamma-ray spectroscopy, Cherenkov light intensity measurements and τ values. Future research will also focus on improving the performance of these algorithms to attune them to experimentally obtained fuel signatures thereby making them useful in development of nuclear safeguards methodologies for safeguards inspectors performing inspections in the field.

REFERENCES

[1] International Atomic Energy Agency (IAEA). *Safeguards Techniques and Equipment:*. Number 1 (Rev. 2) in International Nuclear Verification Series. International Atomic Energy Agency, Vienna, 2011. URL.

[2] P. Jansson. *Studies of Nuclear Fuel by Means of Nuclear Spectroscopic Methods*. PhD thesis, Uppsala University, Department of Nuclear and Particle Physics, 2002.

[3] S. Grape, E. Branger, and Zs. Elter et al. Determination of spent nuclear fuel parameters using modelled signatures from non-destructive assay and random forest regression. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 969:163979, 2020. DOI.

[4] C. Hellesen, S. Grape, and P. Jansson et al. Nuclear spent fuel parameter determination using multivariate analysis of fission product gamma spectra. *Annals of Nuclear Energy*, 110:886–895, 2017. DOI.

[5] A. M. Bachmann, J. B. Coble, and S. E. Skutnik. Comparison and uncertainty of multivariate modeling techniques to characterize used nuclear fuel. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 991:164994, 2021. DOI.

[6] A. Borella, R. Rossa, and H. Zaloun. Determination of 239Pu content in spent fuel with the SINRD technique by using artificial and natural neural networks. *Esarda Bullettin*, 58:41–47, 6 2019. URL.

[7] A. Borella, R. Rossa, and C. Turcanu. Signatures from the spent fuel: simulations and interpretation of the data with neural network analysis. *Esarda Bullettin*, 55:29–39, 12 2017. URL.

[8] Zs. Elter, L. P. Balkeståhl, and E. Branger et al. Pressurized water reactor spent nuclear fuel data library produced with the Serpent2 code. *Data in Brief*, 33, 2020. DOI.

[9] E. Branger. `clip` - python package for estimating the Cherenkov light intensity of a spent nuclear fuel assembly for nuclear safeguards purposes. Software available from URL (accessed 2021-04-17).

[10] E. Branger. *Enhancing the performance of the Digital Cherenkov Viewing Device : Detecting partial defects in irradiated nuclear fuel assemblies using Cherenkov light*. PhD thesis, Uppsala University, 2018. URL.

[11] L. C. Balkeståhl, Zs. Elter, and S. Grape. Parameterization of the differential die-away self-interrogation early die-away time for PWR spent fuel assemblies. In *ESARDA Bulletin*, volume : 8, pages 13–21, 2019. URL.

[12] M. M. Bé, R. Helmer, and V. Chisté. The "NUCLÉIDE" Database for Decay Data and the "International Decay Data Evaluation Project". *Journal of Nuclear Science and Technology*, 39(sup2):481–484, 2002. DOI.

[13] K. Wilhelm. *Pearson's Correlation Coefficient*, pages 1090–1091. Springer Netherlands, Dordrecht, 2008. DOI.

[14] F. Pedregosa, G. Varoquaux, and A. Gramfort et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12, 01 2012. URL.

[15] M. Abadi, A. Agarwal, and P. Barham et al. `tensorflow`: Large-scale machine learning on heterogeneous systems, 2015. Software available from URL.

[16] T. O'Malley, E. Bursztein, and J. Long et al. Keras Tuner, 2019. URL.

[17] J. Snoek, H. Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *Proceedings of the 25th International Conference on NIPS - Volume 2*, 2012. URL.

[18] Vaibhav Mishra. vmishra-uu/INMM-ESARDA-VMishra-et-al: Initial release, August 2021. DOI.