Machine Learning Augmented Nuclear Data Evaluations

by

Pedro Junior Vicente Valdez

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering – Nuclear Engineering

in the

Graduate Division

of the

University of California, Berkeley

Spring 2021

Machine Learning Augmented Nuclear Data Evaluations

Abstract

Machine Learning Augmented Nuclear Data Evaluations

by

Pedro Junior Vicente Valdez

Doctor of Philosophy in Engineering – Nuclear Engineering

University of California, Berkeley

Professor Massimiliano Fratoni, Chair

With the fast increase in computational power over the last decade, including the development of better Graphical Processing Units (GPUs), the field of Machine Learning (ML) and Artificial Intelligence (AI) has improved drastically since its inception in the 1980s. With more efficient algorithms and faster training times, the prominence of ML/AI throughout business and technology sectors has only grown. However, its extension to Nuclear Physics and Engineering remains limited to date. In this work, we explore the possible roles of ML in the Nuclear Data Evaluation pipeline including the development of (1) NucML and (2) an ML-based solution for neutron-induced evaluations. To catalyze future development in the area, NucML, the first end-to-end ML-augmented nuclear data evaluation pipeline was developed. This Python toolbox contains various capabilities ranging from loading ML-ready datasets to automatic validation of trained models using an in-house developed criticality benchmark public repository. It allows any user to focus on model development, in addition, to quickly navigating each step of the proposed enhanced pipeline in a modular easy-to-use fashion. Using the ML-enhanced framework and the newly developed code, several proof-of-concept ML models including Decision Tree, K-Nearest-Neighbor, and Gradient Boosting Machines were fitted to nuclear data from the EXFOR database to infer neutron-induce reaction cross sections. All models were used to predict nuclear data for several well-characterized isotopes in literature including ^{239}Pu, ^{238}U, ^{235}U, and ^{233}U. Afterward, data for several isotopes of interest including ^{35}Cl, a less studied but important nuclide for advanced nuclear reactors, was generated. The predicted values for several of these isotopes were used to create an evaluation and tested using various benchmarks using SERPENT2 including the ^{233}U Jezebel and ^{235}U-reflected ^{233}U spheres. Results for these proof-of-concept models in some cases outperforming the current release of the ENDF library by up to 200%. Once the models were validated, they were used to predict the ^{35}Cl(n,p)^{35}S at the same energy points as the new LBNL/UCB measurements. The predicted values described the cross sections 150% more accurately than any of the evaluated data libraries, which overestimate experimental results by up to a factor of five. The proof-of-concept models explored in this work, reliant on learning underlying patterns of cross section data from other radionuclides, demonstrate evidence that ML models can aid traditional physics-guided models and have a role to play in nuclear data evaluations. Furthermore, incorporating ML models in the nuclear data pipeline can allow evaluators to make faster human-bias-free decisions in areas of uncertainty as well as better inform future data measurement campaigns on areas of greatest sensitivity in EXFOR.

Han sido tiempos difíciles, fáciles, de mucho trabajo, y también de mucha flojera. Fue durante este periodo del doctorado que realmente me di cuenta de que la vida se tiene que disfrutar y que no todo es trabajo. Muchos te lo dicen, pero nunca lo aprendes hasta experiméntalo. Puede que ni si quiera lo creamos pero si existe ese balance entre ambas responsabilidades y una vez encontrada, todo es mejor.

Estoy muy orgulloso de todo lo que se he logrado hasta ahorita. Todo se lo debo a mi padre, madre, mis tres hermanos, mi hermanita, y a mi pareja que me aguanto por todo este proceso. Espero sepan que los quiero. Esta meta se los dedico a ustedes… son mi todo.

Morgan Fox... we did it!

Contents

Contents ii

List of Figures iv

List of Tables viii

1 Introduction 1
1.1 Nuclear Data and Data Evaluation Overview 1
1.2 Machine Learning 5
1.3 Motivation, purpose, and scope of the thesis 6

2 Nucleus and Nuclear Reaction Background 9
2.1 Nucleons 9
2.2 Nuclear Reactions and Reaction Mechanisms 14
2.3 Reaction Types 19
2.4 Cross Section 22
2.5 Current Experimental and Modeling Needs 23
2.6 Summary 24

3 Methodology - NucML 25
3.1 Machine Learning-augmented Nuclear Data Evaluation 25
3.2 NucML: Python-toolbox for Accelerated ML-based Evaluations 28
3.3 Summary 41

4 Data and Benchmarks 42
4.1 EXFOR 43
4.2 Atomic Mass Evaluation 57
4.3 Feature Engineering 57
4.4 Evaluated Libraries 60
4.5 Feature Processing and Data Preparation 61
4.6 Data Sets 70
4.7 Criticality Benchmarks 70
4.8 Summary 75

5 Machine Learning Algorithms **76**
5.1 K-Nearest-Neighbors (KNN) 76
5.2 Decision Tree (DT) 81
5.3 Gradient Boosting Machines (GMB) 88
5.4 Summary 96

6 Results **98**
6.1 NucML 98
6.2 Criticality Benchmarks 99
6.3 ^{35}Cl(n,p)^{35}S Cross Section 111
6.4 Other Individual Isotopes of Interest 116
6.5 EXFOR-SQL 117

7 Conclusion and Future Work **119**

Bibliography **122**

A K-Nearest Neighbors Models **124**
A.1 Performance Metric Plots 124

B Decision Tree Models **127**
B.1 Gini Importance Plots 127

C XGBoost Models **130**
C.1 Other Hyperparameter Optimization Figures 130
C.2 'Gain'- and 'Weight'-based Importance Plots 132

D Incorrect/Outlier EXFOR Datasets **138**

List of Figures

1.1 Nuclear data evaluation pipeline. Taken from Reference [2]. 4
1.2 Proposed machine learning augmented nuclear data evaluation pipeline. 8

2.1 Mean binding energy per nucleon (BE/A) as a function of the mass number (A). Standard deviation shown in light blue. 11
2.2 Packing fraction as a function of the mass number (A). Nuclei above the horizontal line at zero indicate instability and the possibility of energy release by fusion or fission to reach lower packing fraction values (higher binding energy nuclei). . . 12
2.3 Number of Protons vs Number of Neutrons. The orange line represents the $Z = N$ behaviour. Deviations from the trend are mostly in terms of neutron excess. . . 13
2.4 Excitation levels in mirror nuclei ^{37}Ca and ^{37}Cl. Image taken from Reference [9]. 14
2.5 ^{35}Cl(n,total) cross section plot. 16
2.6 Excitation levels for ^{36}Cl. 17
2.7 ^{238}U(n,γ)^{239}U reaction cross section. The first excited levels of ^{239}U are shown connected to the respective resonance peak. Image taken from Reference [9]. . . 21

3.1 NucML Validation and Model Selection Process using the Benchmark Repository. 37
3.2 ML-enhanced Nuclear Data Evaluation Steps and applicable NucML Modules. . 40

4.1 EXFOR's energy histogram before (left) and (after) applying the logarithm transformer. 46
4.2 EXFOR's cross section histogram before (left) and (after) applying the logarithm transformer. 46
4.3 ^{56}Fe(n,total) reaction channel cross section. 49
4.4 ^{35}Cl(n,p)^{35}S reaction channel cross section. 49
4.5 ^{235}U(n,g)^{236}U Reaction Cross Section plot. Left figure shows box plot of the cross section values. 51
4.6 Filtered ^{235}U(n,g)^{236}U Reaction Cross Section plot. Left figure shows box plot of the cross section values. 51
4.7 Outlier data according to different outlier detection algorithms. Contamination was set to 0.3%. 52
4.8 Proton and Neutron Distribution in the extracted EXFOR datapoints. 55

4.9 Nucleus Radius and Neutron-to-Nuclide Radius Ratio Histogram. 59
4.10 ^{99}Tc(n,elastic)^{99}Tc reaction cross section. 61
4.11 AME isotopic data (orange) for all Uranium isotopes. Blue points signify interpolated data. 64
4.12 Missing Uncertainties as a function of the Experimental Campaign's Year. . . . 65
4.13 Beryllium-reflected Plutonium Core Criticality Benchmark Computational Model. Image taken from Reference [3]. 73

5.1 Effect of Different Normalizer/Transformer on the Validation MAE. 79
5.2 Effect of the "Distance Metric" (left) and "Energy" Min-Max Normalization (right) on the Validation MAE. 80
5.3 Performance Metrics as a Function of the k Hyperparameter for KNN Models Trained on Dataset A. 80
5.4 Effect of "Max Depth" on the Training and Validation MAE using Dataset A. . 84
5.5 Effect of the Max Depth and the MSS on the Training MAE using Dataset A. . 84
5.6 Effect of the Max Depth and the MSS on the Validation MAE using Dataset A. 85
5.7 Effect of the Max Depth and the MSL on the Training MAE using Dataset A. . 85
5.8 Effect of the Max Depth and the MSL on the Validation MAE using Dataset A. 86
5.9 Effect of the MSS and the MSL on the Training MAE using Dataset A. 86
5.10 Effect of the MSS and the MSL on the Validation MAE using Dataset A. 87
5.11 Feature Gini Importance for Decision Tree model trained on Dataset A. 88
5.12 Effect of "L2" and "Max Depth" on the Training MAE. 90
5.13 Effect of "L2" and "Max Depth" on the Validation MAE. 91
5.14 Effect of "Max Bin" and "Max Depth" on the Training MAE. 92
5.15 Effect of "Max Bin" and "Max Depth" on the Validation MAE. 92
5.16 Effect of "Max Bin" and "L2" on the Training MAE. 93
5.17 Effect of "Max Bin" and "L2" on the Validation MAE. 93
5.18 Training and Validation Loss as a Function of the Number of Trees (rounds) for the best XGB Model trained on Dataset A. 94
5.19 'Average Gain'-based Feature Importances for the XGBoost model trained on Dataset A. 95
5.20 'Weight/Split'-based Feature Importances for the XGBoost model trained on Dataset A. 96

6.1 Train and validation MAE as a function of the number of neighbors (k) (top). Validation MAE and Multiplication Factor (k_{eff}) error relative to 1 as a function of the number of neighbors (k) (bottom). Both the train and validation MAE are in a $\log_{10}$ scale. 102
6.2 Comparison of KNN generated and ENDF/B-VIII.0 cross sections for fission and radioactive capture. 103
6.3 ^{233}U(n,elastic) reaction channel experimental datapoints in EXFOR vs the ENDF/B-VIII.0 evaluation. 104

6.4 KNN inferred cross section values for the ^{233}U elastic channel vs the ENDF/B-VIII.0 evaluation. 105
6.5 Train and validation mean absolute error (MAE) as a function of the max depth and the multiplication factor error relative to 1. Both the train and validation MAE are in a $\log_{10}$ scale. 106
6.6 ^{9}Be(n,tot) Cross Section Values in EXFOR. 110
6.7 ^{35}Cl(n,p)^{35}S Cross Section Generated using a trained KNN model. 112
6.8 ^{35}Cl(n,p)^{35}S Cross Section Generated using a trained Decision Tree model. . . . 112
6.9 ^{35}Cl(n,p)^{35}S Cross Section Generated using a trained XGBoost model. 113
6.10 Energy Distribution of all the (n,p) reaction datapoints in EXFOR and of the Available ^{35}Cl(n,p)^{35}S values. 115
6.11 ^{12}C(n,tot) Cross Section Values in EXFOR. 116
6.12 ^{35}Cl(n,tot) Cross Section Values in EXFOR. 117

A.1 Performance Metrics as a Function of the *k* Hyperparameter for KNN Models Trained on Dataset B. 124
A.2 Performance Metrics as a Function of the *k* Hyperparameter for KNN Models Trained on Dataset C. 125
A.3 Performance Metrics as a Function of the *k* Hyperparameter for KNN Models Trained on Dataset D. 125
A.4 Performance Metrics as a Function of the *k* Hyperparameter for KNN Models Trained on Dataset E. 126

B.1 Feature Gini Importance for Decision Tree model trained on Dataset B. 127
B.2 Feature Gini Importance for Decision Tree model trained on Dataset C. 128
B.3 Feature Gini Importance for Decision Tree model trained on Dataset D. 129

C.1 Training and Validation MAE as a function of the Max Depth on XGBoost Models trained on EXFOR data. Better performance is achieved as the Max Depth is increased. Higher-memory GPUs should be used to train models with more predicting capabilities. 130
C.2 Training and Validation MAE as a function of the Max Bin on XGBoost Models trained on EXFOR data. The Max Bin is an important hyperparameter to increase for optimal Training and Validation performance. 131
C.3 Training and Validation MAE as a function of the L2 Regularization Strength on XGBoost Models trained on EXFOR data. L2 values of 2 and 3 shown slightly better overall performance. 131
C.4 Gain-based Feature Importances for the XGBoost model trained on Dataset B. . 132
C.5 Weight-based Feature Importances for the XGBoost model trained on Dataset B. 133
C.6 Gain-based Feature Importances for the XGBoost model trained on Dataset C. . 134
C.7 Weight-based Feature Importances for the XGBoost model trained on Dataset C. 135
C.8 Gain-based Feature Importances for the XGBoost model trained on Dataset D. 136

C.9 Weight-based Feature Importances for the XGBoost model trained on Dataset D. 137

D.1 ^{9}Be(n,tot) Cross Section Data from EXFOR. 138
D.2 ^{12}C(n,tot) Cross Section Data from EXFOR. 139

List of Tables

4.1 EXFOR Dataset Features[a]. 44
4.2 Reaction Channel Distribution. 47
4.3 EXFOR Status Distribution. 53
4.4 Center-of-Mass Distribution. 53
4.5 Identifier Tag for Fields 7 and 8 in the EXFOR C4 Files. 54
4.6 Target and Product Meta-Stable State Distribution. 54
4.7 Data point Counts for Various Isotopes. 56
4.8 Natural/Isotopic EXFOR Campaigns. 56
4.9 Natural Target Distribution. 57
4.10 Atomic Mass Evaluation Features[a]. 58
4.11 Neutron and Proton Valence Type Distribution. 60
4.12 Evaluated Libraries Versions. 60
4.13 Dataset Split. 66
4.14 EXFOR and AME features skewness. 67
4.15 Numerical Reaction Channel (MT) Representation. 69
4.16 ^{233}U Jezebel Assembly Characteristics. 71
4.17 ^{233}U Jezebel Composition. 71
4.18 Model Description and Dimensions. 72
4.19 Atom Density for the Uranium Spheres. 72
4.20 Plutonium Core and Beryllium Shell Compositions. 74
4.21 Model Description and Dimensions. 74
4.22 Tungsten-Reflected Plutonium Sphere Composition. 75

5.1 Experimented K-Nearest-Neighbor Model Parameters. 78
5.2 Final Set of KNN Hyperparameters and Performance Metrics. 81
5.3 Experimented Decision Tree Model Parameters. 82
5.4 Final Set of Decision Tree Hyperparameters 87
5.5 Experimented XGBoost Hyperparameters. 89
5.6 Final Set of XGBoost Model Hyperparameters. 94

6.1 ^{233}U Jezebel Benchmark Results. 101
6.2 U233-MET-FAST-002 10kg Benchmark Results. 108

6.3 U233-MET-FAST-002 7.6kg Benchmark Results. 109
6.4 Average Criticality Benchmark Error. 111
6.5 Evaluated Libraries and ML Prediction Values for the ^{35}Cl(n,p)^{35}S LBNL New Measurements. 114

Acknowledgments

I am eternally grateful to my advisor, Massimiliano Fratoni, for believing in me and this project since day one. His support and advice as a mentor, researcher, and instructor throughout both my master's and doctorate studies were invaluable. Thank you for always having your door open (even when I forgot to make an appointment).

Moreover, I want to thank every member of my graduate committee: Lee Bernstein and Zachary Pardos, and Peter Hosemann. This work would not be possible without the education and advice provided by all members hereby mentioned. A special mention goes to Bethany Goldblum for her feedback and for allowing me to gain Machine Learning experience at the Complexity group which catalyzed my future development as a machine learning and data science practitioner.

Chapter 1

Introduction

Nuclear technology is part of many industries ranging from energy generation to agriculture and food. Most popularly, nuclear reactors are used all over the world for clean and reliable power generation. Other applications of reactors include marine and space propulsion and energy generation due to the long-lasting performance and durability of nuclear-based technologies. In nuclear medicine, radioactive tracers are given to patients allowing the generation of images by detecting the outgoing radiation. Other imagining techniques including CT and PET scans also rely on nuclear-based principles. Sterilization and targeted radioisotope therapy are two additional medical applications of nuclear radiation.

In order to design, simulate, and build these systems, accurate nuclear data must be measured, compiled, and evaluated. Many of the previously mentioned and other technologies were successfully built due to the availability of relevant experimental data collected throughout the years. However, appropriate datasets for many of the new prospective technologies are still not available. This limits the possibility of further innovation and development. One of the most important technology concepts includes advanced nuclear reactors (ANR) including the Molten Salt Reactor (MSR). In a world where climate change is a pressing concern, clean energy technologies must be fast-tracked. To bring these concepts to reality, new nuclear data must be gathered and evaluated and state-of-the-art tools must be developed for better decision making in areas of uncertainty.

1.1 Nuclear Data and Data Evaluation Overview

Nuclear data encompasses a variety of data types that represent relevant nuclear physics. These include particle and nuclei cross sections, emitted particle energy spectra, angular distributions, fission product yields, nuclear structure, decay properties, and more. Accurate and representative data is required for the design, simulation, and construction of nuclear systems. The United States government and many other international private and public entities have spent considerable resources collecting experimental nuclear data throughout the decades to better understand the behavior of isotopes important for nuclear power and nuclear weapon development such as uranium and plutonium isotopes. Additionally, data for many of the structural materials used in the current fleet of light water reactors have been extensively researched. Nevertheless, significant gaps remain in cross section data for many elements, isotopes, reaction channels, and energy ranges. Such missing information often constitutes a critical obstacle to the development of new technologies such as ANRs. Filling these gaps, in general, requires lengthy and costly experimental campaigns. Such efforts are often limited by the availability of appropriate measurement facilities and resources. These are consequently modeled using different physics-based tools and models.

There are two types of nuclear data: measured and evaluated. The former consists of experimentally measured data while the latter are recommended set of values based on experimental and theoretical calculations. Not all measured and evaluated data is stored in a single resource. There are a variety of nuclear databases designed for different data types. Each of these "libraries" has different objectives and compilation processes, some more complex than others. Regional evaluated libraries, those containing a curated set of recommended values, are used in various applications including modeling and simulating fission and fusion reactors, shielding and radiation protection design, criticality safety, nuclear physics research, etc. For example, neutron-induced cross sections are needed to simulate various aspects of nuclear power reactors including the temperature and power distribution using either Monte Carlo or deterministic codes. To apply any sort of new approach to this area we need to understand how a nuclear data evaluation is carried including its advantages and disadvantages.

Nuclear Data Pipeline

Regional libraries like the Evaluated Nuclear Data File (ENDF) or the Joint Evaluated Fission and Fusion (JEFF) are end products of what is called a nuclear data evaluation [5] [17]. These evaluations contain recommended data for not only energy differential cross sections but also fission product yields, decay data, covariance for neutron cross section, product energy-angle distributions, etc. These are generated by evaluators using a variety of tools including physics-based model codes (i.e., TALYS and EMPIRE). These codes are guided and constrained by data from experimental databases like the Experimental Nuclear Reaction Data (EXFOR) database and the Experimental Unevaluated Nuclear Data List (XUNDL) [15] [**xundl**]. Bot the ENDF and JEFF library are examples of evaluated libraries while EXFOR and XUNDL fall within the unevaluated database category. Other evaluated libraries include the Japanese Evaluated Nuclear Data Library (JENDL), the Chinese Evaluated Nuclear Data Library (CENDL), and the Talys Evaluated Nuclear Data Library (TENDL) [19] [7] [11].

Derived studies in applications that utilize these libraries consequently rely on the judgments of evaluators (i.e. the person in charge of carrying the evaluation effort) and the available experimental data and benchmarks to provide the best set of recommended values. However, there are many instances where such experimental data is not available for a specific isotope-reaction channel pair or limited to a different energy range. In these significant gaps and areas of uncertainty, values are based purely on known physics and the evaluators' expertise. Without the constraints of experimental data points, these calculations are due to carry on higher human-bias and uncertainties. Furthermore, the current evaluation methodology often requires parameter optimization and fine-tuning to reach an adequate fit of the experimental data. The evaluation process consumes time and resources that are sometimes not readily available. The process of deriving this set of recommendations is called the nuclear data evaluation pipeline (Figure 1.1). Briefly, the steps of the pipeline consist of compilation, evaluation, processing, and validation. More detailed information can be found in Reference [2].

Compilation

The compilation phase is the first step in the evaluation pipeline and it begins with the Nuclear Science References (NSR) which stores sources to publications from journals, internal reports, and proceedings containing experimental results. These publications are ideally generated from carefully designed experimental campaigns. Once in the NSR, numerical results including mean values and uncertainties are extracted and compiled into one of two databases, XUNDL or EXFOR. The EXFOR library is the nuclear reaction compilation database. Not only are reaction cross sections included but also fission yields, resonance integrals, polarization data, and much more. The XUNDL database contains compiled experimental nuclear structure data. Once experimental values are incorporated into the appropriate library the evaluation phase can begin. Since this work deals with neutron-induced cross sections, the rest of the pipeline will be described specifically on this topic.

Evaluation

The evaluation process is carried out by an evaluator to derive a set of recommended values. Depending on the task, appropriate datasets are gathered and their quality is assessed. The selected data is used to guide physics-based model calculations which result in best estimates, dependent on data availability, of mean values including uncertainties and covariances. In other words, the evaluation proceeds through the application of a model designed to reproduce and explain the experimental data as closely as possible. These values can then be processed to form part of one or more regional libraries. Most of the data generated for libraries such as ENDF are mostly relevant to neutron-induced thermal (0.025eV) and fast (1 MeV) energy cross sections. Data at 14.1 MeV is also of interest to counter-proliferation.

Processing

Once the evaluation is finished, the next step is processing. It is in this step where data is processed into a format compatible with a particular application or user code. It is a nontrivial step that requires knowledge of the evaluation process, the physics used, and the formatting limitations presented by application codes (i.e. MCNP, SCALE, Serpent2). It is

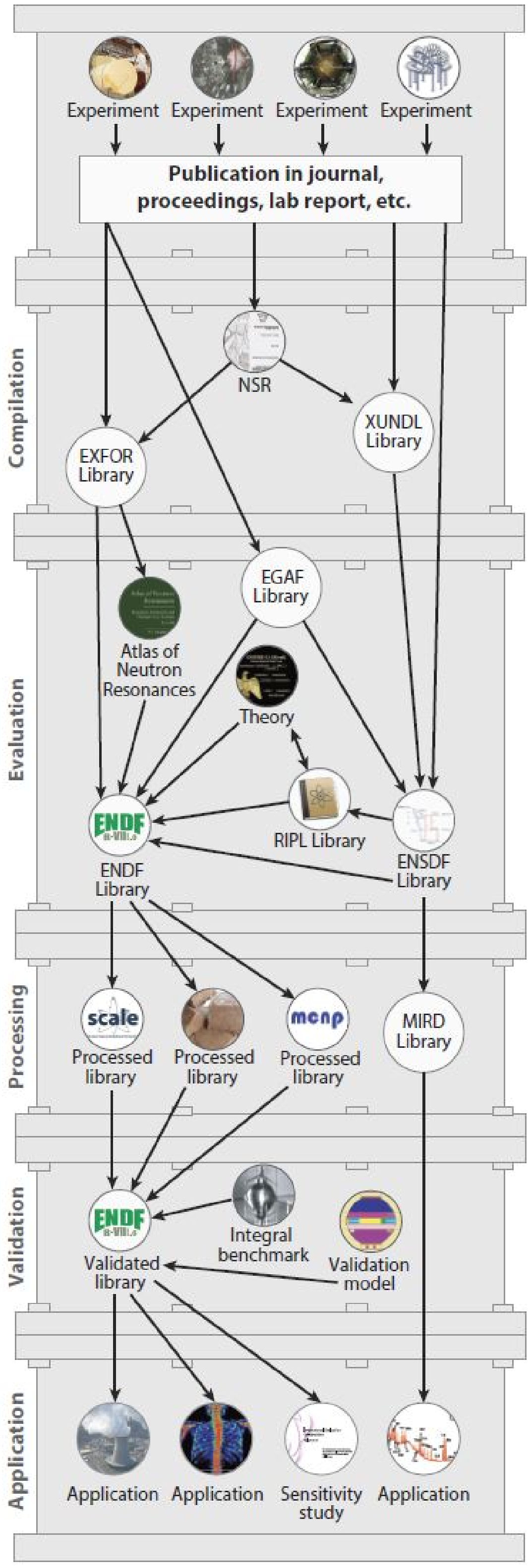


Figure 1.1: Nuclear data evaluation pipeline. Taken from Reference [2].

also a fundamental preceding action to the final step in the nuclear data evaluation pipeline, the validation phase.

Validation

Once the recommended set of values are in a suitable format, the validation step can begin. In this step, user codes and user defined problems are used to validate the generated data. These problems usually consist of integral benchmarks for which the measurements are known to a high degree of accuracy. Benchmark values are dependant on the interplay between different types of data (i.e. cross sections, emitted particle spectra, etc.) making it, in some cases, complicated in the validation of a single interaction type. A common example of benchmarks is critical assemblies for which the ratio of neutron production to neutron loss (k_{eff}) is precisely known. Reaction rate measurements and fission spectrum transmission experiments are other types of benchmarks that can be used to validate neutron interactions. Once a library has been validated, it can be released for use in application codes. The validation step is perhaps the most important phase in the evaluation pipeline.

1.2 Machine Learning

Machine Learning (ML), a subset of Artificial Intelligence (AI), consists of algorithms that can learn from data without explicitly programming them to do so. It is particularly resourceful in cases where it is either difficult or unfeasible to program a conventional algorithm to tackle a given task. The fields of ML/AI have improved drastically since its inception in the 1980s in part thanks to the fast increase in computational power over the last decade. This includes better Core Processing Units (CPU), faster storage Input/Output (I/O) speeds, and the development of better Graphical Processing Units (GPU). These hardware-based developments allow for bigger dataset handling capacities and faster ingestion and computations. High-speed processing times by CPUs and GPUs we're key for the takeoff of ML as a field, especially in the case of Deep Learning, a subset of ML based on artificial neural networks. These developments coupled with the Big Data era, more efficient ML algorithms, and better optimization methods, allows for more accurate models with faster iteration times. As a consequence, the prominence of ML/AI throughout business and technology sectors has only grown translating into a significant impact in our daily lives. Some of the most popular applications include autonomous machines, natural language processing powered speech-to-text and text-to-speech, optical character recognition, image recognition, AI-powered Chatbots, efficient fraud detection, etc.

Nowadays, ML/AI tools and codes are widely accessible in many programming languages. Some of these allow the fast prototyping and testing of ML techniques due to the incorporation of Automatic-ML (AutoML) capabilities. Moreover, powerful clusters are available for use through many cloud-based providers (i.e., Amazon Web Services, Google Cloud Platform, and Microsoft Azure) allowing industries and the scientific community to train very complex models without the need to invest and maintain new clusters. But even with the continuous growth of ML/AI throughout virtually every industry, its extension/application to Nuclear Physics and Engineering remains very limited. To the best of our knowledge, there has not been any work in applied ML to nuclear data evaluations and measurements beyond stating its potential application.

1.3 Motivation, purpose, and scope of the thesis

Data for nuclear reactors are needed for several calculations including reactor and fuel design which in turn are needed for shielding, decay heat, safeguards, and material damage calculations to name a few. While there are extensive documentation and data around light water reactors, big gaps exist in the energy region of interest for new ANRs. Furthermore, the lack of experimental data for new materials used in these concepts acts as a block for innovation and development. Not only is there a lack of experimental cross section data but also benchmark data. Without the latter, the validation phase becomes challenging and resulting evaluations will carry higher than usual uncertainties. Some of the current needs relevant to the advancement of ANRs identified by Berstein et al are:

- Improved ^{238}U(n,n') cross section due to the reaction being a limiting factor in fast spectrum reactors including those using metallic cores and chloride salts.
- Improved ^{35}Cl(n,p) cross section is needed for low uncertainty criticality calculations in molten chloride salt systems.
- Better graphite data for high-temperature reactors.
- Refined thermal neutron scattering law in molecular compounds such as molten salts needed for transport calculations since there are no cross section measurements that account for the molecular bond effects.
- Improved (n,x) cross sections of heavy actinides for high burnup fuel and molten salt reactor buildups.

Currently, modeling neutron-induced reaction cross sections is a nontrivial problem as there is no one-model fits-all approach and the physics are not fully understood. In traditional physics-based tools, several physical parameters need to be adjusted for the model to generate a good function that fits and explains the available experimental cross section data accurately. Unfortunately, in some cases, these parameters can give too much flexibility consequently overfitting the data at the expense of physical meaning.

The fission and capture cross sections are often manually tuned in the final stages of the validation process to ensure that unitarity is conserved and for the benchmark results to be as accurate as possible. The tuning process is a significant source of uncertainty and human bias since this is an evaluator-dependent process. A common product of the validation design for these types of isotopes is the little confidence in the independent fission and capture cross sections. These values are ultimately used for important calculations including reactor power distributions. Accurate data and especially higher confidences in individual cross sections are needed to, for example, ensure materials do not exceed safety limits.

Although there are many potential applications of ML in the field of nuclear data, this work focuses on its potential role in supporting neutron-induced cross section evaluation to support the current nuclear data needs. We believe that the nuclear data users can benefit from the application of ML/AI to better inform cross sections in areas of uncertainty and iterate faster through the evaluation steps, especially in cases where new data is incorporated and quick adjustments need to be made. The possibility of applying these tools comes in big part thanks to the abundance of experimental data collected throughout the years. The EXFOR database contains more than four million data points. Having collected this amount of data makes the nuclear data field a prime candidate for the application of ML methods. These algorithms may be able to leverage data from well explored (isotope, reaction-channel) pairs and learn patterns and behaviors that can be applied to other less measured isotopes including those envisioned to be used in advanced reactor concepts.

For this work, a ML-augmented evaluation pipeline (Figure 1.2) was designed to infer neutron-induced reaction cross sections. Similar to the traditional evaluation methodology, the proposed framework starts with the compilation of all available and applicable datasets. Once collected and processed, a variety of ML models were designed and trained. The performance was evaluated with unseen data and compared relative to the ENDF library. To validate the models, we generated data for several reaction channels including total, inelastic, fission, and elastic cross sections for various isotopes ranging from ^{233}U to ^{35}Cl. Isotopes like ^{233}U and ^{239}Pu were chosen to inspect the performance and demonstrate the reliability of trained ML algorithms in well-researched isotopes. When applicable, the generated cross sections were compiled into .ace files and tested on various benchmarks with SERPENT2 [12]. Once the models were validated, cross sections were predicted for less researched isotopes where data is scarce with the hope that these may be able to extrapolate knowledge from other similar reactions. This included (n,p) cross section data for ^{35}Cl due to the lack of data in the operating region of the Molten Chloride Fast Reactor.

The proposed framework requires specialized tools and software that are currently not available. These tools need to allow researchers to quickly navigate the pipeline seamlessly to allow for fast experimentation using modern tools. For this, we also created and released NucML, the first and only end-to-end python-based supervised machine learning pipeline for enhanced bias-free nuclear data generation and evaluation. NucML's capabilities were designed to support every step of the pipeline and to make it possible to modernize the current data pipelines for compatibility with state-of-the-art ML technologies. These frameworks and software are not meant to replace the evaluator or the physics guided tools but to enhance their analytical power and extract meaningful physics. In this thesis, I describe the proposed ML-augmented evaluation pipeline, the capabilities of NucML, and the application of the developed tools to several benchmarks and reaction cross sections of interest.

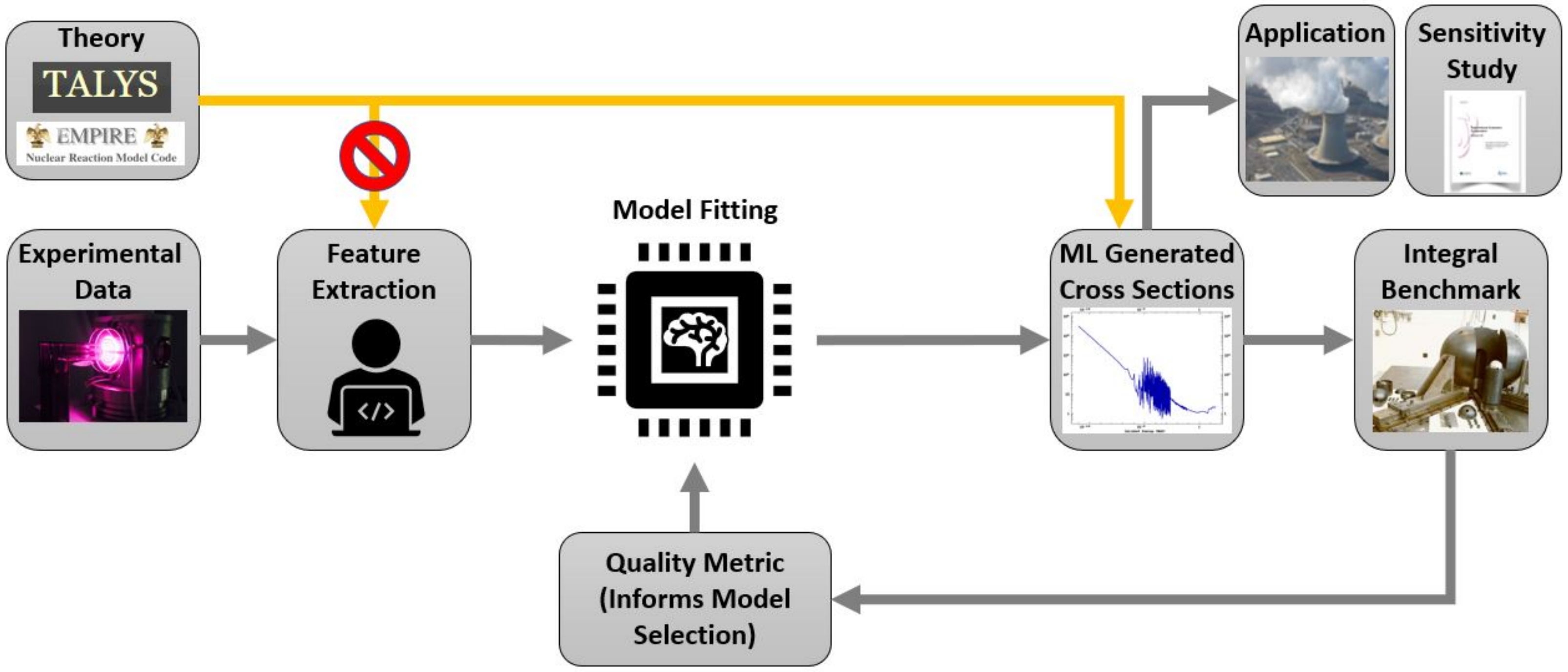


Figure 1.2: Proposed machine learning augmented nuclear data evaluation pipeline.

Chapter 2

Nucleus and Nuclear Reaction Background

In this work, a machine learning-augmented nuclear data evaluation for neutron-induced cross section is proposed with the hope of providing useful information in areas of uncertainty. Understanding the interactions between the neutron and a target nucleus is important so relevant features can be gathered and/or created. In this section the most basic factors influencing neutron-induced cross sections are reviewed. Other non-neutron induced reactions are out of the scope of this work and are therefore not included.

2.1 Nucleons

For a neutron-induced reaction to happen, a traveling neutron and a given nucleus must interact. While the cross section will be dependent on the energy of the incoming projectile, it is more so on the interacting nuclei. It is therefore important to know what characteristics define a nucleus and which features can have an effect on cross section behavior. A nucleus is made of bound protons (Z) and neutrons (N), and an orbiting electron cloud. Protons are positively charged particles that are balanced with negatively charged electrons and are bound to neutrons by the nuclear force, a short-range, charge independent, spin-dependent attractive force. Naturally occurring elements have a different abundance of stable and radioactive isotopes. For example, Uranium has an isotopic abundance of 99.274% ^{238}U, 0.72% ^{235}U, 0.005% ^{234}U and traces of ^{233}U and ^{236}U. Synthetic isotopes that do not occur naturally also exists (i.e. ^{232}U). Using techniques like mass spectrometry, precise atomic masses for these and other isotopes have been measured. Other than their mass, there are many more characteristics and properties that make isotopes unique and different from each other.

Nuclear Radius, Mass Excess, and Binding Energy

Based on the constant density model (Fermi model), the radii of any nucleus is directly related to its mass number by

$$R = r_o A^{1/3} \tag{2.1}$$

where r_o is a charge distribution dependent constant ranging from 1.1-1.5 femtometer (fm) and A is the mass number. The nuclear radius for many isotopes is in the order of 10^{-12} cm. Measuring the mass of the nucleus accurately is important since it is not equal to the sum of the masses of its components but smaller. The difference is called the mass defect (Eq. 2.2) which arises due to the conversion of mass into energy during the formation of the nucleus.

$$\Delta m = [Zm_p + Nm_n] - M(nucleus) \tag{2.2}$$

The mass excess is a measure of the strength of the nuclear force. In energy form, it represents the binding energy (BE) of the nucleus (Eq. 2.3). In other words, it is the energy required to break the nucleus into its constituents. Higher BE translates to increased stability.

$$BE = \Delta mc^2 \tag{2.3}$$

A popular depiction of the BE is the BE per nucleon (BE/A). The mean BE increases along with the number of nucleons until around $A = 50$ (Figure 2.1). Higher stable nuclei have mass numbers ranging from $A = 50$ and $A = 70$ where the BE/A is nearly constant. Above this mass number, coulomb repulsive forces between protons start to counteract the short-range attractive nuclear forces. This causes a decrease in the BE/A in heavy nuclei. Consequently, heavier nuclei with an overall lower BE can be split up into more stable nuclei with higher BE. During a fission process, energy is released due to this difference in BE. For a very low mass number, the surface tension effect causes a reduction in the BE since surface nucleons are less strongly bound than interior ones. The fraction of nucleons on the surface is proportional to $1/R$. Similar to fission, the left-hand-side (LHS) of the graph indicates the possibility of energy release by merging lighter nucleus to form heavier more stable nuclei. This process is called fusion. A related quantity to binding stability is the packing fraction (Eq. 2.4) where positive values indicate instability and can therefore undergo fission and fusion processes. Negative values indicate that there is a mass defect meaning stability (Figure 2.2).

$$P = \frac{M - A}{A} = \frac{\Delta}{A} \tag{2.4}$$

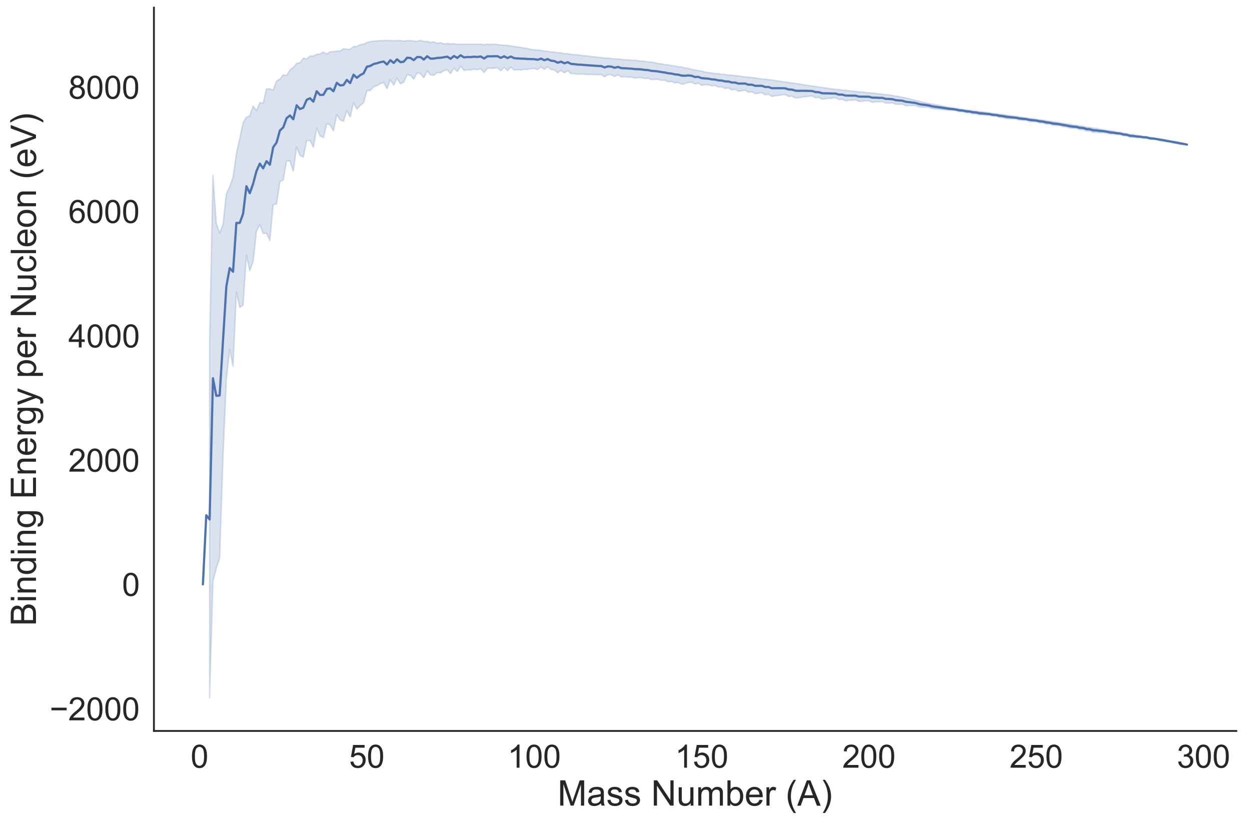


Figure 2.1: Mean binding energy per nucleon (BE/A) as a function of the mass number (A). Standard deviation shown in light blue.

Stability

Stability comes mostly due to certain combinations of protons and neutrons. In light nuclei, both the number of neutrons and protons tend to be equal due to Pauli's principle which states that no more than two protons or neutrons can occupy the given energy state (quantum level). Energy levels are filled up in sequence giving rise to a stable nucleus. Nuclei with an even number of protons and an even number of neutrons have higher stability (symmetry effect). In heavier nuclei, the trend starts to deviate from the $Z = N$ behavior mostly in the form of neutron excess (Figure 2.3). More neutrons are needed to balance the repulsive Coulomb forces between the increasing number of protons which weakens the overall bonds in the nuclei. Isotopes with an unstable nucleus are radioisotopes and experience radioactivity due to the tendency of converting nucleons to reach stability.

If there is an excess of protons, the nucleus will tend to transform some of these to neutrons by positron emission (β^+) (2.6) or by electron capture (EC) (2.7). In the case of neutron excess, the nuclei will tend to gain positive charge by β^- decay (2.7).

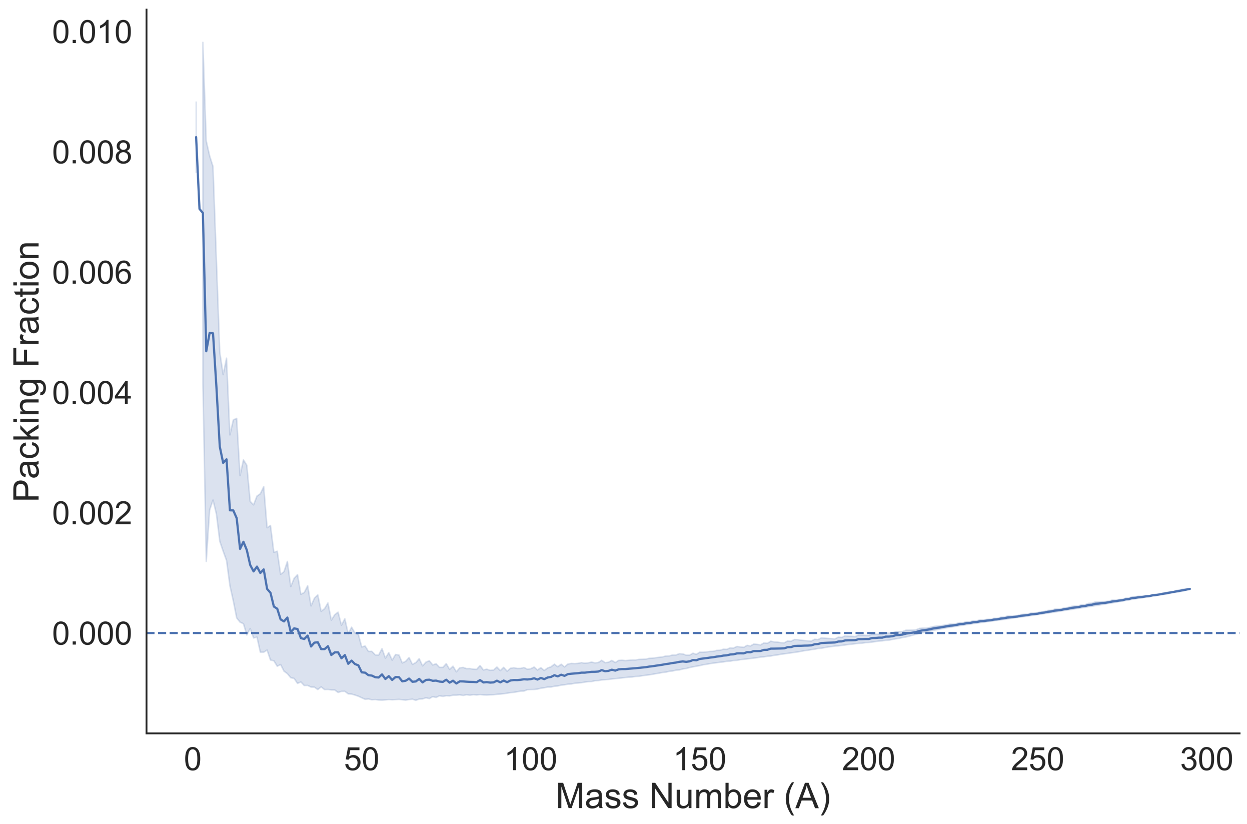


Figure 2.2: Packing fraction as a function of the mass number (A). Nuclei above the horizontal line at zero indicate instability and the possibility of energy release by fusion or fission to reach lower packing fraction values (higher binding energy nuclei).

$$p \rightarrow n + \beta^{+} + v_e \tag{2.5}$$
$$p + e^{-} \rightarrow n + v_e \tag{2.6}$$
$$n \rightarrow p + \beta^{-} + \bar{v}_e \tag{2.7}$$

The energy released or needed for a decay to occur for a given nuclei (Z,A) can be represented in terms of the Q-value for β^{+} decay (Eq. 2.9), β^{-} decay (Eq. 2.8), and EC decay (Eq. 2.10). There are other types of decay including neutron emission, alpha decay, and fission for which the energy released or required can be calculated.

$$Q_{\beta^{-}} = [M(Z,A) - M(Z+1,A)]c^2 \tag{2.8}$$
$$Q_{\beta^{+}} = [M(Z+1,A) - M(Z,A)]c^2 \tag{2.9}$$
$$Q_{\beta^{ec}} = [M(Z,A) - M(Z+1,A)]c^2 \tag{2.10}$$

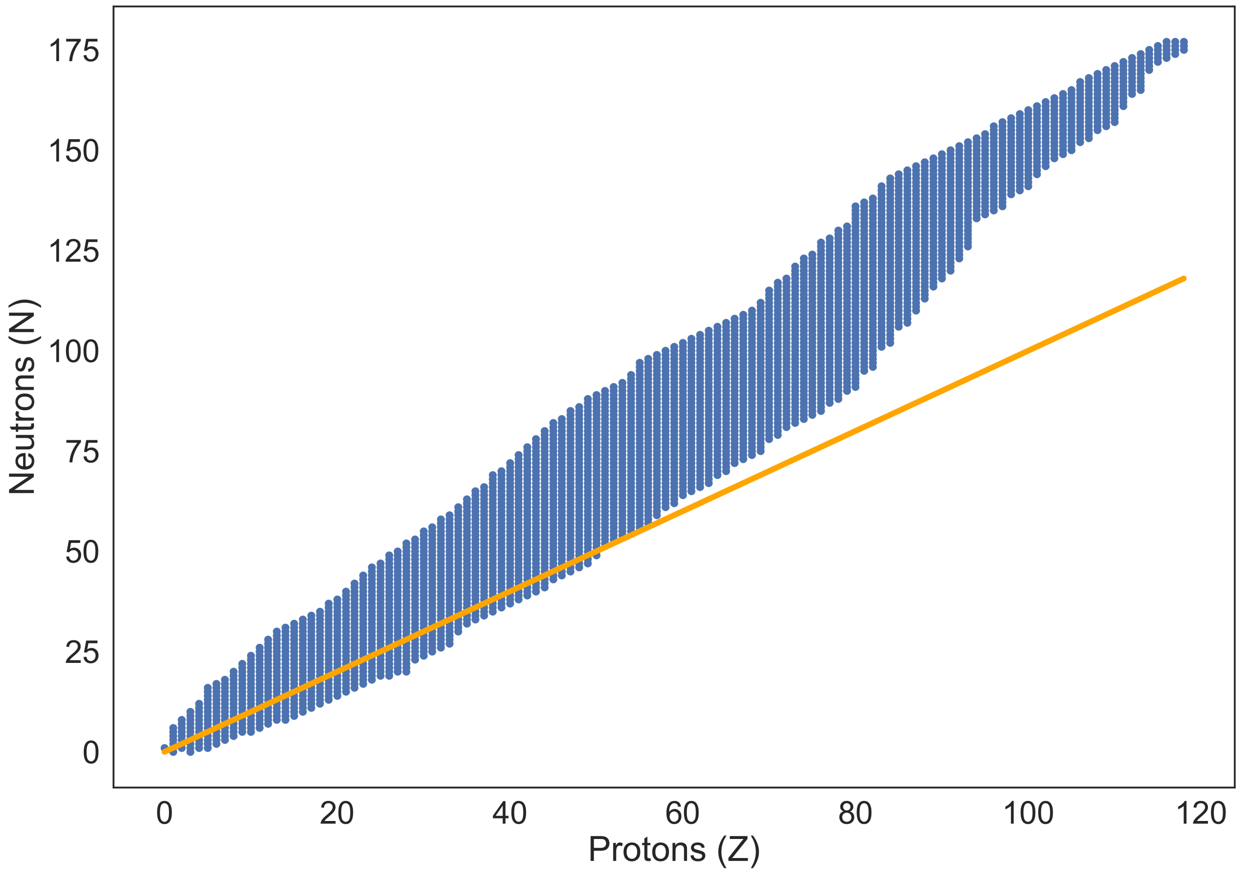


Figure 2.3: Number of Protons vs Number of Neutrons. The orange line represents the $Z = N$ behaviour. Deviations from the trend are mostly in terms of neutron excess.

For the nucleus to be stable against decay via neutron emission, the binding energy of the neutron must be positive (Eq. 2.12). Similarly, stability against proton and alpha decay translates into positive binding energy (Eq. 2.12).

$$B(n) = [M(A-1, Z) + M(n) - M(Z, A)]c^2 \tag{2.11}$$
$$B(\alpha) = [M(A-4, Z-2) + M(\alpha) - M(Z, A)]c^2 \tag{2.12}$$

Energy Levels

A nucleus can be on its ground state or an excited state. The former means all nucleons at their lowest energy levels. Above the ground state, several excited states start to appear at isotope-dependent energies. The distribution of these energy states including the distance between them and the magnitude of the energy levels are nuclei dependent. Heavier nuclei

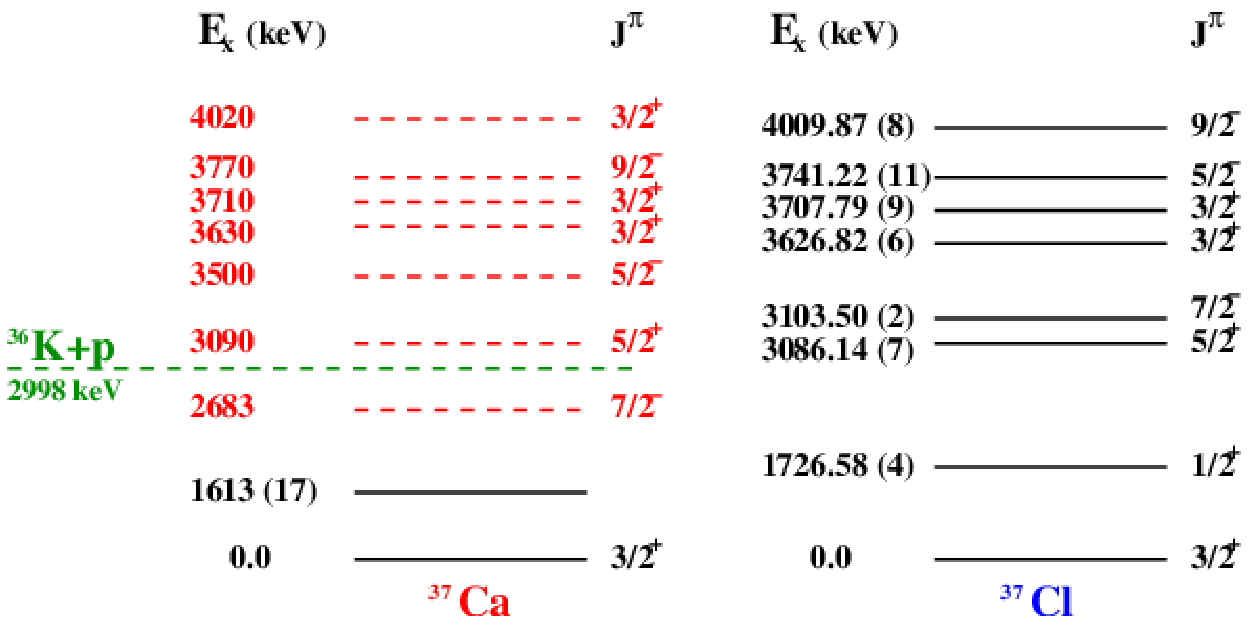


Figure 2.4: Excitation levels in mirror nuclei ^{37}Ca and ^{37}Cl. Image taken from Reference [9].

have more frequent energy levels than lighter nuclei but the energy of the first excited state is higher in the former. The magnitude of the energy levels is also affected by the presence of a magic number of nucleons. In mirror nuclei, where the number of protons in one nucleus equals the number of neutrons in another, the energy levels are found to be very similar (Figure 2.4).

In any case, excited states are not stable and will usually decay to a ground state through the emission of gamma rays or internal conversion. The energy levels correspond directly to the resonances shown in cross sections.

2.2 Nuclear Reactions and Reaction Mechanisms

A collision between a neutron and a target nucleus will produce changes in nuclear composition and/or in the energy state as permitted by the various conservation laws. Due to the neutral nature of the neutron, it does not experience repulsion and is, therefore, useful to penetrate the nucleus and induce reactions. There are different types of interactions that can take place as long as conservation laws are obeyed (nucleons, charge, linear and angular momentum, energy). The energy released or needed for any given reaction (Q) depends on the final (E_f) and initial (E_i) kinetic energy (conservation of energy).

$$Q = E_f - E_i \tag{2.13}$$

The reaction is said to be exothermic if Q is positive or endothermic if negative. In an endothermic reaction, energy must be provided for the process to take place. In these types of reactions the energy Q represents the threshold energy. In an exothermic reaction, mass is converted into released kinetic energy. Thus, the Q-value is useful to predict whether a certain reaction is energetically possible. There are various reaction mechanisms including compound, direct, and pre-equilibrium reactions. The most common mechanism is through the formation of a compound nucleus.

Compound Nucleus Reactions

In compound nucleus reactions, the incident neutron is absorbed by the target nucleus ${}^{A}_{Z}\mathrm{X}$ forming a compound nucleus ${}^{A+1}_{Z}\mathrm{X}$. The kinetic and binding energy of the neutron is distributed among the nucleons until equilibrium is reached. During this transition, nucleons near the surface may escape if the excitation energy meets at least the particle's separation energy. This can continue while the energy is above the threshold requirements. Once below the threshold energy, gamma emission can take place until the ground state is reached. The compound system can decay in several ways including fission, where the nuclei split into two or three large fragments. This is a common decay mechanism in a heavy nucleus. Another possibility is radioactive decay into a more stable nucleus (beta, alpha, etc.). Due to the time involved in the formation of the compound nucleus and the long decay times, the decay mechanisms are thought to be independent of the mode of formation of the compound system (the memory of formation is lost). Of the possible decay modes, the exit channels with neutron emission are more favored since the emission of charged particles involves overcoming the coulomb barrier.

Several aspects need to be understood in compound systems including the properties of energy levels that could be excited, the density of energy states as a function of energy, the mechanisms of de-excitation, etc. Not all states in a nucleus are excited with equal probability and are therefore populated differently depending on the reaction and the energy. This phenomenon allows an experimentalist to get information on the reaction mechanism and the nuclear structure.

Resonances

The probability of interactions between a neutron and a target nuclei is dependent on many variables including the material involve and the energy of the incident neutron. Generally, this probability decreases as the energy increases. However, at low energies spikes may suddenly appear. These resonances are due to quantum mechanics where the neutron wave resonates with the target cavity at certain frequencies. At these points, a large interaction probability exists and therefore a resonance appears. The energy region where cross sections have increased probability of interaction due to these sudden variations is called the resonance region (Figure 2.5).

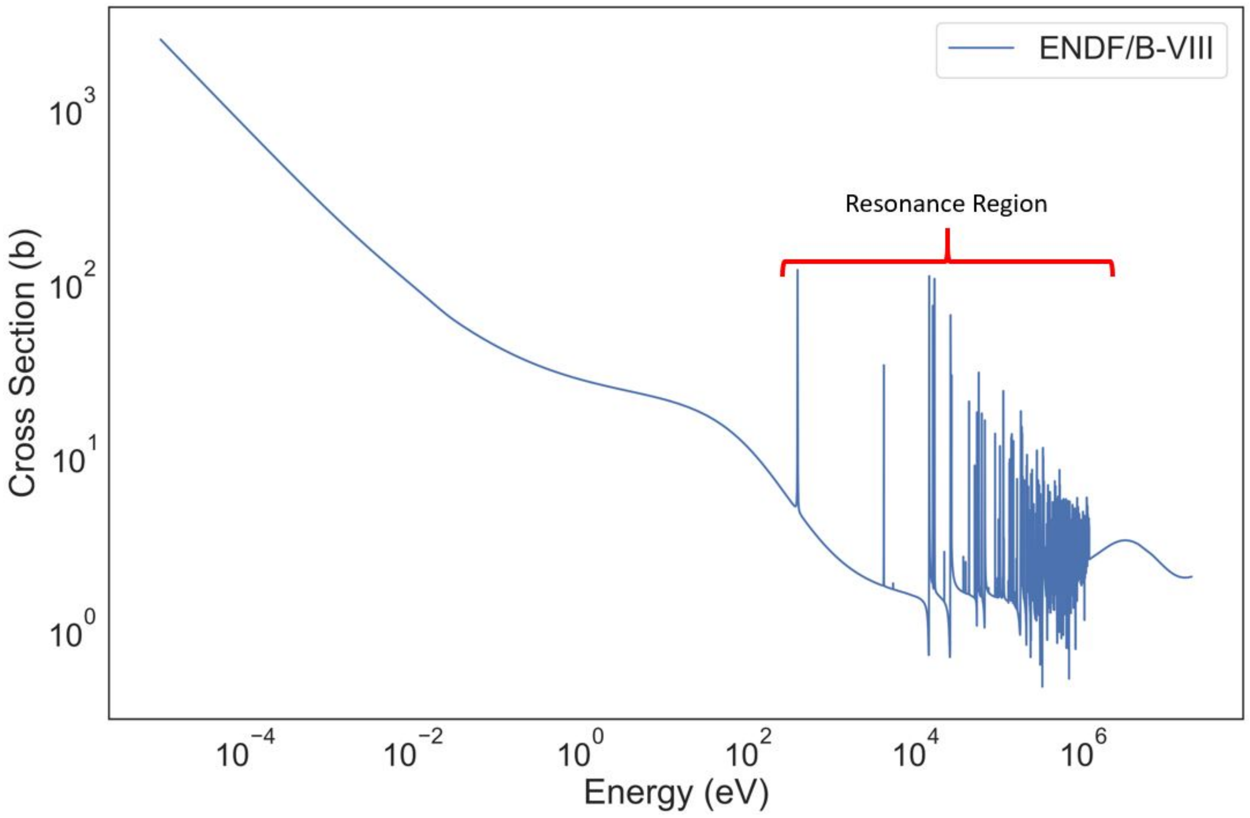
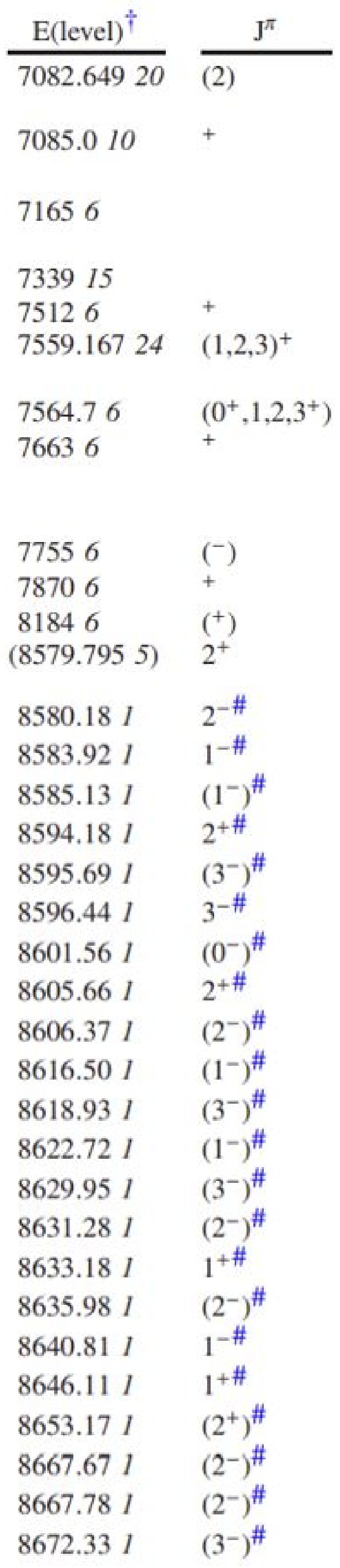


Figure 2.5: ^{35}Cl(n,total) cross section plot.

The energies at which these peaks are located correspond directly to an excited level in the compound nucleus $^{A+1}_{Z}$X. In other words, if the kinetic energy of the incoming neutron (E_n) and the neutron binding energy ($B_n(A+1,Z)$) sum up to a value that corresponds to an excitation level, a resonance will occur with a high probability. Figure 2.5 shows the total cross section plot for ^{35}Cl. The first resonance is located approximately at $4 * 10^{-4}$ MeV. Since the neutron separation energy is 8579.79 keV we should expect to observe the first resonance at approximately 8580.19 keV (the kinetic energy plus the neutron binding energy). Figure 2.6 shows some of the energy levels for ^{36}Cl, the compound nucleus formed upon neutron absorption in the ^{35}Cl nucleus. Indeed, there is an excited state with energy of 8580.18 keV responsible for the first resonance in ^{35}Cl.

With the formation of the compound nucleus, there are several possible subsequent events. One possibility is the de-excitation by the emission of a gamma-ray (photon) of lower energy. In this scenario, the compound nucleus may still be in an excited state and, if the energy allows it, a cascade of gamma-ray emissions can continue. Another possible outcome is the ejection of the same projectile particle by elastic or inelastic scattering.

In any nuclei, the level frequency increases as the energy increases due to the various pos-



E(level)[†]	J^{π}
7082.649 *20*	(2)
7085.0 *10*	$+$
7165 *6*	
7339 *15*	
7512 *6*	$+$
7559.167 *24*	$(1,2,3)^{+}$
7564.7 *6*	$(0^{+},1,2,3^{+})$
7663 *6*	$+$
7755 *6*	$(^{-})$
7870 *6*	$+$
8184 *6*	$(^{+})$
(8579.795 *5*)	2^{+}
8580.18 *1*	2^{-}[#]
8583.92 *1*	1^{-}[#]
8585.13 *1*	(1^{-})[#]
8594.18 *1*	2^{+}[#]
8595.69 *1*	(3^{-})[#]
8596.44 *1*	3^{-}[#]
8601.56 *1*	(0^{-})[#]
8605.66 *1*	2^{+}[#]
8606.37 *1*	(2^{-})[#]
8616.50 *1*	(1^{-})[#]
8618.93 *1*	(3^{-})[#]
8622.72 *1*	(1^{-})[#]
8629.95 *1*	(3^{-})[#]
8631.28 *1*	(2^{-})[#]
8633.18 *1*	1^{+}[#]
8635.98 *1*	(2^{-})[#]
8640.81 *1*	1^{-}[#]
8646.11 *1*	1^{+}[#]
8653.17 *1*	(2^{+})[#]
8667.67 *1*	(2^{-})[#]
8667.78 *1*	(2^{-})[#]
8672.33 *1*	(3^{-})[#]

Figure 2.6: Excitation levels for ^{36}Cl.

sible configurations of the nucleons. This translates into an increase in resonance frequency as a function of energy. At high enough energies, the excited levels will tend to have shorter lifetimes and vice versa. This is due to Heisenberg's uncertainty principle,

$$\Delta E \Delta t \geq \hbar \tag{2.14}$$

where ΔE represents the energy and Δt the time. Similarly, the uncertainty principle may be represented in terms of the level's width Γ and its lifetime τ (Eq. 2.15).

$$\Gamma \tau \geq \hbar \tag{2.15}$$

This latter relationship means the resonance's width increases as the lifetime decreases and therefore as energy increases. There comes a point where the width of the resonances start overlapping each other causing the peaks to soften. The lifetime of high energy resonances is small enough and the uncertainty in energies high that our experimental resolution is unable to capture and therefore measure these values. This leads to the unresolved resonance region where the cross section values are averaged and presented as a smooth line.

As previously mentioned, there are various possible decay modes for a compound system. For every possible mode (j), a partial width Γ_j exists. The total width of the level is the sum of all partial widths (Eq. 2.16).

$$\Gamma = \Gamma_1 + \Gamma_2 + \Gamma_3 + ... \tag{2.16}$$

What the total width characterizes is the total probability of decay which corresponds to the full width of the resonance peak measured at one half the maximum height.

It is evident that the location of the resonances is a direct function of the available excitation levels in the compound nucleus. The density of these nuclear levels is isotope dependent. The general trend is that heavier nuclei start exhibiting resonances at lower energies than lighter nuclei. Additionally, the energy of the excited states decreases with increasing mass number. Energy states for heavy nuclei start to appear as soon as 0.1 MeV. For lighter nuclei, this number is higher and the levels are widely separated by about 1 MeV. In a heavy nucleus, these are approximately 50 keV apart. Magic number nuclei are an exception to these rules. There are several models that attempt to model the level density. One of these is the Fermi's gas model

$$\rho(E) = \rho(0)e^{2(aE)1/2}. \tag{2.17}$$

where $\rho(0)$ and a are empirical constants. Each energy level is characterized by a definite spin, parity, and width. There are other models but none that explains perfectly the experimental measurements. It is not the scope of this thesis to understand other models but to create our own using ML.

Direct Reactions

In direct reactions, emission takes place relatively immediately without the formation of the compound nucleus. The behavior and characteristics of direct reactions differ significantly from compound reactions since the former is a one-step process while the latter is a two-step process. While still possible at low energies, direct reactions are most commonly encountered at high energies while compound nucleus based reactions tend to happen at low energies. Direct reactions include stripping and its inverse, pick-up, knock-out, and heavy-ion reactions.

2.3 Reaction Types

There are many possible reaction types including most popularly elastic and inelastic scattering, transmutation, capture, fission, and charged particle emission channels like (n,α) and (n,p). In this section, we describe the most common reaction types.

Elastic Scattering

In elastic scattering, the kinetic energy of the neutron is redistributed but conserved ($Q = 0$) and internal states are unchanged. An example of elastic collisions is the reaction

$$n + {}^{235}\text{U} \rightarrow n + {}^{235}\text{U}$$

where a neutron hits a ^{235}U nucleus resulting in a loss of neutron kinetic energy and a change in direction. The lost KE is transferred to the target nucleus. Since the neutron lacks charge, it can start scattering elastically at very low energies. There are two possible scattering mechanisms: compound or potential elastic scattering. In the former, the neutron first forms a compound nucleus followed by the re-emission of the neutron. In the latter, the neutron is never absorbed. Instead, it is deflected away from the nucleus surface by the short-range nuclear force. The most common form of scattering is potential scattering although at higher kinetic energies resonance behavior starts to take place via compound scattering. Potential scattering is analogous to billiard balls collisions and takes place at energies up to around 1 MeV. The cross section for potential scattering is almost constant and a function of the nucleus's radius (Eq. 2.18).

$$\sigma_{el} = 4\pi R^2. \tag{2.18}$$

The amount of energy that the neutron loses depends on the target nucleus. A larger target nucleus means the neutron loses a negligible amount of energy while lighter nuclei are more effective in reducing the energy of the neutron. The maximum transfer of energy happens on a head-on collision in which the neutron does not change its initial direction.

Inelastic Scattering

In inelastic scattering, the neutron becomes part of the target nucleus, therefore, being a compound nucleus reaction. For inelastic scattering to occur the energy provided must be close to one of the excited states of the target nucleus. This means kinetic energy is transformed to allow for excitation of the nuclei. The energy release for these type of reactions is negative (endothermic) since energy E_x is required to excite the nuclei ($Q = -E_x$). An example of inelastic collision is the reaction

$$n + {}^{238}\mathrm{U} \rightarrow n^{'} + {}^{238}\mathrm{U}^{*}$$

where the outgoing ^{238}U nuclei is left in an excited state and the neutron loses energy. Inelastic scattering is a threshold reaction and it is more pronounced in heavy nuclei where there are higher levels at lower energies. For lighter nuclei, the inelastic scattering cross section at low energies is virtually zero. When the threshold energy is met, inelastic scattering cross sections are nearly equal to the elastic cross section. For the reaction above the threshold, energy is around 44 keV. For light nuclei like ^{16}O the threshold energy is 6 MeV with magic nuclei behaving similarly.

Transmutation

In a transmutation type of reaction, there is a rearrangement of the nucleons between the incoming neutron and the target. There are many possibilities for the outcome of these reactions as long as the conservation of nucleons law is obeyed. An example is the reaction

$$n + {}^{27}\mathrm{Al} \rightarrow {}^{24}\mathrm{Na} + {}^{4}\mathrm{He}$$

where the number of neutrons and protons are rearranged on the left-hand-side (LHS) and the right-hand-side (RHS) of the reaction.

Capture Reactions

In this type of reaction, the excited compound system is created by the interaction of the incoming neutron. Next, the target nucleus decays into a ground state via one or more gamma rays (cascade emission). An example is the reaction

$$n + {}^{238}\mathrm{U} \rightarrow {}^{239}\mathrm{U}^{*} \rightarrow {}^{239}\mathrm{U} + \gamma + Q$$

whereupon capture of the neutron by ^{238}U, the ^{239}U excited system is created followed by the release of excitation energy in the forms of gamma rays. Figure 2.7 shows the excited levels and their direct relationship with the resonances presented in the ^{238}U cross section.

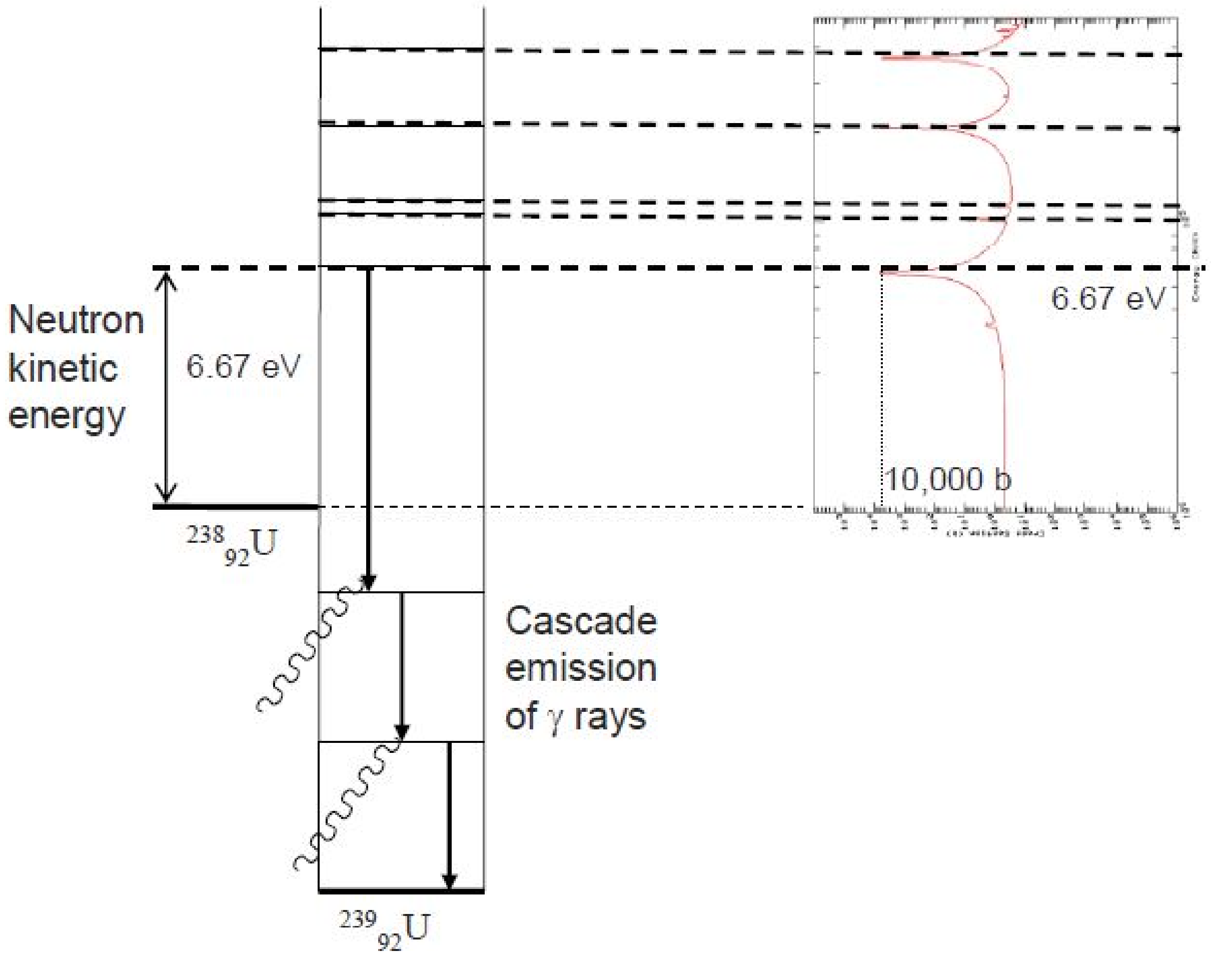


Figure 2.7: $^{238}U(n,\gamma)^{239}U$ reaction cross section. The first excited levels of ^{239}U are shown connected to the respective resonance peak. Image taken from Reference [9].

This type of reaction is often called radiative capture due to the release of gamma rays and can occur at any neutron energies. The capture cross section also follows a 1/v behavior at low energies for most nuclei followed by a resonance region in the same energy range as elastic scattering since the compound nucleus formed is the same in both reaction types. Above the resonance region, the capture cross section drops since other interactions start to open up energetically.

Fission

When a compound nucleus is formed by the absorption of a neutron in a heavy nucleus like ^{235}U, it may split into two or sometimes three fragments with higher probability than in lighter nuclei. At the right energies, this mode becomes competitive with high width reaction channels like capture. This splitting is usually accompanied by a couple of neutrons which allows for chain reactions to exist. An example is the reaction

$$n + {}^{235}\mathrm{U} \rightarrow {}^{236}\mathrm{U}^{*} \rightarrow {}^{139}_{56}\mathrm{Ba} + {}^{94}_{36}\mathrm{Kr} + 3n$$

whereupon capture of the neutron by ^{235}U, the excited ^{236}U nucleus splits resulting in two fission products and three neutrons. Other possible fission products are possible and the probabilities are usually represented as fission yields. The produced fission products can keep decaying depending on their lifetime.

2.4 Cross Section

cross sections are used to quantify the probability of interaction between an incoming projectile and a target. It is a unit of measurement given in length squared. The most common unit is the barn ($1b = 10^{-24}cm^2$). In pure geometric terms, it describes the ratio of the target area to the total area however, due to quantum effects this is not purely dependent on hard properties like the nuclear radius although for some reactions it will come close to the actual area. The cross section captures many properties of the interacting nuclei including the nature and forces between them. For each reaction type, there will be a cross section value at given energy for a given isotope. Non-threshold reactions have large values at low neutron energy due to fewer competing modes while threshold reactions start to play in only above certain energies. One example is ^{238}U for which fission is only possible above 1 MeV.

During the neutron lifetime, it will have a certain total probability of collision (σ_t). Upon interaction, it will either be scattered (σ_s) or absorbed (σ_a):

$$\sigma_{total} = \sigma_s + \sigma_a \tag{2.19}$$

Furthermore, scattering can be either elastic or inelastic.

$$\sigma_s = \sigma_{el} + \sigma_{in} \tag{2.20}$$

Absorption cross sections consist of reactions that terminate the neutron history including capture, charge particle emission, and fission.

$$\sigma_a = \sigma_\gamma + \sigma_f + \sigma_p + \sigma_\alpha + ... \tag{2.21}$$

The resonances of all these happen over the same resonance region while the peak values for each mode may vary (each mode contributes to the total width of the resonance). At low energies, the total cross section for non-threshold interactions behaves as

$$\sigma_{tot} = 4\pi R^2 + \frac{C}{\sqrt{E}} \tag{2.22}$$

where C is a constant. The second term accounts for the radiative capture. Once the resonance region begins there comes an energy point where the resonances are too close together (high width) and experimental values cannot be obtained. This region is called the unresolved resonance region where the smooth values are averages. Recommended cross sections in this area heavily rely on theory and modeling codes.

2.5 Current Experimental and Modeling Needs

Most theoretical models by themselves, especially of low energy neutron-induce reactions, are not ideal for predicting cross sections at the level of accuracy needed for real-world applications. To provide useful information, these models are guided by experimental measurements in the evaluation process to derive a set of recommended values. However, there are still gaps in data or high uncertainties due to the lack of measured data points for specific isotopes and reactions. Even with the extensive experimental campaigns and theoretical developments, the need for improved data and models remains. Berstein et al. reported some of the most important needs in the nuclear data field [2]. Some of the suggestions include:

- Fission reaction mechanisms understanding for better (n,f) evaluations:
  - Measurements are not always easily obtained. Some targets are short-lived leading to high uncertainties and observables are not always easily measured.
  - There is a lack of a theoretical framework and models for fission-related data evaluations. Pre- and post-scission are modeled separately but lack the level of accuracy needed for predicting nuclear quantities.
- Elastic and inelastic scattering improvements need to be made due to their impact on neutron populations and their properties in nuclear applications.
  - Scattering data does not produce easily measured signals leading to a deficiency of high-quality experimental data and disagreement between evaluations.
  - Errors in scattering cross sections are difficult to manage. Compensating errors are resistant to the validation process.
    * Compensating errors occur due to the nature of the validation step. Evaluations using criticality benchmarks risk producing various combination of values that produces the same result of $k_{eff} = 1$. The number of combinations varies causing disagreements between organizations and causes uncertainty in the recommended cross sections.

- Improved (n,p) reaction cross sections are needed due to their potential for the production of therapeutic radionuclides. This is difficult due to the high-energies used leading to orders of magnitude differences between modeling codes. While many reactions of interest take place on stable nuclide, difficulties still arise in experimental settings due to decreasing monitor reactions at high energies.

These are some of the needs that a hybrid ML-based solution could tackle given the correct information.

2.6 Summary

This overview gives a general idea of the factors that influence cross section behavior. It is primarily a function of the projectile, in this case, the neutron and a target particle. The target can be an isotope or a non-pure naturally occurring element. In any case, this means the cross section is at least a function of the number of protons and the number of neutrons. Each nucleus has a set of defined and discrete energy levels that can be excited upon interacting with an energy-carrying projectile. Excited states cause resonances in cross sections which are important since these are the sources of high interaction probability in reactions. While some excited levels have been measured experimentally, there are still uncertainties at higher energy states where our experimental capabilities are unable to measure due to the extremely low lifetimes. Additionally, collectivity in heavy nuclei is difficult to model. The reviewed concepts should help create a usable data set that can help machine learning models model cross section data.

Chapter 3

Methodology - NucML

As previously mentioned, there are several stages in the traditional Nuclear Data Evaluation (NDE) pipeline including compilation, evaluation, processing, and validation [2]. In this work, we proposed a modified Machine Learning augmented Nuclear Data Evaluation (ML-NDE) pipeline for neutron-induced cross sections to help address some of the previously mentioned issues for both the traditional NDE and the nuclear data field including

- Lack of relevant data for important isotopes like ^{35}Cl.
- High uncertainty in data-lacking reaction evaluations
- High consumption of time and resources
- Bias in the generation of solutions
- Benchmark overfitting and compensating error in fissile isotopes with criticality evaluations

In this chapter, we describe in detail the ML-NDE, its advantages, and its disadvantages. We also introduce NucML a python toolbox for accelerated ML-based evaluation created to navigate the ML-NDE seamlessly.

3.1 Machine Learning-augmented Nuclear Data Evaluation

As mentioned previously, it is important to have accurate cross section data and uncertainties to build safe and reliable systems. While extensive experimental reaction data exists for some of the most popular isotopes including ^{233}U and ^{239}Pu, there are still many for which uncertainties are high. These include isotopes like ^{35}Cl, an important material in chlorine-based molten salt reactors. In these technologies where datasets for relevant materials are scarce, high uncertainties impact the operational and safety analysis greatly. Evaluated data relies on the judgments of evaluators to provide the best set of values given known physics and expertise. Missing data causes traditional modeling codes to be unconstrained and results in an increase in human-bias in the generation of solutions. This is especially true in modeling methods that require parameter optimization and fine-tuning to reach an adequate fit of the few experimental data points available. The uncertainty is significantly worse when there are virtually no applicable benchmarks to validate the model with. Furthermore, this process can consume considerable resources that are sometimes not readily available.

Figure 1.2 presents a potential evaluation pipeline augmented by ML. Its steps can be summarized as dataset creation, feature extraction and processing, ML model training and evaluation, hybrid library generation, and validation which in turn informs model selection. Similar to the nuclear data pipeline, it ends on the application side. The main major difference is the use of ML models in the evaluation phase rather than physic-guided modeling codes. What makes an ML-NDE pipeline possible is the extensive set of experimental campaigns and data points collected throughout the years. The major steps can be summarized as follows:

1. The first step consists of collecting the relevant experimental data. For ML to effectively tackle this challenge, a representative dataset containing the physical properties and features that are believed to affect cross section behavior is needed. This mainly includes data from the EXFOR database, the Atomic Mass Evaluation, the Evaluated Nuclear Structure Data File, and any other appropriate experimental data source.

2. After data collection, feature extraction and processing are performed. This step consists mostly of cleaning the data and transform it into a form suitable for ML algorithms. This process is model-dependent and needs to be carefully performed. Some models require stronger normalization and transformation techniques than others. Dataset sparsity can also be detrimental to training optimization for certain types of models.

3. After having all features in an ML-friendly format, the chosen model is trained and its performance is first evaluated using an unseen dataset subset. The trained model can then be used to generate ML-derived recommended values (libraries) based purely on patterns and behaviors it learned from the training dataset. These models are trained using data from the Experimental Nuclear Reaction Data (EXFOR) database, the XUNDL library, and the Atomic Mass Evaluation (AME) [15] [20].

4. These cross sections are then validated using benchmark calculations. In this particular challenge, it is not possible to just rely on the validation subset performance for model selection. The error from the benchmark must have a higher weight on model selection since these are our closest representations to real-world deployment scenarios. However, the benchmark loss is currently not directly taken into account in model training and since it provides essential metrics for further improvement, it must participate in model selection in another way. In this work, the average performance on a set of benchmarks

is used as the selection technique. This process is iterative until an optimal set of model parameters is found.

5. The selected models can then be used to make predictions to inform evaluators in areas of uncertainties. An implicit benefit of ML-based solutions applied to the traditional NDE is the reduction of human bias.

Limitations and Considerations of the ML-NDE

Several aspects require caution when implementing ML pipelines to this area in particular. Theoretically derived data including physical parameters or factors that scientists think can help the model learn behaviors in cross sections must not be included in the dataset. Including them will cause the ML model to have the same constraints as those other models used to derive said theoretical data. For example, filling missing values with ENDF data means the model will not only learn characteristics from the real physical phenomena but also from those models used to create the evaluated library (i.e., TALYS, EMPIRE). Also, it inherits the bias in the creation of these values. Next, splitting must be performed in a stratified manner rather than randomly. It is important to fit the model on a representative diverse dataset, otherwise, it will have poor performance on data types that were not included in the training set. A clear example is elastic scattering of which there are thirty-seven thousand datapoints. Approximately two-thirds of those belong exclusively to ^{237}Np. Random splitting risks the majority of the elastic scattering points in the training subset belonging to ^{237}Np. This can translate into poor predicting capabilities on other non-seen isotopes. Another issue arises in the validation phase. Similar to the current nuclear data pipeline, integral benchmarks are part of the validation step to inform model selection. This means there is a risk of overfitting to a particular benchmark. The model selection process must be based on the average performance on all benchmarks. On the other hand, cleaning the data is a non-trivial process. This step is a hybrid realm between the data sciences and the nuclear data field. Although there might be good causes to discard one dataset versus another, this is outside the scope of the work presented here.

ML applied to cross section and nuclear evaluation presents very unique characteristics that are not usually present in common ML challenges due to the physical nature and meaning of the EXFOR database. EXFOR datapoints consist of experimental measurements of a variety of isotopes undertaken by researchers at different facilities. These carry uncertainty that is a function of the experimental settings, procedures, and more. Measurements are therefore only an approximation of an unknown true cross section value. The evaluation pipeline recognizes this and therefore incorporates the validation phase by the use of benchmarks. Benchmarks provide a very important way of ultimately validating derived cross section data. For example, a critical assembly has a multiplication factor of 1 (critical). This is an actual true value. In other words, the assembly is critical or it is not (there is no in-between) and there exists a set of true cross section values that allows this assembly to exists in a criticality state given the composition and the geometry. These hidden sets of cross section values would ideally be the labels in our datasets, however, we only have measurements that we regard as being close enough to these unknown true values. Competing experimental campaigns and outliers exist in EXFOR and this is why traditionally the evaluator inspects each dataset for quality. To limit bias, the proposed pipeline does not discard data points manually and therefore an odd issue arises.

Most machine learning models try to minimize a given loss function which itself is a function of the predicted value and the given label. Knowing that our labels are not ideal or accurate enough and that the benchmarks are currently not part of the training loss function directly, it may not be in our interest to completely minimize the loss with respect to the EXFOR dataset if this means increasing the loss with respect to the integral benchmarks. Given any model selection process (i.e., cross-validation, train-val-test), an optimal model with the lowest mean absolute error (MAE) and no overfitting will in some cases perform worse in a benchmark than an over-fitted model. This is due to the "approximation" nature of the dataset, the high uncertainties in some measurements, and the frequent presence of outliers. In other words, the EXFOR database converged model might not be the best model. Given these arguments, one must understand the limitations and carefully go about the training process on these types of experimental datasets.

3.2 NucML: Python-toolbox for Accelerated ML-based Evaluations

The application of ML to nuclear data requires specialized tools and software that are currently not available. These tools need to allow researchers to quickly navigate the pipeline seamlessly to allow for fast experimentation using modern tools. For this, we created and released NucML, the first and only end-to-end python-based supervised machine learning pipeline for enhanced bias-free nuclear data generation and evaluation. These tools are not meant to replace the evaluator or the physics-guided tools but to enhance their analytical power, extract meaningful physics, and modernize the current data pipelines for compatibility with state-of-the-art ML technologies. In this chapter, we describe the NucML capabilities for each step of the ML-NDE and the underlying philosophy behind the designed workflow. For information on the implementation, source code, and documentation please visit https://pedrojrv.github.io/nucml/ and https://github.com/pedrojrv/nucml.

(1) Experimentation and Compilation

Nuclear data can be found in a variety of formats and in different data sources. While the modernization of various of these databases is underway, there is a need now for ready-to-use datasets for ML applications. Reaction data from EXFOR is available to download in either various $.x4$ files or a single $.xc4$ file, both of which are in formats not compatible with today's ML technologies. The structure and formats of these files were designed for compatibility with other types of software including codes like EMPIRE which converts EXFOR native files into individual isotopic $.C4$ files for later use. Another database of interest is the Experimental Unevaluated Nuclear Data List (XUNDL). The format in XUNDL files is also incompatible with current ML workflows. The Reference Input Parameter Library (RIPL) offers more easy-to-parse *.dat* files but still offers challenges [4]. These and many other nuclear data sources require modernization.

Identifying, parsing, and formatting all nuclear data sources can be a tedious time-consuming job. NucML contains a variety of utilities that make it easy to download the latest versions of the EXFOR, RIPL, ENDF, and AME libraries easily. To convert these into ML-friendly datasets, parsing utilities are available to read library-native formats, restructure the information, and store the resulting data structure into single easy-to-use files in various formats including CSV, JSON, hdf5, and even parquets and Google BigQuery tables.

NucML Datasets

Transforming these nuclear data sources is just one step towards the integration into ML pipelines. NucML uses the resulting restructured data to make core functionalities available. Datasets like EXFOR, even if converted to ML-friendly formats, cannot be considered $ML-ready$. Feature engineering is an important stage of any ML workflow. In this phase, features are filtered to only include relevant information, new features are engineered, data is transformed according to model-dependent requirements (i.e. one-hot encoding for categorical features), standardization and normalization take place, and much more. Engineering new features is just as important as any other step in the ML workflow and is known to be a quick and easy way to boost model performance.

The AME database contains plenty of information including the precise atomic mass, binding energy, mass excess, beta decay energies, and Q-values for many types of reactions. These can complement EXFOR and the XUNDL/RIPL library. NucML Datasets makes incorporating AME features easy by automatically appending when loading every dataset. In cases where experimental campaigns used a natural target, AME data is interpolated using different methods to create usable information for these data points. Other core capabilities of all NucML Dataset loader functions include easy train-validation-test splits, recommended filters, feature filtering, normalization and standardization (i.e. power transformers, standard scalers, robust scalers), categorical data encoding, and more. Additionally, more features can be added by passing user-defined functions that make use of the already available data. It is through these optional capabilities that truly ML-ready datasets can be prepared and loaded.

Some of the most popular ML libraries including Scikit-learn, XGBoost, TensorFlow, and PyTorch have different data requirements for optimized loading and training. By default, NucML Datasets returns pandas DataFrame objects which are compatible with all scikit-learn models. Although any user can subsequently transform these data structures, several functionalities are provided to return library-dependent objects. For example, for TensorFlow, a tf.data.Dataset objects can be loaded with user-specified options including batch sizes, shuffle buffers, and cache/prefetch instructions. Other objects supported are DataLoader and DMatrix for PyTorch and XGBoost respectively.

NucML's Experimentation and Compilation Capabilities

There is an extensive set of utilities provided by NucML to support the experimentation and compilation phase of the ML-NDE. The most important loader functions support many data formats including simple NumPy arrays and TensorFlow Dataset objects. Some of these functions include:

- nucml.datasets.generate_exfor_dataset: Generates all needed EXFOR datasets for neutron-, proton-, alpha-, deuterons-, gammas-, and helion-induce reactions. This function can be used to update EXFOR whenever new C4 files are added.
- nucml.datasets.load_ame: Loads any of the three Atomic Mass Evaluation 2016 files. It also offers some capabilities to generate synthetic data for natural targets which is needed to join with EXFOR.
- nucml.datasets.load_xundl: Loads the XUNDL structure levels data. It also provides utilities to use user-defined cut-off parameters.
- nucml.datasets.load_xundl_ground_states: Loads only ground state information for all available isotopes in XUNDL.
- nucml.datasets.load_xundl_headers: Loads headers from RIPL .dat files. It contains basic information including the number of gamma rays measured, levels detected, and more.
- nucml.datasets.load_xundl_isotopic: Loads level or gamma records for any given isotope.
- nucml.datasets.load_xundl_ml: Loads an ML-ready dataset for inference of nuclear level's energies and nuclear level density. Categorical data is one-hot-encoded by default and standardization is applied with parameters fitted on the training set.
- nucml.datasets.load_ripl_parameters: Loads the provided RIPL cut-off parameters provided by the Reference Input Parameter Library (RIPL). The returned parameters can be used with the nucml.datasets.load_xundl to return a filtered set of nuclear levels.
- nucml.datasets.load_evaluation: Reads an evaluation file for a specific isotope, reaction channel, and evaluated library. Supported regional libraries include endfb8.0, jendl4.0, jeff3.3, and tendl.2019.
- nucml.datasets.load_exfor: This is the main loading function for EXFOR. It loads the EXFOR dataset in its various forms including ML-ready datasets for different particle

induce reactions or all of them. A set of recommended transformations is applied by default including energy thresholds for low energy inference, unskewing functions, standardization and normalization, filters for wrongly reported data, among others. Visit the documentation for more.

While these are the main loader functions for nuclear data, a set of general and prepossessing utilities are used to support these capabilities. Some of which include:

- nucml.general_utilities.parse_isotope: Helps standardize isotope formatting to work with all NucML's utilities.
- nucml.general_utilities.parse_mt: Helps standardize the reaction type formatting to work with all NucML's utilities.
- nucml.general_utilities.save_obj: Saves a python object including trained ML models, arrays, lists, and any other python-based generated object for later use.
- nucml.processing.impute_values: An isotope-specific data imputation function. It uses all available data for a given proton (Z) and mass number (A) to linearly interpolate missing values. This is the main process behind the Atomic Mass Evaluation natural target data generation.
- nucml.processing.normalize_features: Applies a transformer or normalizer to a set of specific features in the user-provided data. Different normalizers are supported including "poweryeo", "standard", "minmax", "maxabs", "robust", and "quantilenormal". This is the main normalizer engine behind all ML-ready datasets.

(2) Evaluation using ML

In the evaluation phase of the traditional NDE pipeline, relevant data is used to guide physics-based model calculations which result in best estimates, dependent on data availability, of mean values including uncertainties and covariances. These values can then form part of one or more regional libraries (i.e., ENDF). As previously mentioned, the ML-NDE pipeline instead makes use of trained ML models to create reaction cross section data and therefore to generate ML-based libraries.

The NucML Model utilities provide various python scripts to train various ML algorithms including scikit-learn models (i.e. K-nearest-neighbors, Decision Trees), Gradient Boosting Machines, and Neural Networks. It is built around a strict ML management philosophy by keeping track of model hyperparameters and resulting performance metrics for the supported models. The files created from the training iterations can then be used to analyze the impact of different hyperparameters on model performance. These scripts also take care of saving the model, the scaler/normalized used, and the path to all needed objects for future persistence. If the model object needs to be accessed later, the path for both the model and scaler is saved within the management file. There is no need to retrain the models. Other ML management tools like Comet ML and Weights and Biases can be configured and used using user-provided credentials. Based on the management files, different plotting utilities are offered to make analyzing the generated data easier. It is the goal of NucML to first and foremost provide researchers the framework and tools to create, train, and analyze their models rather than providing a set of optimized algorithms.

NucML's ML Evaluation Capabilities

Most of NucML's capabilities deal with data, processing, and validation. The Evaluation phase consists mainly of effectively training ML models. This is user-dependent but included is a list of the training scripts used in this work to help anyone get started. All scripts are built on top of a strict experiment tracking framework. Every model trained is saved for later loading and validation. In each script execution, a specific scaler is fitted and it is saved along with the model. Similarly, the performance metrics calculated on the train, test, and validation subset are saved under the same file. At the end of the experimentation iterations the results file will contain:

- Model Name
- Scaler Type
- Performance Metrics (i.e. MAE, MSE, etc.)
- Model-specific Hyperparameters
- Location of Saved Model and Scaler
- Training Time

This strict management is important for post-training analysis of hyperparameters and will be useful for benchmark validation but more on the latter later. Models included in NucML and command line arguments include:

- K-Nearest-Neighbor (Scikit-Learn):
  - Distance Metric
  - Nearest Neighbors (K)
- Decision Tree (Scikit-Learn):
  - Distance Metric (Euclidean, Manhattan)
  - Nearest Neighbors (K)
  - Weights (Uniform, Distance)

- Gradient Boosting Machine (XGBoost):
  - Max Bin
  - Max Depth
  - L2 Regularization
  - Objective Function
- Neural Network (TensorFlow)

Common command-line arguments between all training scripts are the EXFOR dataset mode, normalizer type, and the MT strategy. The higher the EXFOR dataset mode, the more features included in the resulting dataset. In the next Chapter, these features will be defined along with the normalizers supported. Even though every user should develop their training scripts NucML does contain some functions that can make training models on EXFOR easier. These include:

- nucml.model.model_building.compile_and_fit: This utility allows the user to train any TensorFlow model by simply passing the model architecture, the training and testing data, training parameters (i.e. batch size, epochs, learning rate, and methods). If ML management utilities such as Weights and Biases or Comet ML are being used, the function also enables to append service-specific callbacks. It also allows continuing training if checkpoints have been saved.
- nucml.model.model_building.get_callbacks: Returns a list of TensorFlow callbacks. Useful when custom or module supported learning rate adjustments are to be made during training (i.e. reduce on plateau, exponential decay, etc.). Other callbacks included by default are checkpointing and early stopping which allows training to be stopped if the validation loss has stopped declining.
- nucml.model.model_building.get_xgboost_params: XGBoost training require model parameters to be passed. This function returns a dictionary containing all hyperparameters needed to train a Gradient Boosting Machine model with EXFOR. Its customization allows the user to define the learning rate, regularization strength, max depth, grow policy, max bin, objective function, and GPU specification.

Furthermore, utility functions are given so that any user can keep careful track of their experimental campaigns including:

- nucml.model.utilities.create_error_df: When a user starts training their models using EXFOR, the validation utilities still require a carefully formatted DataFrame with model information and performance metrics. This function transforms the calculated metrics into a pandas DataFrame for insertion into the main results file.

- nucml.model.utilities.train_test_error_df: Once all performance metrics have been calculated on all three subsets (train, validation, testing), this utility joins the results and returns a single concise DataFrame row.
- nucml.model.utilities.load_model_and_scaler: Since the results file contains the path to both the saved model and scaler, this utility will load and return both objects.
- nucml.model.utilities.make_predictions: Given a numpy array, the function will transform the data into an appropriate format for the given model (DMatrix, TensorFlow, etc.) and pass it to the model for predictions which are then returned.
- nucml.model.utilities.regression_error_metrics: Calculates the Mean Absolute Error (MAE), Mean Squared Error (MSE), EVS, MAEM, and R2 between two given vectors. Useful to calculate performance metrics when coupled with model predictions and true data.
- nucml.model.utilities.get_best_models_df: Given the results file generated by the training iterations, this function filters and returns the three best models based on the main performance metric on the test, train, and validation set.
- nucml.model.utilities.cleanup_model_dir: After some time, accumulated files can take considerable storage. This utility allows the user to delete all models except for the best models in terms of the main performance metric on the train, validation, and testing set.

(3) Processing

In the processing stage, evaluated data is transformed into formats (i.e. ACE files) readable by user codes like SERPENT and MCNP. Having a great model is only half the process. Tools are needed to efficiently create entire evaluated libraries consisting of cross sections for all needed isotopes and reaction channels. Currently, NucML offers the capabilities to create isotopic ACE files by simply supplying a user-trained model. The python package will take care of querying the model for predictions using the same energy grid as the original ACE files for compatibility. Additionally, some stability options are included. Since these algorithms are not expected to be perfect, NucML Ace is built under the assumption that the best solution is a hybrid solution. In other words, traditional tools must work together with ML models to create a good evaluation. By default, NucML Ace will stabilize the 1/v region using evaluated library values and all values after the first resonance are filled with ML generated values.

NucML's Processing Capabilities

- nucml.ace.data_utilities.convert_dos_to_unix: Useful if working on both Windows and Linux, it allows to convert any file from DOS to UNIX. NucML uses this to make new benchmark inputs compatible with transport codes.

- nucml.ace.data_utilities.create_new_ace: Generates a new ACE file ready to be used in transport codes from ML generated data. It utilizes both an XSS array using the original ACE format for a given isotope or a DataFrame as long as the length matches the original ACE energy array.

- nucml.ace.data_utilities.create_mt2_mt3_mt101: This is an experimental feature made available for cases where there is lacking data for either the MT2 or MT3 reaction channel. The user is responsible for checking the consistency of the generated cross sections. An easy alternative is to adjust generated cross sections. This utility function calculates the (n,nonelastic) reaction channel (MT3) as the summation of 4, 5, 11, 16, 17, 18, 22, 23, 24, 25, 26, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 41, 42, 44, 45, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117. Subsequently, the (n,disappearance) cross section (MT101) is calculated as:

  $$MT_{101} = \sum_{N=102}^{117} MT_N$$

  After calculating both the (n,nonelastic) and (n,disappearance), then the (n,elastic) scattering is calculated as

  $$MT_2 = MT_1 - MT_3$$

  This process enforces these physical constraints at the expense of noise introduction to either the (n,nonelastic) or (n,elastic) cross sections.

- nucml.ace.data_utilities.enforce_unitarity: In cases where unitarity has not been enforced, the MT2 cross section can be calculated as

  $$MT_2 = MT_1 - MT_3$$

  However, this will lead to some negative values which are not valid. This utility function will add back the negative values to the total cross section to effectively conserve unitarity. This adjustment is optional and is performed automatically if the user does not provide an already constrained set of cross sections.

- nucml.ace.data_utilities.fill_ml_xs: ML models (other than neural networks) are expected to be unstable at very low energy regions, especially in the 1/v region. KNN models, for example, are purely a function of dataset completeness and Decision Trees have a square-like behavior due to the underlying feature-splitting process. This means that even though the trend may be predicted correctly, smooth values may not be predicted for these energy regions where the cross section behaves somewhat predictably. Therefore, a hybrid approach can be taken to correct this behavior. The user can do it manually or NucML will find the nearest resonance peak on the evaluated data and replace lower energy cross section values up to the peak energy point with evaluated information. This provides the stability of physics-based models while conserving ML-generated information in the resonance and fast energy region. NucML uses scipy's implementation where a peak is defined as:

...any sample whose two direct neighbors have a smaller amplitude. For flat peaks (more than one sample of equal amplitude wide) the index of the middle sample is returned (rounded down in case the number of samples is even).

Since we are only searching for the first resonance peak, a set of default parameters is provided that has worked with many common nuclei.

- nucml.ace.data_utilities.get_energies: Retrieves the energy array for the original evaluated files for a given isotope. This is useful for users to have their ML models predict the needed data at these specific energies which are required by SERPENT2 since custom energy grids are not supported.

- nucml.ace.data_utilities.get_final_ml_ace_df: Given a set of ML generated XS (adjusted), this function allows to fill in other reaction channels not included by the ML model predictions. This is useful since for some calculations some reaction channels are not required but are still present in the ACE files. This allows making ML-generated ACE files compatible.

- nucml.ace.data_utilities.modify_xss_w_df: Using the ML generated cross section, this function modifies the original ACE XSS array with the ML values.

- nucml.ace.data_utilities.reduce_ace_filesize: The size of the generated and copied ACE files can be reduced considerably by eliminating other temperature cross sections (i.e. .06c). This utility function will eliminate all but room temperature cross sections.

- Other useful functions to help users explore and modify ACE files include get_xs_for_mt, get_to_skip_lines, get_pointers, get_nxs_jxs_xss, get_nxs_dictionary, get_mt_xs_pointers_array, get_mt_array_w_pointers, get_mt_array, and get_jxs_dictionary.

(4) Validation and Application

Validation is perhaps the most important step in the entire pipeline. The evaluated data must perform at least better in applicable benchmarks than the previously released version

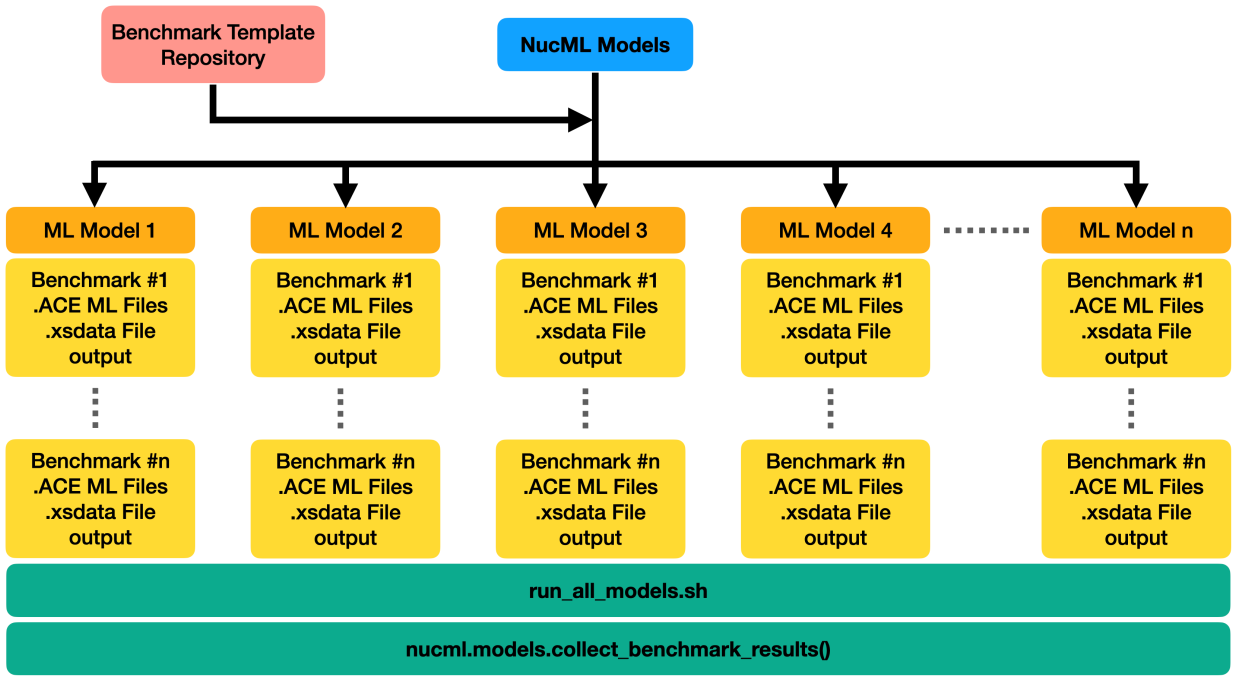

Benchmark Template Repository
NucML Models
ML Model 1
ML Model 2
ML Model 3
ML Model 4
ML Model n
Benchmark #1 .ACE ML Files .xsdata File output
Benchmark #n .ACE ML Files .xsdata File output
run_all_models.sh
nucml.models.collect_benchmark_results()

Figure 3.1: NucML Validation and Model Selection Process using the Benchmark Repository.

of the library. In ML solutions, generalization is the end-goal. As previously mentioned, algorithms usually optimize and try to minimize a loss function but EXFOR is far from perfect. A model that generalizes well to other EXFOR datapoints is not necessarily the best. Therefore, we cannot rely on performance metrics derived from EXFOR data (even if splitted or cross-validated) to select the best performing model. Instead, a series of trained models need to be tested and validated using all available benchmarks. The model with the best average performance can be regarded as the best model. This is quite different from both established NDE and traditional ML but so is the nature of the challenge.

The benchmarking utilities provided are built on top of this methodology. Once a series of models are trained there are various steps needed to set up the validation phase. These are depicted in Figure 3.1 and include:

1. Querying the model for relevant cross section data for a given benchmark: NucML will look for a metadata file associated with every available benchmark containing the composition. This information is used to query the model for cross section data for each relevant isotope and each needed reaction channel. The predicted values are then passed into the processing step.

2. Processing the results and converting them into appropriate user-code formats: The predicted values are processed according to user-provided methodologies. The default

mode consists of correcting the 1/v region for all reactions (when applicable) using evaluated data. Every value after the first resonance peak is filled using the ML-generated values. After all reactions for all isotopes have been adjusted, the original ACE file is copied and modified using the new data.

3. Building metadata files used by Monte-Carlo codes: The .xsdir file contains the ACE files path for every isotope. This is used by SERPENT2 to load the applicable reaction information when needed. NucML will keep track of the previously ML-generated ACE files to create a new .xsdir file that can be used by SERPENT2 to find the ML-generated files.

4. Modify the benchmark inputs accordingly: Once all cross section related files are created, the code will modify the template benchmark input file with the appropriate paths and store them in user define directories.

5. Run the benchmark files with SERPENT2: NucML will also keep track of generated benchmark input files. Once all previous steps are finished, the code will create a single bash script that contains all needed instructions for running each benchmark and converting the output files into more friendly formats for later use.

6. Analyze the output: Once all simulations have finished running, all results can be easily gathered and analyzed in conjunction with model hyperparameters and performance metrics to find the best overall algorithm. Additional capabilities include a fast and easy comparison between ML- and ENDF-based benchmark calculations for fast reporting.

This process has to be performed for each trained model and each benchmark. By making use of a benchmark template repository, NucML can seamlessly run all steps of the validation phase for all trained models and all available benchmarks. The capability of adding user-defined benchmarks is also available.

NucML's Validation and Application Capabilities

- nucml.ace.data_utilities.generate_bench_ml_xs: This is one of the core functions of the validation stage NucML utilities. By providing simple information such as the benchmark name, NucML will generate all needed cross sections. By setting a threshold, cross sections will be generated using ML models for all isotopes with composition above the defined threshold. There are instructions available to add user-defined benchmark inputs. By providing a list of trained ML models along with the storing information, NucML will automatically search for all models, load them, and use them to predict the needed cross sections using the original isotopes ACE energy grid.

- nucml.ace.data_utilities.generate_serpent_bash: To automate running all generated benchmark model cases, this function searches for all benchmark inputs and builds a bash script with desired specifications that contains all needed commands to run all monte-carlo simulations and convert results using a simple MATLAB conversion script.

- nucml.ace.data_utilities.copy_ace_w_name: When creating new benchmark inputs it is important to provide any cross section not provided by ML for the benchmark calculation to run without any error. This function allows copying evaluated ACE files to cover those needed cross sections.

- nucml.ace.data_utilities.copy_benchmark_files: NucML's validation utilities are built on top of a template benchmark repository. For every ML model to be tested a new set of benchmark input and supporting files are needed. This function takes care of the transfer from the repository to a location of choice.

- nucml.ace.data_utilities.gather_benchmark_results: This utility function gathers the criticality results from all resulting MATLAB (.mat) files from SERPENT2 in a given directory. This makes it easy to collect and analyze hyperparameters as a function of the criticality benchmark results. While the intended capability is to simply collect the multiplication factor, other information (i.e. detectors) can be easily extracted by modifying the source code for this function.

- nucml.ace.data_utilities.generate_sss_xsdata: SERPENT2 needs a .xsdir file containing the path to the needed ACE files. This function generates the metadata file using the appropriate directories.

(5) Other Capabilities

The result of this effort is a comprehensive suite of tools to aid the current nuclear data evaluation pipeline. Each of the NucML methods contains utilities that can be used by many activities of the NDE pipeline (Figure 3.2) and the traditional ML workflow. To allow researchers to quickly set up and begin working on their ML models, a variety of tutorials are included in the code's GitHub repository. These notebooks allow anyone to explore all core functionalities along with more miscellaneous functions that allow easy exploration of EXFOR, AME, ENSDF, and ENDF.

One major objective was not only to aid the NDE pipeline but also to make data and benchmark information available in a modern easy-to-access way. As a result, several ready-to-use particle-induce reaction ML-datasets are provided through a Google Cloud storage bucket making it available to all the community without the need to install any tools or codes. These include neutron-, proton-, gamma-, and -alpha induce reaction cross section datasets. Furthermore, a public repository has been created to store community-made benchmarks. While a working version NucML can easily be installed through the Python Package Index, any user can easily modify the source code to suit any of their needs.

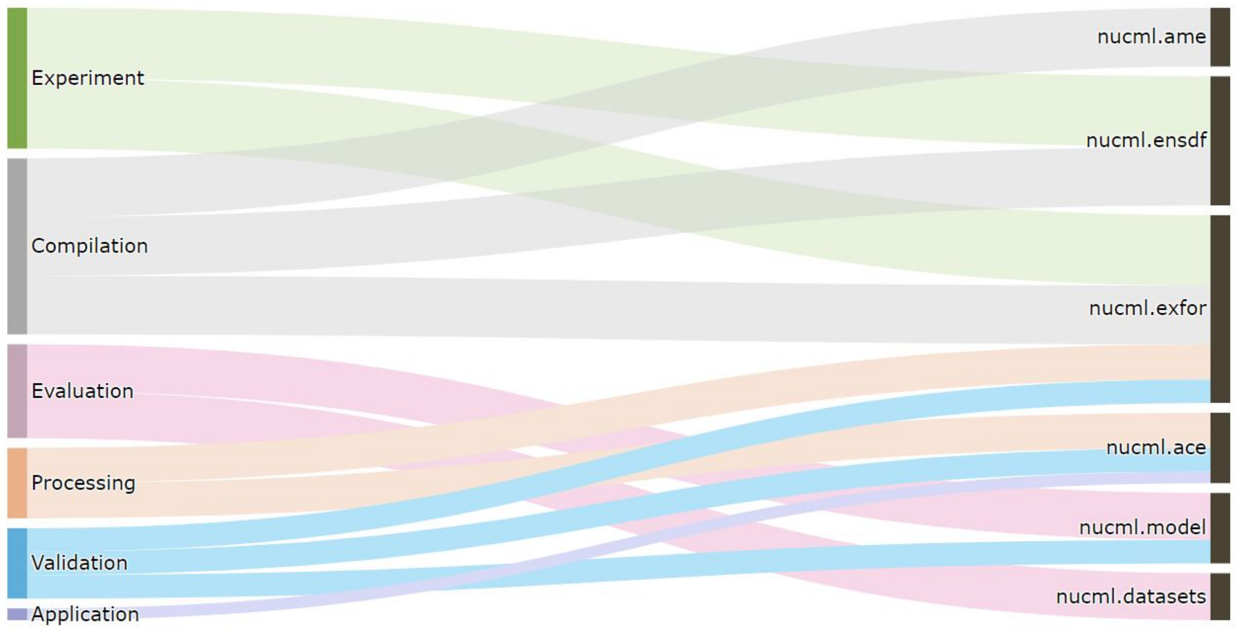


Figure 3.2: ML-enhanced Nuclear Data Evaluation Steps and applicable NucML Modules.

Data Analysis Speed Improvements

Currently, the online EXFOR user interface (UI) provides users the ability to extract data for quick analysis. Unfortunately, there is no API or modern interface to quickly extract data using the most popular programming languages. Furthermore, miscellaneous activities like plotting reaction cross sections can be slow for widely experimented isotopes and reaction channels. NucML, taking advantage of data locality, can accelerate data analysis and exploratory data analysis. Utility functions are included to help with everyday tasks such as plots of experimental data from EXFOR, comparison of different evaluated libraries, error calculations between EXFOR and ENDF datapoints using custom and standard energy grids, and much more. For example, the time needed to plot all experimental points for ^{233}U(n,f) using NucML was approximately 10 seconds, a 95% improvement in speed relative to the EXFOR user interface (28 seconds). NucML does not only support static plotting but also interacting Plotly plots. These types of speed improvements in nuclear data analysis are noticeable across all of NucML functionalities. As for modeling improvements, there are no other software or pipelines to compare with. This is the first ML-evaluation software of its kind.

3.3 Summary

Nuclear data evaluations are important for every industry and technology that utilizes nuclear-based technology. The next generation of nuclear reactors including Molten Chloride Fast Reactors needs accurate cross section data to be designed safely and reliably. Organizations in charge of producing regional evaluated libraries have an important task at hand. Evaluators must be given all available tools to aid this difficult process.

NucML is the first and only end-to-end python-based supervised machine learning pipeline for enhanced bias-free nuclear data generation and evaluation to support the advancement of next-generation nuclear systems. It offers capabilities that allow researchers to navigate through each step of the ML-based nuclear data cross section evaluation pipeline. Some of the supported activities include dataset parsing and compilation of reaction data, exploratory data analysis, data manipulation, feature engineering, model training and evaluation, and validation via criticality benchmarks. Some of the inherent benefits of this approach are the reduced human-bias in the generation and solution and fast iteration times. The resulting data from these models can aid the current NDE and help decisions in uncertain scenarios. Future work includes adding more capabilities to support other modeling codes, create ACE files with user-provided energy grids, incorporate more databases, and directly integrate traditional modeling codes.

Chapter 4

Data and Benchmarks

To provide useful information to a model for cross section inference, a dataset with relevant features must be built. There are many factors that cross sections depend on including the incident neutron's energy and the target isotope. As reviewed in Chapter 2, an isotope, in turn, has a variety of top-level properties including the:

- Number of Protons (Z)
- Number of Neutrons (N)
- Mass Number (A)
- Precise Atomic Mass (M)

From these isotopic properties, several more features and quantities can be calculated including the:

- Radius of the Nuclei (R)
- Surface Nuclei Fraction in the Nucleus (1/R)
- Binding energy (BE) and the binding energy per nucleon (BE/A) including the neutron and proton binding energy.
- Packing Fraction (PF)
- Mass Excess (Δ)
- Q-value for β^+, β^-, α, etc.

This section provides a brief overview of the databases used for gathering the data to train the ML models: the EXFOR database and the Atomic Mass Evaluation 2016 (AME2016). Additionally, a brief description is given for the criticality benchmark cases used to validate trained models. Data in the EXFOR and AME2016 libraries were used to compile a working dataset that was manipulated and transformed for modeling using machine learning algorithms using NucML. The ENDF library was used as a benchmark to compare ML-solutions.

4.1 EXFOR

The nuclear reaction compilation database, known as the EXFOR library was collected from the International Atomic Energy Agency (IAEA) [15]. As mentioned previously, it is the database for experimental nuclear reaction data. In addition to reaction cross sections, this database contains resonance integrals, fission yields, polarization data, etc. For this work, only neutron-induce reaction cross sections were extracted. All other data types were discarded from the dataset and are therefore out of the scope of the following exploration and discussion.

Through international collaboration between the Nuclear Reaction Data Centres (supervised by the IAEA Nuclear Data Section), the compilation of experimental data takes place by identifying the literature for publications where reaction measurements have been made. Once identified, the National Nuclear Data Center (NNDC), the institution responsible for data measured in the United States and Canada, creates an EXFOR entry number for the identified experimental campaign. The reported data is revised between institutions for incorporation into the EXFOR Master File. The database contains approximately 4.5 million reaction data points covering a variety of isotopes and reaction channels across different facilities.

Before attempting to apply any ML-based solutions it is important to explore and understand the strengths and limitations of this database. An ML model is only as good as the dataset is trained on meaning that in an ideal scenario, the dataset must be representative. In this particular field, obtaining new experimental data is not a trivial task. It takes resources, including facilities, target material, and time to perform new measurements. These resources are sometimes not readily available.

Table 4.1 contains a list of the extracted features from EXFOR. In the following subsections, a brief description and exploration of each of these will be carried.

Energy

In the dataset, there are a couple of measurements with energies of 0 eV. Data points with zero energy are invalid and were therefore filtered out. The Energy distribution has a high Fisher-Pearson coefficient of skewness of 3.85 and, as expected, a high standard deviation (2.62E+06) both of which are detrimental to model training and optimization. In later sections in feature engineering and processing, all features will be normalized/standardized to bring features to approximately the same order of magnitudes with similar standard deviations. The skewness of the Energy feature is however resistant to these normalizers. Both a Box Con power transformer and simple logarithm transformation have a more significant im-

Table 4.1: EXFOR Dataset Features[a].

Feature Name	**Values (min, max)**	**Input Type**
Incident Energy (eV)[b]	5.763000e-10, 1.017000e+11	Float
Cross Section (barns) [b]	0.000000e+00, 2.311600e+08	Float
# of Protons [b]	1, 99	Integer
# of Neutrons [b]	0, 156	Integer
Atomic Mass Number [b]	1, 255	Integer
Target Meta-State [b]	M, M1, M2, All	Categorical
Reaction Number (MT) [b]	1-4, 16-18, 22, 24, 28-29 32-33, 37, 41, 51, 101-108 111-113, 152-153, 155 158-161, 203, 1003, 1108 2103, 9000, 9001	Categorical or Integer
Product Meta State [b]	All, L, M, Ground, M2, Unknown, More than 1, M1	Categorical
Center-of-Mass Flag [b]	Lab, Center-of-Mass	Categorical
Target [b]	Isotope, Natural	Categorical
Target Atomic Radius (fm)[c]	1.25 - 7.92	Float
Neutron/Target Atomic [c] Radius Ratio (fm)	1.009253e-01, 6.400000e-01	Float

[b] Source: EXFOR
[c] Source: Calculated

pact on skewness resulting in a Fisher-Pearson coefficient of approximately 0.096 and 0.626 respectively. Both will be explored when pre-processing the data in a later section.

Cross Section

The cross section statistics are similar to those of the Energy feature although in different magnitudes. The standard deviation is very high (6.18E05) and has an FP-coefficient of 138.58. For this particular data feature, we must be careful not to change the nature of the data since this is our regression label. These values represent physical reaction probabilities (although not in the usual 0-1 scale). Similarly to the Energy feature, there are several data points where the cross section is 0. These are not incorrect but are still filtered out to be able to apply a log transformation. The fraction of data points with a cross section value of zero is minimal. A log transformation brought down the skewness coefficient to 1.01, still high but much better. A log transformation also creates correlation. The Pearson correlation coefficient is -0.0192 and -0.43 for non-transformed and transformed data respectively for the ^{235}U(n,g)^{236}U reaction. This means ML models will at least be able to learn the general trend and location in terms of energy and cross section. A Box Con transformer was able to achieve a much lower skewness coefficient, however, it changes the nature of the data due to the parametric nature of the function. One non-model related benefit of applying a log transformation is the easy direct visualization of results obtained by ML models.

Reaction Channel

The reaction type is one of the most important features as it identifies what reaction channel each measurement belongs to. Table 4.2 shows the distribution of reaction channel points across EXFOR and the reaction notation translation for each ENDF integer-code (MT). A big portion of the neutron-induce reactions belongs to the (n,total) channel covering approximately 75% of the entire dataset. There is also a decent amount of information for fission and capture.

In general, class imbalances need to be kept in mind when developing models. Performance in highly explored reaction channels will certainly be strong provided the quality of the data is high while the performance in other less studied channels might be low due to the lack of training data. This is true especially for some channels for which we have less than one thousand data points. Predictions on these will be challenging, may be inaccurate, and will carry higher uncertainty. However, if the reaction channel can be represented in a non-categorical format that allows us to represent the channel as a set of an arbitrary number of outgoing particles, the model may be able to extrapolate knowledge from other reactions channels. Not only does model development affect generalization but also feature engineering and data representation.

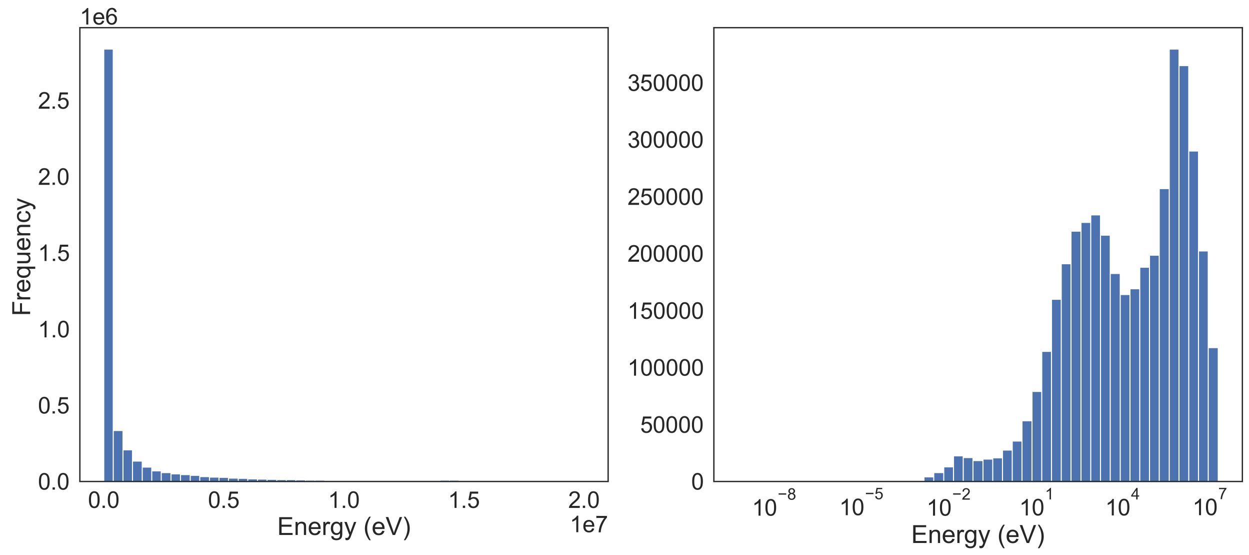


Figure 4.1: EXFOR's energy histogram before (left) and (after) applying the logarithm transformer.

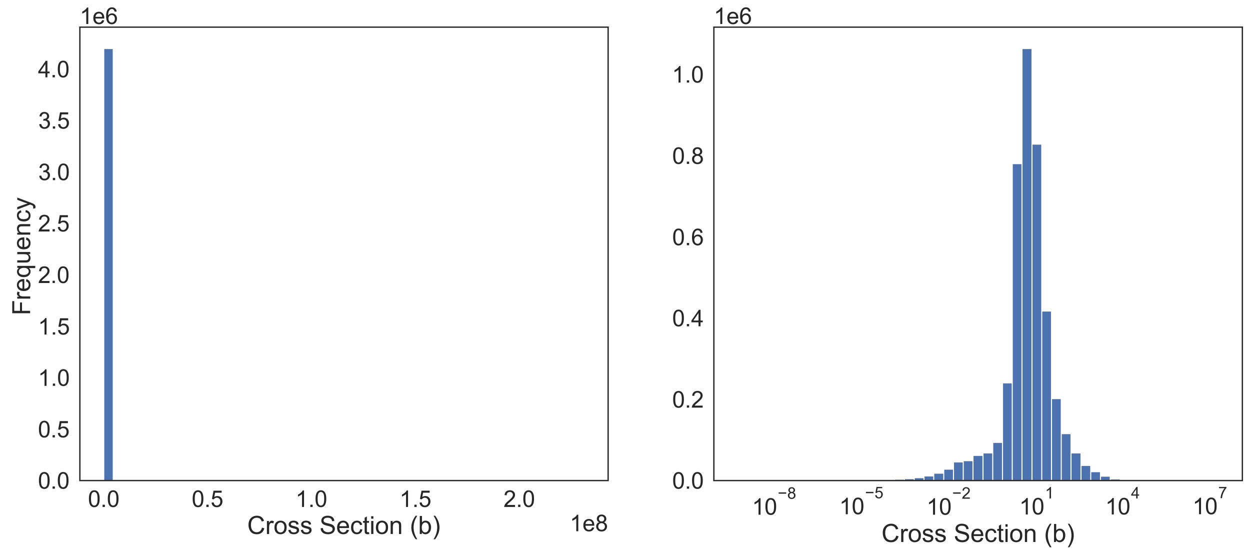


Figure 4.2: EXFOR's cross section histogram before (left) and (after) applying the logarithm transformer.

Table 4.2: Reaction Channel Distribution.

ENDF integer-code (MT)	**Reaction Notation**	**Datapoints**
1	(n,total)	3229171
18	(n,fission)	514890
102	(n,g)	216201
9000	Production Data	99895
51	(n,n1)	50571
4	(n,inelastic)	37751
2	(n,elastic)	37417
103	(n,p)	17574
107	(n,a)	15669
16	(n,2n)	11754
9001	Production Data	11028
101	(n,disap)	7896
105	(n,t)	2086
3	(n,nonelastic)	1290
17	(n,3n)	601
104	(n,d)	275
28	(n,n+p)	237
155	(n,t+a)	141
22	(n,n+a)	140
37	(n,4n)	135
33	(n,n+t)	128
1003 [a]	Unknown/Unassigned	103
24	(n,2n+a)	98
106	(n,helion)	62
112	(n,p+a)	42
29	(n,n+2a)	42
108	(n,2a)	39
111	(n,2p)	36
32	(n,n+d)	23
2103 [a]	(n,p)DI	21
153	(n,6n)	18
152	(n,5n)	17
1108 [a]	(n,2a)CALC	13
160	(n,7n)	11
41	(n,2n+p)	11
203 [a]	(n,Xp)	9
161	(n,8n)	6
158	(n,n+a+d)	5
113	(n,t+2a)	2
159	(n,2n+p+a)	1

[a] These reaction channels were dropped out

Energy, Cross Section, and Reaction Channel

Energy and cross sections are usually visualized as a function of the isotope and reaction channel pair. For example, Figure 4.3 shows the available data points for the ^{56}Fe(n,total) reaction channel. In this example, data comes from a variety of experimental campaigns, each color representing a different one. These are usually performed by different authors in different institutions and even using different measurement methods. This increases the complexity of the challenge since experimental methods are not standardized nor they can be since the method is dependent on the phenomena being measured, the energy region of interest, etc.

Due to the experimental nature of the database, the quality standards by which measurements are incorporated are not high. There exists a variety of conflicting experimental campaigns in many researched reactions. For example, Figure 4.4 shows the ^{35}Cl(n,p)^{35}S reaction channel cross section. In the 10^1 to 10^6 eV energy region, the campaigns performed by Koehler and Popov differ significantly in almost every region including the $1/v$ region and the magnitude of the first two resonance peaks [10] [18]. In the traditional evaluation pipeline, it is the job of the evaluator to analyze the information and methodology of every dataset and discard or adapt them to come up with a set of recommended values. In this work, we rely on the algorithm's ability to be robust against outliers. In a later section, outlier detection techniques are explored.

Additionally, there are many scenarios where collected data only covers very specific energy regions. Calculations performed during the traditional evaluation process make theoretical-based assumptions in these areas of uncertainty. The ^{35}Cl(n,p)^{35}S is a good example of a reaction where the available experimental points cover just a small fraction of the energy range of interest for many applications (see Figure 4.4). These are situations where a machine learning solution can help provide information by extrapolating learned knowledge from other isotopes and reaction channels. The model must be able to navigate through this field of measurements of varying quality and specifications.

Novelty and Outlier Detection

Discarding EXFOR datasets is part of the evaluation pipeline performed by an evaluator. However, this introduces person-specific biases. ML-based solutions have the advantage of providing human-bias free inferences, provided the data is also as free from bias as possible. Still, some pre-processing techniques have been proven to be useful for novelty and outlier detection.

In ML and statistics, there are a variety of useful metrics that can be calculated to detect outliers, however, due to the experimental nature of this particular dataset and the resonance behavior of many reaction channels, it becomes extremely difficult to do this systematically and automatically. Figure 4.5 shows the ^{235}U(n,g)^{236}U reaction cross section plot. This is an example of a single reaction for a single isotope. Unfortunately, typical statistics techniques for outlier detection may categorize resonances (peaks) points as outliers. These are the

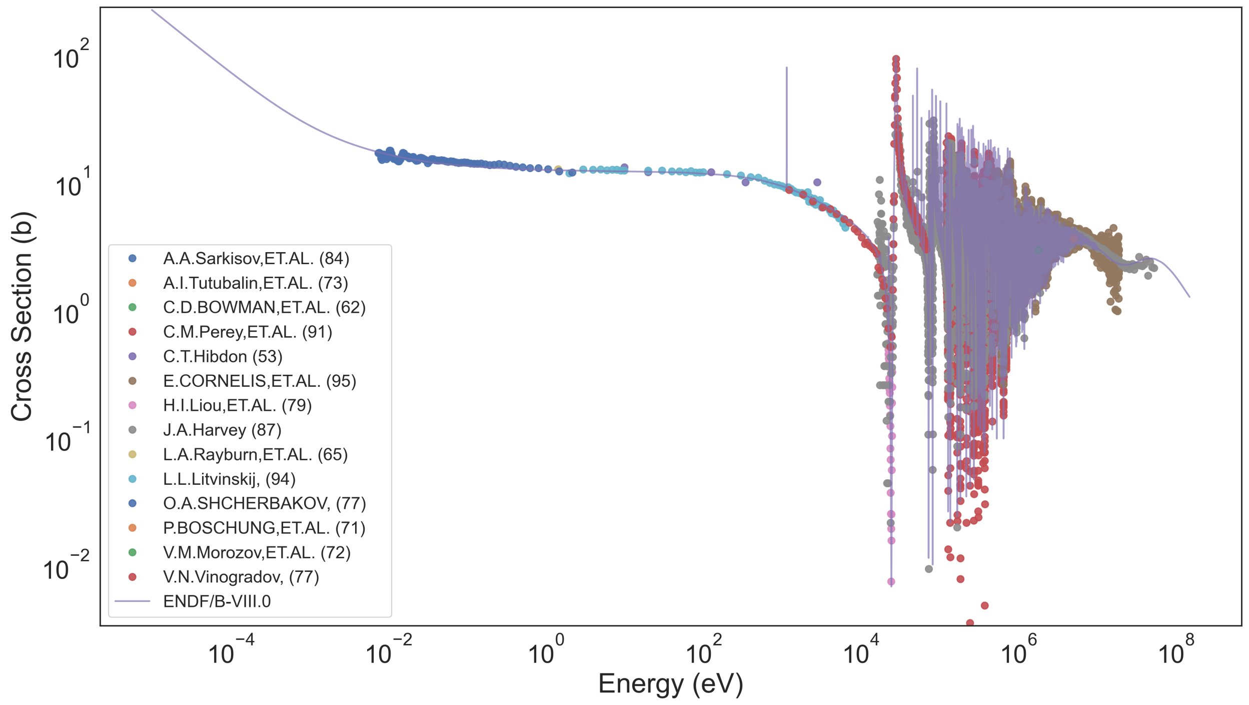


Figure 4.3: ^{56}Fe(n,total) reaction channel cross section.

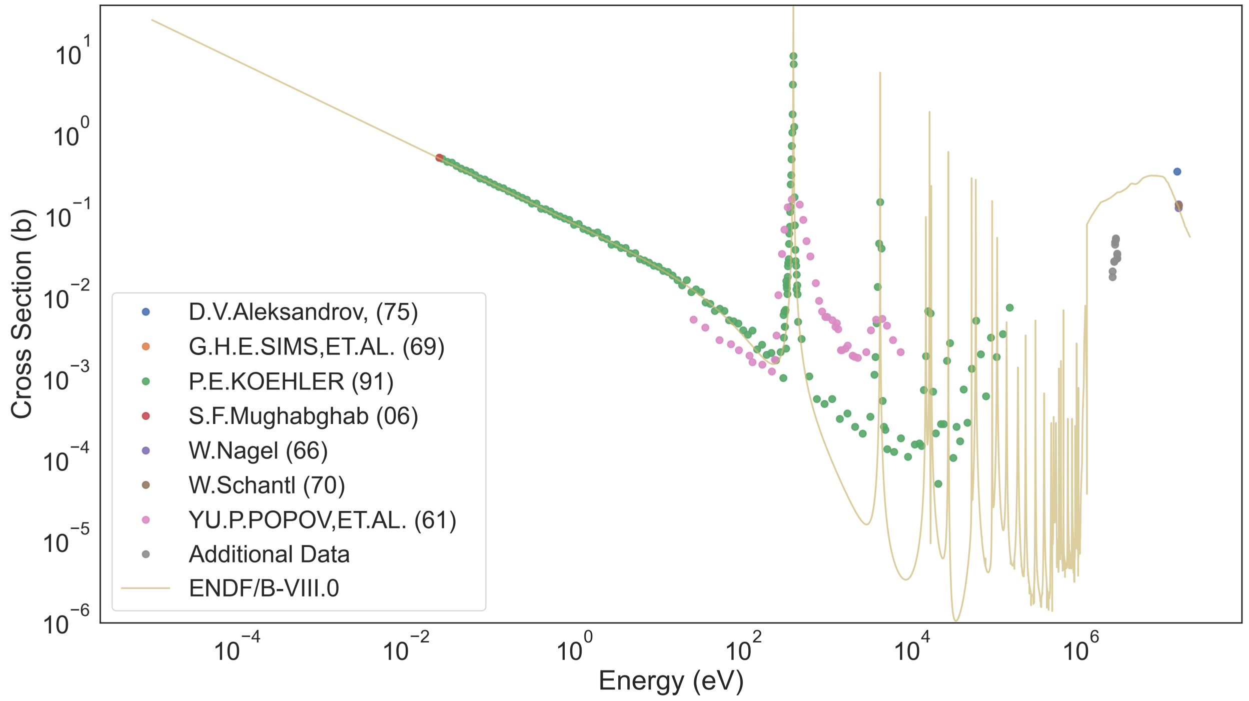


Figure 4.4: ^{35}Cl(n,p)^{35}S reaction channel cross section.

most important quantum physics phenomena that we need to model and must therefore be treated and manipulated carefully. Common techniques like filtering base on the first and third quartile, as shown in Figure 4.6, results in loss of important data not only from the resonance region but also the 1/v region which is not acceptable.

These traditional statistics-based techniques analyze the distribution of the data and typically "shave-off" data from both maximums and minimums. For these types of challenges, more advanced methods can be employed including robust covariance, one-class Support Vector Machines (SVM), Local Outlier Factor (LOF) algorithms, and Isolation Forests (IF) to try and clean the data better. Figure 4.7 shows the identification of outliers in the resonance section of the ^{235}U(n,g)^{236}U reaction by all four algorithms given a contamination parameter of 0.3%.

Each algorithm has its advantages and disadvantages and assumptions about the data. One class SVM works generally well but its performance is limited by the presence of outliers. This fact makes them suitable for novelty detection (outlier detection after fitting to a non-contaminated sample). RC assumes the data is Gaussian thus degrading all multi-modal data. Still, it works well as an outlier detector. IF and LOF algorithms are generally better suited relative to SVM and RC for multi-modal data sets. The LOF algorithm, for example, is resistant to density variations as a function of energy since from this model's perspective, outliers are a function of nearby neighbors. LOF appears to work better than all three previously mentioned models. The IF, RC, and SVM algorithms consider some of the top cross section points as outliers but LOF since there are enough neighboring samples locally. In general, these are more optimal than traditional statistics techniques.

One major disadvantage of all four methods is the contamination parameter. These algorithms require to be given the expected fractional amount of outliers/noise. This is difficult to determine or calculate for datasets like EXFOR. The contamination parameter will be reaction and isotope dependent. Additionally, this type of filtering operation would need to be optimized to act on each reaction channel for each isotope independently since outliers are not a function of other experimental measurements. For example, ^{35}Cl(n,p)^{35}S data points should not be provided when detecting outliers for the ^{235}U(n,g)^{236}U reaction. Unexpected and incorrect outcomes would result from these types of interactions. This makes this problem complex but still accomplishable. A viable approach is the tuning of the contamination parameter as a function of benchmark performance. This limits this approach to isotopes for which significant benchmark data exists. In the absence of benchmark data, the problem is completely unsupervised so outlier detection model selection and tuning can be a challenge.

Even without the utilization of these types of data cleaning techniques, many ML models are by nature resistant to outliers provided enough data is present and can provide bias-free predictions. In this work, the dataset was only checked for consistency and converted into a clean numerical format.

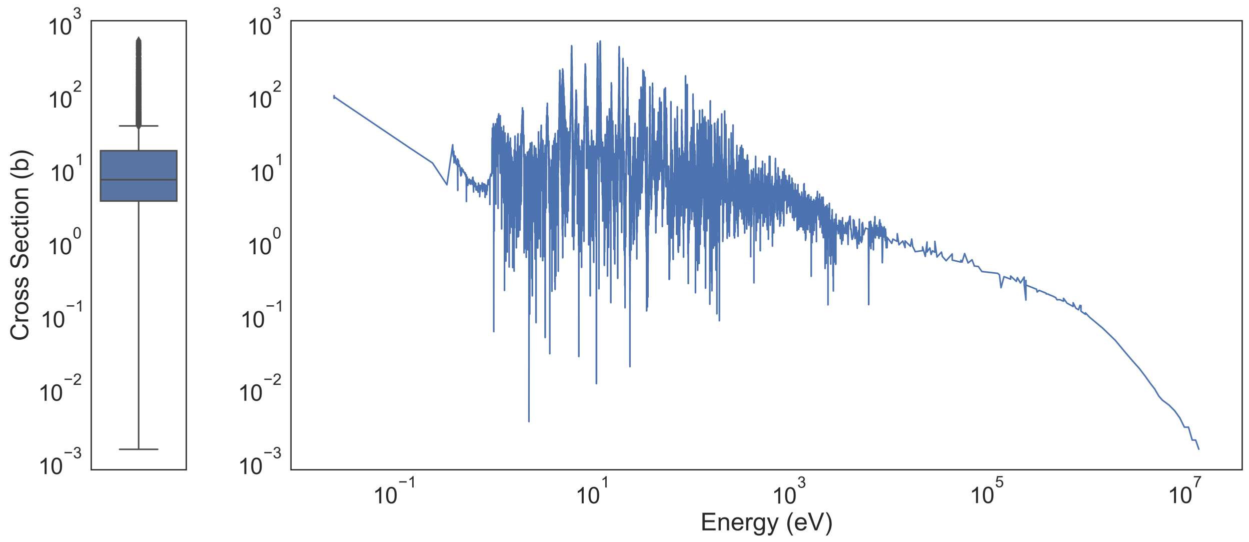


Figure 4.5: ^{235}U(n,g)^{236}U Reaction Cross Section plot. Left figure shows box plot of the cross section values.

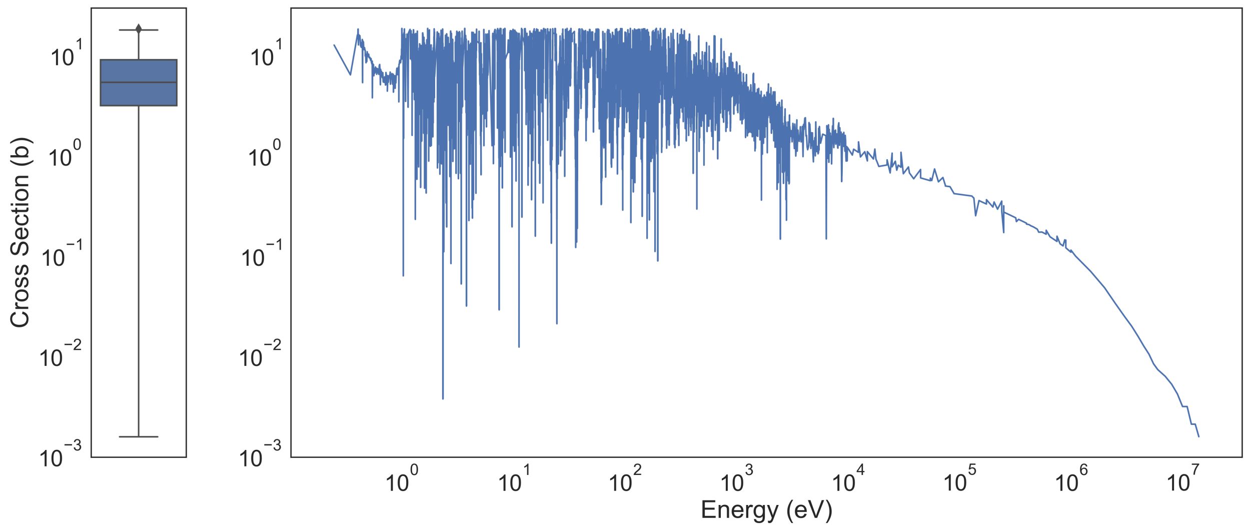


Figure 4.6: Filtered ^{235}U(n,g)^{236}U Reaction Cross Section plot. Left figure shows box plot of the cross section values.

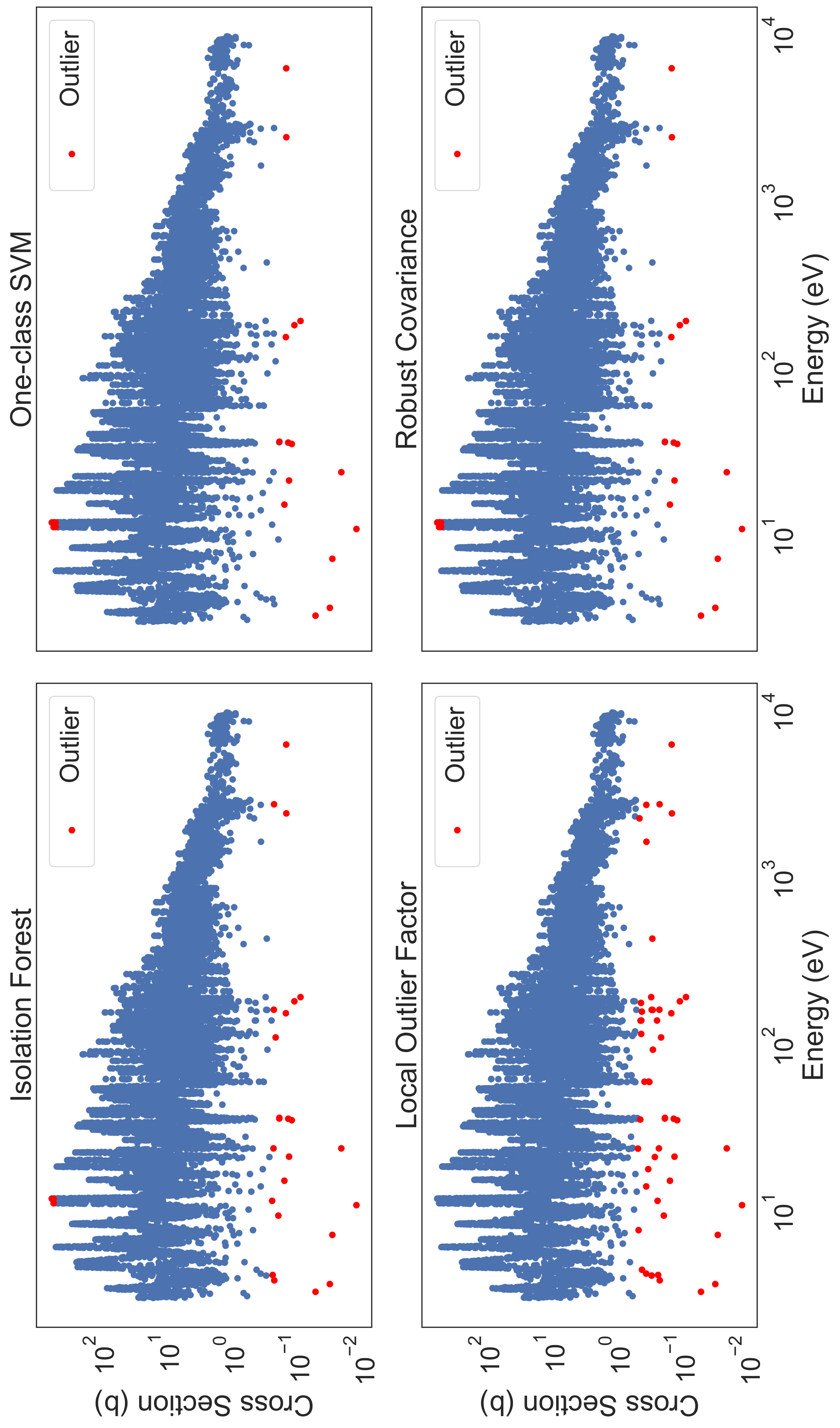


Figure 4.7: Outlier data according to different outlier detection algorithms. Contamination was set to 0.3%.

Table 4.3: EXFOR Status Distribution.

EXFOR Status Tag	**Datapoints**
Other	2181990
Approved-by-Author	1691538
Correlated	231849
Dependent	126722
Preliminary	20622
Outdated	2513
Re-normalized	175

Table 4.4: Center-of-Mass Distribution.

Center-of-Mass Tag	**Datapoints**
Lab	4250745
Center of Mass	4664

EXFOR Status and the Center-of-Mass Flag

While the EXFOR Status is not included in the final dataset it is good practice to understand where the data comes from. It is assumed that all missing tags are either "approved by authors" or "accepted" but not outdated and therefore filled with the key "Other" (see Table 4.3). A very small fraction (around twenty thousand data points) are categorized as outdated. One would be tempted to drop these datapoints but this will only introduce the bias of both the model developers and the people responsible for evaluating the experimental data points using this schema. Unless an experimental campaign was evidently erroneously performed, then no data deletion should take place. The database was left as is in terms of the EXFOR status.

As for the center-of-mass flag, most of the experimental campaigns are taken at the lab frame (see Table 4.4).

EXFOR C4 Fields 7 and 8

The EXFOR C4 files contain a pair of flexible data input columns that are author- and experiment-dependent. A tag is included to help identify the meaning of the values in these two input fields. Table 4.5 contains the tag value counts for this field. Missing tags were filled with "Other".

Target and Product Meta-stable State

The meta-stable state of both the target and the residual nucleus form part of the limited information provided also in the EXFOR C4 files. According to the documentation, data-

Table 4.5: Identifier Tag for Fields 7 and 8 in the EXFOR C4 Files.

Identification Tag	**Datapoints**
Other	4116656
Secondary Energy	93019
Level	42778
Excitation	2288
Secondary Energy Range	666
Half Life	2

Table 4.6: Target and Product Meta-Stable State Distribution.

Meta-Stable State Tag	**Target Datapoints**	**Product Datapoints**
All	4243232	4238714
M	12140	11162
M1	32	214
M2	5	317
Ground	0	4549
More than 1	0	333
L	0	101
Unknown	0	19

points with no specified meta-stable state are defined to cover all available states. Table 4.6 indicates the datapoints distribution of the meta-stable state values across the dataset.

Proton and Neutron Distribution

As mentioned previously, the (n,tot) reaction channel accounts for a big portion of the dataset causing a class imbalance. However, not only will the predictive power of a model depend on the amount of information on exit channels but also on the distribution of the target isotopes. Therefore, to further understand the possible model limitations and other imbalances included in the dataset, the distribution of proton and neutron numbers must be understood. Figure 4.8 shows the distribution of protons and neutrons across EXFOR.

Four main peaks appear in the distribution, two for protons and two for neutrons. The majority of the experimental campaigns seem to be focused at around 26 and 90 protons, and 26 and 145 neutrons. These peaks specifically correspond to high research including ^{56}Fe and ^{235}U. This translates into higher predictive power for isotopes having a similar or near number of protons and neutrons. Predictive potential on mirror nuclei might also increase since excitation levels between these are highly similar.

We can further explore what other isotopes have the highest and lowest number of exper-

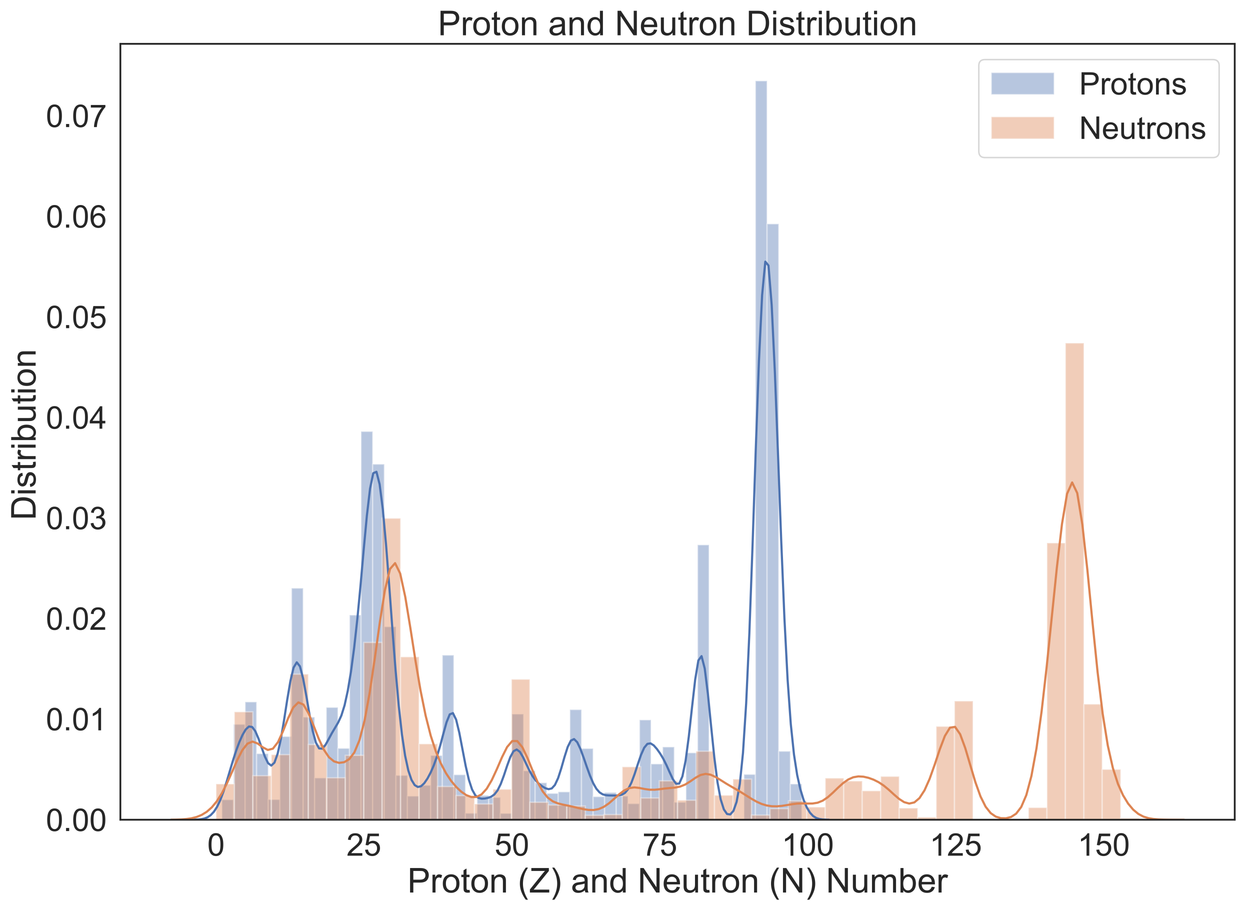


Figure 4.8: Proton and Neutron Distribution in the extracted EXFOR datapoints.

imental data points (see Table 4.7). Most of the data focuses on ^{56}Fe, an important isotope used for structural materials in all kinds of nuclear technologies, and fissile and fertile isotopes like ^{239}Pu, ^{235}U, ^{233}U, and ^{238}U. Additionally, amongst the most explore isotopes, there are many alloying materials including ^{58}Ni and ^{52}Cr. Shielding isotopes like ^{207}Pb have also been highly researched. On the other extreme, there are a lot of isotopes that do not have any measurements at all. Table 4.7 shows some of the lowest researched materials for which only one measurement exists. The majority of these are radioactive and therefore hard to handle in production and experimental settings. There are other isotopes for which no measurements have been performed. These types of isotopes are ideal to apply learned patterns from other highly researched materials using models that might be able to extrapolate knowledge.

Table 4.7: Data point Counts for Various Isotopes.

Highest Researched Isotopes		Lowest Researched Isotopes	
56Fe	238200	132Ce	1
239Pu	213599	199Pt	1
235U	186577	220Rn	1
233U	132923	146Pm	1
238U	130275	121Sn	1
237Np	125097	127Xe	1
91Zr	102540	151Pm	1
28Si	99885	249Cm	1
207Pb	96396	125Sn	1
241Pu	95073	135Ce	1
27Al	91938	169Er	1
12C	83096	99Mo	1
240Pu	81318	95Zr	1
52Cr	77458	88Rb	1
58Ni	74314	76Br	1

Table 4.8: Natural/Isotopic EXFOR Campaigns.

Target Sample Type	**EXFOR Distribution**
Isotopic	3218578
Natural	1036831

Natural and Isotopic Experimental Campaigns

Not all experiments take place in mono-isotopic target samples. Approximately 25% of all experimental campaigns in EXFOR were performed on naturally occurring samples (see Table 4.8). A model must be able to distinguish between natural and isotopic reactions. The handling of these natural samples will be extremely important.

Table 4.9 shows the highest researched naturally occurring target samples. These are not the usual fissile and fertile isotopes but rather elements that are usually used in bulk including structural materials like Iron. Others include Zirconium, Germanium, and Lead which are used for fuel cladding, detectors, and shielding respectively. Including a ”flag” feature that identifies if the target is isotopic or naturally-occurring will help our model distinguish them with the additional advantage of providing flexibility for cross section inference for both types of materials.

Table 4.9: Natural Target Distribution.

Target Element	**EXFOR Datapoints**
Fe	185562
Si	80428
Zr	79841
C	58210
Cu	36163
O	36101
Sn	35144
Te	29506
Mg	26676
Ni	26032
Pb	25273
Ge	23033

4.2 Atomic Mass Evaluation

The Atomic Mass Evaluation 2016 (AME2016) contains 3436 data points, each concerning a particular isotope. It includes a variety of experimental data used to derive a set of atomic mass values. Also, various reaction, separation, and decay energies for each isotope were extracted. A list of the extracted features and their value ranges can be found in Table 4.10 along with the corresponding uncertainties. The description of each feature will not be described here. For more information on the database and its contents, see Reference [20].

4.3 Feature Engineering

To create a more complete dataset with more relevant data for models to use, a couple of features were calculated and added.

Nucleus Radius and Neutron to Nucleus Radius Ratio

As mentioned in the previous chapter, scattering reactions at low energies are in some cases an approximate function of the target nucleus radius. To provide the model with a sense of the geometric aspect of the nuclei and the neutron, a pair of features were added to the dataset: the nuclide radius (Eq. 4.1) and the neutron-to-nuclide radius ratio (Eq. 4.2).

$$\text{Nuclide Radius} = r_o \cdot A^{1/3} \tag{4.1}$$

Table 4.10: Atomic Mass Evaluation Features[a].

Feature Name	Values (min, max)	Input Type
# of Protons	0, 9	Integer
# of Neutrons	0, 178	Integer
Atomic Mass Number	1, 295	Integer
Mass Excess (keV)	-9.165285e+04, 2.01512e+05	Float
Uncertainty (keV)	0, 2.003e+03	Float
Binding Energy (keV)	-2267, 8794.553	Float
Uncertainty (keV)	0, 667	Float
β^- Decay Energy (keV)	-28945, 31687	Float
Uncertainty (keV)	0, 2003	Float
Atomic Mass (micro-amu)	1.007000e+06, 2.952163e+08	Float
Uncertainty (micro-amu)	0, 2150	Float
S(2n) Energy (keV)	-3120, 40541	Float
Uncertainty (keV)	0, 2.003000e+03	Float
S(2p) Energy (keV)	-7630, 55187	Float
Uncertainty (keV)	0, 2014	Float
Q(a) Energy (keV)	-25474.73, 11920	Float
Uncertainty (keV)	0, 2042	Float
Q(2B-) Energy (keV)	-37359, 52098	Float
Uncertainty (keV)	0, 2003	Float
Q(ep) Energy (keV)	-52959, 28352	Float
Uncertainty (keV)	0, 2003	Float
Q(B-n) Energy (keV)	-39622, 31755	Float
Uncertainty (keV)	0, 2003	Float
S(n) Energy (keV)	-2488, 27715	Float
Uncertainty (keV)	0, 2011	Float
S(p) Energy (keV)	-4527, 31008	Float
Uncertainty (keV)	0, 2832	Float

[a] The reported uncertainties are with respect to the preceding feature name
Other Q-values include: Q(4B-), Q(d,a), Q(p,a), Q(n,a), Q(g,p), Q(g,n), Q(g,pn), Q(g,d), Q(g,t), Q(g,He3), Q(g,2p), Q(g,2n), Q(g,a), Q(p,n), Q(p,2p), Q(p,pn), Q(p,d), Q(p,2n), Q(p,t), Q(p,3He), q(n,2p), Q(n,np), Q(n,d), Q(n,2n), Q(n,t), Q(n,3He), Q(d,t), Q(d,3He), Q(3He,t), Q(3He, a), and Q(t,a).

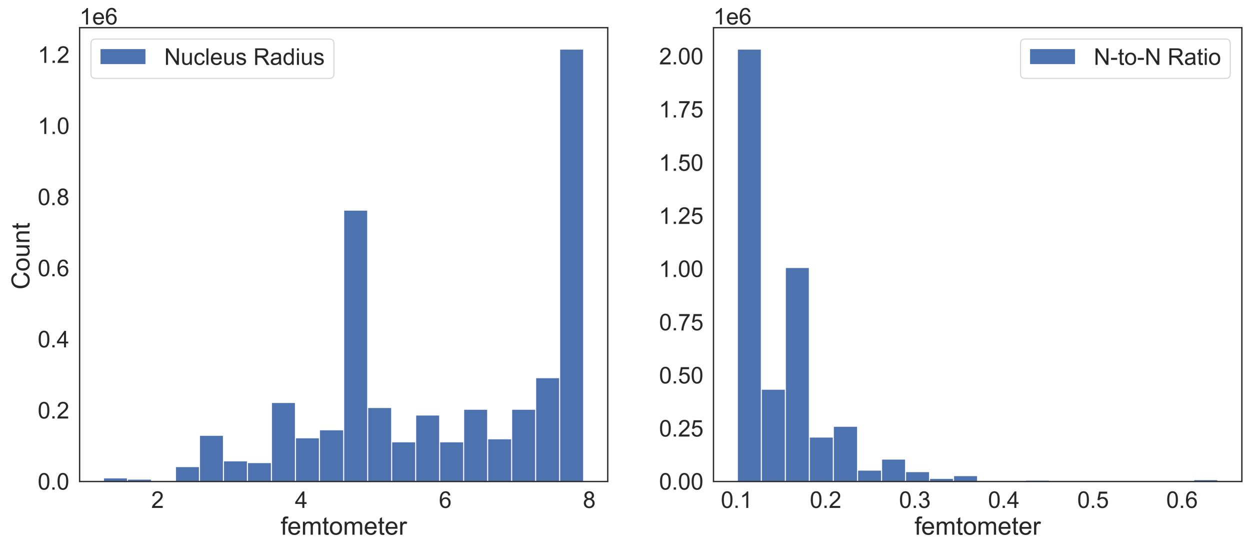


Figure 4.9: Nucleus Radius and Neutron-to-Nuclide Radius Ratio Histogram.

$$\text{Nuclide to Neutron Radius Ratio} = \frac{r_o \cdot A^{1/3}}{r_n} \tag{4.2}$$

The neutron radius was assumed to be 0.8 fm and the r_o parameter was set to 1.25 fm. The resulting distribution can be observed in the plotted histograms in Figure 4.9. As expected, due to the high concentration of heavy nuclei data points in the dataset, the skewness of the nucleus radius histogram is noticeable.

Neutron/Proton Valence Number and Promiscuity Factor

As an indication of possible collectivity to aid resonance predictions, the number of valence neutrons and protons relative to the magic numbers (2, 8, 20, 28, 40, 50, 82, 126, and 184) was added. Having obtained the valence numbers, the promiscuity factor was also calculated and added (Eq. 4.3).

$$\text{P-factor} = \frac{N_n * N_p}{N_p + N_n} \tag{4.3}$$

Additionally, even-even, odd-odd, and even-odd nuclei have different behaviors in terms of nucleus stability. To represent this, three categorical variables capturing their applicable group were added (see Table 4.11).

Table 4.11: Neutron and Proton Valence Type Distribution.

Neutron-Proton Valence Type	**EXFOR Distribution**
even-even	1764332
odd-even	1307899
even-odd	998656
odd-odd	184522

Table 4.12: Evaluated Libraries Versions.

Library	**Version**
Evaluated Nuclear Data File (ENDF)	B-VIII.0
TALYS Evaluated Nuclear Data Library (TENDL)	2019
Joint Evaluated Fission and Fusion (JEFF)	3.3
Japanese Evaluated Nuclear Data File (JENDL)	4.0

Other characteristics incorporated into the dataset included the fraction of nucleons in the surface (1/R), the packing fraction, the level density (using fermi's gas model), and the geometric elastic scattering cross section ($4\pi R^2$).

4.4 Evaluated Libraries

Different evaluations for comparison are needed since the evaluation is not a perfect standardized process. It is organization- and evaluator-dependent and there are always disagreements between these. Figure 4.10 shows the ^{99}Tc(n,elastic)^{99}Tc reaction cross section. The ENDF evaluation predicts a higher base for the resonances than the other evaluations. Furthermore, the thermal and fast energy regions are in disagreement between all evaluations. Comparing to only one would introduce bias in the validation process.

These regional libraries often need to be processed further into a format suitable for application codes. For example, the ENDF library is often processed with tools like NJOY and AMPX into a format suitable for transport codes like MCNP, SERPENT2, and SCALE. Once in the appropriate format, these libraries can be tested on the validation step using integral benchmarks (i.e. critical assemblies). The validation step is used to verify that the generated nuclear data performs as expected. As a measure of current predicting power, various evaluated libraries were downloaded. These provide a benchmark for our ML models to compare to. Table 4.12 shows the versions of the libraries used in this work. The JEFF evaluation contains mostly fission data and was therefore only use when applicable.

Not only was the ENDF evaluation use for comparison but the same energy grid was used to query cross section from the ML models in addition to the energy points of the experimental datapoints. This was performed to be able to calculate errors relative to ENDF and ML respectively.

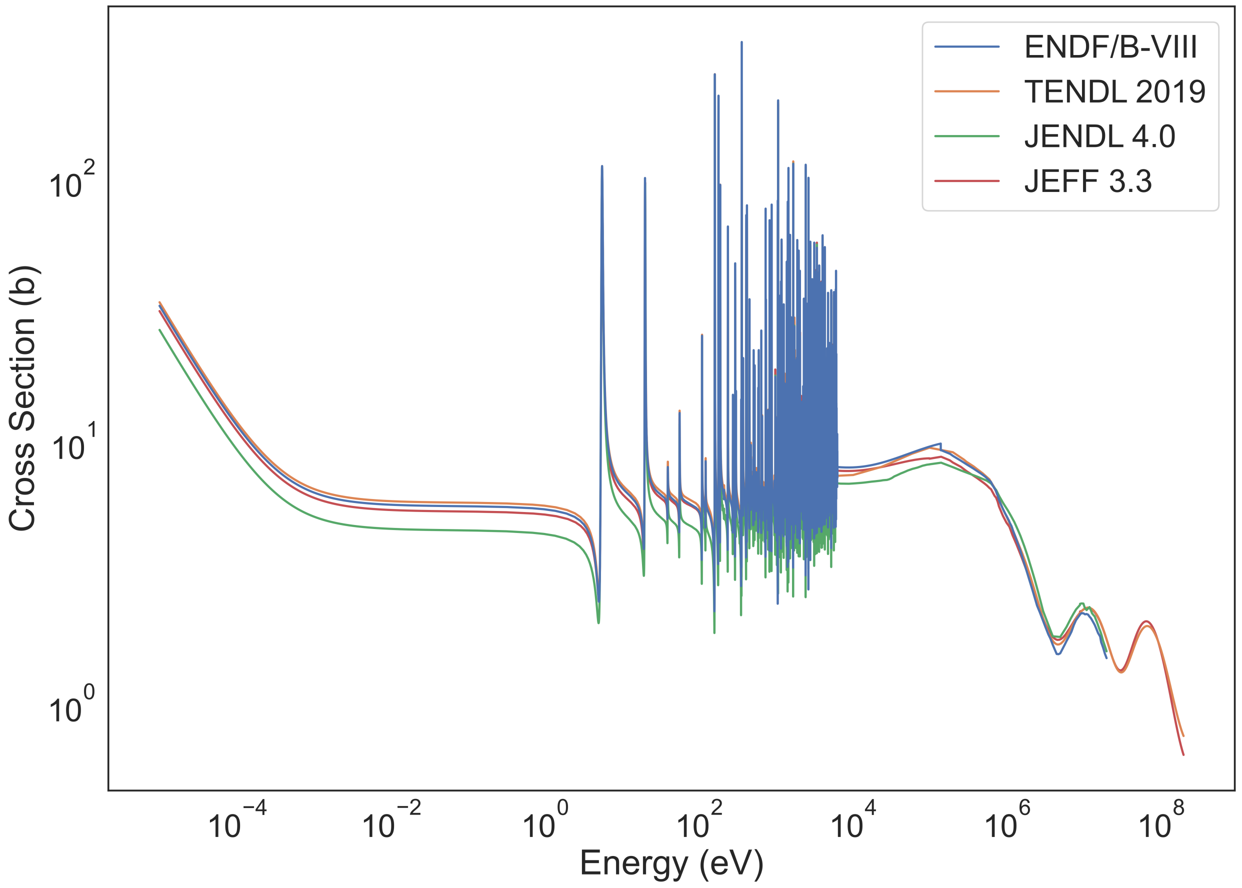


Figure 4.10: ^{99}Tc(n,elastic)^{99}Tc reaction cross section.

4.5 Feature Processing and Data Preparation

In this section, we give an overview of the dataset transformations and standardization procedures which are important for model optimization and performance. As mentioned previously, the original EXFOR dataset consists of approximately six million experimental data points. The general goal focuses on providing validated cross sections for isotopic and naturally occurring samples. Due to the restrictions on data representation, which are discussed in later sections, all molecular data points were dropped. These include light and heavy water measurements.

Scalers, Transformers, and Normalizers

The optimization of machine learning algorithms has various aspects, amongst which is data representation. The majority of the ML algorithms can behave unexpectedly when features are not transformed correctly. The general practice is to standardize or normalize the data, however, the selected method is often model- and challenge-dependent. Algorithms like SVM assume features are centered around zero and have a variance of the same order. Similarly, Nearest Neighbors and other dimensionality-distance-driven algorithms assume higher variance features are more important. In other words, larger variance features will dominate the objective function and the model will be unable to learn from lower magnitude variance features. Consequently, different scalers and transformers were tested. In all cases, the normalization technique was fitted on the training set of the data. The parameters obtained were used to transform the validation and test set.

Standard Scaler

Standardization is one of the most popular normalization technique (Eq. 4.4). This transformation gives the feature x the properties of a standard normal distribution with $\mu = 0$ and $\sigma = 1$ where μ is the mean and σ is the standard deviation from the mean. This prevents the high-low variance issue that many ML models present. This normalization is applied feature-wise meaning each has its own set of parameters calculated during the scalers fitting.

$$x^{'} = \frac{x - \bar{x}}{\sigma} \tag{4.4}$$

Min-Max Scaler and Max-Abs Scaler

An alternative to standardization is the min-max normalizer (Eq. 4.6 which scales all features (X) between a given minimum (min) and maximum (max). The range is often between zero and one. This scaler provides robustness to very small standard deviations of features and preserves dataset sparseness.

$$x_{std} = \frac{X - X.min()}{X.max() - X.min()} \tag{4.5}$$

$$x_{scaled} = X_{std} * (max - min) + min. \tag{4.6}$$

The max-abs scaler is a similar normalizer but transforms the data to lie in the [-1, 1] range. It is suited for data that is already centered at zero and/or for sparse data.

Robust Scaler

In datasets where the number of outliers is suspected to be high, normalizing using data-dependent parameters is not recommended since mean and variances are heavily impacted. The robust scaler uses more robust estimates for the center and range of the data. Specifically, it removes the median and scales the data according to the interquartile range. The interquartile range is the range between the 1st and 3rd quartile. Similar to other scalers, the median and interquartile range parameters are stored for later use.

Power Transformer

Highly positively or negatively skewed features can heavily impact the training in many ML algorithms. The tail region of skewed features may act as an outlier for a statistical model, especially regression-based. Although there are models that are robust to skewness and outliers, like Tree-based models, this type of data limit the possibility to try other ML-models effectively. A common solution consists of transforming skewed data into a Gaussian- or Normal-like distribution using power transformers. In this work, the Yeo-Johnson transformer (Eq. 4.7) was tested, a parametric-monotonic transform function that aims to preserve the rank of the values along with each feature. This method finds an optimal parameter (λ) that minimizes skewness through maximum likelihood estimation. Once fitted to the training set, the same parameters were used to transform the testing and validation set.

$$x_i^{(\lambda)} = \begin{cases} -\frac{[(-x_i+1)^{2-\lambda}-1]}{(2-\lambda)} & \text{if } \lambda \neq 2, x_i < 0, \\ \frac{[(x_i+1)^{\lambda}-1]}{\lambda} & \text{if } \lambda \neq 0, x_i \geq 0, \\ \ln(x_i+1) & \text{if } \lambda = 0, x_i \geq 0 \\ -\ln(-x_i+1) & \text{if } \lambda = 2, x_i < 0 \end{cases} \tag{4.7}$$

Natural Data in the Atomic Mass Evaluation

As mentioned, the EXFOR database does not only contain pure isotopic targets but also naturally available samples.. Since the AME library only contains isotopic information, one entry per element was created to represent natural element data. The atomic masses were filled using generally accepted values.

Other features in the AME, including Q-values and separation energies, are calculated using isotopic information. Leaving empty values for these features in the newly created entries for natural data is not an option due to the abundance of natural sample data in EXFOR and the formatting requirements for many ML algorithms. Two options exist: (1) drop all the natural target data in EXFOR or (2) impute these properties using isotopic information for each element. The former means losing valuable data while the latter creates uncertainty. The latter approach was taken to avoid the loss of information. Data for the newly created entries were calculated by linear interpolation using all the available isotopes for each natural element. This approach is not meant to yield highly accurate information, but to provide acceptable imputed values that captures the general trend. Figure 4.11 depicts the imputed values for natural uranium and also for missing isotopic data. For any other missing values, a value of zero was set.

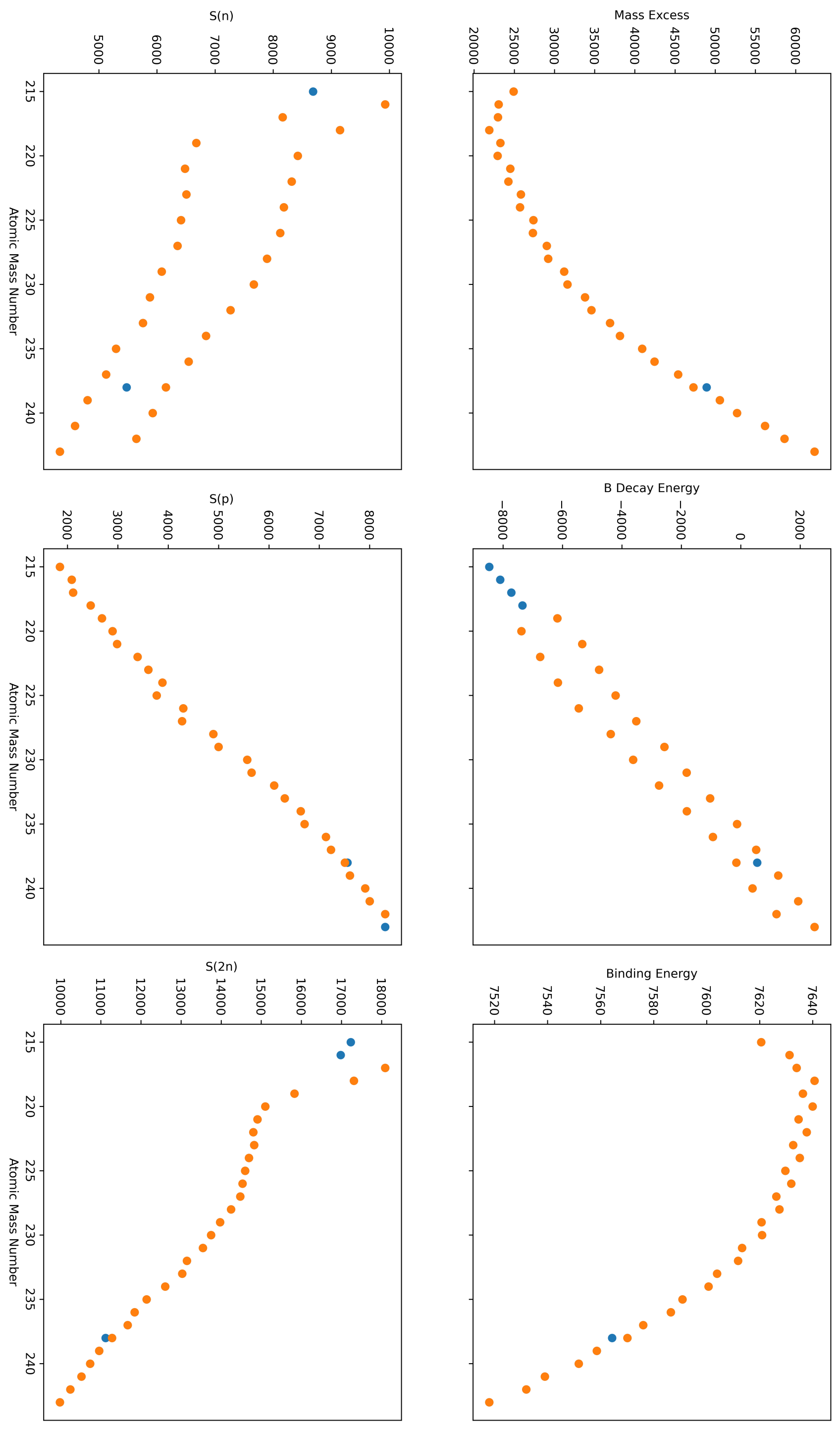


Figure 4.11: AME isotopic data (orange) for all Uranium isotopes. Blue points signify interpolated data.

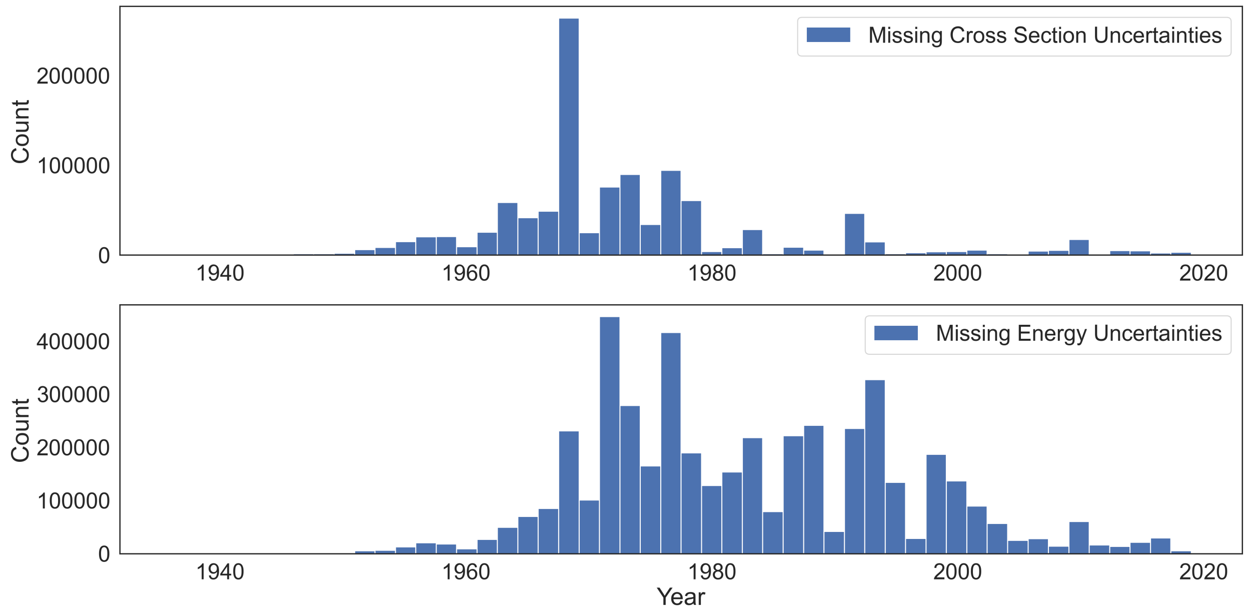


Figure 4.12: Missing Uncertainties as a function of the Experimental Campaign's Year.

Energy and Cross Section

As mentioned earlier, the cross section is directly dependent on the energy and it is therefore important for the model to process these features in the best way possible. A big portion of the measurements in EXFOR was performed around the thermal energy region (0.023 eV). Machine learning algorithms, especially neural networks suffer from vanishing and exploding gradient problems where the multiplication of many small numbers causes gradients to default to zero and exploding gradient where the multiplication of many big numbers causes gradients to numerically explode. In this particular challenge, numerical instabilities quickly arise due to the small numbers used to represents the energy. Converting the energy from the eV to MeV will give rise to the opposing issue of exploding gradients. To address this the logarithm transformation was applied to both the Energy and the Cross Section. While the variance has been minimized, it is still big enough to prompt for the application of an additional normalizer such as those described in previous sections.

Uncertainty

While it is of interest to take into account and calculate the uncertainty in the ML-model predictions, it is not possible to do so accurately given the current state of EXFOR. Of the 4.5 million data points, approximately 88% and 17% of the energy and cross section uncertainty values respectively were either not reported, missing, or documented in another format not easily accessible. Figure 4.12 shows a histogram of missing uncertainty values as a function

Table 4.13: Dataset Split.

Split	**Datapoints**
Train	3347293
Validation	418411
Test	418411

of the year the experimental campaign was undertaken. Evidently, many experiments from the 60s to the 80s did not report cross section uncertainties. For the Energy, the matter is worse. Even today, the uncertainty in energy is not appropriately reported. This may not be an artifact of the experimental campaign but of the EXFOR database.

This makes it harder for the model to accurately infer or propagate uncertainties since not enough accurate information to train on is available. In an attempt to partially solve this issue an imputing framework was created.

1. First, the uncertainty as a percentage was obtained for each entry when available.
2. The dataset was grouped by reaction channel and the missing uncertainty values were filled by the groups mean. For example, all (n,total) data points were extracted and the average uncertainty percentage among all these points was used to fill missing values for other data points of the same reaction type.
3. Persistent missing values were filled similarly only this time grouped by institutions.
4. The third round of imputation was performed using the mean grouped by isotope.
5. Any remaining missing uncertainty values were filled by the dataset mean.

In a second version of the dataset, both the uncertainty in energy and in cross section were dropped and not used.

Categorical Variables

All features marked as categorical in both Table 4.1 and Table 4.10 were one-hot encoded. These were therefore not subject to any transformer or normalizer to preserve sparsity.

Splitting the Dataset: Train, Test, and Validation

Previous to any further transformations the dataset was split into a training, validation, and testing set. It is common practice to split a dataset in 60-20-20 or 70-15-15 but for datasets where samples are upwards of a million, it is becoming common practice to split the data in 80-10-10. This latter approach was taken (see Table 4.13).

Table 4.14: EXFOR and AME features skewness.

Feature	Calculated Skewness
ELV/HL	724.315247
Energy	689.584529
Data	139.742977
Neutron Nucleus Radius Ratio	2.652843
S(2n)	1.001953
S(p)	1.001594
Q(p,2p)	-1.001594
Q(g,p)	-1.001594
Q(n,np)	-1.001594
Q(n,d)	-1.001668
Q(d,3He)	-1.001710
Q(t,a)	-1.001902
Q(g,2n)	-1.001953
Q(p,t)	-1.002031
Q(p,2n)	-1.593202
Q(B-n)	-1.593206
Q(4B-)	-1.677550
Q(ep)	-1.696255
Q(n,2p)	-1.711368
Q(2B-)	-1.715373
Beta Decay Energy	-1.924420
Q(p,n)	-1.924420
Q(3He,t)	-1.924420
Binding Energy	-3.828846

Numerical Features

The numerical features in the training set were standardized/normalized using one of the described scalers/transformers mentioned in Section 4.5. The choice of scaler will be important, particularly in this challenge where the majority of the features have a certain degree of skewness and high variance. Table 4.14 shows the calculated skewness using the Fisher-Pearson coefficients. For normally distributed data the value should be zero. Values greater than zero represent more weight in the right tail and vice-versa. All features not listed have a skewness value between plus and minus one.

Reaction Channel (MT)

How the reaction channel is represented in the dataset will play an important role in the designed ML-base solutions. It can be treated either as a categorical or a numerical feature.

Treating it as a categorical variable is a simple and elegant solution but comes with some disadvantages. The solution will be limited to reactions that were only available at the time of training. Adding other reaction channels will not be possible. For example, the (n,n+3a) reaction channel is not included in the training set, however, it is part of the listed reactions in ENDF. Similarly there are many others not included like (n,3na), (n,2n+2a), (n,n+3He), and (n,n+d+2a). Still, the one-hot encoding representation might suffice for inferences relevant to power reactor applications since rare reactions do not play an important role.

An alternative representation might be needed for other applications and to better represent the interaction physics. In this work, one numerical approach was tested. First, a feature is created for every possible outgoing particle, one for a neutron, a proton, a gamma, and so on. For the (n,2n) reaction, for example, the number two will be placed in the outgoing neutrons feature. Similarly, for the (n,3n) and (n,4n) reactions, a three and a four are placed respectively in the outgoing neutron feature. For (n,g), a one is placed under the outgoing gammas feature. In this approach, one important issue arises. Multi-nuclei outgoing particles like alpha and tritons also need to be represented. Two options exist. The first option consists of also creating an outgoing feature for these particles. Alternatively, these particles need to be decomposed into their individual outgoing nuclei. In the latter approach, a new feature indicating nuclei binding might be appropriate, else distinguishing between multi-nuclei and single multiple nuclei particles will not be possible. Data representation issues will always arise. There are other reactions like (n,n+a+d) where the number of outgoing particles is not one but multiple. The representation is limited and this reaction cannot be included.

Decisions need to be made on a goal-basis on how the data will be represented. Whatever approach is taken, the limitations need to be understood throughout not only training but also production. One big advantage of the numerical representation is the possible extrapolation of knowledge to other outgoing particles that have not been considered. An example scheme of a numerical representation can be seen in Table 4.15.In this work both, numerical and categorical approaches, were tested.

Specific Filtering

A filtering framework was written to discard several datapoints that were deemed inconsistent. Measurements dropped included:

- Molecular data points such as those concerning heavy and light water measurements.
- Datapoints with the tag "DERIV".
- Measurements with energy or cross section reported as zero.
- Experiments with MT values of 1003, 2103, 1108, and 203. These amounted to a total of 147 data points.

Table 4.15: Numerical Reaction Channel (MT) Representation.

Reaction	out_neutron	out_proton	out_alpha	out_gamma	out_fission	out_triton	out_helion	out_deuteron
(n,total)	0	0	0	0	0	0	0	0
(n,elastic)	1	0	0	0	0	0	0	0
(n,nonelastic)	1	0	0	0	0	0	0	0
(n,inelastic)	1	0	0	0	0	0	0	0
(n,2n)	2	0	0	0	0	0	0	0
(n,3n)	3	0	0	0	0	0	0	0
(n,fission)	0	0	0	0	1	0	0	0
(n,n+a)	1	0	1	0	0	0	0	0
(n,2n+a)	2	0	1	0	0	0	0	0
(n,n+p)	1	1	0	0	0	0	0	0
(n,n+2a)	1	0	2	0	0	0	0	0
(n,n+d)	1	0	0	0	0	0	0	1
(n,n+t)	1	0	0	0	0	1	0	0
(n,4n)	4	0	0	0	0	0	0	0
(n,2n+p)	2	1	0	0	0	0	0	0
(n,n1)	1	0	0	0	0	0	0	0
(n,disap)	0	0	0	0	0	0	0	0
(n,g)	0	0	0	1	0	0	0	0
(n,p)	0	1	0	0	0	0	0	0
(n,d)	0	0	0	0	0	0	0	1
(n,t)	0	0	0	0	0	1	0	0
(n,helion)	0	0	0	0	0	0	1	0
(n,a)	0	0	1	0	0	0	0	0
(n,2a)	0	0	2	0	0	0	0	0
(n,2p)	0	2	0	0	0	0	0	0
(n,p+a)	0	1	1	0	0	0	0	0
(n,t+2a)	0	0	2	0	0	1	0	0
(n,5n)	5	0	0	0	0	0	0	0
(n,6n)	6	0	0	0	0	0	0	0
(n,t+a)	0	0	1	0	0	1	0	0
(n,n+a+d)	1	0	1	0	0	0	0	1
(n,2n+p+a)	2	1	1	0	0	0	0	0
(n,7n)	7	0	0	0	0	0	0	0
(n,8n)	8	0	0	0	0	0	0	0

4.6 Data Sets

In order to test the usefulness of new features, three different datasets where compiled. These will be referred to in the following sections by their number.

- Dataset A: "Energy", "Data", "Z", "N", "A", "MT", "Center of Mass Flag", "Element Flag"
- Dataset B: Dataset 0 Features, "Atomic Mass Micro", "Nucleus Radius", "Neutron Nucleus Radius Ratio"
- Dataset C: Dataset 1 Features, "Mass Excess", "Binding Energy", "B Decay Energy", "S(n)", "S(p)", "S(2n)", "S(2p)"
- Dataset D: Dataset 2 Features, "N valence", "Z valence", "P factor", "N tag", "Z tag", "NZ tag"
- Dataset E: Dataset 3 Features plus all Q-values.

4.7 Criticality Benchmarks

During the last phase of the nuclear data evaluation, it is important to validate our models with benchmarks, especially critcality assemblies. In this section, the benchmarks that were created as part of this work are described which included:

- U233-MET-FAST-001: ^{233}U Jezebel
- U233-MET-FAST-002: ^{235}U Surrounded ^{233}U Core (10kg)
- U233-MET-FAST-002: ^{235}U Surrounded ^{233}U Core (7.6kg)
- PU-MET-FAST-019: Sphere of Plutonium Reflected by Beryllium
- PU-MET-FAST-005: Sphere of Plutonium Reflected by Tungsten

^{233}U Jezebel

The ^{233}Jezebel critical assembly (U233-MET-FAST-001) was an unreflected (bare) near-spherical criticality benchmark operated at the Los Alamos National Laboratory. Two different configurations were tested:

- Configuration A:
  - Mass: 16.556kg
  - Average Density: 18.424g/cm3

Table 4.16: ^{233}U Jezebel Assembly Characteristics.

Description	Value
Core Density	18.424 g/cm3
^{233}U Core Mass	16.535 +- 0.4 wt.% +- 0.066 kg
Core Radius	5.9838 cm

Table 4.17: ^{233}U Jezebel Composition.

Isotope	Density (atoms/barn-cm)
^{233}U	4.6712E-02
^{234}U	5.9026E-04
^{235}U	1.4281E-05
^{238}U	2.8561E-04

  - Supercritical: 25 Cents
  - Calculated Critical Mass: 16.235kg
  - Calculated Average Density: 18.424g/cm3

- Configuration B:
  - Mass: 16.651kg
  - Average Density: 18.424g/cm3
  - Supercritical: 1 Cent
  - Calculated Critical Mass: 16.269kg
  - Calculated Average Density: 18.424g/cm3

Some corrections were performed to come up with an idealized bare sphere of a homogeneous material with composition as described in Table 4.17 and configuration as described in Table 4.16. More information on these and several other model assumptions and derivations can be found in Reference [3].

^{233}U Spheres Surrounded by ^{235}U

Two critical experiments using spherical masses of uranium highly enriched in ^{233}U surrounded by uranium highly enriched in ^{235}U were undertaken at Los Alamos Scientific Laboratory using the Planet universal assembly.

The "10kg Experiment" and the "7.6kg Experiment" contained 10.012kg and 7.601kg of core mass respectively. Both used a ^{210}PoBe neutron source with a strength of approximately 10^5 neutrons/sec. The core density for the "10" and "7" kg experiments was 18.621 and

Table 4.18: Model Description and Dimensions.

The 10kg Experiment	
Description	Value
Core Density	18.621 g/cm3
Core Mass	10012 grams Uranium
Core Radius	5.0444 cm
Surrounding Uranium Shell Thickness	1.2217 +- 1%
Surrounding Uranium Shell Density	18.8 g/cm3
Configuration Outer Radius	6.2661 cm
The 7.6kg Experiment	
Description	Value
Core Density	18.644 g/cm3
Core Mass	7601 grams Uranium
Core Radius	4.5999 cm
Surrounding Uranium Shell Thickness	1.9888 +- 1%
Surrounding Uranium Shell Density	18.8 g/cm3
Configuration Outer Radius	6.5887 cm

Table 4.19: Atom Density for the Uranium Spheres.

^{233}U **Core**		
Isotope	10kg Experiment (atoms/barn-cm)	7.6kg Experiment (atoms/barn-cm)
^{233}U	$4.7253 * 10^{-2}$	$4.7312 * 10^{-2}$
^{234}U	$5.2705 * 10^{-4}$	$5.2770 * 10^{-4}$
^{238}U	$3.2975 * 10^{-4}$	$3.3015 * 10^{-4}$
Surrounding ^{235}U **Sphere**		
Isotope	10kg Experiment (atoms/barn-cm)	7.6kg Experiment (atoms/barn-cm)
^{235}U	$4.4892 * 10^{-2}$	$4.4892 * 10^{-2}$
^{238}U	$3.2340 * 10^{-3}$	$3.2340 * 10^{-3}$

18.644 g/cm3 respectively while the surrounding ^{235}U sphere's density was the same for both experiments (18.8 g/cm3).

Based on data collected throughout the benchmark experiment, an idealized model was derived for both the "10" and "7" kg experiments. The model dimensions and characteristics are described in Table 4.18. The core and surrounding Uranium atomic number densities are given in Table 4.19. More information on these and several model assumptions and derivations can be found in Reference [3]

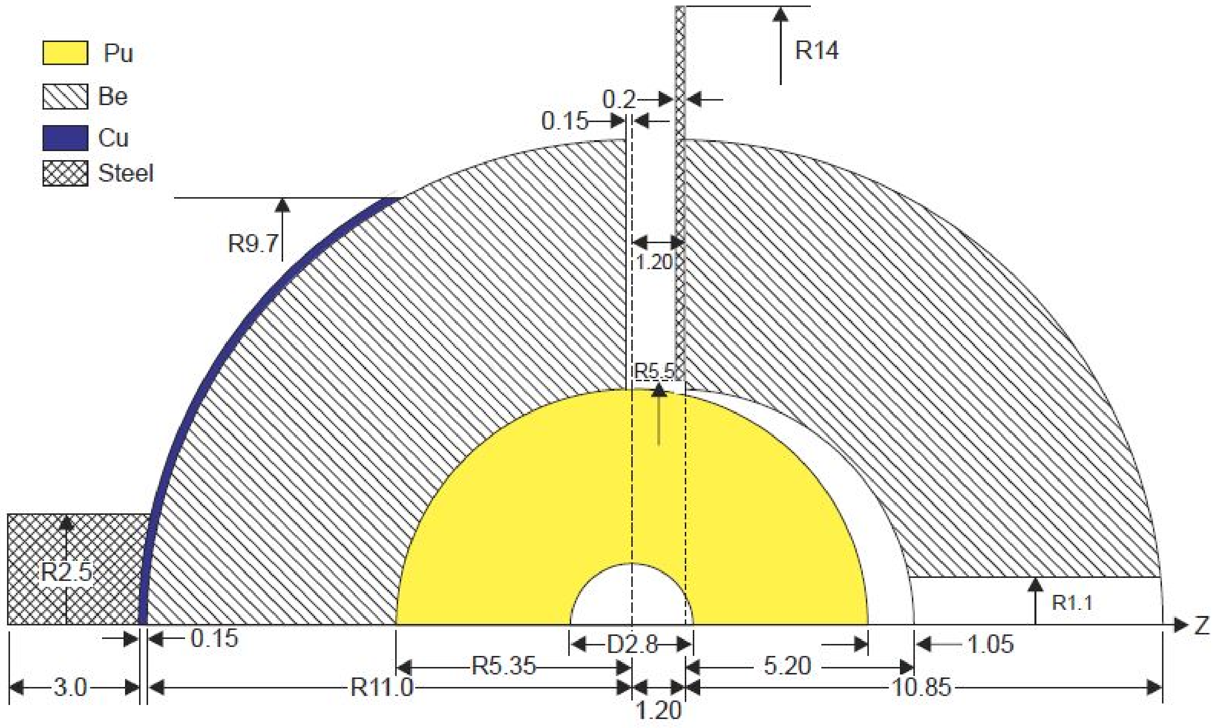


Figure 4.13: Beryllium-reflected Plutonium Core Criticality Benchmark Computational Model. Image taken from Reference [3].

Sphere of Plutonium Reflected By Beryllium

Undertaken using the VNIITF Criticality Test Facility, this assembly consisted of a sphere of plutonium reflected by beryllium. While the original geometry is quite complex some assumptions and corrections were made during the criticality benchmark evaluation. The homogenized model can be seen in Figure 4.13. More information on these and several model assumptions and derivations can be found in Reference X.

The inner core and outer radii are 1.4 cm and 5.35 cm. The reflector inner and outer radii are 5.35 cm and 11.0 cm. The plutonium atom densities used can be found in Table 4.20 along with the beryllium shells values as well. Since delayed critical was not obtained the experimental k_{eff} was less than 1.00. The benchmark model k_{eff} is 0.9992 +- 0.0015.

Plutonium Sphere Reflected by Tungsten

Performed at Los Alamos Scientific Laboratory using the Planet universal assembly machine, a slightly subcritical spherical mass of delta-phase plutonium reflected by tungsten was built and measured. The core was composed of two hemispheres of delta-phase plutonium alloy

Table 4.20: Plutonium Core and Beryllium Shell Compositions.

Plutonium Core	
Isotope	Atom Densities (atoms/barn-cm)
^{239}Pu	$4.7253 * 10^{-2}$
^{240}Pu	$5.2705 * 10^{-4}$
^{241}Pu	$3.2975 * 10^{-4}$
Ga	$2.2105 * 10^{-3}$
C	$3.0246 * 10^{-4}$
Fe	$3.2525 * 10^{-4}$
W	$7.4100 * 10^{-5}$
Ni	$1.4187 * 10^{-3}$
Surrounding ^{235}U Sphere	
Isotope	Atom Densities (atoms/barn-cm)
Be	$1.2081 * 10^{-1}$
O	$8.2064 * 10^{-5}$
C	$1.0020 * 10^{-4}$
Fe	$5.0939 * 10^{-5}$

Table 4.21: Model Description and Dimensions.

Description	Value
Core Density	15.778 g/cm3
Core Mass	8471 grams Plutonium Alloy
Core Radius	5.0419 cm
Surrounding Tungsten Shell Thickness	4.699 +- 1 cm%
Surrounding Tungsten Shell Density	17.21 g/cm3
Configuration Outer Radius	9.7409 cm

having a diameter of 3.970 inches with a 0.85-inch diameter source cavity at the center. For the tungsten reflector, several thicknesses were constructed to enclose the plutonium hemispheres. A ^{210}PoBe neutron source with a strength of approximately 10^5 neutrons/sec was used.

Based on data collected throughout the benchmark experiment, an idealized model was derived. The model dimensions and characteristics are described in Table 4.21. The core and surrounding geometries atomic number densities are given in Table 4.22. More information on these and several model assumptions and derivations can be found in Reference [3].

Table 4.22: Tungsten-Reflected Plutonium Sphere Composition.

Plutonium Alloy Core	
Isotope	Atom Densities (atoms/barn-cm)
^{239}Pu	$3.7291 * 10^{-2}$
^{240}Pu	$1.9277 * 10^{-3}$
^{241}Pu	$1.2196 * 10^{-4}$
Ga	$1.3628 * 10^{-3}$
Surrounding Tungsten Sphere	
Isotope	Atom Densities (atoms/barn-cm)
W	$5.1468 * 10^{-2}$
Ni	$9.7124 * 10^{-3}$
Cu	$4.0774 * 10^{-3}$
Zr	$7.9528 * 10^{-4}$

4.8 Summary

Implementing and designing data pipelines is important for future data incorporation into any nuclear data-driven effort. In this section, an overview of the EXFOR, AME, RIPL, and ENDF databases relevant to this work was described. The EXFOR library is the main source of neutron-induced reaction data. Approximately 4.5 million data points were gathered and formatted for reaction modeling. Additionally, isotopic information including Q-values, Beta-decay energies, precise atomic masses, and more were gathered from the Atomic Mass Evaluation. To take advantage of natural target experimental campaigns in EXFOR, AME information was used to impute natural target information. RIPL contains isotopic energy level information. Resonances in cross sections are directly related to energy level location. To take advantage of the information in XUNDL, the RIPL cut-off parameters were used to obtain a limited dataset of nuclear levels. Linear interpolation was then used to calculate nuclear level densities at the appropriate experimental energies from EXFOR. Information from EXFOR, AME, and RIPL was eventually joined to build 5 different datasets with different features. To validate models trained on the gathered data, several criticality benchmarks were used. These will also allow the models to be evaluated at different energy ranges and using different materials.

Chapter 5

Machine Learning Algorithms

Having explored and transformed the data, the machine learning model training (evaluation) phase can begin. This challenge falls under the supervised model regime meaning our dataset has labels. The labels are cross sections, a continuous numerical value making it a regression-based supervised challenge.

In general, a model is a mathematical structure that predicts an output $\hat{y}$ based on an input x_{ij} where i represents the row number and j the features. The most popular example is a simple linear regression model where the output is described as $\hat{y} = \sum_j \theta_j x_{ij}$. In all models, the parameters/coefficients are the unknown variables that need to be learned from the training data. In the case of a simple linear model, the parameters are the weights θ_j. There are a variety of regression algorithms including K-Nearest Neighbors (KNN), Linear and Logistic Regression, Support Vector Machines (SVM), Decision Trees (DT), Random Forests (RF), Gradient Boosting Machines (GBM), Neural Networks (NN), etc. The true goal of any ML algorithm is a generalization, meaning adequate performance when deployed in real-world conditions. The model selection will depend on many factors including the type and complexity of the problem. As the complexity increases a more powerful model is needed but also more data is required for training to prevent overfitting.

In this chapter, a very brief overview of the KNN, DT, GBM, and NN algorithm implementations used in this work is given. In each section, the hyperparameter optimization task is explained and the final parameters are presented. Further details on all these models can be found in References [8][14][6]. For both the KNN and DT models, the Scikit-Learn implementations were used [16]. For the GBM and NN, the XGBoost and TensorFlow implementations were used respectively [6][13].

5.1 K-Nearest-Neighbors (KNN)

The KNN regression algorithm calculates a new data point's value by first querying the K nearest data points available in the training set and sorting them by increasing distance, hence the model's name. Here, K is a hyperparameter that has to be set before training.

There are a variety of available distance metrics that can be used to measure the distance between points in the i dimensional space where i is the number of training features including the Euclidean and Manhattan distance. The Euclidean distance is calculated between points p and x by taking the square root of the squared difference between $(p_1, p_2, ..., p_n)$ and $(x_1, x_2, ..., x_n)$ (Eq. 5.1). The Manhattan distance is calculated as the absolute difference between points x and p (Eq. 5.2).

$$d(p, q) = \sqrt{\sum_{i=1}^{n} (p_i - x_i)^2}. \tag{5.1}$$

$$d(p, q) = \sum_{i=1}^{n} |p_i - x_i|. \tag{5.2}$$

By default, the weight of neighboring points is uniform and the value of a new data point is calculated simply as the mean of the K nearest points (Eq. 5.3).

$$\text{New Point} = \frac{1}{K} \sum_{i}^{K} k_i. \tag{5.3}$$

Alternatively, the K nearest points can be weighted according to distance, meaning the closest points have a heavier influence on the new data point's value (they are weighted by the inverse of their distance). The model performance will not only depend on the chosen distance metric and weight function but also on the variance and the scales/magnitudes of the dataset features. In general, features with higher variance and higher magnitudes will tend to be given preference by distance-drive models. Important information in other features might be missed if the dataset is not correctly transformed and normalized.

There are many types of nearest neighbor algorithms including brute force, k-d tree, and ball tree. The most accurate, but also the most computationally expensive, is the brute force algorithm. In contrast to other models, the training stage of the KNN brute-force algorithm consists of computing the distances between all points in the dataset. As the number of samples (N) and the number of dimensions (D) increases, so does the complexity by $O[DN^2]$. Consequently, the application of this algorithm becomes unfeasible for large datasets.

A trained model will store the multi-dimensional feature vector and the corresponding label for each of the training samples. The actual query and distance computation takes place once an inference is requested. Even if an optimal k hyperparameter is found, care must be taken not to incorporate irrelevant or extra features as it can have a significant impact on the model's accuracy while increasing complexity. In the case of high-dimensional data, it is worth exploring dimensionality reduction techniques (DRT). These can help avoid the

Table 5.1: Experimented K-Nearest-Neighbor Model Parameters.

Parameter	Values Tested
Number of Neighbors (K)	1-20
Weight Function	Distance
Algorithm	Brute
Distance Metric	Manhattan, Euclidean
Normalizer/Standardizer	SS, QN, RS, MAXABS, MINMAX, PT

curse of dimensionality which states that as the dimensions increase, the euclidean distance cannot be reliably used as all vectors become almost equidistant to the search query vector. The most popular DRT's is Principal Component Analysis (PCA).

Although a very simple algorithm, KNN has been successfully applied to a variety of challenges including satellite imagery classification and fraud detection. Simple models also allow studying the characteristics of a dataset with less interference. A major advantage is the non-parametric nature of the algorithm. This makes it suitable for problems where decision boundaries are not well defined. The range of parameters tested in this work is specified in Table 5.1.

Training

While the size of the EXFOR dataset is large, brute-force calculations are still feasible. Even though the training and inferences time will be large, the challenge's objective does not carry a time constraint. This makes the training scenarios simpler by limiting the number of hyperparameters to the number of neighbors (K), the distance metric, and the weight function. Just by observing a cross section plot, it is evident that the weight function needs to be a function of the distance. Datapoints closer to the query energy should have a higher influence than data points farther away.

Before training the model with Datasets B-E, the Dataset A was used to test the impact of different scalers and normalizers in generalization performance. Figure 5.1 shows the validation MAE for several models trained with different scalers. From this information, it is evident that the Min-Max scaler generally achieved better performance.

All these normalization objects were fitted to all but the Energy feature. The only process applied to the Energy was a logarithmic transformation. To investigate the effect of Energy normalization, the Min-Max transformer was applied sequentially after the log transformation in a set of training iterations and compared to the un-transformed results. The validation MAE of these training processes can be observed in Figure 5.2. Based on these results, the Min-Max scaler was chosen and the Energy was included in the scaled features in addition to the logarithmic transformation. Similarly, to choose the optimal distance metric, several models were trained using either the euclidean or Manhattan distance. Figure 5.2 also shows the effect of both on the Validation MAE. The best distance metric is less obvious,

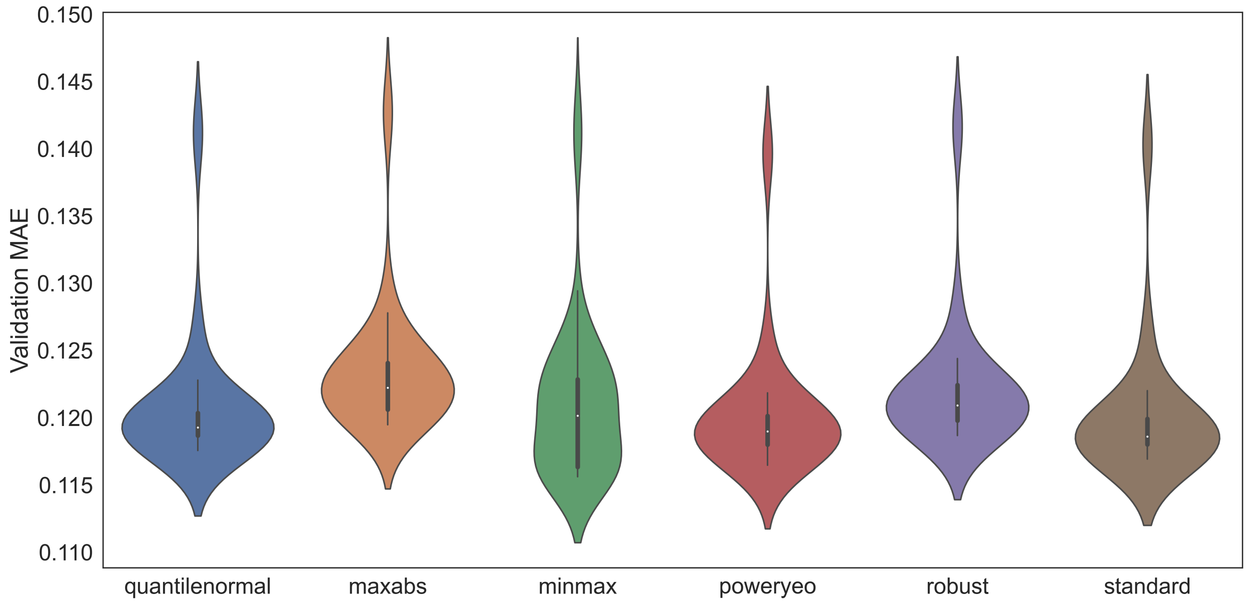


Figure 5.1: Effect of Different Normalizer/Transformer on the Validation MAE.

however, in general, the manhattan distance seemed to achieve better overall performance and was therefore chosen for all other models.

For model selection, the best performing training iterations in terms of validation performance were chosen. Since KNN are not loss-driven algorithms, training cannot be visualized as usual (loss vs time step). Instead, we can visualize a set of similar models by comparing the effect of K on the overall validation MAE. Since the Brut Force approach was chosen, other hyperparameters are not valid. Figures 5.3 represents the mean absolute error (MAE), for the Train, Test, and Validation sets, as a function of the parameter K (number of nearest neighbors) for all models fitted to datasets A.

In this case, it can be observed that at $k = 9$ both the validation and test MAE starts to increase and deviate from a downwards trend. This, coupled with a steady decrease in train MAE, indicates overfitting. Based on this information, the model with $k = 9$ was selected for Dataset A. Similarly, the best models were selected for the other modeling scenarios with Datasets B-E. Table 5.2 summarizes the performance metrics for these models. Appendix A contains the training curves for all other models fitted to Datasets B through E.

There are just so many combinations of parameters that can be tested using KNN algorithms. Performance and learning limits quickly reach a plateau translating into constrain predicting power. Still, in the Results section, we will explore how these models can perform adequately in terms of cross section inferences and benchmark performance.

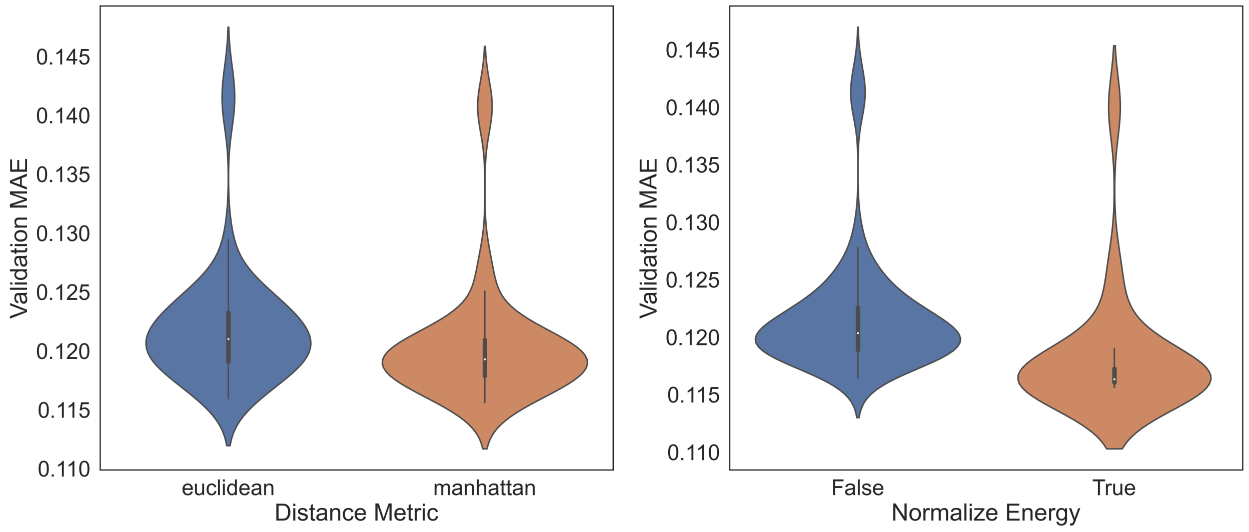


Figure 5.2: Effect of the "Distance Metric" (left) and "Energy" Min-Max Normalization (right) on the Validation MAE.

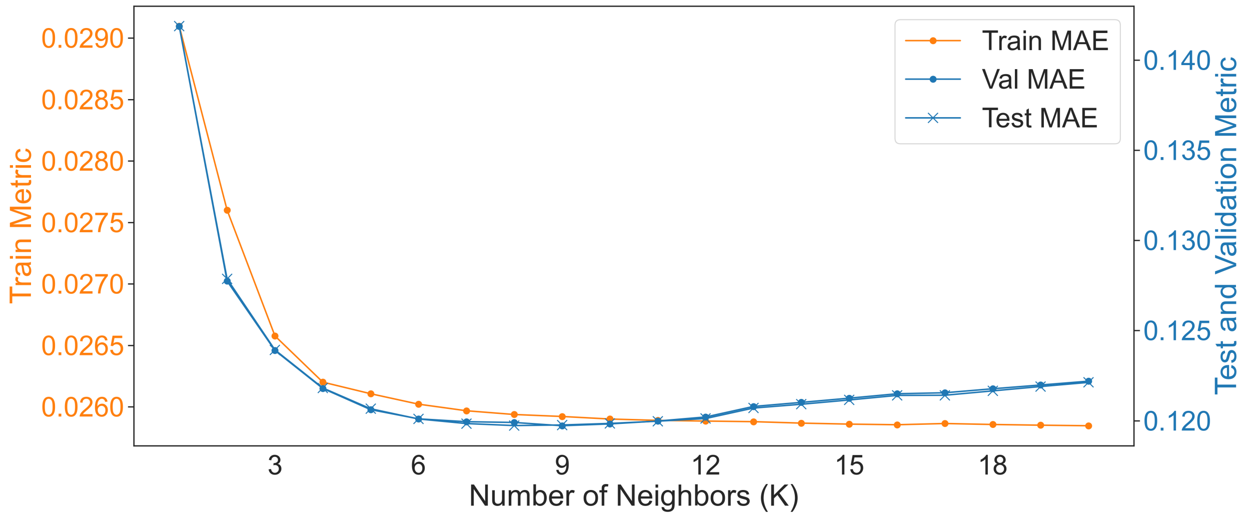


Figure 5.3: Performance Metrics as a Function of the k Hyperparameter for KNN Models Trained on Dataset A.

Table 5.2: Final Set of KNN Hyperparameters and Performance Metrics.

Dataset	K Nearest Neighbors	Train MAE	Val MAE	Test MAE
A	9	0.025922	0.119721	0.119763
B	9	0.025921	0.119010	0.118578
C	11	0.025873	0.115434	0.115954
D	13	0.025876	0.115373	0.115669
E	13	0.025852	0.115209	0.116056

5.2 Decision Tree (DT)

Decision Tree (DT) models also belong to the family of non-parametric methods and can be applied to both classification and regression tasks. The model approaches both tasks by learning simple if-then-else decision rules from the available features. DT's are therefore white-box models meaning any prediction can be explained by Boolean logic. The deeper the tree is allowed to be built, the more complex the rules become translating into a higher risk of overfitting.

Decision Trees are very powerful and have been used successfully in many applications. Splits are created feature-wise independent of variances and ranges meaning very little data preparation is needed. Additionally, these can handle multi-output problems which means that both the cross section and uncertainties can be predicted. If built appropriately and limiting tree complexity, generalization can be achieved. DTs are built based on the training data and are very sensitive to variations in the data. Training on a slightly different dataset might result in a completely different model. Ensemble trees and gradient boosting trees tackle this issue by building multiple trees.

The training process starts by feeding in the training vectors x_i and the label vector y_i. The decision tree will try's to find an optimal point $\theta = (j, t_m)$ to split the data Q at node m by setting up a threshold t_m for each feature j in such a way that the impurity $G(Q, \theta)$ is minimized (Eq. 5.5). For regression tasks, the impurity criteria to minimize is the Mean Squared Error or the Mean Absolute Error.

$$\theta^* = \operatorname{argmin}_\theta G(Q, \theta) \tag{5.4}$$

$$G(Q, \theta) = \frac{n_{left}}{N_m} H(Q_{left}(\theta)) + \frac{n_{right}}{N_m} H(Q_{right}(\theta)) \tag{5.5}$$

where:

$$H(X_m) = \frac{1}{N_m} \sum_{i \in N_m} (y_i - \bar{y}_m)^2 \tag{5.6}$$

Table 5.3: Experimented Decision Tree Model Parameters.

Parameter	Values Tested
Criterion	Mean Squared Error
Splitter	Best
Max Depth	10-400
Minimum Samples for Split	2-15
Minimum Samples for Leaf	2-15

$$\bar{y}_m = \frac{1}{N_m} \sum_{i \in N_m} y_i \tag{5.7}$$

An advantage of DT relative to KNN is the little data preparation needed (i.e. normalization, one-hot-encoding, etc.). The best splits are found feature-wise independent of each others scales. While there are a variety of DT implementations that are capable of creating trees with multiple branches, the scikit-learn implementation is built around an optimized version of the CART algorithm meaning only binary trees are created through the training process. There are many mechanisms to prevent overfitting in DT's during and post-training. As mentioned, decision trees will create splits (internal nodes) feature-wise to reduce the MSE (impurity). In addition to setting the maximum depth the tree is allowed to have, two other hyperparameters can help prevent overfitting. These are the minimum number of samples required to split (MSS) an internal node and the minimum number of samples required to be a leaf node (MSL). The former is simple, if X amount of samples are not above the MSS then the internal node becomes a leaf. In the latter, the algorithm only allows a leaf to be created if there are MSL numbers of samples minimum. A popular post-training technique is pruning, where the tree is allowed to build as many decisions as possible followed by the fractional removal of some of the last leaf nodes. More information on the DT can be found in reference [14].

Training

For all Decision Tree models trained the "Best" splitter algorithm was used given the size of EXFOR. The training scenarios are more complex relative to the KNN model. There are four main hyperparameters to tune: the training criterion, max depth, MSL, and MSS. Furthermore, these parameters will be dataset-dependent. As mentioned previously, the DT algorithm does not require that data be normalized. Other than taking the logarithm for both the Energy and Cross Section, no scalers or normalizers were applied to the dataset. Datasets A-E were used to train a variety of decision tree models with varying hyperparameters in our effort to find the best model. Table 5.3 shows the hyperparameters tweaked along with their explored ranges.

Figure 5.4 shows the Train and Validation MAE as a function of the Max Depth, the main hyperparameter in terms of model complexity, for all different models trained on Dataset A. As expected, as max depth increases, the easier a low Train MAE is achieved. Even with a low Max Depth of 50, a low enough error is achieved as long as low regularization is applied.

It is difficult to visualize the impact of more than two hyperparameters on the performance metric without 3D visualizations. To tune the decision tree hyperparameters visualizations can help guide future training iterations. Figure 5.5 shows the effect of the Max Depth and the MSS, one of the main DT model regularizers, on the Training MAE. Confirming our earlier statement, Max Depths as low as 50 can achieve performance as long as no regularization is applied. The minimum MSS value allowed is two. This is because no splits can happen with only one value left. As soon as the MSS value is greater than two, the Training MAE increases. Figure 5.6 shows the same hyperparameters as a function of the Validation MAE. The trend here is rather different than that observed in Figure 5.5. Regularization is usually needed for generalization and this trend is evident in Figure 5.6. As the MSS increases, the Validation MAE decreases. The best models appear to be located at Max Depth of 50 and MSS of 10.

As for the MSL, Figure 5.7 shows the effect of the Max Depth and the MSL, the next main regularizer, on the Training MAE. The trend, in this case, is more dramatic. As soon as the MSL increases so does the Training MAE with a steeper slope independent of the Max Depth. As for the effect on the Validation MAE (Figure 5.8), models with low Max Depth and somewhat high MLS are more optimal. The best models appear to be located at Max Depth of 50 and MSL of 6. The same trend is observed in models train on Datasets B-E.

Low MSS and MSL clearly lead to model overfitting when comparing the training and testing metrics. This can also be verified by visualizing the Training and Validation MAE as a function of both the MSS and MSL. Figure 5.9 and 5.10 show the effect of both these hyperparameters on the Training and Validation MAE. Opposite trends are observed. Higher MSS and MSL does lead to better generalization.

With this information and trends in mind, several models were trained using a standard Grid Search algorithm to search for the best combination of hyperparameters. Table 5.4 shows a list of different performance metrics for the best overall models trained in Datasets A-E. It can be observed that the best max depth for all datasets is around 70-80 while the best MSS and MSL are 10 and 7 respectively for all cases.

Decision Tree models can be used to get feature importance's. Per the documentation:

"The importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature. It is also known as the Gini importance."

Figure 5.11 shows feature importances extracted from the best model trained on Dataset A. The Energy, Number of Protons, Number of Neutrons, and Mass Number appear to be some of the most important features. While the Gini Importance for the reaction channels is not high, it does not mean their not important. Feature importances can be misleading for

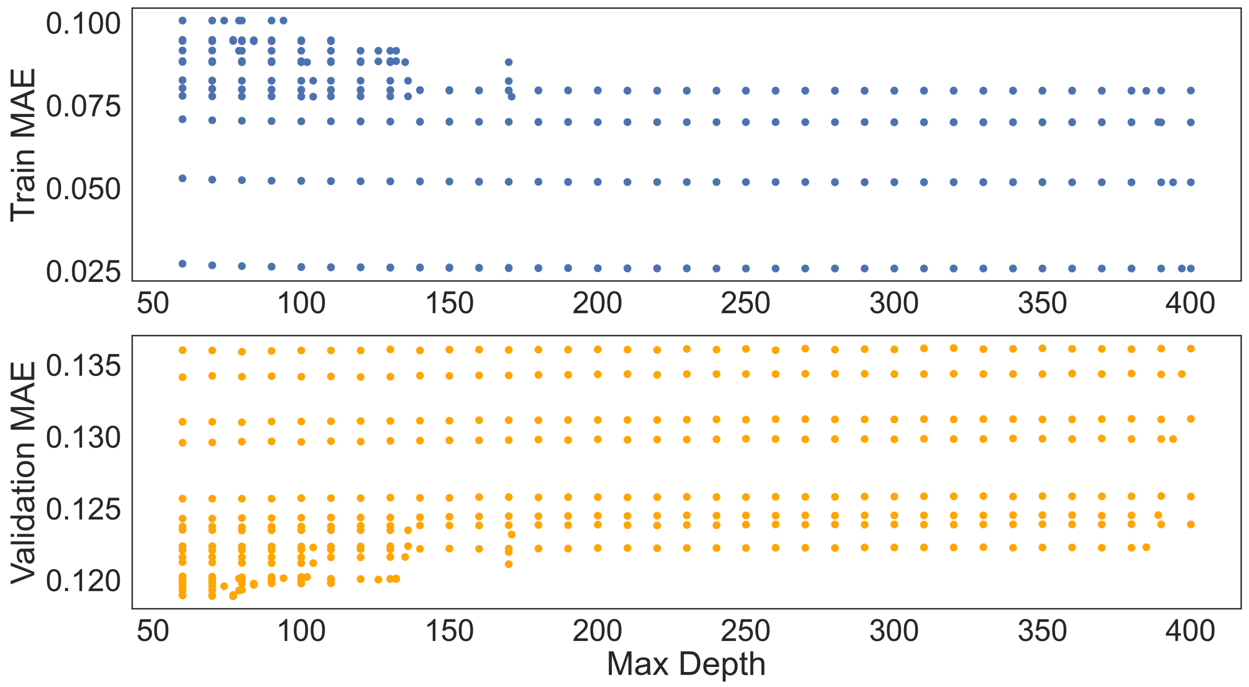


Figure 5.4: Effect of "Max Depth" on the Training and Validation MAE using Dataset A.

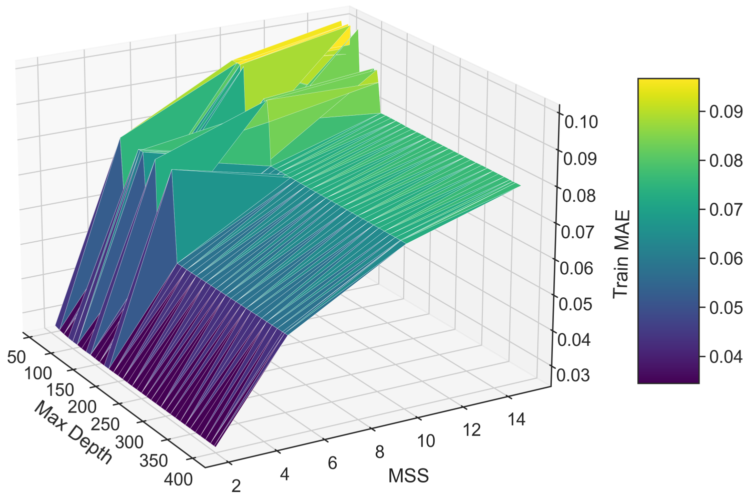


Figure 5.5: Effect of the Max Depth and the MSS on the Training MAE using Dataset A.

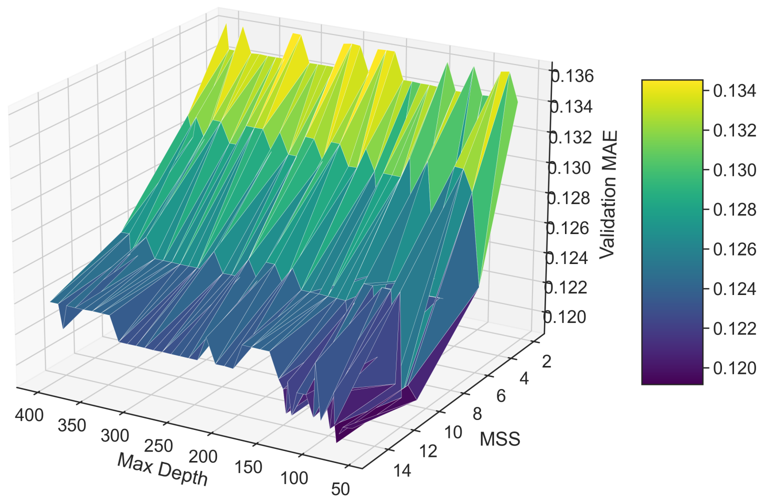


Figure 5.6: Effect of the Max Depth and the MSS on the Validation MAE using Dataset A.

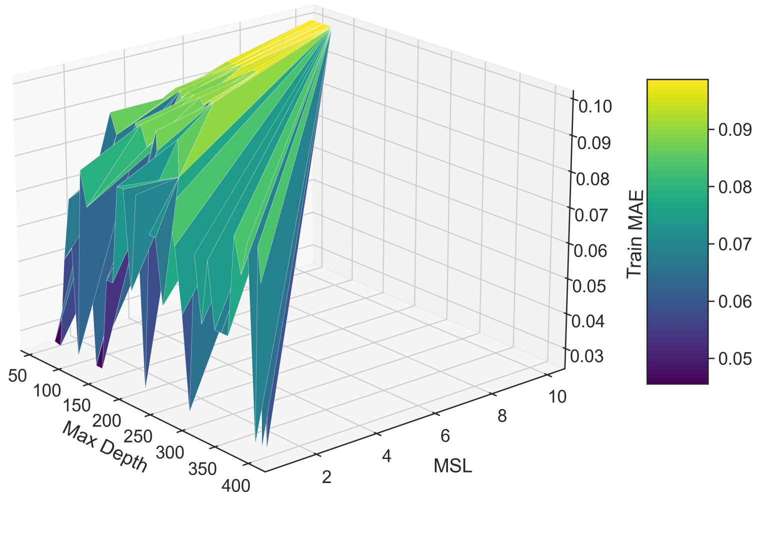


Figure 5.7: Effect of the Max Depth and the MSL on the Training MAE using Dataset A.

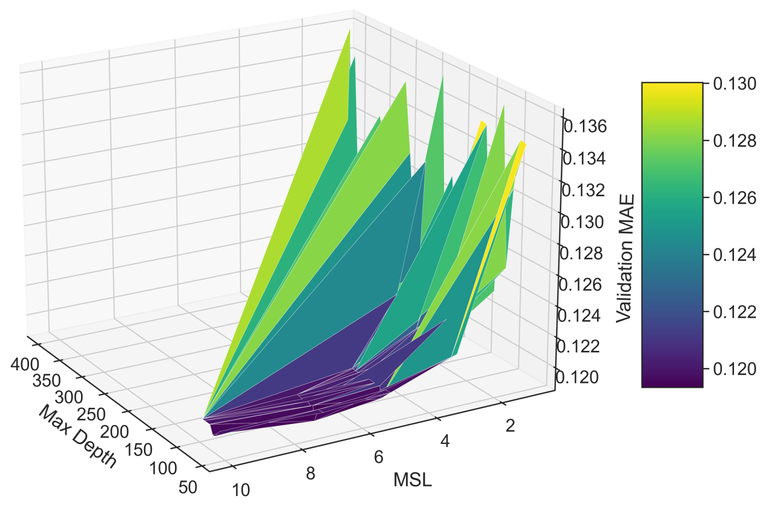


Figure 5.8: Effect of the Max Depth and the MSL on the Validation MAE using Dataset A.

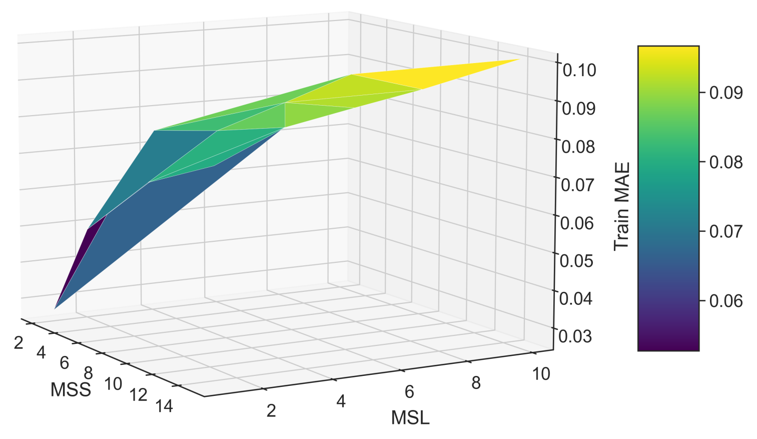


Figure 5.9: Effect of the MSS and the MSL on the Training MAE using Dataset A.

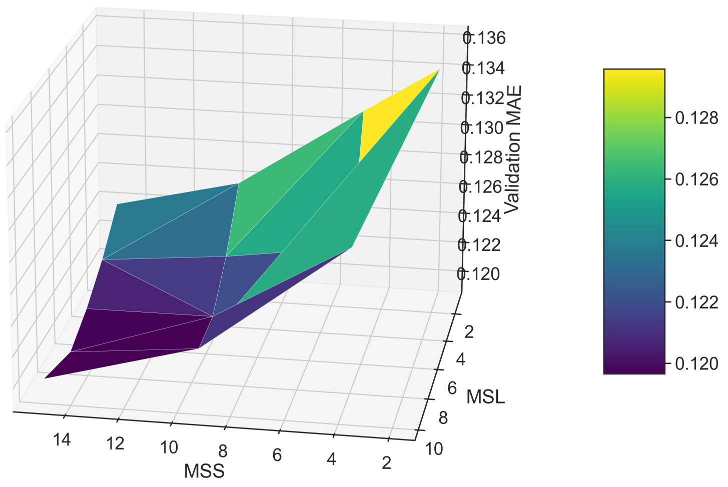


Figure 5.10: Effect of the MSS and the MSL on the Validation MAE using Dataset A.

Table 5.4: Final Set of Decision Tree Hyperparameters

Dataset	Max Depth	MSS	MSL	Train MAE	Val MAE	Test MAE
A	77	10	7	0.094472	0.118891	0.119736
B	70	10	7	0.094443	0.118699	0.119142
C	79	10	7	0.094436	0.119583	0.119327
D	70	10	7	0.094523	0.118764	0.119485
E	80	10	7	0.094559	0.118741	0.119347

high cardinality features. Without the reaction channels, it would be impossible for a model to distinguish between them. In this particular scenario, the relative importance between the reaction channels is based on the amount of data present in EXFOR as shown by the big Gini Importance value for MT1. Calculated feature importances for Dataset B-E can be found in Appendix B.

Decision Trees are very powerful models but these also have limitations in terms of generalization. Still, these and all other trained models set the baseline for more complex models and will also be tested using benchmark calculations in later sections.

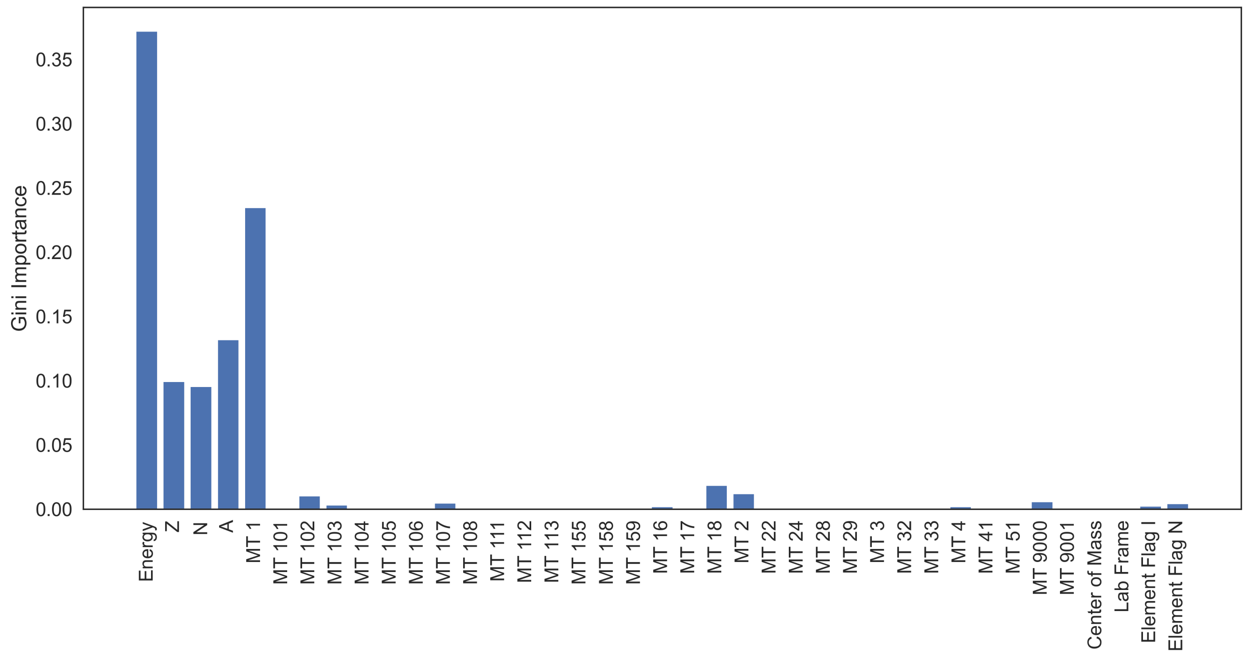


Figure 5.11: Feature Gini Importance for Decision Tree model trained on Dataset A.

5.3 Gradient Boosting Machines (GMB)

Gradient Boosting Machines are amongst the most powerful algorithms for structure data. In this work, we use the XGBoost implementation of GBMs. From the official documentation:

> XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. The same code runs on major distributed environment (Hadoop, SGE, MPI) and can solve problems beyond billions of examples.

The previously described Decision Trees are based on the CART type regression trees. This means only binary trees are built during the training process. XGBoost trains an ensemble of CART type regression trees meaning multiple trees are built. Contrary to traditional optimization problems where gradients are taken this is a more complex task since all trees cannot be built at once. XGBoost relies on the additive strategy meaning learned trees are never refitted; rather new trees are built afterward and added. The training procedure is rather complex and out of the scope of this work. For more information on the training methods and parameters, please see Reference [6].

Table 5.5: Experimented XGBoost Hyperparameters.

Parameter	**Experimental Ranges**
Objective Function	RMSE, Huber, RMSLE
Booster	GBTree
Learning Rate	0.1-0.5
Number of Trees (rounds)	5-60
Max Depth	5-30
Tree Method	GPU Histogram
Grow Policy	Depth-wise, Loss Guided
Max Leaves	0
Max Bin	0-30000
Gamma (Minimum Loss Reduction)	0-0.001
L2 Regularization	0-1
Deterministic Histogram	True or False
Early Stopping Method	True or False

Training

For all XGB models trained the GBTree booster algorithm was used given the size of EXFOR. To accelerate training the GPU Histogram tree method implementation was used. The training scenarios are more complex relative to the KNN and DT algorithm implementations due to the number of available hyperparameters. The two main hyperparameters of an XGBoost model are the Max Depth (MD) and the Number of Trees (NT). The former controls the maximum allowed depth of the individual trees built during the training process while the latter controls the total number of individual trees to be built. XGBoost is an ensemble technique meaning it sequentially adds new trees, limited by the set NT, to correct the performance of prior models. As both number increases, higher complexity is given to the model consequently increasing the risk of overfitting increases as well.

As more powerful models are trained, regularization will become important for increased generalization. Regularization can be applied not only by reducing the NT and the maximum allowed depth but also by increasing *Gamma* and the L2 Regularization strength. The Gamma hyperparameter specifies the minimum reduction in loss required to make a further partition on a leaf node. In XGBoost models and Neural Network, large weights can have a detrimental effect on the model. As the L2 Regularization strength increase, the more constrained these weights and consequently the model. To avoid overfitting, early stopping techniques can also be used to stop training and spending unnecessary computational resources when overfitting signs start to appear by setting an evaluation metric, an evaluation dataset, and a given number of iterations to wait for the performance metric to decrease. In early stopping techniques, the learning rate becomes important. Too large and the model will be unable to keep learning. To small and the training might take longer than needed. A combination of these hyperparameters can be used to find an optimal design that

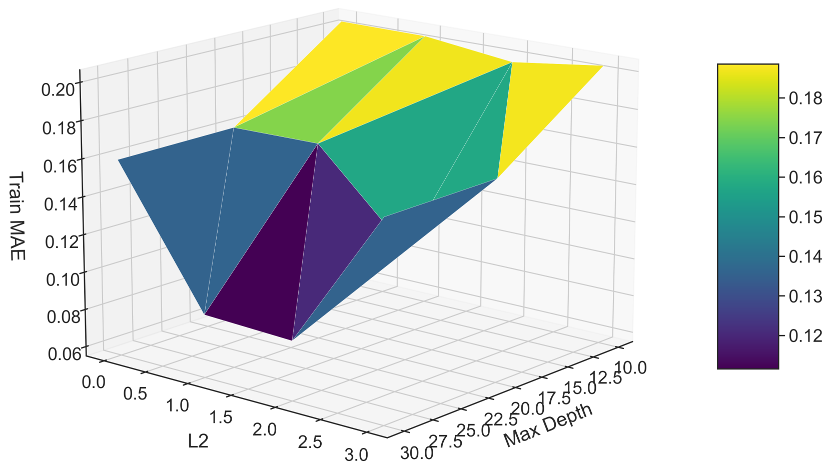


Figure 5.12: Effect of "L2" and "Max Depth" on the Training MAE.

generalizes well. Similar to DT algorithms, XGBoost models do not require that data be normalized. Other than taking the logarithm for both the Energy and Cross Section, no scalers or normalizers were applied to the dataset. Datasets A-E were used to train a variety of algorithms in our effort to find the best model.

Before applying a Grid Search algorithm to find the optimal hyperparameters, a parametric approach was taken to observe the effect of the different hyperparameters on the Training and Validation MAE. Figure 5.12 and 5.13 shows the Train and Validation MAE as a function of the Max Depth and L2 Regularization for all different models trained on Dataset A. The number of rounds was fixed to 10. As expected, a low Train MAE is achieved as the max depth increases. Similarly, as this number increases the Validation MAE also decreases. Due to memory constraints, Max Depths greater than 30 cannot be tested therefore setting a maximum allowed number. However, higher-memory GPUs should be used to train these models with Max Depths greater than 30 as there are no signs of overfitting at this point. The role of L2 regularization on generalization is also shown in both these figures. Several training iterations were performed using 0, 1, 2, and 3 as the L2 Regularization strength hyperparameter. A regularization strength between 1 and 2 seems to offer the best performance in terms of both the Training and Validation MAE. Both Figure 5.12 and 5.13 show the interactive effect between L2 and Max Depth on model performance. The surface plots show that max depths greater than 30 should be explored with L2 Regularization strength set to 1.5. No signs of overfitting are observed, in contrast to Decision Trees. In short, even with a max depth as high as 30, low enough error and no overfitting is achieved as long as

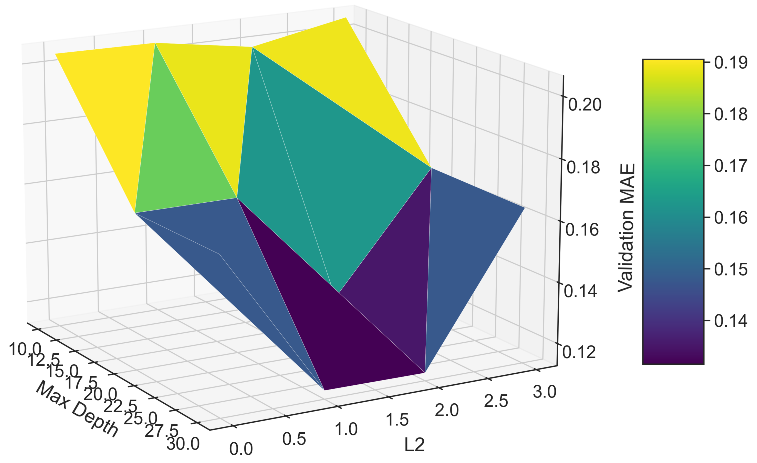


Figure 5.13: Effect of "L2" and "Max Depth" on the Validation MAE.

some regularization is applied.

A particular hyperparameter that was fundamental for model generalization is the Maximum Bin (MB). This parameter specifies the maximum number of bins by which a feature will be split to find the best split point when constructing the tree nodes. The default value of the XGBoost implementation is 256, a generally accepted value for most applications. However, nuclear data requires very fine grids. The level of resolution must be high enough that it allows to capture and separate overlapping resonances. Increasing the MB will therefore be important for optimal use of XGBoost models in the evaluation pipeline. To accurately model cross sections occurring at different energies for any isotope and reaction channel, a fine grid must be created. This means that 256 splits will not suffice to accurately describe the needed energy/cross section behavior (i.e. too coarse). To investigate the effect of the Max Bin parameter on both the Training and Validation MAE, several models were trained with differing Max Bin values. A 3D plot showing the effect of the Max Bin and Max Depth is shown in Figure 5.14. Increasing the number of bins will allow us to fit better models to this particular dataset along with a higher max depth. Confirming our expectations, as MD increases, performance quickly reaches a plateau for a low MB. Quick performance gains are observed with a steep gradient for increasing MB. A similar trend can be observed on the Validation MAE (Figure 5.15).

The interaction between the Max Depth and the L2 Regularization strength indicated an optimal value between 1-2. This was confirmed by plotting the interaction between the Max Bin and the L2 Regularization strength. As observed in Figure 5.16 and 5.17, local minima

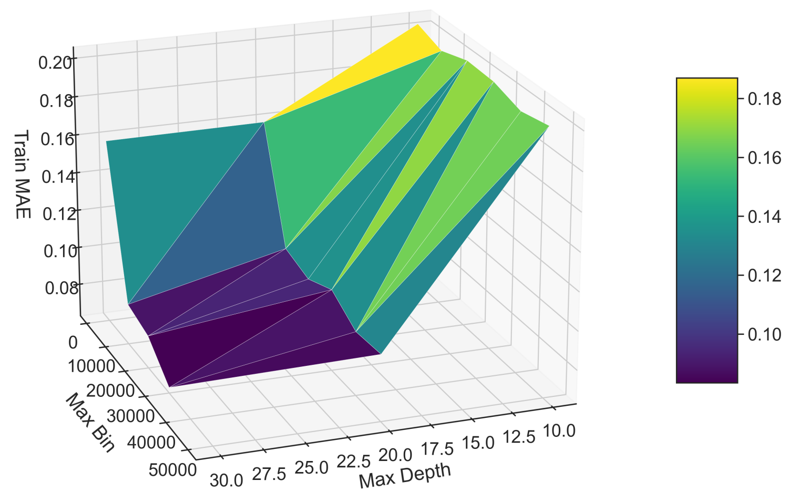


Figure 5.14: Effect of "Max Bin" and "Max Depth" on the Training MAE.

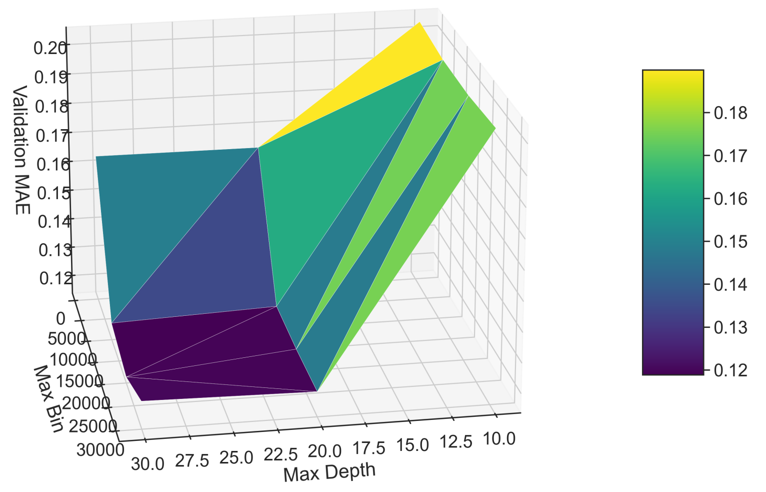


Figure 5.15: Effect of "Max Bin" and "Max Depth" on the Validation MAE.

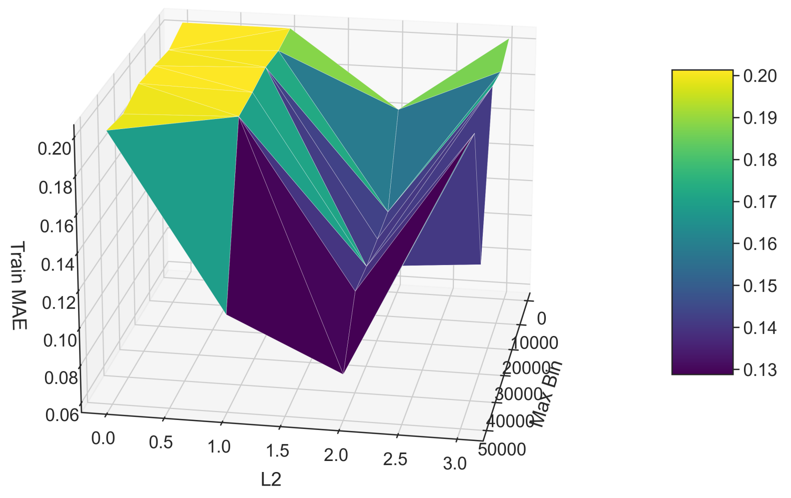


Figure 5.16: Effect of "Max Bin" and "L2" on the Training MAE.

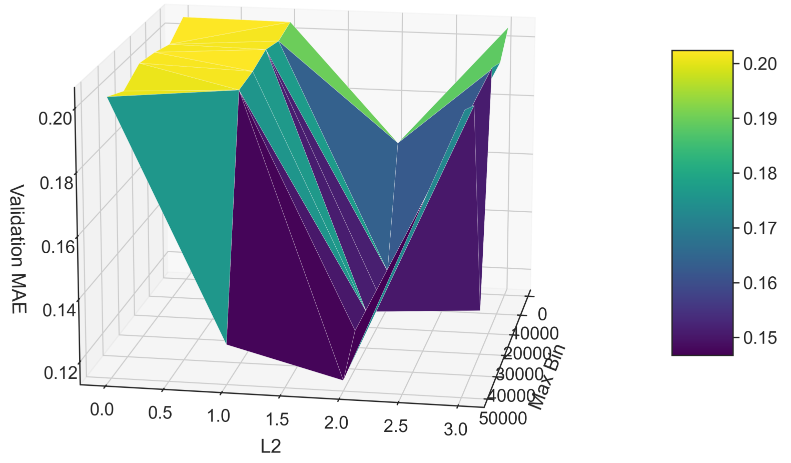


Figure 5.17: Effect of "Max Bin" and "L2" on the Validation MAE.

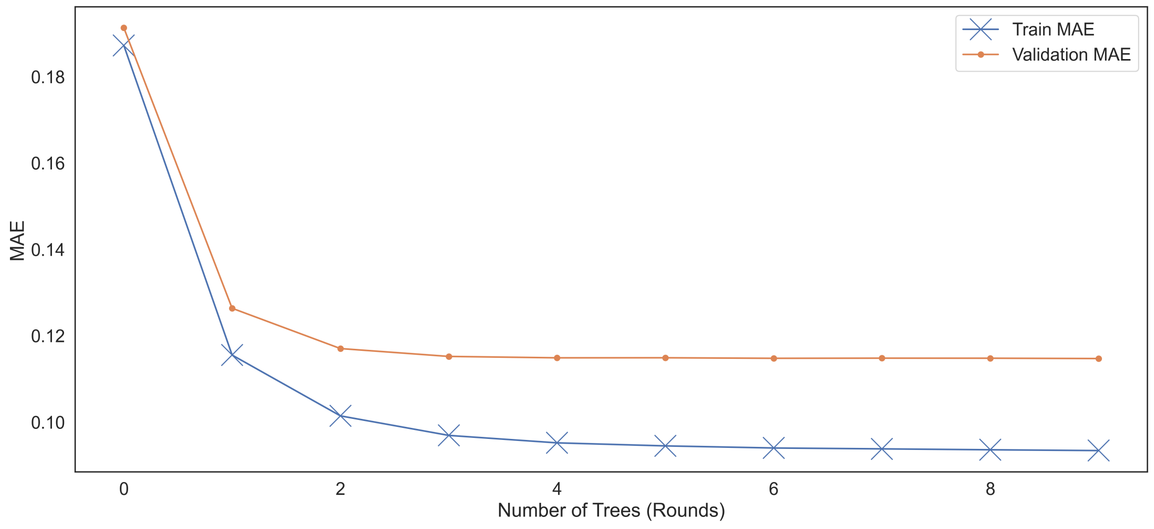


Figure 5.18: Training and Validation Loss as a Function of the Number of Trees (rounds) for the best XGB Model trained on Dataset A.

Table 5.6: Final Set of XGBoost Model Hyperparameters.

Dataset	Max Depth	L2 Regularization	Max Bin	Train MAE	Val MAE	Test MAE
A	30	3	10000	0.093399	0.114572	0.114719
B	30	1	20000	0.079715	0.114399	0.115458
C	30	3	10000	0.094528	0.114533	0.115727
D	30	1	10000	0.088638	0.114844	0.114683
E	30	3	20000	0.088554	0.114577	0.115488

in terms of Training and Validation performance seem to be located at L2 values of 2 for any Max Bin value. From all this information, it is evident that future training iterations should focus on exploring higher max depths.

The Learning Rate for all iterations was set to 0.8. Figure 5.18 shows the training and validation loss as a function of the number of rounds (trees) for the best model based on Dataset A. While the loss decreases quickly at first, it reaches a plateau at approximately 8 rounds. Based on this, the number of rounds was set to 10 for all training iterations. More figures exploring the hyperparameter optimization challenge can be found in Appendix C.

Based on the information gathered through the parametric exploration of hyperparameters, several models were trained using a Grid Search algorithm to search for the most optimal combination of hyperparameters using the ranges given in Table 5.5. Table 5.6 shows a list of different performance metrics for the best overall models trained in Datasets A-E.

XGBoost models can also be used to get feature importances. While Decision Tree's offer

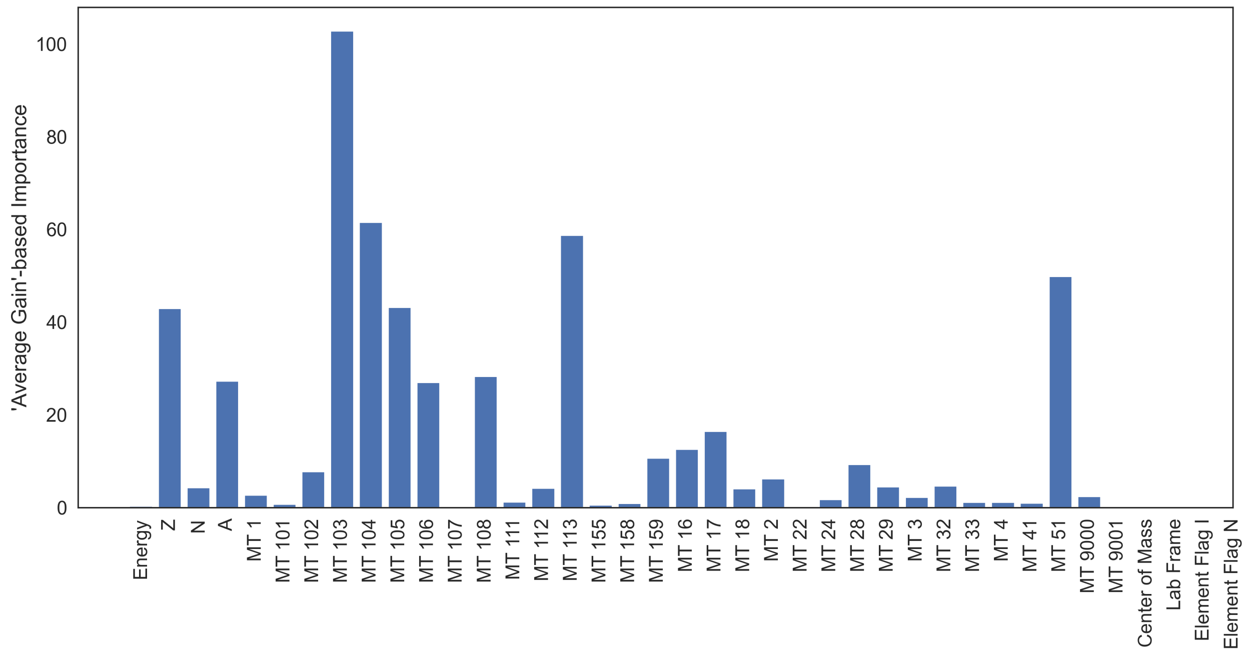


Figure 5.19: 'Average Gain'-based Feature Importances for the XGBoost model trained on Dataset A.

the same capability, it is only computed based on a single fitted tree. XGBoost models are an ensemble of trees. Consequently, the feature importance of each feature can be defined differently. Two popular options include 'weight' and 'gain'. The former calculates the importance based on the number of times a feature is used to split the data across all trees while the latter gets the average gain across all splits the feature is used in.

Figure 5.19 and 5.20 shows feature importance's extracted from the best XGBoost model trained on Dataset A. As expected, Energy appears to have no importance in the Gain-based calculations. Energy is simply a query point in space that is common for all cross sections and reaction channels. Instead, Figure 5.19 shows the features that help reduce the overall loss in this regression-based challenge. Contrary to the Decision Tree importances, here the reaction channels are quite important. These are fundamental to distinguish between different reaction scenarios in addition to the number of protons, neutrons, and the mass number. Figure 5.20 on the other hand shows reduced importance for the MT features. Since Weight-based importances are calculated in terms of the number of splits, it is expected that the reaction channels with most data are more important since most of the splits will be done for these scenarios. The Energy is not shown in this plot since the magnitude of the importance was too high to be plotted but it presented itself as the most important 'weight'-based feature. Calculated 'gain' and 'weight'-based feature importances for Dataset B-E can be found in Appendix C.

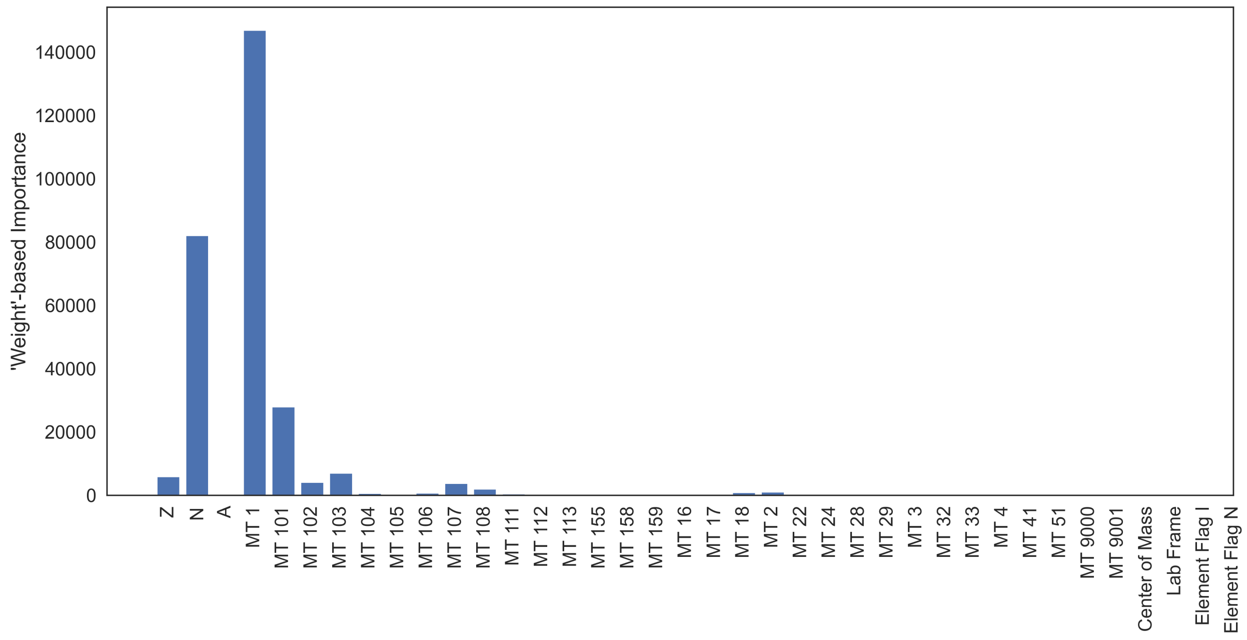


Figure 5.20: 'Weight/Split'-based Feature Importances for the XGBoost model trained on Dataset A.

XGBoost models are very powerful, often regarded as the best for structure data. The complexity of the model unfortunately also means a higher number of hyperparameters combinations to search for and to optimize. In this section, we discussed the hyperparameter optimization challenge for the Dataset built here that led to a set of best models. Reaching an optimization limit, increasing the number of rounds proved to be not beneficial. The performance seems to be plateauing after a couple of rounds. Since there are no signs of overfitting, a higher max depth should provide better overall performance. The selected best-performing models for each Dataset were saved and prepared for later testing using criticality benchmarks. Feature importance's calculated here will also help later experimental iterations using neural networks.

5.4 Summary

In this chapter, an overview of three popular algorithms (KNN, DT, and XGboost) was given. The different hyperparameters available in each of these were explored at first parametrically to understand the impact on both the training and validation loss using Datasets A-E. This analysis started with a rather simple algorithm (KNN) and moved towards more complex and powerful models. After a detailed analysis of hyperparameter interactions using 3D visualizations, a Grid Search algorithm was used with a limited search space provided by the parametric exploration. Tables 5.2, 5.4, and 5.6 show the resulting best models for Datasets A-E. Between all models, XGBoost showed the best performance in the validation set. Comparison of losses with previously published models is not possible as these set the first-ever benchmarks of ML applied to the EXFOR database.

Chapter 6

Results

The main goal is the development of proof-of-concept ML-NDE pipelines with validated models using the designed framework. Chapter 4 contains detailed information regarding the different ML models trained using data mainly from EXFOR. The traditional performance metrics and generalization potential evaluations are given and a set of models were selected based on hyperparameter optimization. These models set the first-ever benchmark of ML models applied to neutron-induced cross sections. Iterating through the models and coming up with a set of performance metrics usually marks the end of the ML pipeline (not accounting for deployment). However, in the ML-NDE pipeline, these models are to be validated using criticality benchmarks. Even though the previous three chapters mark important achievements in the path towards ML-enhanced nuclear data evaluations, here we additionally provide proof-of-concept criticality benchmark model validation. The potential impact of ML to support cross section data generation for use in modeling and simulation is demonstrated with five main examples. First, ^{233}U cross section data was predicted and tested using the ^{233}U Jezebel benchmark (U233-MET-FAST-001). ^{233}U was chosen for validation due to the extensive data available in terms of both EXFOR data points and benchmarks. Moreover, this benchmark is almost monoisotopic presenting a good isolated environment for model testing. Other criticality benchmarks presented here include U233-MET-FAST-002 (10kg), U233-MET-FAST-002 (7.6kg), and PU-MET-FAST-019. The ^{35}Cl(n,p)^{35}S reaction cross section was also chosen as an example of prediction capabilities in an energy region lacking experimental data important for new reactor concepts like the MCFR. Other individual isotopes of interest are explored, and dataset issues are raised. All of these experiments were possible thanks to NucML, the main deliverable as part of this work.

6.1 NucML

Due to the lack of specialized tools for the application of ML to the Nuclear Data Evaluation field, this work aimed first and foremost at providing a complete ML-enhanced evaluation pipeline framework for inference of neutron-induced reaction cross sections and the tools necessary to seamlessly navigate it. From this endeavor, NucML was born. NucML is the first and only end-to-end python-based supervised machine learning pipeline for enhanced bias-free nuclear data generation and evaluation to support the advancement of next-generation nuclear systems. It offers capabilities that allow researchers to navigate through each step of the ML-based nuclear data cross section evaluation pipeline. Some of the supported activities include dataset parsing and compilation of reaction data, exploratory data analysis, data manipulation, feature engineering, model training and evaluation, and validation via criticality benchmarks. NucML's source code can be found in GitHub and has been made public to the community through the Python Package Index manager. The code is thoroughly documented making it easy to use. For information on the implementation, source code, and documentation please visit `https://pedrojrv.github.io/nucml/` and `https://github.com/pedrojrv/nucml`.

Datasets

In Chapter 4, a detailed exploration of feature impact on model performance was described. Part of NucML's ML-ready datasets include 5 different combination of features:

- Dataset A: "Energy", "Data", "Z", "N", "A", "MT", "Center of Mass Flag", "Element Flag"
- Dataset B: Dataset A Features, "Atomic Mass Micro", "Nucleus Radius", "Neutron Nucleus Radius Ratio"
- Dataset C: Dataset B Features, "Mass Excess", "Binding Energy", "B Decay Energy", "S(n)", "S(p)", "S(2n)", "S(2p)"
- Dataset D: Dataset C Features, "N valence", "Z valence", "P factor", "N tag", "Z tag", "NZ tag"
- Dataset E: Dataset D Features plus all Q-values.

For information on the data type meaning please refer to Chapter 3. After analyzing the results of all trained models including performance metrics and feature importances, the best dataset was found to be Dataset C. While dataset D and E contained some features that might be relevant for future exploration of ML-based solutions, their impact on model performance was marginal and in some cases negative. The criticality benchmark calculations were performed using all models trained on Dataset C.

6.2 Criticality Benchmarks

All benchmark performances will be compared to the ENDF library and in the appropriate cases to other evaluated libraries like JEFF and JENDL. For an overview of the benchmark geometry and composition please refer to Chapter 3. The 1/v region for all ML-generated cross section was adjusted in post-processing due to the instabilities presented by both models. This goes in line with the expected hybrid implementation. In other words, the solution presented here consists of a machine learning and traditional tools hybrid modeling technique. Furthermore, only isotopes for which the total fraction is above 10% were considered for ML-based cross section generation. Trace quantities of other isotopes were left as is and imported from the ENDF library. During the compilation process, the cross sections were run through standard checks (i.e. making sure the appropriate reaction channels sum up to the total cross section). All KNN, DT, and XGB models were used to generate these cross sections for the needed reactions, not just the best models. The reason will become apparent in a moment. The generated cross sections were compiled into ACE format and metadata files needed by SERPENT2 were generated for all cases using NucML's automatic framework. Other data files including fission product yields we're not modified.

U233-MET-FAST-001

The ^{233}U Jezebel benchmark was selected due to the simple, almost mono-isotopic nature. The generated cross sections included ^{233}U (n,tot), (n,γ), (n,inelastic), (n,elastic), and (n,f). All other trace uranium isotopes were left unchanged due to the small contribution to the total composition.

KNN Model Selection and Performance

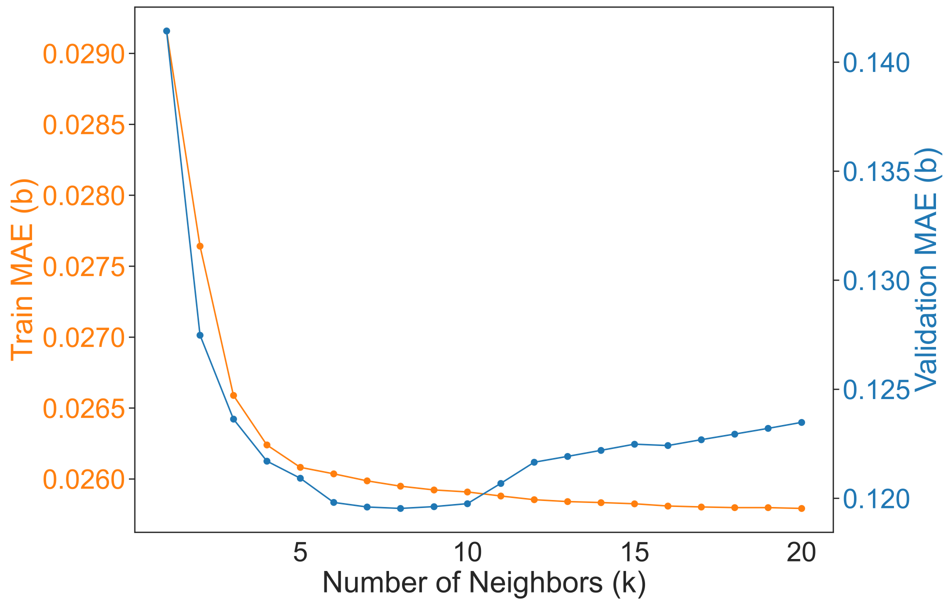
Of the KNN parameters tested the "distance" weight function and the Manhattan distance worked best. As for the scalers, the min-max achieved better results. The top graph in Figure 6.1 represents the MAE for the train and validation sets as a function of the parameter K (number of nearest neighbors) for the models trained on dataset A. The model with $k = 8$, as seen in Table 6.1, served as our traditionally selected model since it is at this point that the validation MAE starts to increase and deviate from a downwards trajectory while the train MAE kept decreasing indicating overfitting. Using this model, the Jezebel benchmark was run using the ^{233}U KNN-generated cross sections. The results for this model are listed in Table 6.1. The performance is adequate with an error of 0.3870%.

To find if the traditionally selected model was indeed the best model in terms of benchmark performance, ^{233}U cross sections were generated using all KNN models. The bottom graph in Figure 6.1 depicts the validation MAE and the multiplication factor as a function of K. As expected, the traditional model selection technique does not seem to correlate with benchmark performance. Still, in this specific case, the validation set provides better benchmark results than that of the overfitted model ($k = 20$) by around 41%. On the other side, handpicking the model with the best benchmark performance ($k = 10$) leads to an improvement of 98% relative to the validation-based selected model. This is a big difference and already outperforms the ENDF-based benchmark results. One drawback of performing this kind of analysis in KNN models is that there is no loss function being optimized. The per-

Table 6.1: ^{233}U Jezebel Benchmark Results.

Library	k_{eff}	Uncertainty	Error (%)	Train MAE	Validation MAE	Test MAE
ENDF	1.000200	+/- 0.00114	0.0200	N/A	N/A	N/A
KNN[a] ($k = 8$)	1.003870	+/- 0.00140	0.3870	0.025949	0.119539	0.121166
KNN[b] ($k = 20$)	0.993452	+/- 0.00158	0.6548	0.025793	0.123483	0.123792
KNN[c] ($k = 10$)	1.000040	+/- 0.00133	0.0040	0.025909	0.119757	0.121358
DT[a, d]	1.00011	+/- 0.00137	0.011	0.094400	0.119509	0.119396
DT[b, e]	1.00396	+/- 0.00139	0.396	0.025784	0.135760	0.135082
DT[c, f]	0.999986	+/- 0.00134	0.0014	0.094735	0.119564	0.119406
XGB[a, g]	0.977935	+/- 0.00044	2.2065	0.079715	0.114399	0.115458
XGB[b, h]	1.004940	+/- 0.00044	0.4940	0.065501	0.121320	0.122416
XGB[c, i]	0.997123	+/- 0.00043	0.2877	0.196566	0.197323	0.197410

[a] Validation-based selected model
[b] Train-based selected model
[c] Hand-picked model
[d] Max Depth = 106, MSS = 10, MSL = 7
[e] Max Depth = 335, MSS = 2, MSL = 1
[f] Max Depth = 100, MSS = 15, MSL = 7
[g] Max Depth = 30, Max Bin = 20000, L2 = 1
[h] Max Depth = 30, Max Bin = 30000, L2 = 0
[i] Max Depth = 10, Max Bin = 20000, L2 = 3



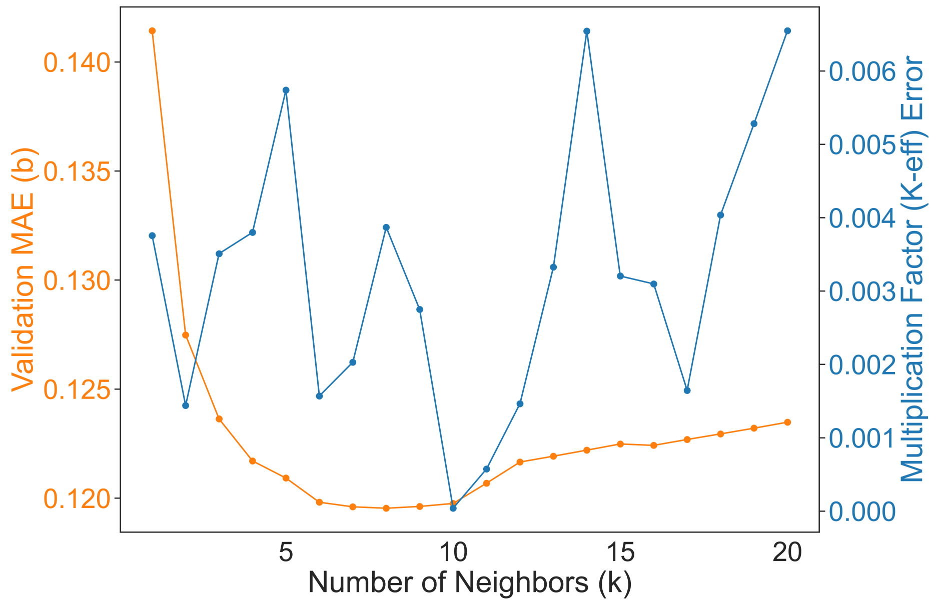


Figure 6.1: Train and validation MAE as a function of the number of neighbors (k) (top). Validation MAE and Multiplication Factor (k_{eff}) error relative to 1 as a function of the number of neighbors (k) (bottom). Both the train and validation MAE are in a $\log_{10}$ scale.

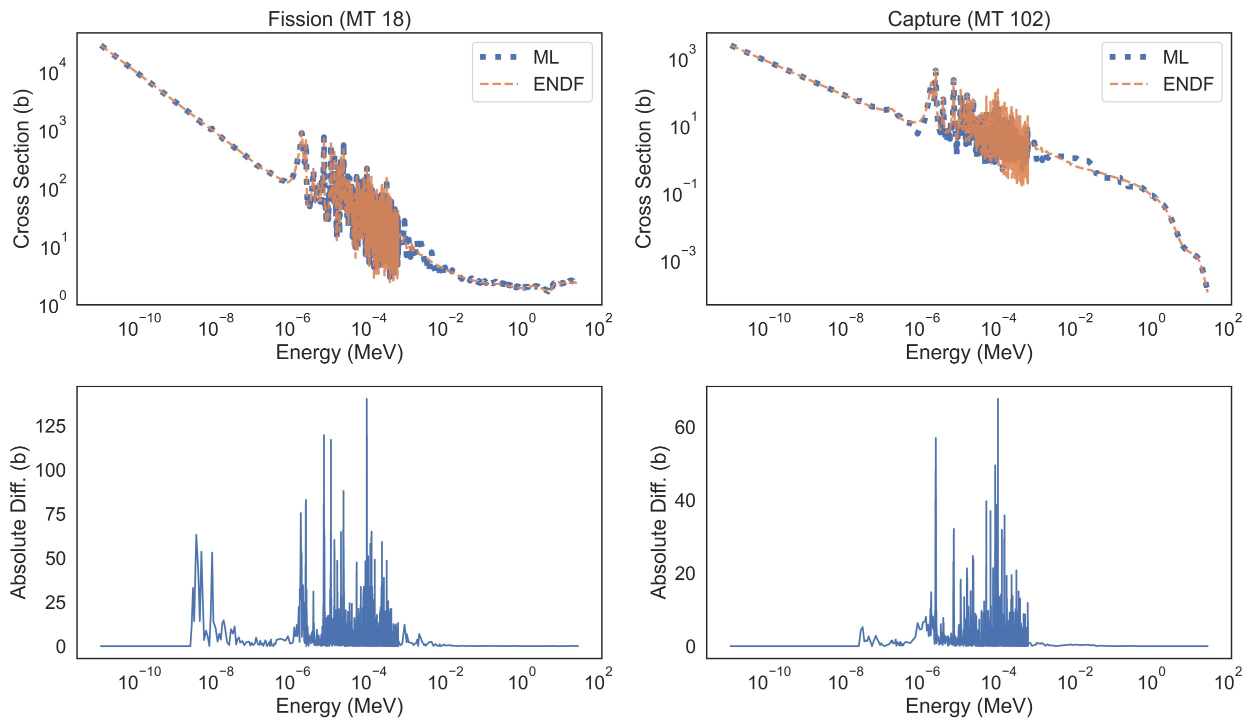


Figure 6.2: Comparison of KNN generated and ENDF/B-VIII.0 cross sections for fission and radioactive capture.

formance comes strictly from the dataset completeness. Still, it provides important insight into the dataset issues causing deviation from the expected validation MAE/multiplication factor correlation.

The deviations could be explained by a variety of factors. Let us take the best benchmark model ($k = 10$) for example. Figure 6.2 shows the fission and capture cross section results along with the absolute differences relative to the ENDF cross sections. As expected, the resonance region is where most of the differences arise. Overall, the generated cross sections for ^{233}U compare well to those in ENDF with the exception of the (n,elastic) cross section (Figure 6.4). As seen in Figure 6.3, the experimental datapoints available are scarce. This limits the model's performance in this particular (isotope, reaction-channel) pair. Because of the high availability of other reaction channels, the MT 2 cross section was calculated as the difference between the ML generated (n,tot) and (n,nonelastic) cross sections. The resulting values show some erratic behavior at low energies. In intermediate energies, the resonances seem to have a higher magnitude in both directions relative to ENDF. This is a clear example of the limitations of using this type of model and performance metrics when working with databases like EXFOR. There are also differences in the transition between the resonance and fast energy regions. This is to be expected since resonances exist in the fast

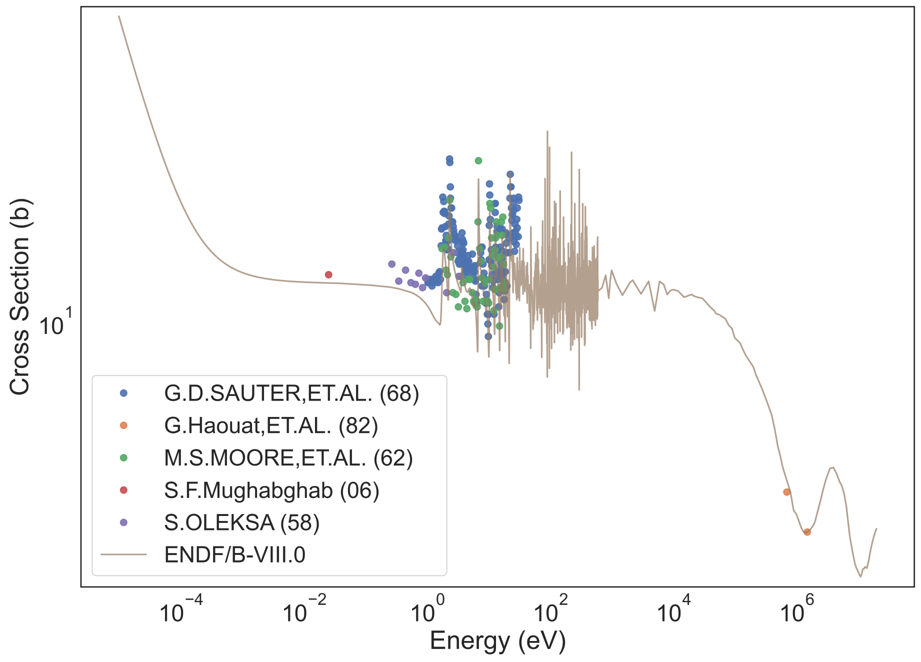


Figure 6.3: ^{233}U(n,elastic) reaction channel experimental datapoints in EXFOR vs the ENDF/B-VIII.0 evaluation.

energy region but our capabilities/resolution do not allow us to measure them and therefore an average is taken in ENDF. In a real ML-hybrid modeling scenario, the evaluator would need to make a decision based on benchmark performance.

DT Model Selection and Performance

Decision Trees on the other hand do optimize for MSE. As mentioned, a balance between the max depth, MSS, and MSL must be found to achieve good generalization and performance. Table 6.1 shows the results on the Jezebel benchmark for the validation-based and train-based selected models along with the final model parameters. Similar to the KNN results, the performance is adequate for the validation-based model with an error of around 0.011%. The error is already better than that of the ENDF-based benchmark calculations. Having observed that traditional model selection techniques may not yield the best model in terms of benchmark performance, a parametric study was performed by generating the ^{233}U cross sections and running the benchmark using every trained DT model.

Figure 6.5 depicts the train (top) and validation (bottom) MAE as a function of the max depth and the multiplication factor. The MSS and the MSL parameters were also changed. The validation-based selected model achieved much better performance in the benchmark relative to the train-based selected model. However, similar to the outcome in the KNN

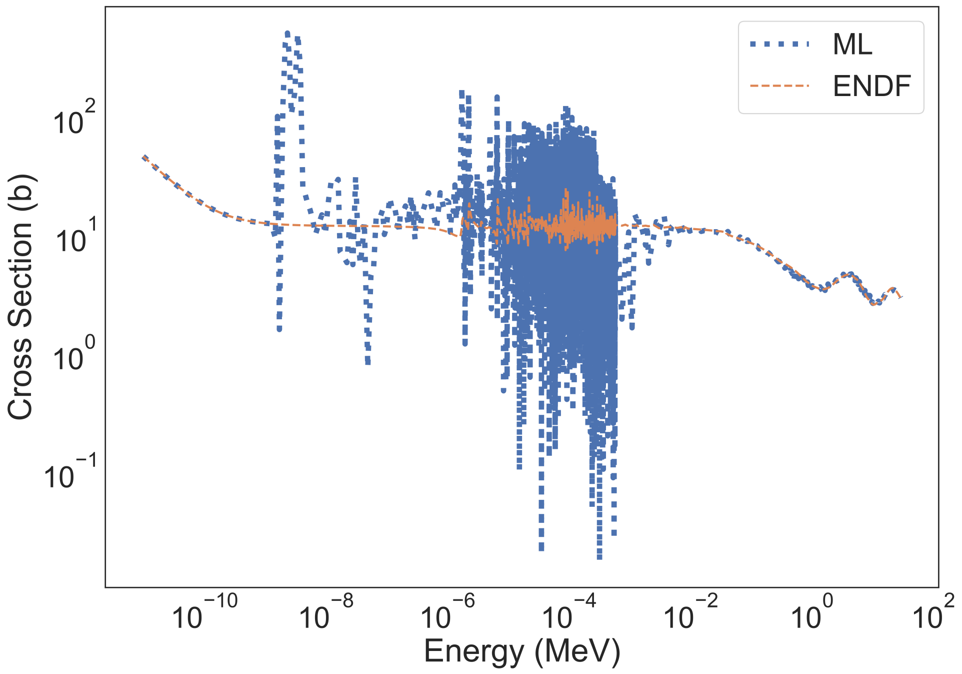


Figure 6.4: KNN inferred cross section values for the ^{233}U elastic channel vs the ENDF/B-VIII.0 evaluation.

based models, the figure shows that the lowest validation MAE does not yield the best benchmark performer. For both the validation and train MAE, there is a value where the optimal benchmark performer can be found. Handpicking the model based on benchmark performance leads to an improvement of 87% relative to the validation-based model.

XGBoost Model Selection and Performance

XGBoost models did not seem to have the same performance as simpler models. There exists the possibility that XGBoost benchmark performance is affected by the dataset selected. As we have seen so far, the correlation between validation MAE and benchmark performance is not always so straightforward. Additionally, we did not see signs of overfitting on even the best models. Higher complexity designs were not tested due to the memory requirements. High-Performance Computing systems should be used to train more complex models that can potentially outperform KNN and DT in benchmark calculations.

U233-MET-FAST-002 (7.6kg and 10kg)

The next tested benchmark were both the 10kg and 7.6kg ^{235}U-reflected ^{233}U spheres. This case is one step more complex since cross section data needs to be generated not only for

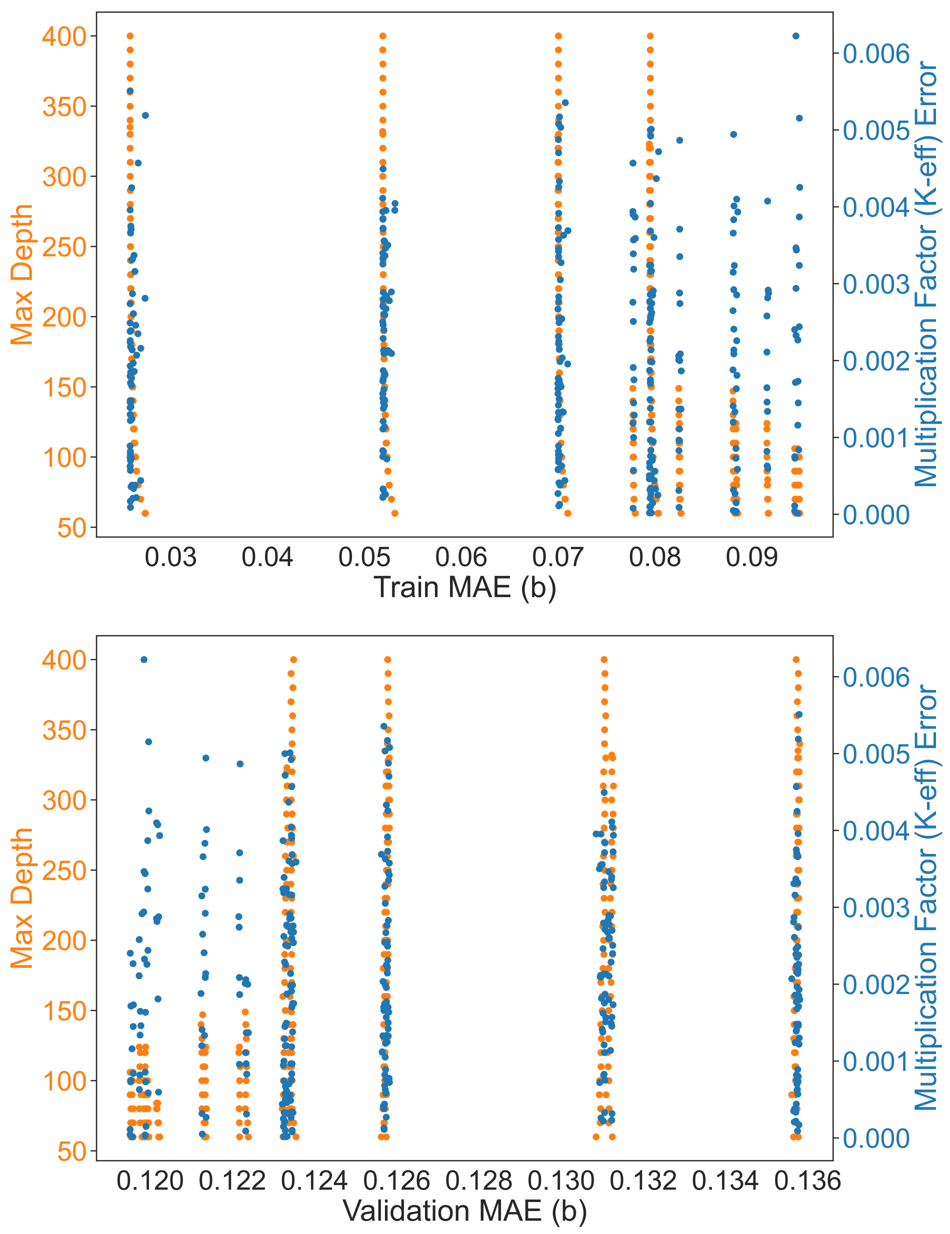


Figure 6.5: Train and validation mean absolute error (MAE) as a function of the max depth and the multiplication factor error relative to 1. Both the train and validation MAE are in a $\log_{10}$ scale.

^{233}U but also for the surrounding ^{235}U shell. The same methodology taken during the U233-MET-FAST-001 benchmark validation was taken and three models for each algorithm were taken. Table 6.2 and 6.3 shows the results for the train-, validation-, and hand-picked-based KNN, DT, and XGBoost models.

Similar conclusions can be drawn from these results. In all cases, the Train-based model outperforms the Validation-based model. For the KNN algorithm, both the train- and hand-picked-based models outperform the evaluation showing improvements of 39% and 160% respectively. In this particular benchmark, in contrast to the U233-MET-FAST-001, the hand-picked XGBoost model also outperformed the ENDF evaluation in this more complex scenario by 54%. As for the decision tree models, the hand-picked model generated library achieved a multiplication factor of 1 (zero error) making it the most accurate model in terms of benchmark performance in this work. This is a boost in performance of 200%. This further reinforces our hypothesis that outlier detection will become important for ML applications to the nuclear data field.

Having some of these ML models outperformed the evaluation in the 10kg assembly of the U233-MET-FAST-002 criticality benchmark, the expectation is to further outperform the regional library in the 7.6kg assembly. The KNN algorithm did not continue to outperform the evaluation in this case, not even the hand-picked model. The same occurred for the XGBoost models. While the results do not outperform the evaluation in this case, it is still promising to see models performed adequately given that this is the first iteration of this effort setting the benchmark for future work. Decision Tree models in this case continue to outperform the evaluation proving to be one of the most robust algorithms here. Both the validation and the hand-picked model manage to get a performance boost of 23% and 118% respectively.

PU-MET-FAST-019

This benchmark presents itself as an excellent example of the needed improvements in EXFOR. The models were successfully compiled and run for criticality calculations using the Beryllium-reflected Plutonium Sphere. However, the results were highly inconsistent and erroneous. After inspection, the Plutonium cross sections appear to be processed correctly, however, inferred values for Beryllium were unstable.

Figure 6.6 shows all available datapoints for the ^{9}Be(n,tot) reaction in the EXFOR library. It is evident that wrong datasets are forming part of the training dataset at low-energy regions. There are 4.5 million datapoints and inspecting each one manually is not possible for a single person. Unfortunately, all models trained here used some of these datapoints for training and therefore predict wrongly the correct cross section values. Re-training a set of Machine Learning models can take weeks and these types of mistakes simply set back the overall goal and waste computational resources. Figure 6.6 contains the ^{9}Be(n,tot) predictions performed by a sample DT model that was trained on an EXFOR version of the dataset containing this erroneous points. The following benchmark was therefore not tested.

Table 6.2: U233-MET-FAST-002 10kg Benchmark Results.

Library	k_{eff}	Uncertainty	Error (%)	Train MAE	Validation MAE	Test MAE
ENDF	0.998075	+/- 0.00042	0.1925	N/A	N/A	N/A
KNN[a] ($k = 9$)	1.006410	+/- 0.00043	0.6410	0.025921	0.119010	0.118578
KNN[b] ($k = 20$)	0.998708	+/- 0.00044	0.1292	0.025814	0.121110	0.120706
KNN[c] ($k = 17$)	1.000210	+/- 0.00044	0.0210	0.025822	0.120527	0.120098
DT[a, d]	0.997767	+/- 0.00044	0.2233	0.094443	0.118699	0.119142
DT[b, e]	1.003330	+/- 0.00044	0.3330	0.025773	0.136140	0.135027
DT[c, f]	1.000000	+/- 0.00041	0.0000	0.051870	0.131216	0.129827
XGB[a, g]	0.980368	+/- 0.00046	1.9632	0.079715	0.114399	0.115458
XGB[b, h]	1.006340	+/- 0.00044	0.6340	0.065501	0.121320	0.122416
XGB[c, i]	1.001100	+/- 0.00044	0.1100	0.196566	0.197323	0.197410

[a] Validation-based selected model
[b] Train-based selected model
[c] Hand-picked model
[d] Max Depth = 70, MSS = 10, MSL = 7
[e] Max Depth = 400, MSS = 2, MSL = 1
[f] Max Depth = 280, MSS = 5, MSL = 1
[g] Max Depth = 30, Max Bin = 20000, L2 = 1
[h] Max Depth = 30, Max Bin = 30000, L2 = 0
[i] Max Depth = 10, Max Bin = 20000, L2 = 3

Table 6.3: U233-MET-FAST-002 7.6kg Benchmark Results.

Library	k_{eff}	Uncertainty	Error (%)	Train MAE	Validation MAE	Test MAE
ENDF	1.000860	+/- 0.00043	0.08600	N/A	N/A	N/A
KNN[a] ($k = 9$)	1.008740	+/- 0.00045	0.87400	0.025921	0.119010	0.118578
KNN[b] ($k = 20$)	1.001540	+/- 0.00044	0.15400	0.025814	0.121110	0.120706
KNN[b] ($k = 20$)	1.001540	+/- 0.00044	0.15400	0.025814	0.121110	0.120706
DT[a, d]	1.000680	+/- 0.00044	0.06800	0.094443	0.118699	0.119142
DT[b, e]	1.005650	+/- 0.00042	0.56500	0.025773	0.136140	0.135027
DT[c, f]	1.000220	+/- 0.00042	0.02200	0.082419	0.121982	0.121942
XGB[a, g]	0.985785	+/- 0.00044	1.42150	0.079715	0.114399	0.115458
XGB[b, h]	1.008750	+/- 0.00039	0.87500	0.065501	0.121320	0.122416
XGB[c, i]	0.998213	+/- 0.00042	0.17870	0.196634	0.197023	0.197088

[a] Validation-based selected model
[b] Train-based selected model
[c] Hand-picked model
[d] Max Depth = 70, MSS = 10, MSL = 7
[e] Max Depth = 400, MSS = 2, MSL = 1
[f] Max Depth = 170, MSS = 10, MSL = 3
[g] Max Depth = 30, Max Bin = 20000, L2 = 1
[h] Max Depth = 30, Max Bin = 30000, L2 = 0
[i] Max Depth = 10, Max Bin = 40000, L2 = 2

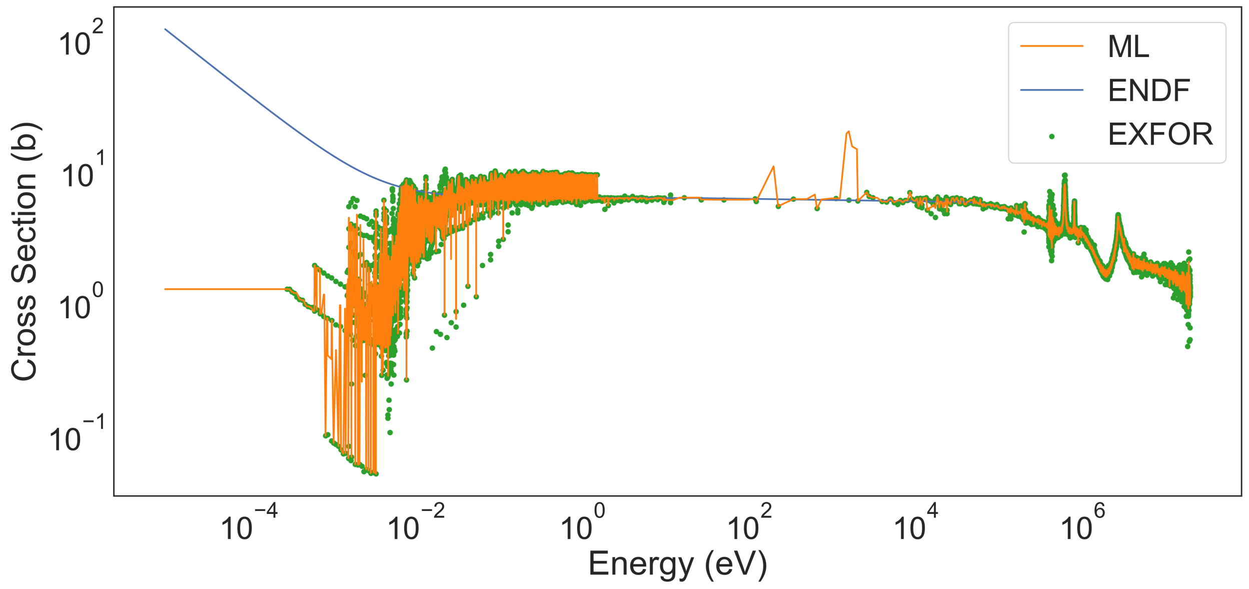


Figure 6.6: ^{9}Be(n,tot) Cross Section Values in EXFOR.

Overall Performance Comparison Between ENDF and ML-enhanced Evaluations

A model cannot be selected and compared based on a single benchmark calculation. This can make the process overfit to these singly selected scenarios. To more accurately select the best models, the average performance between all available benchmarks should be used as a key metric for model selection. In this work, all models were used to generate cross sections and tested using all three benchmarks presented here. Using this more rigorous criterion, there are 43 models of all parameters explored able to outperform the ENDF library. Table 6.4 shows only the best models for each algorithm. The XGBoost models were not able to outperform ENDF on average compare to the KNN and DT both of which outperform the regional library by 105% and 200% respectively. Still, these are only three benchmarks. Future work should focus on gathering more criticality benchmarks and splitting them into a train and validation set. Otherwise, even though this average performance model selection, we can overfit to a set of benchmarks.

Other Benchmarks

NucML's benchmark repository is aimed to be an open source, collaborative effort where users can add their own inputs to form part of the ML-NDE pipeline project. Any valid model added can be used by NucML's pipeline to measure the impact of ML-generated cross sections and validate models more accurately. As mentioned previously, NucML's current

Table 6.4: Average Criticality Benchmark Error.

Library		Average Error	Standard Deviation
Number of Models			
ENDF	0.028833	0.009654	N/A
KNN ($k = 19$)	0.008967	0.000440	1
DT (MD=110, MSS=5, MSL=1)	0.000100	0.000430	42
XGB (MD=10, MB=20000, L2=3)	0.115767	0.16541	0

benchmarks include:

- U233-MET-FAST-001: ^{233}U Jezebel
- U233-MET-FAST-002: ^{235}U Surrounded ^{233}U Core (10kg)
- U233-MET-FAST-002: ^{235}U Surrounded ^{233}U Core (7.6kg)
- PU-MET-FAST-019: Sphere of Plutonium Reflected by Beryllium
- PU-MET-FAST-005: Sphere of Plutonium Reflected by Tungsten

Other sets of benchmarks of interested forming part of the source code include all eight cases of Water-moderated U(2.35)O2 Fuel Rods in 2.032-cm Square-pitch Arrays. These are thermal benchmarks that can immensely aid the validation of models in the thermal region. Not only are these benchmarks included in their original form but also tailored to NucML's needed format for validation. Detailed instructions are provided for any user to add their own even they wish to not be included as part of the public repository but needed to validate their models. Visit `https://github.com/pedrojrv/ML_Nuclear_Data/tree/master/Benchmarks` for more information.

6.3 ^{35}Cl(n,p)^{35}S Cross Section

^{35}Cl, instead, has far less experimental data and in particular, the lack of data for the (n,p) reaction in the energy range above 0.1 MeV has a substantial impact on the development of molten chloride fast spectrum reactors. Recently, a measurement performed at the University of California, Berkeley at about 2.5 MeV has shown that existing data libraries overestimate measured points by up to a factor of five [1]. Therefore, ^{35}Cl was chosen to demonstrate the capability of the ML model to generate cross section data for data-starved reactions, and its viability was evaluated against the new measurements whose results were not known to the algorithm at any stage. The cross section for ^{35}Cl(n,p)^{35}S was inferred using the validated KNN, DT, and XGBoost models.

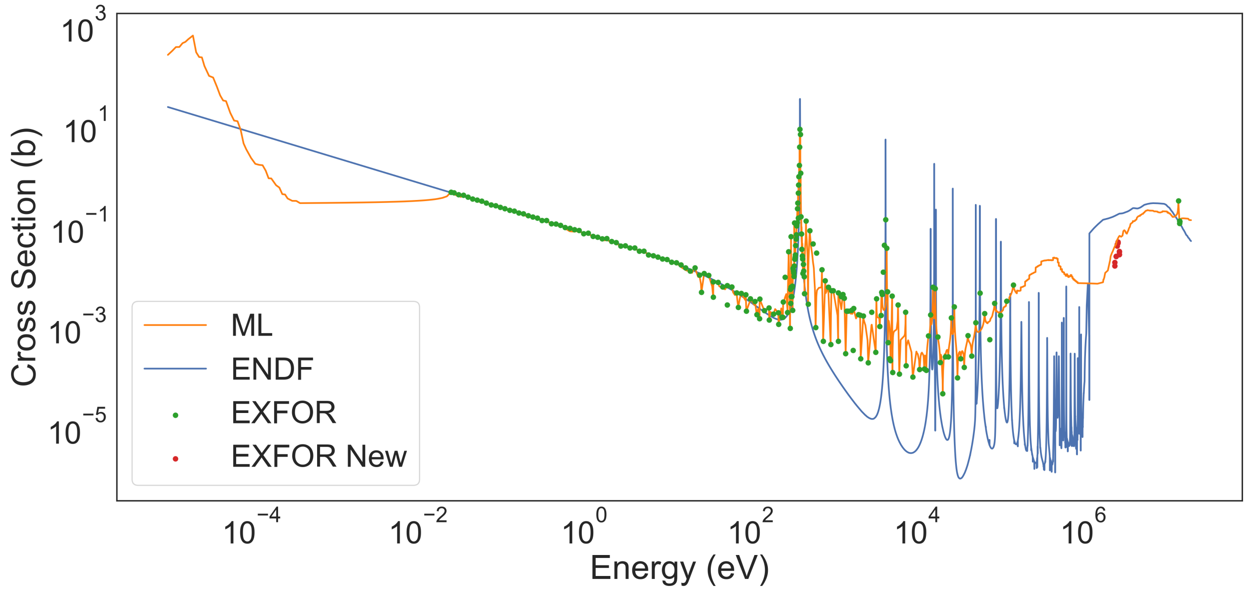


Figure 6.7: ^{35}Cl(n,p)^{35}S Cross Section Generated using a trained KNN model.

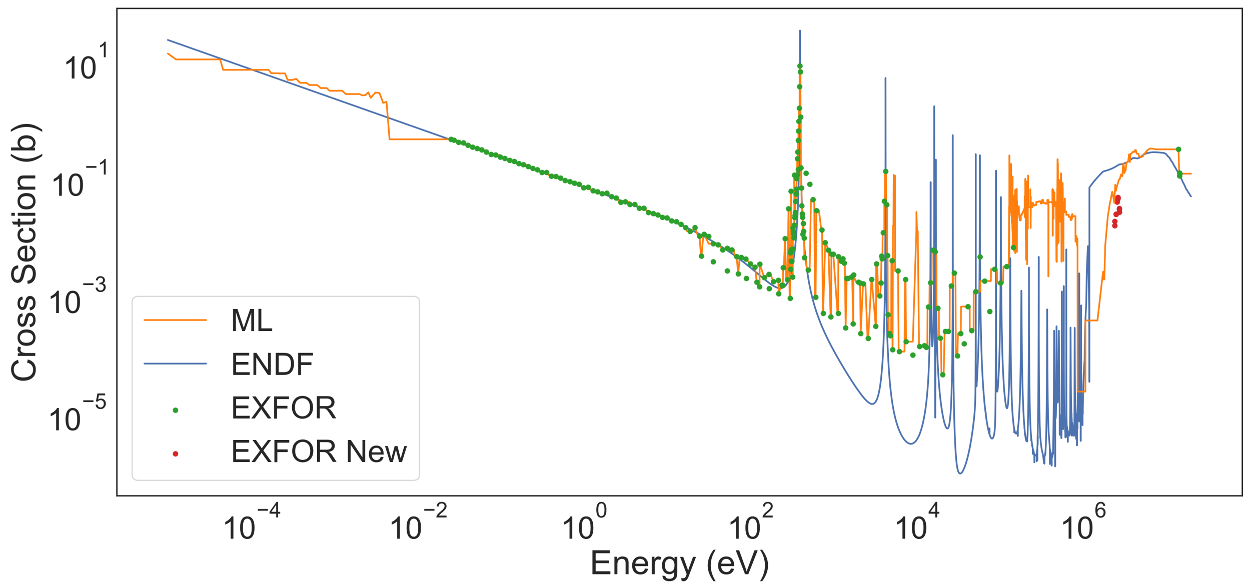


Figure 6.8: ^{35}Cl(n,p)^{35}S Cross Section Generated using a trained Decision Tree model.

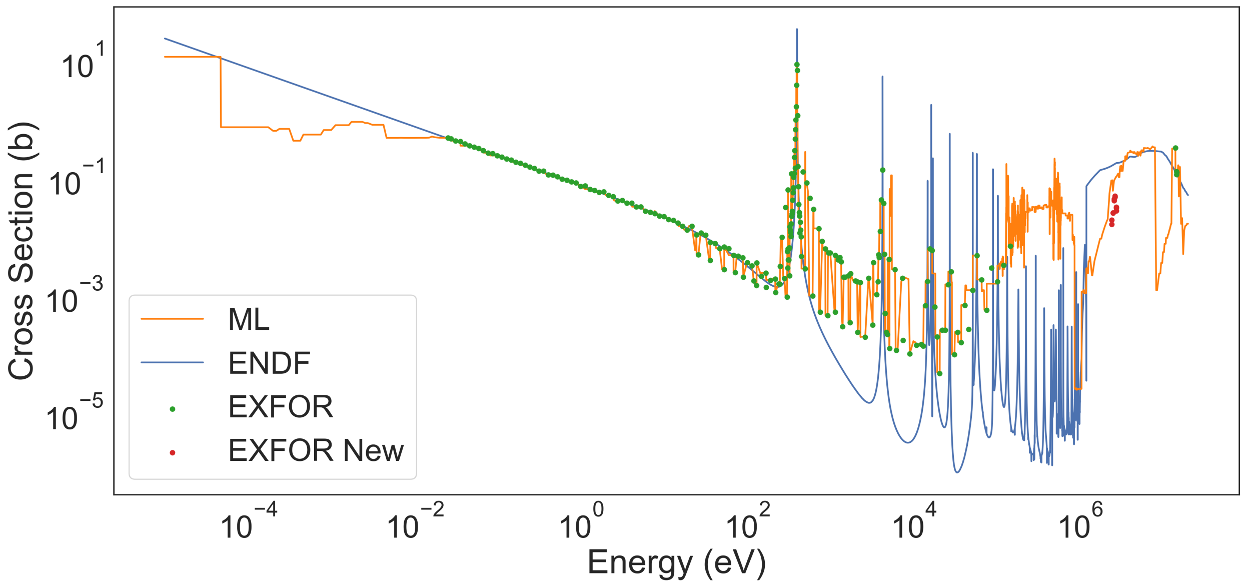


Figure 6.9: ^{35}Cl(n,p)^{35}S Cross Section Generated using a trained XGBoost model.

Figures 6.7-6.9 show the inference values for the best KNN, DT, and XGBoost model respectively along with the available EXFOR datapoints, the new LBNL/UCB measurements, and the ENDF/B-VIII.0 evaluation. Table 6.5 shows the errors and overall MAE of these models with respect to the LBNL/UCB measurements for the ML-generated values, the ENDF/B-VIII.0 library, and the JENDL-4.0 library.

All models presented here were able to predict the cross section values better than any of the evaluated libraries. The KNN models were able to do so by looking at other nearby points in the multi-dimensional space, not easily observed by humans. Decision Trees and XGBoost models outperformed ENDF by learning a set of if-then-else rules based on important features from not only other X(n,p)Y reactions but also other reaction channels. Not only was the magnitude of the newly measured datapoints more accurately described by all three models, but the expected trend after 10^6 eV seems similar to that of the ENDF evaluation even when no known datapoints were available in the entire dataset constraining the calculations. There is certainly room for improvement. Due to the if-then-else nature, the DT predictions have a step-like behavior. Post-processing smoothing can be applied to make the $1/v$ region and the resonance peaks smoother. The same can be applied to the KNN model predictions.

The KNN model predictions in the low energy region are surprisings. Being a simple model, the queries at points below 10^2 eV should be flat as there are no other nearby points. The model seems to have found closer points using the features provided for training. The smoothness at these energies and at higher energies are due to the low amount of data available. As mentioned previously, the KNN model is simply as good as the dataset completeness. It does not "learn" rules but rather looks at the nearby space to come up

Table 6.5: Evaluated Libraries and ML Prediction Values for the ^{35}Cl(n,p)^{35}S LBNL New Measurements.

Energy (eV)	Cross Section (b)	KNN (b)	Decision Tree (b)	XGBoost (b)	ENDF (b)
2420000.0	0.0166	0.040723	0.082000	0.099926	0.164866
2420000.0	0.0196	0.040723	0.082000	0.099926	0.163792
2520000.0	0.0261	0.045894	0.091000	0.072726	0.171915
2520000.0	0.0257	0.045894	0.091000	0.072726	0.170763
2580000.0	0.0446	0.054121	0.078307	0.091611	0.174243
2580000.0	0.0417	0.054121	0.078307	0.091611	0.173075
2640000.0	0.0500	0.059396	0.070000	0.060848	0.176602
2640000.0	0.0478	0.059396	0.070000	0.060848	0.175418
2740000.0	0.0284	0.064900	0.101000	0.104728	0.177794
2740000.0	0.0281	0.064900	0.101000	0.104728	0.178993
2740000.0	0.0315	0.064900	0.101000	0.104728	0.181417
2740000.0	0.0324	0.064900	0.101000	0.104728	0.180201
MAE		0.020658	0.065100	0.06111953	0.145430

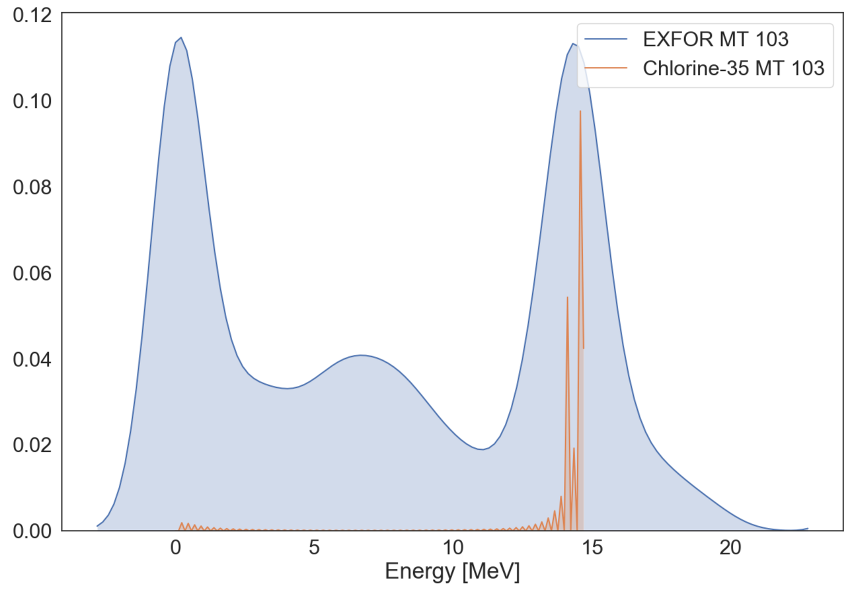


Figure 6.10: Energy Distribution of all the (n,p) reaction datapoints in EXFOR and of the Available ^{35}Cl(n,p)^{35}S values.

with a value based on model parameters. The DT model on the other hand can take further advantage of the dataset due to its higher complexity and learning capabilities. The XGBoost model, being an ensemble technique where new trees are built to correct previous mistakes, is more powerful and behaves somewhat similar to the DT model. The DTs used here are based on the scikit-learn CART implementation. XGBoost models also built CART-type trees. Interestingly, both the DT and XGBoost models predict resonances in the unknown fast energy region with similar magnitudes and overall treed. Of course, without any experimental data, it is rather difficult to validate these models. However, the fact that various models of different architecture and design come to similar conclusions might mean that the evaluation needs to be revisited. Overall, the KNN model predicted the new LBNL/UCB measurements more accurately but the XGBoost model seems to be more generalizable.

Figure 6.10 shows an energy distribution of all the (n,p) reaction datapoints in the entire EXFOR library and for the ^{35}Cl(n,p)^{35}S data point energy values. It is evident that these models might also be informed by not only other X(n,p)Y reactions in the vicinity but also other reaction channels. This makes any of these algorithms capable of predicting the cross section values rather accurately.

There is certainly room for improvement. The XGB, DT, and KNN model seem to be overfitting some of the EXFOR datapoints in the 1/v region. The DT model is more resistant to the other conflicting experimental campaigns but presents step-wise behavior, similar to the XGBoost model. More ”smooth” models like Neural Networks might create more consistent and stable results in all energy regions. In any case, these proof-of-concept

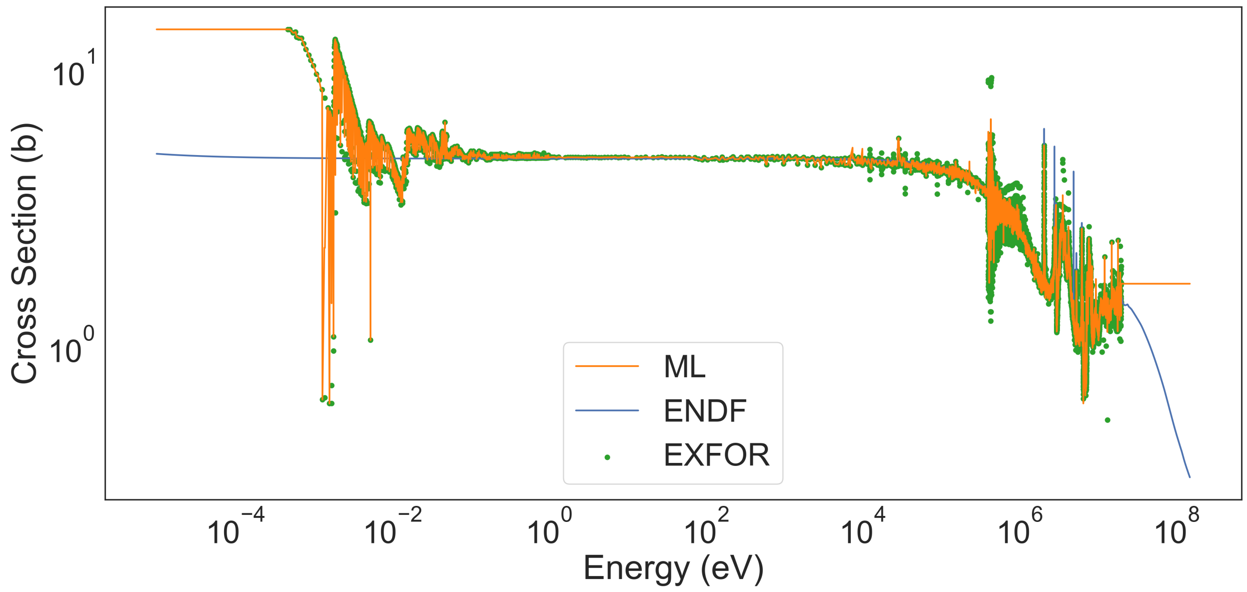


Figure 6.11: ^{12}C(n,tot) Cross Section Values in EXFOR.

models demonstrate the capability for aiding cross section evaluations in areas of uncertainty. While the generated values are not completely correct, it can assist the evaluator in key decision-making scenarios. Benchmark data using Chlorine-35 is also needed for validation of predicted cross sections.

6.4 Other Individual Isotopes of Interest

These trained models can be used to generate data for any isotope of interest. For simplicity, in this section, we predict cross section values for some of the most popular isotopes and reaction channels. The trained models can be found and downloaded from the public Google Storage Bucket. For more information please visit `https://pedrojrv.github.io/nucml/`.

Carbon

There are various great benchmarks making use of graphite. While the cross section for several reactions of interest is small, the great amount of graphite used in some reactor concepts is thought to be big enough to make a difference. It was the intention to test graphite cross section in some graphite-containing benchmarks, however, similar to the Beryllium case, erroneous datasets were found in EXFOR. Figure 6.11 shows the available experimental points in EXFOR. Some low-energy values appear to be incorrect measurements or unprocessed/raw data types. Unfortunately, the current database does not allow us to distinguish

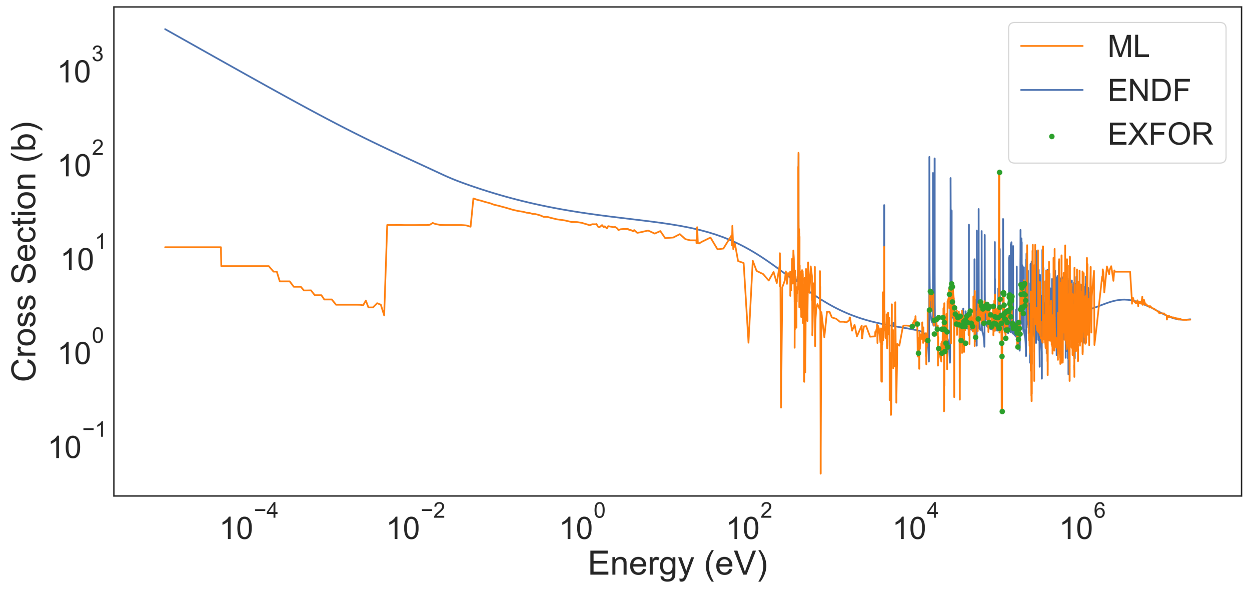


Figure 6.12: ^{35}Cl(n,tot) Cross Section Values in EXFOR.

these efficiently for all reactions. As part of the modernization effort, these datasets were manually tagged as outliers. New models should be trained on cleaner datasets.

Chlorine

Another reaction of interest is ^{35}Chlorine(n,tot). Similar to the (n,p) reaction case, there is very little data as shown by Figure 6.12. While these models are rather coarse and unstable, it is interesting to see that some resonance rules are being taken across the epithermal/fast energy regions. The predictions in the low energy are unreliable and not smooth. Still, for the small amount of data it is promising to see the correct trend being predicted. More powerful models should be able to deal with the unstable characteristics at low energies.

6.5 EXFOR-SQL

As part of the ongoing nuclear data modernization effort performed as part of this project. EXFOR SQL, a modernized version of the EXFOR database hosted on Google BigQuery for easy data analysis and extraction was designed, built, and publish. Google BigQuery is:

> Serverless, highly scalable, and cost-effective multi-cloud data warehouse designed for business agility. BigQuery is an in-memory analysis service built into BigQuery that enables users to analyze large and complex datasets interactively with sub-second query response time and high concurrency.

This will allow researchers to quickly navigate and extract data from EXFOR in a much faster and flexible manner compared to the current EXFOR database implementation. Google BigQuery has a variety of APIs for all the most popular algorithms making it more accessible to a range of expertise. It is through this type of modernization that ML can start to be applied successfully to the nuclear data field. Not only is EXFOR part of EXFOR-SQL but also the Atomic Mass Evaluation and the Reference Input Parameter Library.

One additional advantage of these types of deployments is the easy insertions and updates to any data element or group of elements. The datasets identified as outliers in this work including those of Beryllium and Iron are marked as "outliers" in EXFOR SQL. These types of tags are purely based on personal assessments but have had a positive impact on model performance relative to using all raw data points. More information can be found at `https://pedrojrv.github.io/projects/`.

Chapter 7

Conclusion and Future Work

Nuclear data evaluations are important for every industry and technology that utilizes nuclear-based technology. For example, the next generation of nuclear reactors including Molten Chloride Fast Reactors needs accurate cross section data to be designed safely and reliably. Organizations in charge of producing evaluated libraries have an important task at hand and evaluators must therefore be given the right tools to aid this difficult process. Current tools include physics-based reaction modeling codes (i.e EMPIRE and TALYS). These are guided by experimental databases like EXFOR and are used to create an appropriate fit that explains most of the experimental data points. However, some evaluations are inevitably created with higher uncertainty and human bias, especially in reactions where experimental data is not available to constrain the model calculations. In this work, we attempt to provide ML-based tools to aid the evaluator in these areas of uncertainty. To tackle this challenge, data sources and pipelines need to be modernized as the existing infrastructure is not compatible with state-of-the-art industry tools. For example, simple utilities or APIs for access to ML-ready reaction data are not available. The lack of research in the area can be attributed to many factors including the lack of specialized tools to navigate the current and future proposed pipelines in a modular and modern way.

In this work, we have (1) propose and built a framework for expedited ML-augmented nuclear evaluations and (2) trained various algorithms to infer neutron-induce cross sections. The proposed pipeline consists of data collection and processing, model training, evaluation and compilation, and validation using criticality benchmark calculations. To navigate seamlessly through these steps, NucML was created. NucML is the first and only end-to-end python-based supervised machine learning pipeline for enhanced bias-free nuclear data generation and evaluation. These tools are not meant to replace the evaluator or the physics-guided tools but to enhance their analytical power, extract meaningful physics, and modernize the current data pipelines for compatibility with current ML technologies. It offers various capabilities including dataset parsing and compilation of reaction data functionalities, exploratory data analysis tools, data manipulation and featuring engineering utilities, model training and evaluation scripts, and automatic validation via criticality benchmark frameworks.

Using this framework, several ML algorithms were trained using data from EXFOR, AME, and RIPL. Several sub-datasets were created to test feature importances. The results obtained show the capability of ML algorithms to inform evaluations. K-Nearest-Neighbor, Decision Tree, and XGBoost models performed adequately compared to the EXFOR datapoints and similar to the ENDF evaluation as shown by both the MSE and MAE performance metrics in several reactions of interest. Several optimized models also produced satisfactory results on all three tested criticality benchmarks. Some models even outperformed the current release of the ENDF library by up to 170% in the ^{233}U Jezebel Benchmark case, a well-characterized assembly in literature. On average, the Decision Tree models outperformed the evaluation on all three tested criticality benchmarks. Furthermore, the (n,p) cross section for ^{35}Cl, a less studied but important nuclide for advanced nuclear reactors, was investigated. Optimized models, reliant on learned patterns and behaviors of other X(n,p)Y reactions, predicted the LBNL/UCB measurements more accurately than any of the evaluated data libraries which overestimate experimental results by up to a factor of five. These and other examples presented in this work demonstrate the potential for ML models to aid traditional physics-guided models.

Future work includes using more powerful algorithms that can generalize better. The models presented here present instabilities that can be fixed by (1) developing custom loss functions and (2) implementing physics constraints on several elements of the model development pipeline. With the correct data processing schemes, more complex algorithms including RNN, LSTM, and even Transformer may be trainable if Energy is treated as a sequential (time-series type) feature. The current state of EXFOR also needs to be improved to allow multi-output models to accurately propagate uncertainties. Other potential development efforts include:

- Full implementation of the theoretical-ML hybrid modeling system. This includes the incorporation of physics-guided tools like TALYS and EMPIRE into the cross section library generation suite.
- Monte-Carlo-based loss function for Criticality Benchmark guided training. Custom loss functions can initiate external processes including criticality benchmark calculations. The loss can then be calculated based on the determined multiplication factor. While this is computationally expensive, results will be physics-guided and constrained.
- Dataset quality-aware training and assessment. Many outliers exist on the EXFOR database. ML-based outlier detection can aid models to score better validation performance metrics.
- Support for processing codes like NJOY into the ML-NDE pipeline.
- Future ML-guided proposals. Optimized models can help future efforts locate areas of interest and need that can in turn benefit the ML-NDE pipeline.

- Data Augmentation for high energy reaction. Knowing that (n,2n), (n,4n) are high energy reactions, augmented the data with cross section values of 0 for low energies can provide stability to the training process. It is a valid and easy way of constraining the model to learn these physics.
- Incorporate proton-induced reactions.

By having a role in the nuclear data evaluation pipeline, Machine Learning based solutions can help take human-bias-free informed decisions in areas of uncertainty. These tools aim to augment and enhance our learning capabilities, not to replace existing methods and pipelines.

Bibliography

[1] J. C. Batchelder, S. A. Chong, et al. “Possible evidence of nonstatistical properties in the ^{35}Cl(n, p)^{35}S cross section”. In: *Physical Review C* 99.4 (2019), pp. 1–12. ISSN: 24699993. DOI: 10.1103/PhysRevC.99.044612.

[2] Lee A. Bernstein, David A. Brown, et al. “Our Future Nuclear Data Needs”. In: *Annual Review of Nuclear and Particle Science* 69.1 (2019), pp. 109–136. ISSN: 0163-8998. DOI: 10.1146/annurev-nucl-101918-023708.

[3] J. Blair Briggs, Lori Scott, and Ali Nouri. “The International Criticality Safety Benchmark Evaluation Project”. In: *Nuclear Science and Engineering* 145.1 (2003), pp. 1–10. DOI: 10.13182/NSE03-14. eprint: https://doi.org/10.13182/NSE03-14. URL: https://doi.org/10.13182/NSE03-14.

[4] R. Capote et al. “RIPL – Reference Input Parameter Library for Calculation of Nuclear Reactions and Nuclear Data Evaluations”. In: *Nuclear Data Sheets* 110.12 (2009). Special Issue on Nuclear Reaction Data, pp. 3107–3214. ISSN: 0090-3752. DOI: https://doi.org/10.1016/j.nds.2009.10.004. URL: https://www.sciencedirect.com/science/article/pii/S0090375209000994.

[5] M B Chadwick, P Oblo, et al. “ENDF / B-VII . 0 : Next Generation Evaluated Nuclear Data Library for Nuclear Science and Technology”. In: *Nuclear Data Sheets* 107 (2006), pp. 2931–3060. DOI: 10.1016/j.nds.2006.11.001.

[6] Tianqi Chen and Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’16. San Francisco, California, USA: ACM, 2016, pp. 785–794. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: http://doi.acm.org/10.1145/2939672.2939785.

[7] Zhigang Ge et al. “CENDL-3.2: The new version of Chinese general purpose evaluated nuclear data library”. In: *European Physical Journal Web of Conferences*. Vol. 239. European Physical Journal Web of Conferences. May 2020, p. 09001. DOI: 10.1051/epjconf/202023909001.

[8] Jacob Goldberger et al. “Neighbourhood Components Analysis”. In: *Advances in Neural Information Processing Systems* 17 (2005), pp. 513–520.

[9] Tatjana Jevremovic. *Nuclear Principles in Engineering.* Boston, MA: Springer US, 2009. ISBN: 978-0-387-85607-0. DOI: 10.1007/978-0-387-85608-7. URL: http://link.springer.com/10.1007/978-0-387-85608-7.

[10] P. E. Koehler. "^{35}Cl(n,p)^{35}S cross section from 25 meV to 100 keV". In: *Phys. Rev. C* 44 (4 Oct. 1991), pp. 1675–1678. DOI: 10.1103/PhysRevC.44.1675. URL: https://link.aps.org/doi/10.1103/PhysRevC.44.1675.

[11] A.J. Koning et al. "TENDL: Complete Nuclear Data Library for Innovative Nuclear Science and Technology". In: *Nuclear Data Sheets* 155 (2019). Special Issue on Nuclear Reaction Data, pp. 1–55. ISSN: 0090-3752. DOI: https://doi.org/10.1016/j.nds.2019.01.002. URL: http://www.sciencedirect.com/science/article/pii/S009037521930002X.

[12] Jaakko Leppänen, Maria Pusa, et al. "The Serpent Monte Carlo code : Status , development and applications in 2013". In: *Annals of Nuclear Energy* 82 (2015), pp. 142–150. DOI: 10.1016/j.anucene.2014.08.024.

[13] Martın Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[14] G G Moisen and U S Forest Service. "Classification and Regression Trees". In: *Ecological Informatics* 2000 (2008), pp. 582–588.

[15] N. Otuka, E. Dupont, et al. "Towards a More complete and accurate experimental nuclear reaction data library (EXFOR): International collaboration between nuclear reaction data centres (NRDC)". In: *Nuclear Data Sheets* 120 (2014), pp. 272–276. ISSN: 00903752. DOI: 10.1016/j.nds.2014.07.065.

[16] Fabian Pedregosa, Ron Weiss, and Matthieu Brucher. "Scikit-learn : Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[17] Plompen, A. J. M. et al. "The joint evaluated fission and fusion nuclear data library, JEFF-3.3". In: *Eur. Phys. J. A* 56.7 (2020), p. 181. DOI: 10.1140/epja/s10050-020-00141-9. URL: https://doi.org/10.1140/epja/s10050-020-00141-9.

[18] Y. P. Popov and Shapiro F. L. "The Cl35(n,p) reaction and neutron resonance parameters of chlorine". In: *Journal of Experimental and Theoretical Physics* 13 (6 1961), pp. 1132–1135.

[19] Keiichi SHIBATA et al. "JENDL-4.0: A New Library for Nuclear Science and Engineering". In: *Journal of Nuclear Science and Technology* 48.1 (2011), pp. 1–30. DOI: 10.1080/18811248.2011.9711675. eprint: https://doi.org/10.1080/18811248.2011.9711675. URL: https://doi.org/10.1080/18811248.2011.9711675.

[20] Meng Wang, G. Audi, et al. "The AME2016 atomic mass evaluation (II). Tables, graphs and references". In: *Chinese Physics C* 41.3 (2017). ISSN: 16741137. DOI: 10.1088/1674-1137/41/3/030003.

Appendix A

K-Nearest Neighbors Models

A.1 Performance Metric Plots

Dataset B

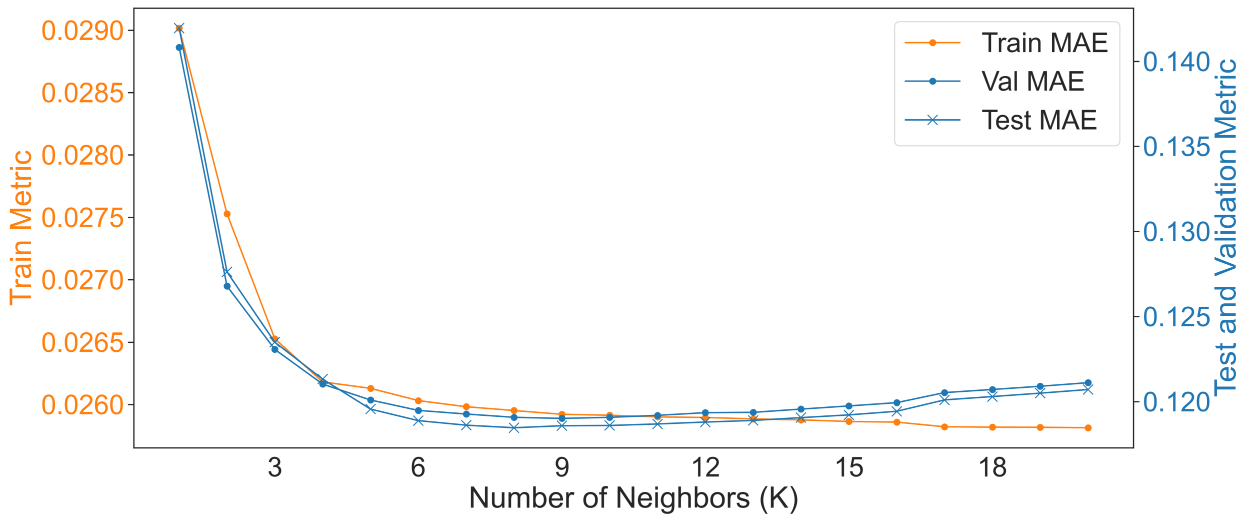


Figure A.1: Performance Metrics as a Function of the k Hyperparameter for KNN Models Trained on Dataset B.

Dataset C

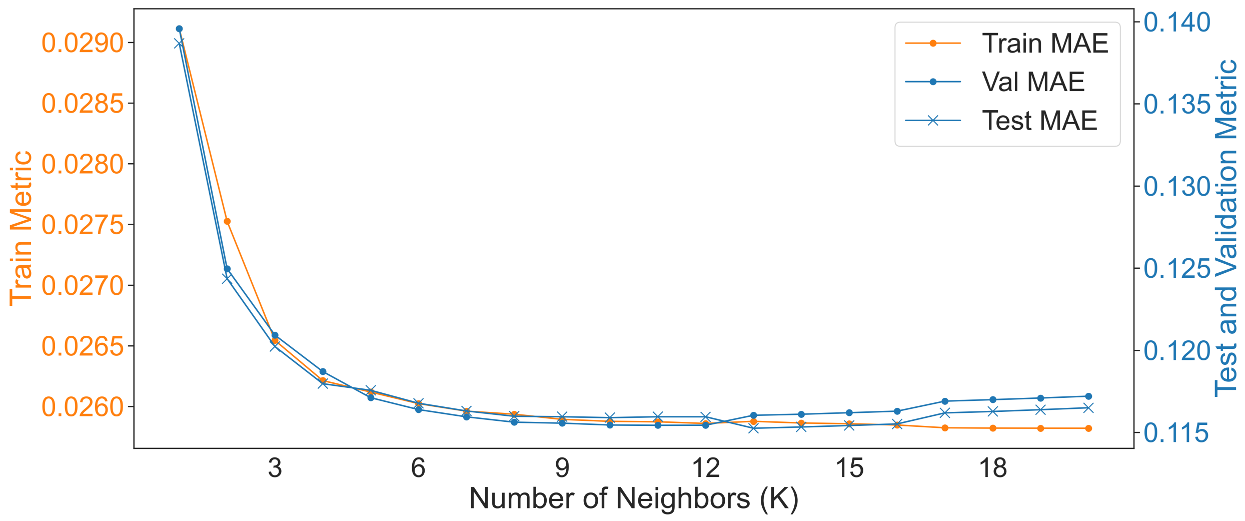


Figure A.2: Performance Metrics as a Function of the k Hyperparameter for KNN Models Trained on Dataset C.

Dataset D

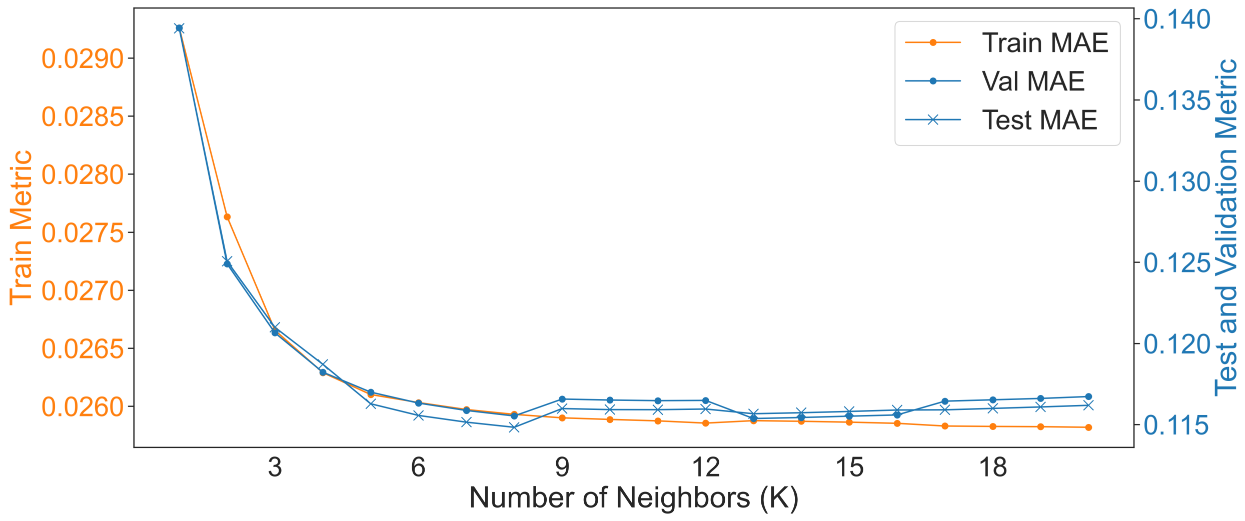


Figure A.3: Performance Metrics as a Function of the k Hyperparameter for KNN Models Trained on Dataset D.

Dataset E

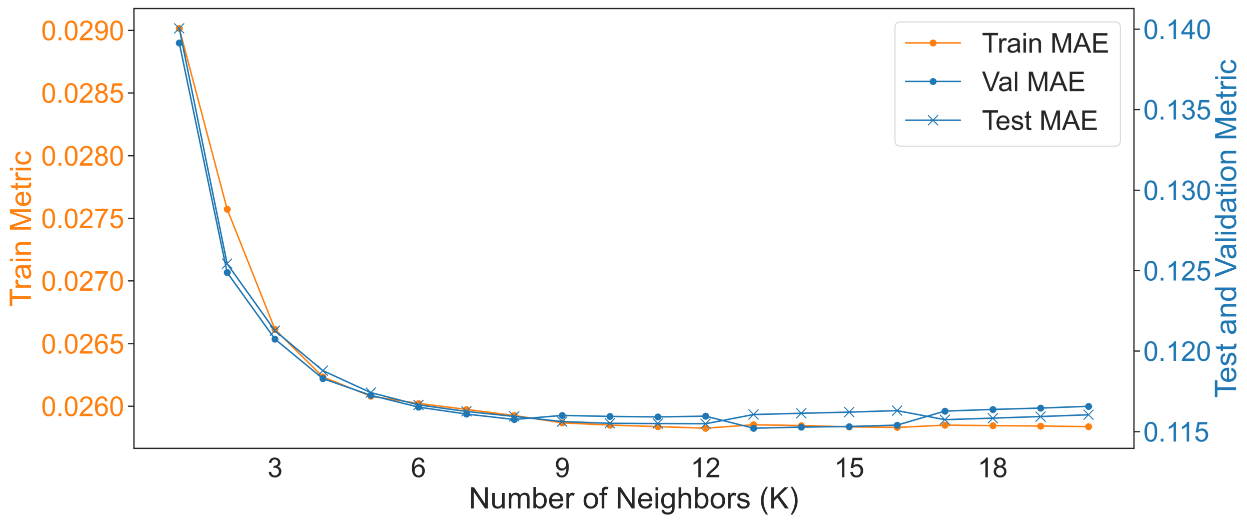


Figure A.4: Performance Metrics as a Function of the k Hyperparameter for KNN Models Trained on Dataset E.

Appendix B

Decision Tree Models

B.1 Gini Importance Plots

Dataset B

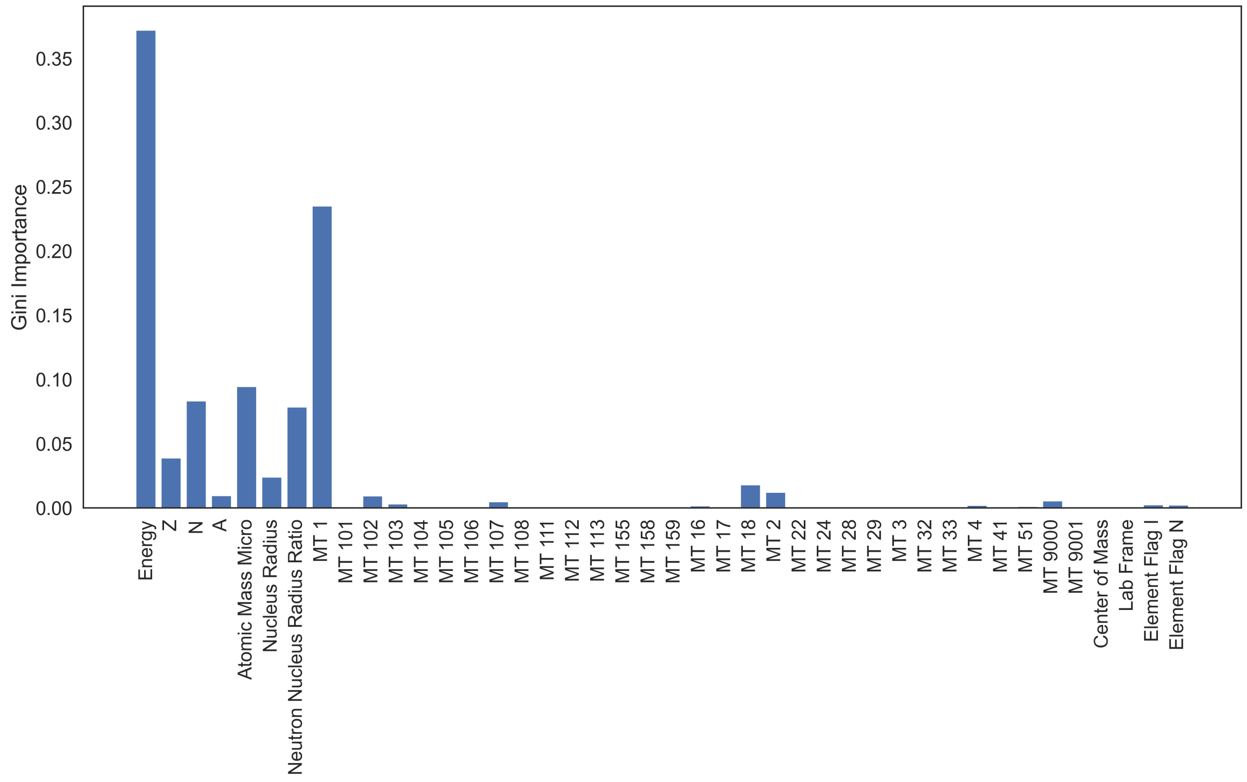


Figure B.1: Feature Gini Importance for Decision Tree model trained on Dataset B.

Dataset C

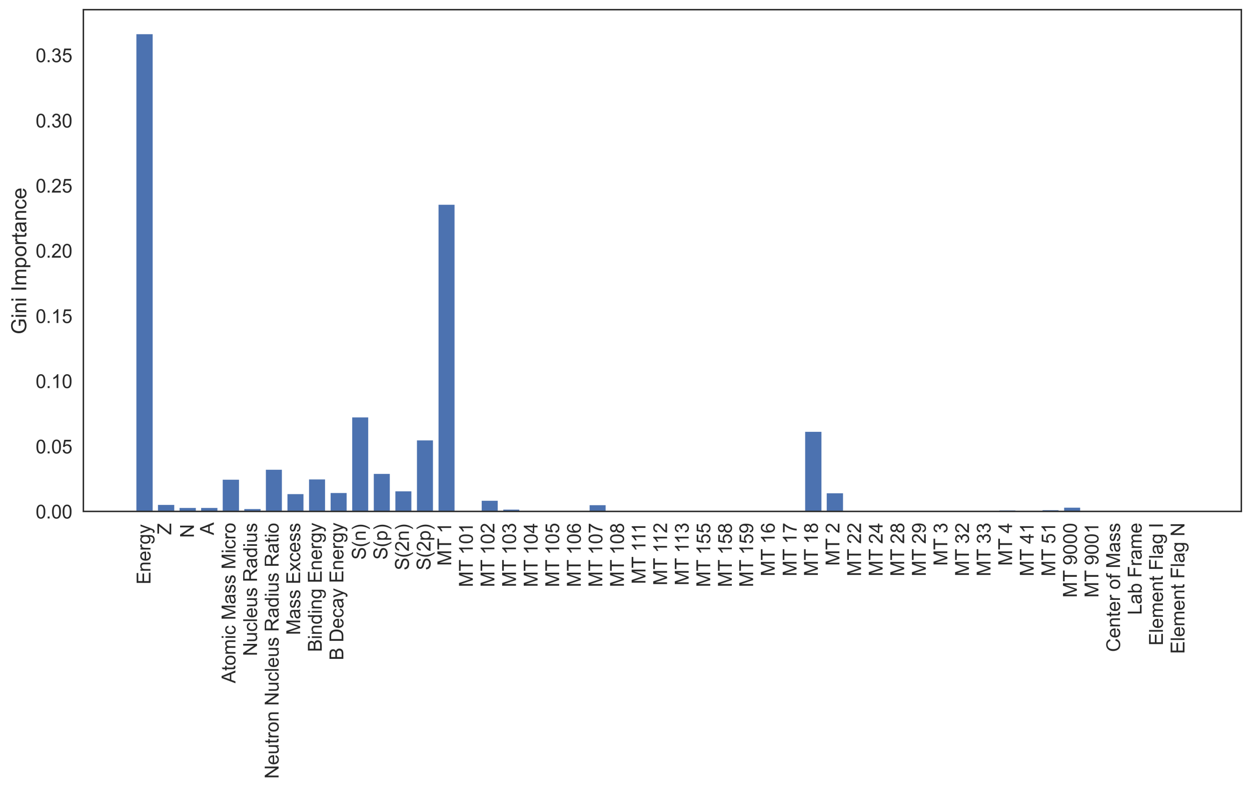


Figure B.2: Feature Gini Importance for Decision Tree model trained on Dataset C.

Dataset D

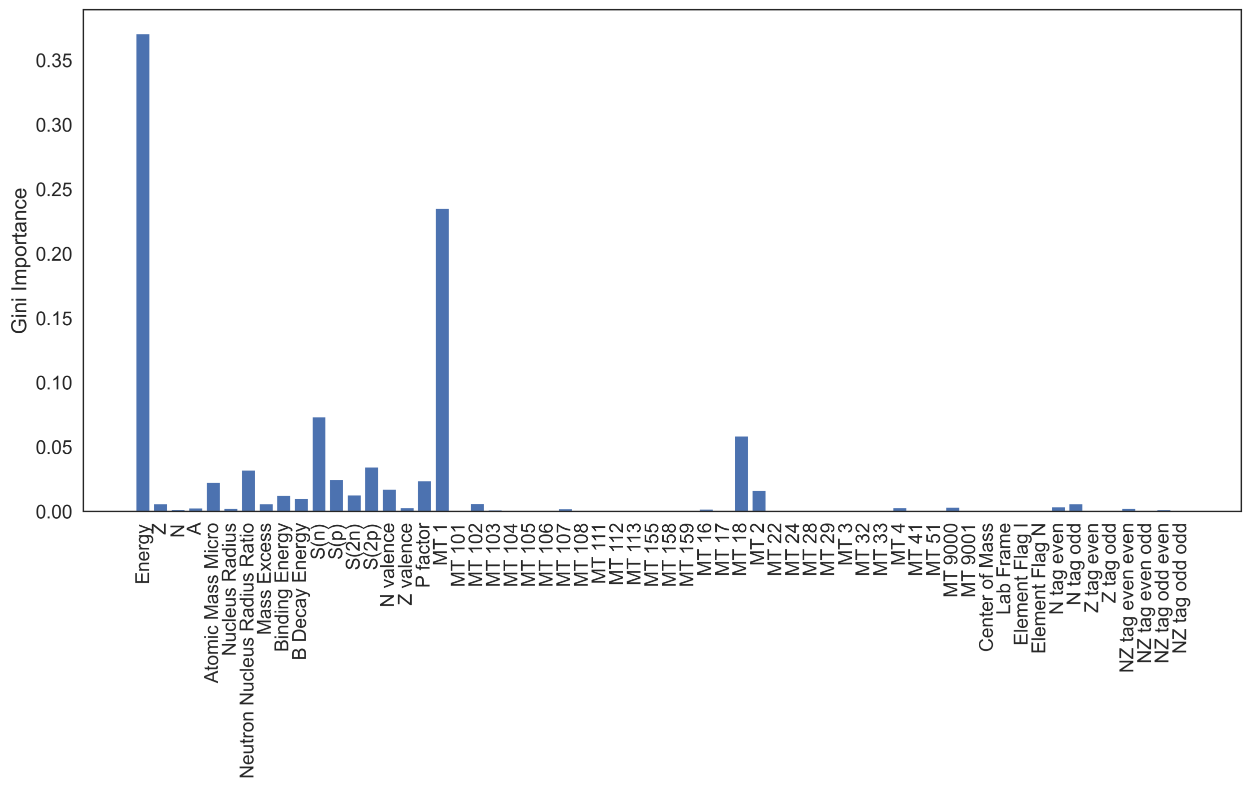


Figure B.3: Feature Gini Importance for Decision Tree model trained on Dataset D.

Appendix C

XGBoost Models

C.1 Other Hyperparameter Optimization Figures

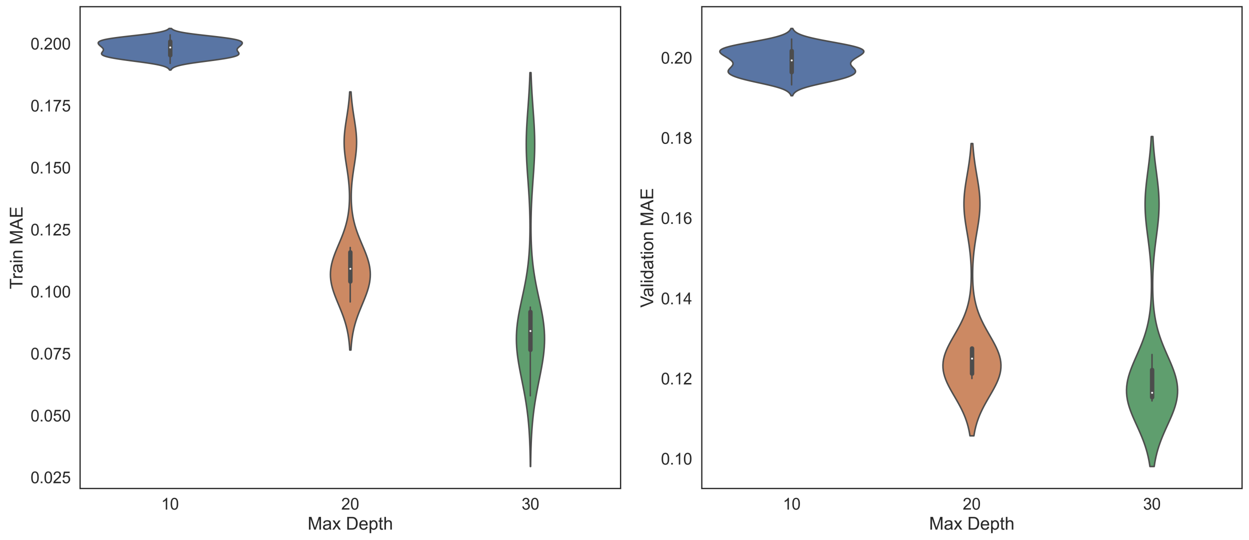


Figure C.1: Training and Validation MAE as a function of the Max Depth on XGBoost Models trained on EXFOR data. Better performance is achieved as the Max Depth is increased. Higher-memory GPUs should be used to train models with more predicting capabilities.

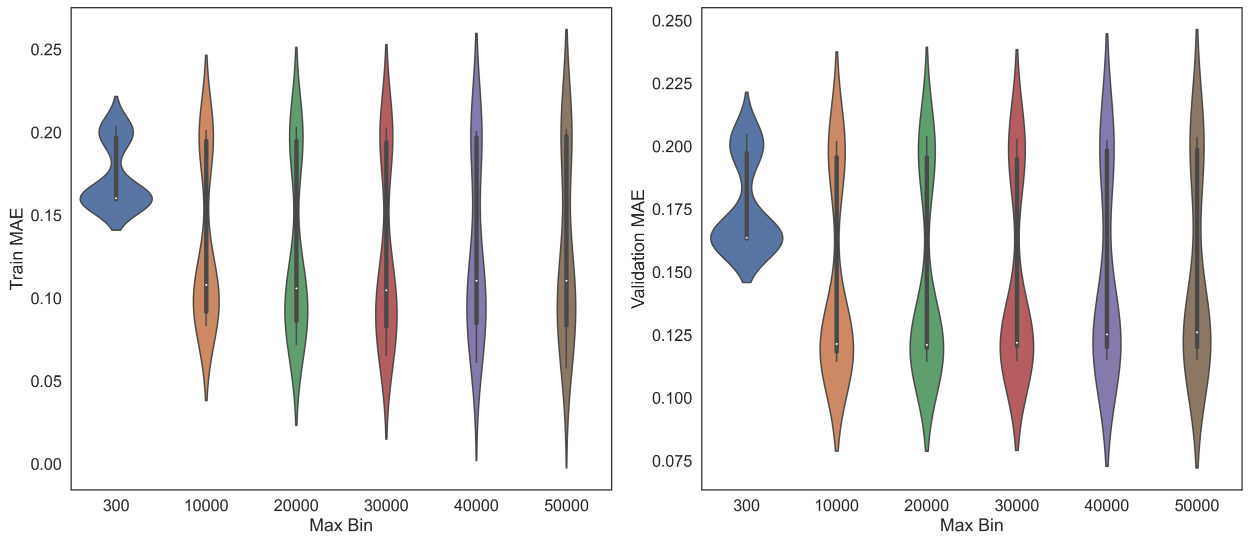


Figure C.2: Training and Validation MAE as a function of the Max Bin on XGBoost Models trained on EXFOR data. The Max Bin is an important hyperparameter to increase for optimal Training and Validation performance.

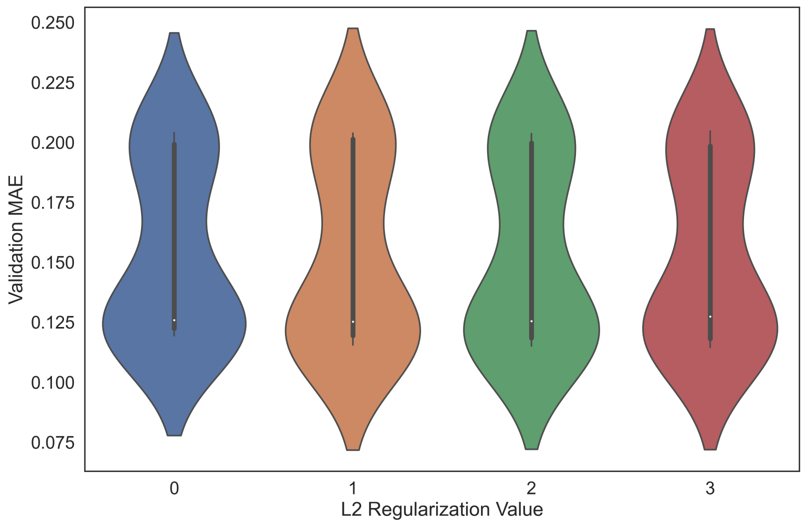


Figure C.3: Training and Validation MAE as a function of the L2 Regularization Strength on XGBoost Models trained on EXFOR data. L2 values of 2 and 3 shown slightly better overall performance.

C.2 'Gain'- and 'Weight'-based Importance Plots

Dataset B

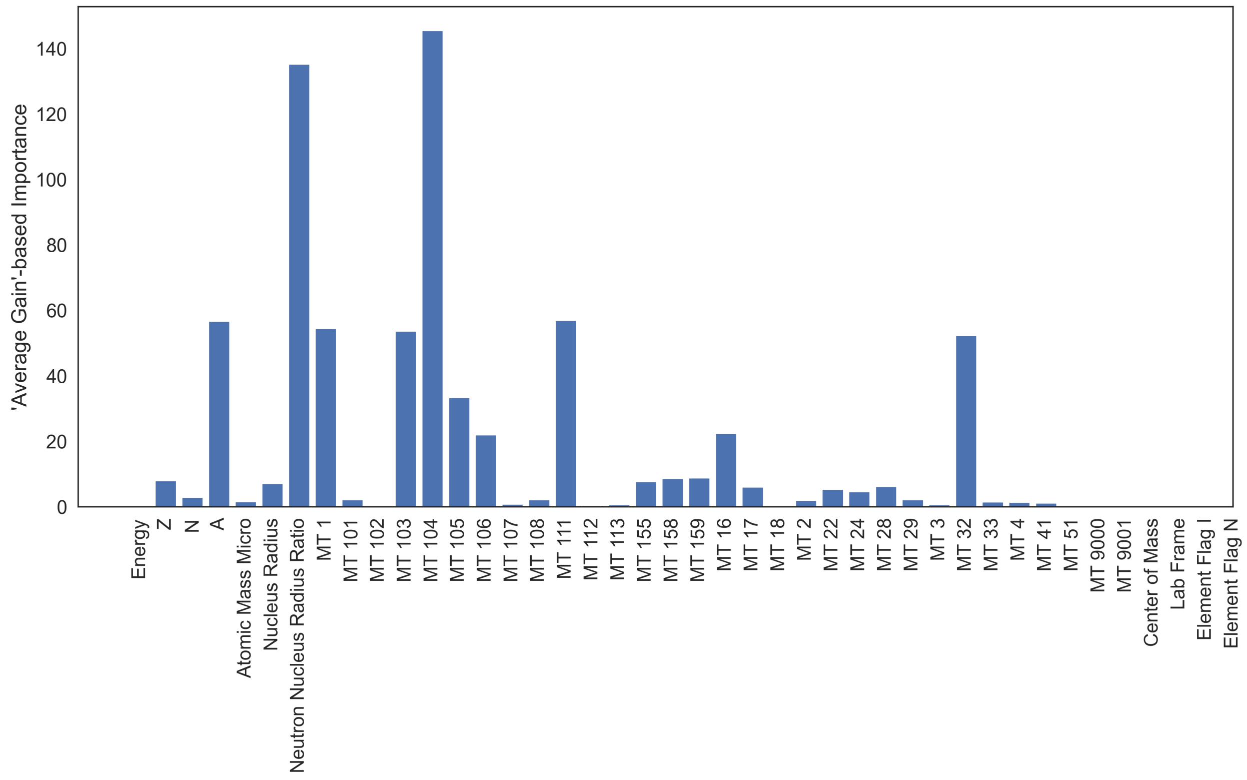


Figure C.4: Gain-based Feature Importances for the XGBoost model trained on Dataset B.

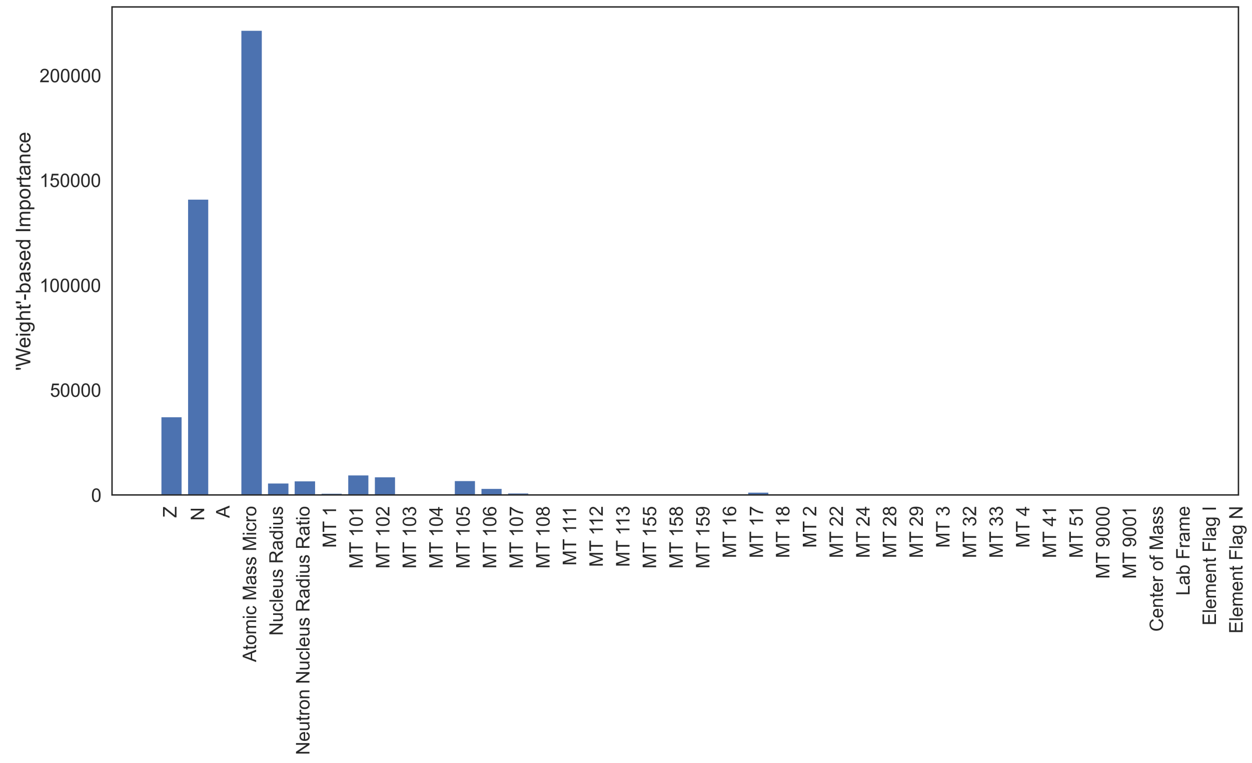


Figure C.5: Weight-based Feature Importances for the XGBoost model trained on Dataset B.

Dataset C

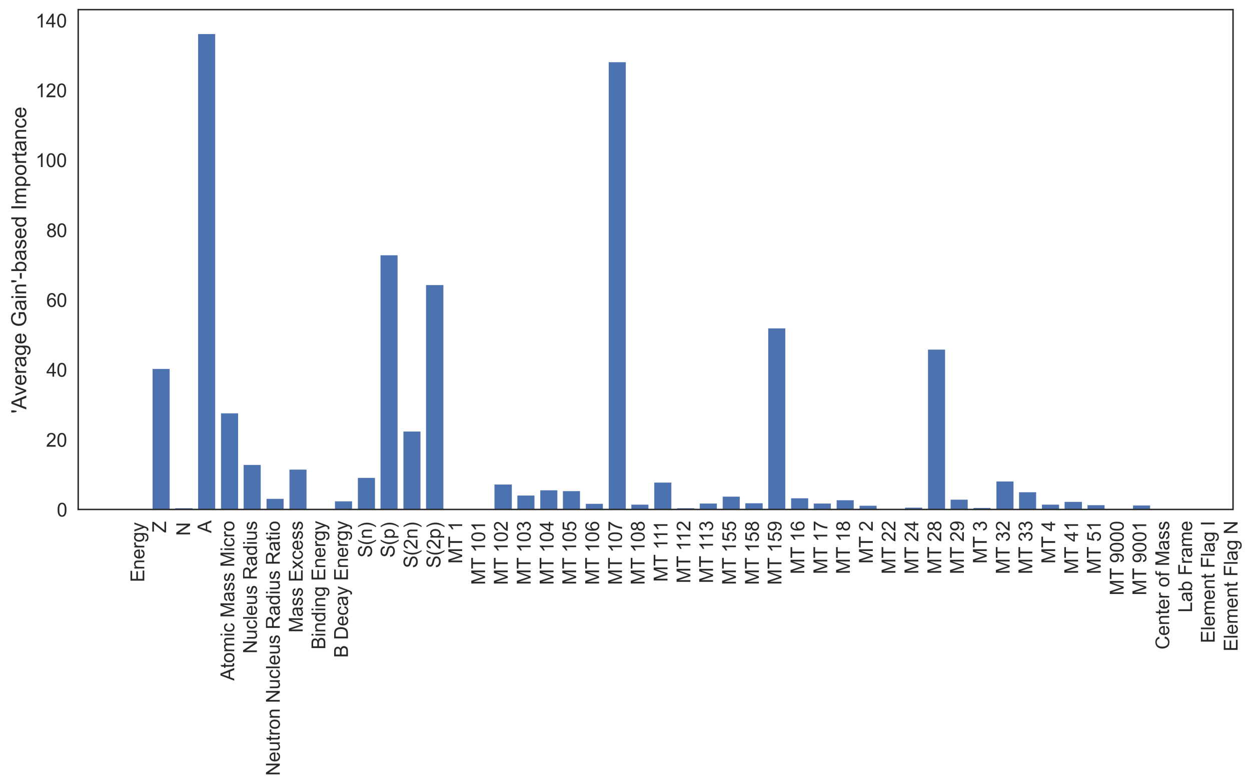


Figure C.6: Gain-based Feature Importances for the XGBoost model trained on Dataset C.

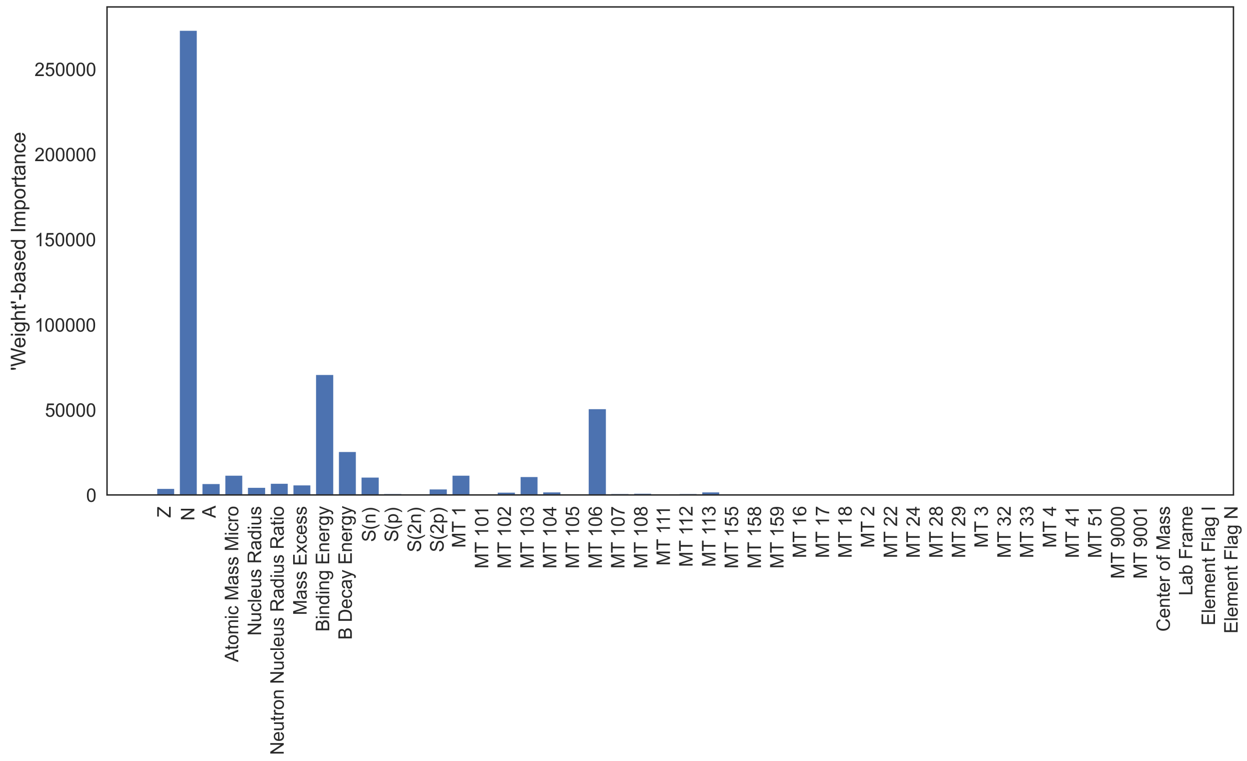


Figure C.7: Weight-based Feature Importances for the XGBoost model trained on Dataset C.

Dataset D

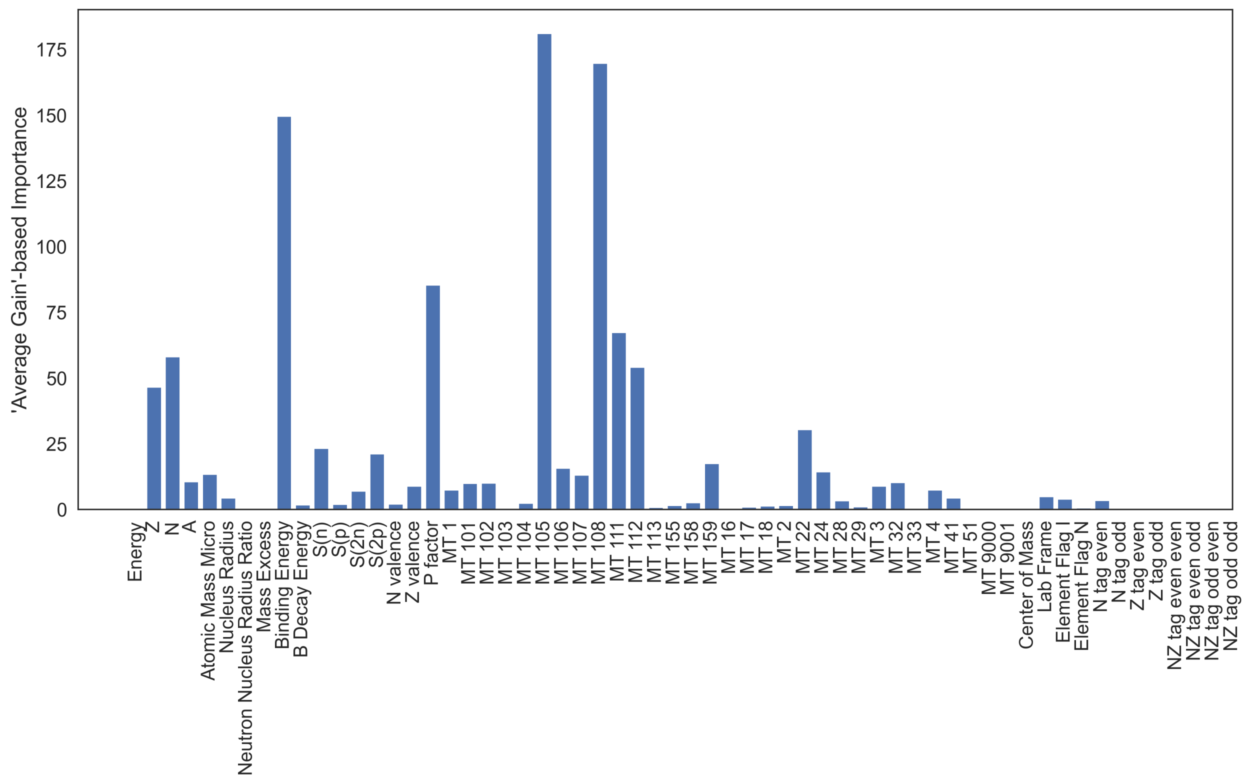


Figure C.8: Gain-based Feature Importances for the XGBoost model trained on Dataset D.

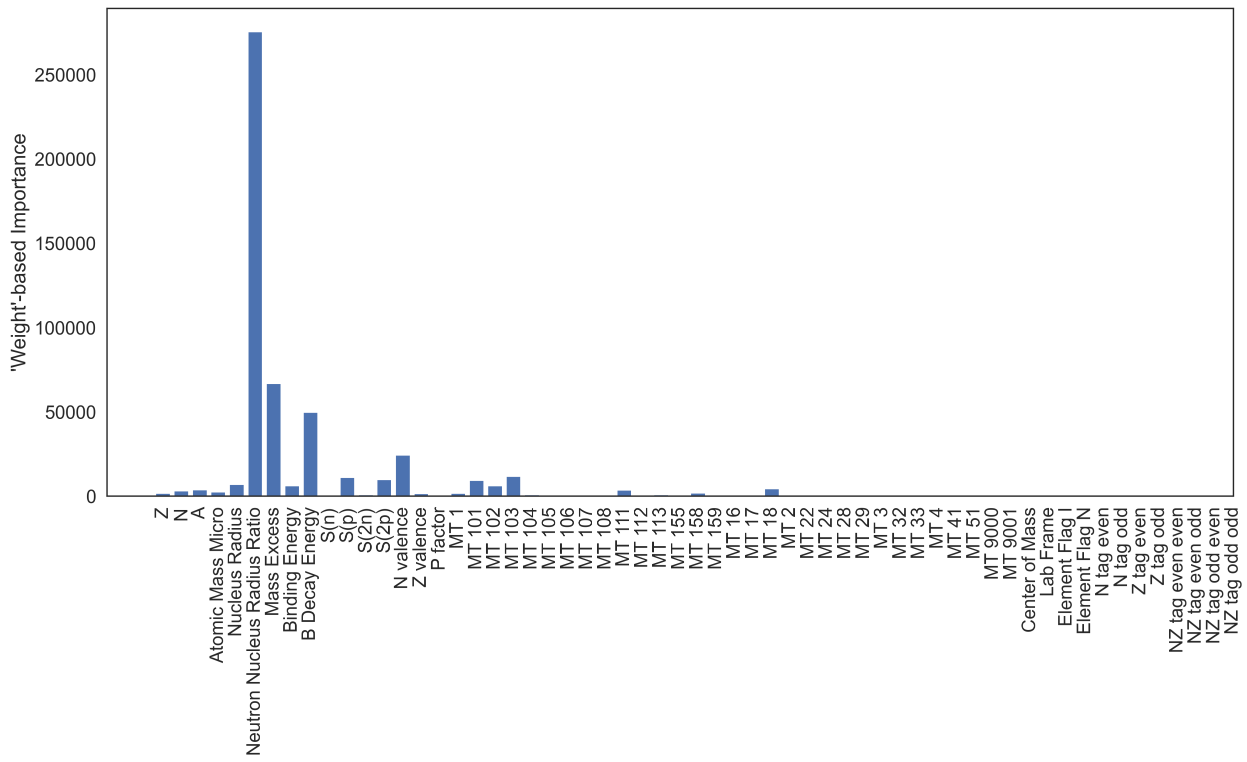


Figure C.9: Weight-based Feature Importances for the XGBoost model trained on Dataset D.

Appendix D

Incorrect/Outlier EXFOR Datasets

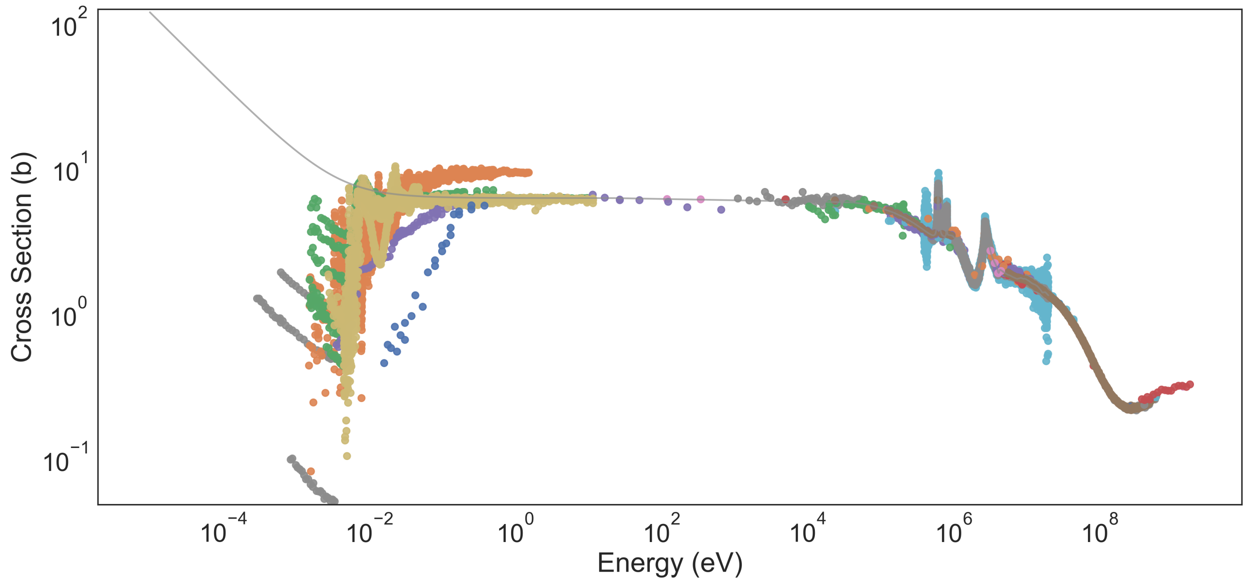


Figure D.1: ^{9}Be(n,tot) Cross Section Data from EXFOR.

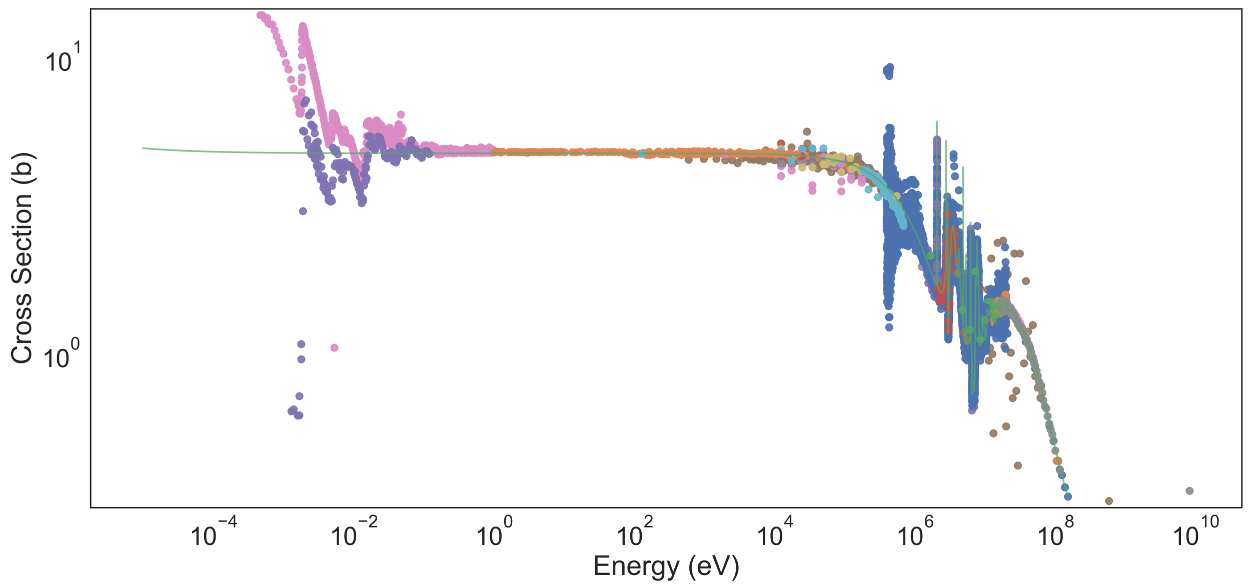


Figure D.2: ^{12}C(n,tot) Cross Section Data from EXFOR.