LA-UR-18-24800 (Accepted Manuscript)

Using Machine Learning Methods to Predict Bias in Nuclear Criticality Safety

Grechanuk, Pavel Aleksandrovi
Rising, Michael Evan
Palmer, Todd

To be published in: Journal of Computational and Theoretical Transport

DOI to publisher's version: 10.1080/23324309.2019.1585877

Permalink to record: http://permalink.lanl.gov/object/view?what=info:lanl-repo/lareport/LA-UR-18-24800

25th International Conference on Transport Theory, Monterey, California, 16-20 October 2017

Using Machine Learning Methods to Predict Bias in Nuclear Criticality Safety

Pavel Grechanuk[1], Michael E. Rising[2], and Todd Palmer[3]

[1]*grechanp@oregonstate.edu,* [3]*palmerts@engr.orst.edu, Nuclear Science and Engineering Department, Oregon State University, Corvallis, OR, 97331, USA*

[2]*mrising@lanl.gov, XCP-3 Computational Physics Group, Los Alamos National Laboratory, MS B283, Los Alamos, NM 87545, USA*

Abstract

The manuscript provides a description of the application of machine-learning (ML) tools to the prediction of bias in criticality safety analysis. In particular, a set of over 1000 experiments included in the Whisper package were fed into a variety of ML algorithms (notably Random Forest and AdaBoost) implemented in SciKit-Learn using k-eigenvalue sensitivities (with and without energy dependence) for individual nuclides, and optionally, the simulated k_{eff} as the training features. Ultimately, the ML model was used to predict the bias (k_{sim} - k_{exp}). The results indicate that use of energy-integrated sensitivity profiles with k_{sim} as training features led to the best predictions as quantified by root-mean square and mean absolute errors. In particular, the best-case estimates came from AdaBoost, with a mean absolute error of 0.00174, which is less than the mean experimental uncertainty of 0.00328 for the experiments included.

1. Introduction

For real-world applications involving criticality safety, having a neutron multiplication factor $k < 1$ is not enough to ensure safety. Each application requires a validation study that incorporates uncertainty in both the nuclear data and the model to set an upper subcritical limit (USL), which represents an acceptable value for k_{sim} to ensure safety. The bias (simulated - experimental k_{eff}) in the calculation is the most important quantity used for code validation. The objective of this project is to accurately predict the bias of *MCNP6® [1] criticality calculations using machine learning (ML) algorithms, with the intention of creating a tool that can complement current nuclear criticality safety methods. A secondary objective of this study is to identify isotopes and reactions of interest that could be significantly influencing bias. In the latest release of MCNP6, version 6.2 [2], the Whisper tool version 1.1 [3] is available for criticality safety analysis and includes a large catalog of experimental benchmarks, sensitivity profiles, and nuclear data covariance matrices. The data originates from 1,100+ benchmark cases in Whisper, and is used in this study for predicting criticality simulation bias. The remainder of this paper includes background information in Section 2, methodology in Section 3, the results are detailed in Section 4, followed by a brief conclusion, and future work.

2. Background

2.1 Whisper Tool

Historically it was labor intensive, time consuming, and difficult to quantitatively defend an entire criticality safety validation study, and it was only done when an application was very different from

*MCNP® and Monte Carlo N-Particle® are registered trademarks owned by Los Alamos National Security, LLC, manager and operator of Los Alamos National Laboratory. Any third party use of such registered marks should be properly attributed to Los Alamos National Security, LLC, including the use of the designation as appropriate. For the purposes of visual clarity, the registered trademark symbol is assumed for all references to MCNP within the remainder of this paper.

previous validation efforts [3]. Whisper was developed at Los Alamos National Laboratory (LANL) to automate, speed up, and quantify in a defensible and reproducible way a significant portion of a validation study [4]. Historically, sensitivity and uncertainty methods are used to identify the area of applicability for a particular benchmark [5,6]. Recent incorporation of continuous-energy Monte Carlo adjoint-weighted sensitivity methods in production level codes like MCNP6 has made it easy to calculate sensitivity coefficients of the k-eigenvalue to nuclear data quantities of interest for many applications [7]. In general, the sensitivity coefficient is defined as:

$$S_{k,x} = \frac{\Delta k/k}{\Delta x/x}$$

where k is the neutron multiplication factor, and x is some parameter that is perturbed (cross section, material density, etc.). The sensitivity coefficient has the property of being additive, meaning the sum of the sensitivities over energy groups for a particular reaction is the total sensitivity to that reaction. The sensitivity profile for a particular benchmark incorporates a lot of information about the neutronics of the system. An example of a sensitivity profile can be seen in Figure 1.

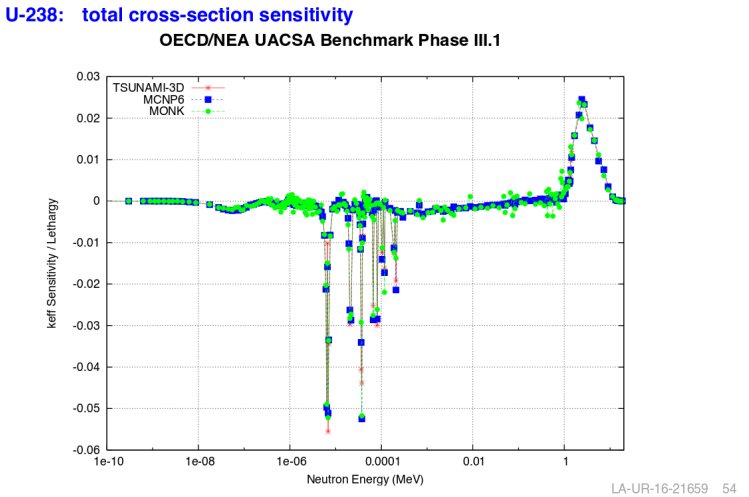

Figure 1: Example sensitivity profile of ^{238}U obtained from LANL Whisper presentation [8]. The effects of perturbing the total cross section at various energies is visible as increased parasitic absorption in the intermediate energy range and increased fast fission at high energy.

From the sensitivity profile it can be seen that the k_{eff} is sensitive to perturbations in the intermediate energy range, which physically makes sense, as that is where resonance absorption occurs. Additionally there is a peak in the fast energy ranges, which comes from increased fast fission. Essentially the sensitivity profiles detail what materials and reactions are important to the neutron multiplication of a given benchmark, and can be thought of as a fingerprint of the problem. Thus, Whisper uses the sensitivities along with nuclear data covariances to calculate similarity measures between applications and benchmarks [9]. The benchmarks with the highest similarity to the applications are then used to build an extreme value distribution for the bias, and the expectation is taken to obtain a conservative estimate of bias [4]. While having a conservative estimate for the bias is extremely important, knowing the expected bias provides another piece of information for the criticality safety analyst. This study aims to create a ML tool to complement Whisper, and to identify isotopes of interest when trying to reduce the bias of MCNP6 criticallity calculations.

2.2 Machine Learning

Machine learning methods allow computers to learn a specific task without explicitly being programed to do so by learning patterns present in data. A type of ML called supervised learning feeds the solutions, called labels, to the algorithm with the training data. A common supervised learning task is regression, which tries to predict a numeric value (example: home value) based on a given set of features (example: number of bedrooms, year of construction, square footage, etc.). During the training process, different parameters within the model are slowly optimized to reduce a cost function (mean squared error for regression) at the model output. Typically a ML algorithm performs best when there is a large quantity of data available from which to learn. If the training sample is too small, the data is non-representative of the whole picture, and the algorithm under-performs. This can also occur with a large dataset, if the data is not sampled properly (sampling bias) [10].

Typically the data fed to a ML algorithm is split into test and training subsets. The standard practice is to train on approximately 80% of the data, and then to evaluate the model on the remaining 20% of the data. The error rate on the testing subset is called the generalization error, which is the error rate to expect from the model on instances it has never encountered before. The best way to evaluate the performance of a model is to use a technique called cross validation [11]. This method works by subdividing the training set into a certain number of subsets (called folds), and the model is evaluated on one fold and trained on the remaining folds. This process is repeated for every possible combination until every fold has been evaluated, with the other folds acting as training data. The measures of fit from each round are averaged to calculate a more representative value of model generalization error.

2.3 Decision Trees

Decision Trees are powerful ML algorithms that are used for regression and classification. Scikit-Learn [12], an open source ML module in Python, uses the Classification and Regression Tree (CART) algorithm to "grow" decision trees [13]. The algorithm first takes the training data, and tries to split it into two subsets using a feature k and a threshold t. This is a numerical procedure where many threshold points are tested based on a feature in order to minimize a cost function, which for regression is either the mean squared error or the mean absolute error.

Once the dataset is split into two, each subset undergoes the same operation, until further splitting does not reduce the cost function. This eventually leads to a decision tree that has many branches (node splits) and leaves (node ends). The model works by starting at the top of the tree, and undergoing logical operators at each branch node until arriving at a leaf node which has an associated value with it. Decision trees are used in this study because of their simplicity and quick training.

A very useful feature of decision trees is that they can calculate the relative importance of each feature on predicting the estimated value. The most important features are more likely to appear near the start of the tree, while the less important features will appear further down in the tree. Scikit-Learn automatically estimates the feature importance by computing the average depth of each feature across all of the trees [12]. The feature importances show which pieces of data are important to the ML model in predicting the quantity of interest.

2.4 Random Forest

The Random Forest regressor is an ensemble method that trains each decision tree on a random subset of the dataset, with the intention of reducing over-fitting to the training set [14]. The individual trees also differ because they search through a random subset of features to find the best split. Building the trees in such a manner introduces extra randomness into the ensemble, and results in increased tree diversity citebreiman2001random. Restricting the problem space for each individual tree makes them specialists on a small subset of the dataset. When all of the individual tree predictions are averaged, a stronger model results because of the "wisdom of the crowds" effect. This method of grouping many predictors to make a single estimate is called ensemble learning [15]. The random forest algorithm trades a higher bias for a reduced variance, which reduces over-fitting to the training dataset and most often results in a better overall model.

2.5 Adaptive Boosting Ensemble

The Adaptive Boosting (AdaBoost) ensemble method goes through multiple iterations of training, where the next iteration tries to correct the predecessor [16]. The method works by first training a decision tree with all of the training instances having uniform weights on a subset of the training samples. The mean squared error is calculated for the model, and a predictor weight is extracted [17]. Using the predictor weight the instance weights are adjusted and the instances with higher error are boosted. The next iteration is trained on the data with the updated instance weights, the new predictor weight is calculated, the feature weights are updated, another predictor is trained, and so on. This algorithm forces the model to increasingly pay more attention to the cases that it gets wrong. The process is repeated until the desired number of iterations is reached or the error converges. To make the predictions on new instances the algorithm calculates the predictions of every predictor, weighs them using their predictor weights, and calculates the mean. This method produces results that are more accurate than the standard decision tree model, by increasingly focusing on the cases that it gets wrong.

3 Methods

3.1 Features and Target

The main objective of this study is to predict the bias ($k_{sim} - k_{exp}$) of MCNP6 criticality calculations using the Random Forest and AdaBoost ensemble methods. The ML algorithms were implemented in Python using the Scikit-Learn toolbox. The sensitivity vectors generated by MCNP6, which describe how the neutron multiplication factor (k) is impacted by the nuclear data of a given application, are chosen as the features for learning. In making this choice, we make the assumption that they inherently carry enough information to characterize a system, and consequently can be used to find patterns that influence bias. This assumption is valid as sensitivities have been historically used to find neutronically similar benchmarks in validation studies [5]. Each benchmark sensitivity profile contains data for 172 isotopes with each isotope having 12 reactions, and each reaction is broken into 44 energy bins. Note that each benchmark contains only a subset of the total 172 isotopes, resulting in nonzero sensitivity profiles for only the isotopes present in that benchmark. Since the features are broken into 44 energy groups, the relative importance of each isotope-reaction pair can be analyzed at the energy level. This provides insight into what cross sections are important to the ML model when predicting bias at the thermal, intermediate, and fast energies.

The models were also evaluated on a modified set of sensitivity vectors, where the sensitivity values for each isotope-reaction pair are summed over energy. This removes the energy dependence, making it so that the models are trained on the benchmark's sensitivity to each isotope-reaction pair. Using these energy-integrated sensitivity vectors significantly reduces training time and speeds up optimization. In order to generate the sensitivity profiles, a MCNP6 calculation must be performed, which also generates k_{sim}, another piece of information that can be used as a feature to improve the accuracy of the model.

Ten-fold cross validation was used to evaluate the mean average error (MAE) and the root mean squared error (RMSE) of each algorithm. Each model has a set of hyper-parameters that act as tuning knobs, which affect how closely the model fits to the training instances (regularization) [18]. Some of these hyper-parameters are the maximum features each tree learns on, the total number of trees, and the maximum size of a tree. To find an optimal set of hyper-parameters for the models, a grid search was performed in which the cross validation scores were computed for different combinations of parameter values specified by a grid.

3.2 Removing Outliers

The cases on which the models make the greatest errors are effectively outliers, since there are not enough similar cases to make accurate predictions. To better learn on those difficult cases more training data is needed. The presence of these outliers can skew the model during training, so that the validation error increases, since the models tend to focus on the more numerous classes to minimize the overall cost [19]. As a result, the top 50 cases with the greatest error initially are removed from training and testing in this study (approximately 5% of the data), which leaves 1,051 cases for training and testing. There is nothing qualitatively wrong with the removed benchmarks, other than being dissimilar to the majority of the cases. Of the 50 cases that were excluded here, 34 are also excluded (treated as outliers) from the Whisper library by using an iterative Chi-squared rejection method prior to validation.

4 Results

4.1 Sensitivity Vectors

The first thing to verify was the assumption that the k_{eff} sensitivities are good features for ML. To verify that assumption the models were only trained on the sensitivities without k_{sim}, and the results can be seen in Table I.

Model	RMSE	MAE
R. Forest (I)	0.00499	0.00350
AdaBoost (I)	0.00498	0.00352
R. Forest (D)	0.00572	0.00397
Adaboost (D)	0.00537	0.00374

Table I: Error statistics for both models on the energy independent (I) and dependent (D) datasets. Obtained from 10 fold cross validation.

Table I demonstrates that the models are learning to predict the bias accurately using the sensitivities as features, which demonstrates their utility as features [20]. The mean experimental uncertainty in the benchmark measurements of k used in this study is 0.00328, and the models are approaching that value on

average. Both models perform better on the energy integrated dataset, which is most likely a result of not enough training instances to learn on the full dataset, as it is very high dimensional (approximately 90,000 features per benchmark).

To evaluate the models, a plot of the errors on each benchmark, and the distribution of the error is plotted in Figure 2 for the AdaBoost model trained on the summed dataset.

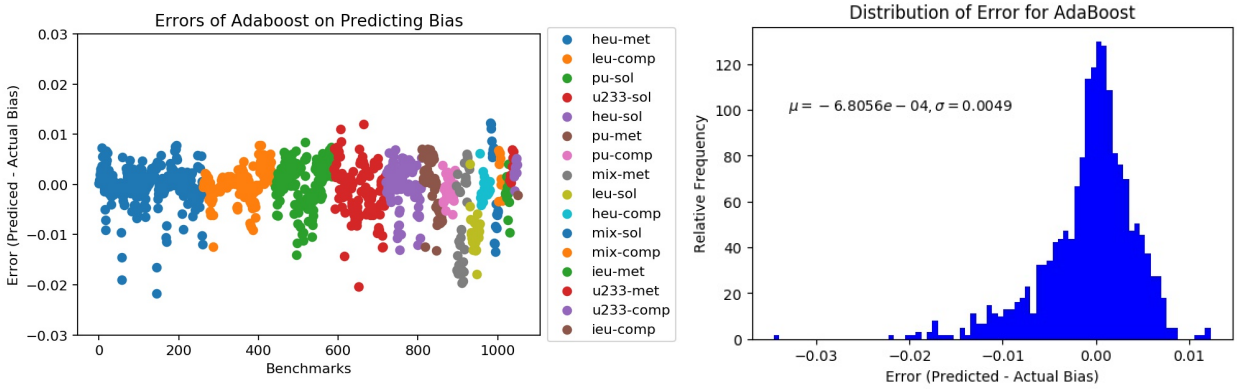

Figure 2: Errors of the AdaBoost model on each benchmark (left), and the distribution of the error (right), when trained and tested using the energy independent data. Each color represents a material classification for the benchmark in the Whisper library.

The plot of the error distribution demonstrates that the model is learning the dataset, because the error forms a distribution around zero. Methods to estimate the confidence intervals and variance of decision tree predictions exist in literature, but were not implemented in this study [21,22]. The model performs well on the benchmark classes that are numerous, and experiences significant error for some of the rare cases (seen on the right hand side of the left plot). These results show that the sensitivities of k_{sim} perform well as features for ML. In general, these results imply that sensitivities of global quantities calculated with respect to the inputs of a particular model have the potential to be good features for ML. Performing ML on physics models using this methodology could provide insights about the problem space that would be difficult to obtain otherwise.

4.2 Sensitivity Vectors and k_{sim}

Now that the sensitivities have been shown to be good features for ML, more information can be included during training, like the calculated value of k_{sim} for each benchmark. The models were trained on the sensitivities along with k_{sim}, and the results are summarized in Table II.

Model	RMSE	MAE
Random (I)	0.00376	0.00241
AdaBoost (I)	0.00292	0.00174
Random (D)	0.00537	0.00360
Adaboost (D)	0.00522	0.00352

Table II: Error statistics for both models on the energy independent (I) and dependent (D) datasets when using k_{sim} as a feature. Obtained from 10 fold cross validation.

The performance of these models reinforces that the sensitivity profiles carry enough information to characterize a system, which is then used by the models to find patterns to predict the bias accurately. The models training and testing on the energy dependent dataset performed worse, which could mean the decision tree forests lack the complexity needed to sufficiently learn on the full dataset. A plot of the errors on each benchmark, and the errors on each benchmark type are included in Figure 3 for the AdaBoost model trained on the energy independent dataset and k_{sim}.

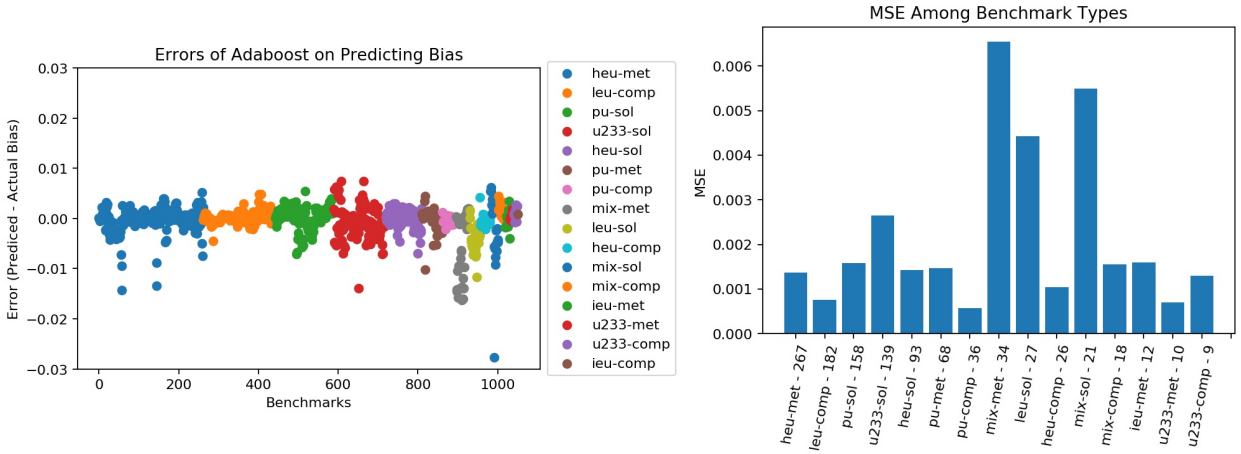

Figure 3: Errors of the AdaBoost model on each benchmark (left), and the errors on each benchmark type (right), when trained and tested using the energy independent data and k_{sim}. Each color represents a material classification for the benchmark in the Whisper library.

The error decreases significantly after including k_{sim}, which is observed in the tighter distribution of the error around zero. The box plot (right side of Fig. 3) shows that the models perform well on most of the benchmarks, but a few benchmarks generate high relative error. Both of the models make similar errors on both of the datasets, which means that either the models are not complex enough to learn the features or there are not enough benchmarks of that type. It is most likely a combination of the two, since the feature space is quite large, and there are only a few training examples for those rare benchmark classes. Overall the models perform accurately in predicting the bias, and the sensitivities perform well as features for ML. These results show that the bias can be accurately estimated when using the sensitivities and k_{sim} as the features for ML.

4.3 Comparison to Whisper

Whisper calculates bias using extreme value theory in order to provide a conservative estimate for criticality calculations. Additionally Whisper also uses the generalized linear least squares method (GLLSM) to calculate a minimum margin of sub-criticality due to nuclear data uncertainties [9]. Specifically the GLLSM method is used to obtain an adjusted nuclear data covariance library, which can then be used to modify the cross-sections in order to increase the agreement between calculated and experimental k_{eff} measurements. This produces a modified value for k_{sim} that is closer to the experimental one, and has a reduced uncertainty. A comparison of the error statistics between Whisper, GLLSM, and the ML models can be found in Table III.

The error statistics for Whisper are relatively large, as expected since Whisper uses extreme value theory to provide a conservative estimate of bias, which tries to be larger than the actual bias. The comparison

Model	RMSE	MAE
Whisper	0.01329	0.00906
GLLSM	0.00959	0.00645
R. Forest (I)	0.00499	0.00350
AdaBoost (I)	0.00498	0.00352
R. Forest (D)	0.00572	0.00397
Adaboost (D)	0.00537	0.00374

Table III: Error statistics for the ML models trained only on the sensitivities from 10 fold cross validation, GLLSM, and Whisper.

against Whisper and the GLLSM method is not direct because the ML models are not bounded by the uncertainty of the cross section data. Nonetheless, if one wishes to know the bias accurately the ML algorithms are shown to be the superior method when compared against Whisper and the GLLSM.

4.4 Feature Importances

Relative feature importances can be calculated by taking the average depth of every feature across the trees in an ensemble. This value represents the relative importance of that feature in predicting the desired output. In the context of this study, the relative importance represents how influential an isotope-reaction sensitivity is on the bias prediction. Feature ranking with recursive feature elimination and cross-validated selection (RFECV) was used to find the best features in the dataset to use in the study [12]. This method works by first fitting a model to all of the features, and the two least important features to predicting the bias are dropped. This continues as an iterative process using cross validation at each step to evaluate model performance until the model error converges. This approach was used with the summed sensitivity vectors and the Random Forest regressor.

Of the 2,066 unique isotope reaction pairs, 584 are dropped, and the modified dataset is fed through the models again. This removes both redundancy and noise from the data set, as the model is only trained on the features that are important to predicting the bias. The top ten important isotope-reaction pairs to the Random Forest regressor on this dataset are included in Table IV. These are the isotopes and reactions that the model considers first, when trying to predict the bias.

Isotope, Reaction	Importance
^{233}U, n-gamma	0.04681
^{232}U, total nu	0.04510
^{232}U, fission	0.03933
^{234}U, n-gamma	0.03528
Carbon, n-gamma	0.03235
^{234}U, fission	0.03165
^{234}U, total nu	0.03093
^{232}U, n-gamma	0.02773
Carbon, n-alpha	0.02552
Carbon, inelastic	0.02441

Table IV: Top ten relative importances of each isotope reaction to the Random Forest regressor after performing RFECV. There are only four nuclides present in this list: natural carbon, ^{232}U, ^{233}U, ^{234}U.

Because the sensitivities are a function of energy (44 groups), the importance of each isotope-reaction pair can be analyzed over energy group ranges, which provides insight into which reactions and neutron energies are important to predicting the bias. Many cross sections have high relative uncertainty, and it is difficult to decide which experiments (evaluating nuclear data) to run in order to reduce MCNP6 bias effectively. The energy discretized importance combined with cross section uncertainties could inform which cross sections or secondary distributions should be improved to most efficiently reduce the bias in MCNP6. The top five most important reactions in each of three energy group ranges can be found in Table V.

Thermal (0 - 0.625 ev)	Intermediate (1.0 ev - 0.1 Mev)	Fast (0.4 Mev - 20 Mev)
Carbon, n-gamma, 0.014562	^{233}U, n-gamma, 0.018457	^{233}U, fission, 0.015264
^{233}U, total nu, 0.011437	^{233}U, fission, 0.015724	^{233}U, inelastic, 0.013543
^{233}U, n-gamma, 0.010641	^{233}U, total nu, 0.012844	^{233}U, n-gamma, 0.012739
^{234}U, n-gamma, 0.009479	^{234}U, n-gamma, 0.011945	^{233}U, total nu, 0.012644
^{1}H, n-gamma, 0.009069	^{239}Pu, n-gamma, 0.011687	^{19}F, inelastic, 0.010355

Table V: Top five relative importances of each isotope-reaction by energy group to the Random Forest regressor in calculating bias. These importances make physical sense: capture at low energies, capture and fission for actinides at intermediate energies, inelastic scattering and fission at high energies.

These results are physically explainable, as they are common reactions and isotopes expected in nuclear materials. One isotope that is not typically expected to be of high importance is U-234, since it generally only appears in trace amounts. However, the ^{234}U n-gamma reaction is the fourth most important isotope-reaction pair, as seen in Table IV. Interestingly, the nuclear cross section for the ^{234}U n-gamma reaction has a high relative uncertainty over the same energy range in which it is important to the ML model. This suggests that some of the high importance reactions are indeed important to predicting bias, due to inaccuracies in the cross sections, which are leading to increased bias. While it is an intuitive result, it shows that the models are learning which reactions are influencing the bias. Another interesting result is the presence of multiple ^{233}U reactions over all three energy ranges, since it is also a trace isotope. To investigate this isotope further the uncertainty over the energy range is plotted alongside the relative importance in Figure 4.

As can be seen in the figure the uncertainty and feature importance are fairly correlated, which is an interesting result, since the machine learning algorithms are not trained on the uncertainties. The uncertainty of other high importance isotopes is also strongly correlated with the feature importances, and typically the more important an isotope is to the ML model, the stronger the correlation. Investigating reactions that are both important and uncertain could provide insights about where effort should be focused to reduce bias efficiently.

5 Conclusion

Machine learning models have been shown to accurately predict the bias of MCNP6 criticallity simulations, by using the k eigenvalue sensitivities as features for ensembles of decision trees. The methodology developed could be using in conjunction with Whisper in order to provide more information to the criticallity safety analyst. The mean experimental uncertainty in the benchmark measurements of k used in this study is 0.00328, and the mean absolute error is below that for all of the models. On average the error of the ML models is approaching the uncertainty of experimental k measurements. The accurate

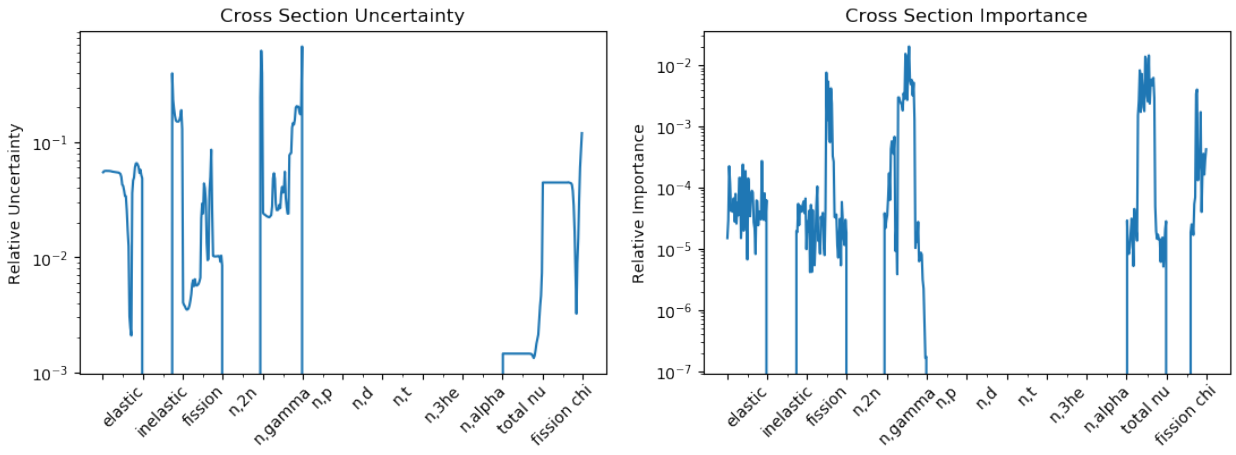

Figure 4: Uncertainties for the various reactions for ^{233}U (left), and feature importances for the same reactions obtained from the Random Forest regressor (right). The Spearman correlation coefficient between the uncertainty and feature importance is 0.7894.

performance of these models shows that the sensitivity coefficient is able to characterize the problem effectively, and thus provides a strong feature for ML. In general these results imply that sensitivities of global quantities of interest (calculated with respect to the inputs of a model), might serve as good features for ML. Exploring this concept using validation studies for other physics modeling software is of interest. Additionally the feature importances show which isotopes and reactions are the most important to predicting the bias, which in turn can inform which cross sections and models should be prioritized when trying to reduce bias.

6 Future Work

Additional research is needed to investigate the correlation between the feature importances and cross section uncertainty, and to investigate the performance of ML algorithms on a more diverse set of benchmarks. A study similar to this could be repeated with more features like the average neutron energy causing fission, the prompt removal lifetime, the percentages of fissions caused by neutrons in the thermal, intermediate, and fast neutron ranges, and other data provided by MCNP6 outputs. Currently work is underway on applying convolutional neural networks on this dataset, which should be able to learn on the full dataset effectively. Additionally uncertainty analysis can be explored, in order to provide confidence intervals for predictions made by the ML tools.

Acknowledgements

This work was supported by the DOE Nuclear Criticality Safety Program, funded and managed by the National Nuclear Security Administration for the Department of Energy.

References

[1] J. T. Goorley, M. James, T. Booth, F. Brown, *et al.*, "Initial MCNP6 Release Overview," *Nuclear Technology*, vol. 180, Jan 2012.

[2] C. Werner, J. Bull, C. Solomon, *et al.*, "MCNP6.2 Release Notes," Tech. Rep. LA-UR-18-20808, Los Alamos National Laboratory, 2018.

[3] F. B. Brown, M. E. Rising, and J. L. Alwin, "User Manual for Whisper-1.1," Tech. Rep. LA-UR-17-20567, Los Alamos National Laboratory, 2017.

[4] B. C. Kiedrowski, F. B. Brown, *et al.*, "Whisper: Sensitivity/uncertainty-based computational methods and software for determining baseline upper subcritical limits," *Nuclear Science and Engineering*, vol. 181, no. 1, pp. 17–47, 2015.

[5] B. L. Broadhead, B. T. Rearden, *et al.*, "Sensitivity-and uncertainty-based criticality safety validation techniques," *Nuclear science and engineering*, vol. 146, no. 3, pp. 340–366, 2004.

[6] B. T. Rearden, "Perturbation theory eigenvalue sensitivity analysis with Monte Carlo techniques," *Nuclear science and engineering*, vol. 146, no. 3, pp. 367–382, 2004.

[7] F. B. Brown, M. E. Rising, *et al.*, "MCNP-WHISPER Methodology for Nuclear Criticality Safety Validation," Tech. Rep. LA-UR-16-23757, Los Alamos National Laboratory, 2016.

[8] F. B. Brown, M. E. Rising, and J. L. Alwin, "Lecture notes on criticality safety validation using mcnp & whisper," tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2016.

[9] B. Kiedrowski and F. Brown, "Methodology, verification, and performance of the continuous-energy nuclear data sensitivity capability in mcnp6," in *Proceedings of the 2013 International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, 2013.

[10] B. Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *Proceedings of the twenty-first international conference on Machine learning*, p. 114, ACM, 2004.

[11] P. Refaeilzadeh, L. Tang, and H. Liu, "Cross-validation," in *Encyclopedia of database systems*, pp. 532–538, Springer, 2009.

[12] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[13] D. G. Denison *et al.*, "A bayesian CART algorithm," *Biometrika*, vol. 85, no. 2, pp. 363–377, 1998.

[14] A. Liaw, M. Wiener, *et al.*, "Classification and regression by randomForest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[15] T. G. Dietterich, "Ensemble methods in machine learning," in *International workshop on multiple classifier systems*, pp. 1–15, Springer, 2000.

[16] D. P. Solomatine and D. L. Shrestha, "AdaBoost. RT: a boosting algorithm for regression problems," in *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, vol. 2, pp. 1163–1168, IEEE, 2004.

[17] H. Drucker, "Improving regressors using boosting techniques," in *ICML*, vol. 97, pp. 107–115, 1997.

[18] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[19] R. C. Prati *et al.*, "Data mining with imbalanced class distributions: concepts and methods.," in *IICAI*, pp. 359–376, 2009.

[20] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial intelligence*, vol. 97, no. 1-2, pp. 245–271, 1997.

[21] T. G. Dietterich and E. B. Kong, "Machine learning bias, statistical bias, and statistical variance of decision tree algorithms," tech. rep., Technical report, Department of Computer Science, Oregon State University, 1995.

[22] S. Wager, T. Hastie, and B. Efron, "Confidence intervals for random forests: The jackknife and the infinitesimal jackknife," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1625–1651, 2014.