Prediction of Reactor Core Neutronics Parameters Based on Neural Network Hyperparameter Optimization Methods

Authors: Zhang Fan, Zhang Junda, Sun Qizheng, Xiao Wei, Liu Xiaojing, Zhang Tengfei, Tengfei Zhang

Date: 2024-06-29T00:00:00+00:00

Abstract

Neural networks can learn the relationship between input and output variables from large datasets, exhibit strong approximation capabilities, and are frequently employed as surrogate models for programs in domains such as nuclear engineering calculations. Neutron transport calculation, as one of the core components of neutronics simulation, suffers from the problem of long computational times that can be alleviated through the utilization of neural network models. However, neural network models entail a series of hyperparameters requiring configuration, and manual tuning of these hyperparameters is labor-intensive, repetitive, tedious, relies solely on empirical experience, and such hyperparameters are not transferable across different problems. To address these challenges, this paper proposes a methodology employing the Bayesian Optimization algorithm to tune neural network hyperparameters, incorporating learning rate decay and loss function optimization techniques. This approach can automatically search for optimal hyperparameter combinations tailored to datasets from different problems to achieve peak performance, exhibiting high flexibility, efficiency, and strong generalization capability. This paper fits key core parameters obtained from the TAKEDA benchmark problem, with results demonstrating that the average error of the effective multiplication factor keff remains within 150 pcm, the average error rate of the regional integrated flux Φ on the TAKEDA1 dataset is 1.72%, and the maximum error rate is 7.56%. This study can provide a reference for the application of artificial intelligence in core physics calculation theory.

Full Text

Research on Core Neutronic Parameter Prediction Based on Neural Network Hyperparameter Optimization Method

ZHANG Fan[1], ZHANG Junda[2], SUN Qizheng[2], XIAO Wei[2], LIU Xiaojing[2], ZHANG Tengfei[2]

[1] College of Smart Energy, Shanghai Jiao Tong University, Shanghai 200240, China

[2] School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China

Abstract

[**Background**] Neural networks possess powerful fitting capabilities and can learn relationships between input and output variables from large datasets, making them effective surrogate models for physics-based codes in engineering calculations, including nuclear engineering. Neutron transport calculations, as a core component of neutronics simulations, suffer from lengthy computational times that can be mitigated through neural network models. However, neural networks require careful configuration of numerous hyperparameters. Manual hyperparameter tuning is labor-intensive, repetitive, and relies heavily on empirical experience, with hyperparameters lacking transferability across different problems.

[**Purpose**] This study seeks to develop an efficient surrogate model for VITAS to provide a reference for artificial intelligence applications in reactor physics calculations.

[**Methods**] We propose a Bayesian optimization algorithm for neural network hyperparameter tuning, integrated with learning rate decay and loss function optimization techniques.

[**Results**] Using datasets generated from VITAS calculations of the TAKEDA benchmark problems, the results demonstrate that the average error of the effective multiplication factor (k_{eff}) remains within 150×10^{-5}. For the TAKEDA1 dataset, the average error rate of regional integral flux (ϕ) is 1.72%, with a maximum error rate of 7.56%.

[**Conclusions**] This approach can automatically search for optimal hyperparameter combinations tailored to different datasets, achieving superior performance with high flexibility, efficiency, and strong generalization capability.

Keywords: Bayesian optimization for hyperparameter tuning, FCNN, Neutron transport computation, Learning rate decay, Loss function optimization methods

Introduction

Monte Carlo methods and deterministic approaches represent the two primary classes of methods for solving neutron transport problems in nuclear reactor physics calculations. Commonly used deterministic neutron transport methods include the Finite Element Method (FEM), Method of Characteristics (MOC), and Variational Nodal Method (VNM) [?, ?, ?]. These traditional neutron transport methods are characterized by low computational efficiency and long execution times, making them impractical for frequent core design optimization and fuel reloading studies.

Machine learning and artificial intelligence offer significant advantages in handling complex big data compared to traditional Monte Carlo and deterministic neutron transport methods. In recent years, interdisciplinary applications between nuclear physics and machine learning have emerged with promising prospects. Song et al. [?] developed a core parameter prediction program based on BP (back propagation) artificial neural networks achieving relative errors within 10%, which sparked subsequent research interest. Akkoyun [?] employed artificial neural networks to predict total fusion and fusion-evaporation reaction cross-sections, while Guo et al. [?] used a hybrid artificial neural network to simulate the thermodynamic behavior of the Sequoyah nuclear power plant.

Among these approaches, the Fully Connected Neural Network (FCNN) [?] represents a relatively simple artificial neural network architecture belonging to the feedforward neural network family. FCNNs consist solely of an input layer, hidden layers, and an output layer, with multiple neurons possible in each hidden layer. This versatile learning method can be applied to nearly all tasks, including classification, regression, and unsupervised learning. Machine learning has experienced rapid development due to its ability to fit complex analytical computational processes while reducing manual and computational costs.

VITAS [?, ?] is a general-purpose computational program developed based on VNM for accurately solving neutron transport problems. It integrates multiple computational methods and employs matrix operations and numerical integration techniques to handle multi-dimensional, multi-group, steady-state, and transient neutron transport problems with various mesh types. To address the inherent low efficiency and long computation times of traditional transport methods, this study develops a neural network-based surrogate model for core transport calculations to replace the computational functionality of the VITAS program.

However, neural networks require configuration of numerous hyperparameters that critically determine FCNN performance. Manual hyperparameter adjustment demands substantial effort, requires considerable experience, and incurs high costs. With the continuous evolution of machine learning, deep learning [?, ?] modeling involves far more hyperparameters than traditional machine learning methods, rendering manual tuning inadequate for both precision and time constraints. Bayesian optimization [?] has gained increasing popularity for solving black-box function problems and has become the mainstream approach for hyperparameter optimization [?]. Known alternatively as Sequential Kriging Optimization (SKO), Sequential Model-based Optimization (SMBO), or Efficient Global Optimization (EGO) in different domains, Bayesian optimization is a global optimization method that uses a surrogate model to approximate the expensive-to-evaluate objective function. It actively selects the most "promising" evaluation points based on the surrogate model's posterior information, leveraging complete historical information to improve search efficiency and obtain optimal solutions for complex functions with minimal evaluations. Consequently, Bayesian optimization is also termed active optimization. The Bayesian optimization framework effectively utilizes complete historical information to enhance search efficiency.

This study employs Bayesian optimization combined with learning rate decay and loss function optimization methods to optimize FCNN hyperparameters. Using experimental data generated from VITAS calculations of TAKEDA benchmark problems as datasets, we evaluate the prediction accuracy of the optimized FCNN and compare the results with manually tuned networks. The objective is to provide a more efficient method for solving core reloading optimization problems and explore further applications of artificial intelligence in the nuclear industry.

1.1 Bayesian Optimization Method

The Bayesian optimization framework comprises two key components: (1) using a probabilistic model to surrogate the computationally expensive complex objective function, and (2) constructing an acquisition strategy (acquisition function) based on the surrogate model's posterior information to actively select evaluation points. In practical applications, appropriate models must be selected for specific problems. This section defines the hyperparameter optimization problem and outlines the Bayesian optimization procedure, concluding with the selected FCNN hyperparameters and their search ranges.

The hyperparameter optimization problem is defined as:

$$\arg\min_{x \in X} f(x)$$

where x represents a set of hyperparameter values and X denotes the hyperparameter search space. The function $f(x)$ represents the objective to be optimized in hyperparameter tuning. In this study, the loss function value serves as $f(x)$, with loss function selection discussed in Section 1.2 (2) on loss function optimization. The goal of hyperparameter algorithm optimization is to find the global optimum most efficiently.

The Bayesian optimization procedure for hyperparameters is as follows:

1. Randomly initialize n_0 sets of hyperparameters x_{init} in the search space X;
2. Obtain their corresponding function values $f(x)$ to form the initial point set $D_0 = \{(x_i, f(x_i))\}$, and set $t = 0$;
3. Construct a surrogate model $g(x)$ based on the currently obtained point set distribution;
4. Maximize the acquisition function based on the surrogate model $g(x)$ to obtain the next evaluation point: $x_t = \arg\max_{x \in X} \alpha(x; D_t)$;
5. Evaluate the function value $f(x_t)$ at point x_t and add it to the current evaluation set: $D_{t+1} = D_t \cup \{(x_t, f(x_t))\}$. If t is less than the maximum iteration number N, return to step (3);
6. Upon reaching the maximum iteration count, output the optimal evaluation point $\{x^*, f(x^*)\}$.

This study implements the Bayesian optimization process using TensorFlow's Bayesian Optimization function [**?**]. By inputting hyperparameters and their value ranges, the optimization is initiated. The selected FCNN hyperparameters and their ranges are presented in Table 1 .

Table 1 The selected hyperparameters and their ranges

Hyperparameters	Ranges
Batch size (batch_{size})	[1, 1000]
Number of hidden layers (num_{{{hidden}}_{{layers}}})	[1, 5]
Threshold (min_{delta})	[1e-6, 1e-4]
Decay factor (factor)	[0.1, 0.9]
Number of neurons in hidden layers (num_{neurons})	[1, 2000]
Loss function parameter (loss_{delta})	[0.1, 10]

All hyperparameters listed in the table are critical parameters in FCNN, as discussed below.

1.2 Fully Connected Neural Network

For the surrogate model in neutron transport solving, this work selects the Fully Connected Neural Network (FCNN). FCNN is a relatively simple artificial neural network architecture belonging to the feedforward neural network family, consisting only of an input layer, hidden layers, and an output layer, with multiple neurons possible in each hidden layer, as shown in Figure 1 [Figure 1: see original paper].

The entire network is built using TensorFlow, an open-source framework developed by Google in 2015 for deep learning. The FCNN training process involves the following parameters:

Epoch: The process of feeding the entire training dataset through the neural network model once for complete training is called an epoch.

Batch_{Size}: Due to computational limitations or other constraints, when data cannot be passed through the neural network all at once, the dataset must be divided into multiple batches. A small portion of data samples from the training set is used for one backpropagation parameter update.

Learning Rate (lr) refers to the step size used to adjust fitting parameters during neural network training, determining the magnitude of parameter adjustments during each gradient update. The choice of learning rate directly affects model performance and training effectiveness.

(1) Adaptive Learning Rate Adjustment

An excessively large learning rate increases the step size, causing oscillations that make it difficult to find high-precision solutions. Figure 2 [Figure 2: see original paper] (left) illustrates loss function oscillations caused by an overly large learning rate, preventing stable convergence. Conversely, an excessively small learning rate slows convergence and risks getting trapped in local optima. To address this, we propose an adaptive learning rate adjustment method that begins with a larger learning rate for rapid approximation, then gradually reduces it to accommodate model parameter update patterns, eliminating late-stage loss function oscillations and stabilizing the loss curve. Figure 2 (right) shows the loss function decay curve after adaptive learning rate adjustment, which effectively eliminates oscillations.

This study employs the ReduceLROnPlateau function from TensorFlow, a learning rate scheduler that automatically adjusts the learning rate based on monitored metric changes. When performance metrics on the validation set cease to improve, ReduceLROnPlateau gradually reduces the learning rate to facilitate better model convergence. Its parameters include:

- **lr**: Initial learning rate value;
- **factor**: The factor by which the learning rate is reduced each time, with the learning rate decreasing as $lr \times factor$;
- **patience**: The number of epochs without improvement after which the learning rate reduction is triggered;
- **min_{delta}**: Threshold; when the model performance improvement is less than min_{delta}, it is considered no improvement.

Both factor and min_{delta} are selected as hyperparameters for Bayesian optimization in this study.

(2) Loss Function Optimization

Mean Square Error (MSE) represents the average of squared differences between model predictions $f(x)$ and true sample values y, as shown in Equation (2):

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}(y_i - f(x_i))^2$$

Mean Absolute Error (MAE) represents the average of absolute differences between model predictions $f(x)$ and true sample values y, as shown in Equation (3):

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m}|y_i - f(x_i)|$$

In Equations (2) and (3), y_i and $f(x_i)$ represent the true and predicted values of the i-th sample, respectively, and m is the number of samples.

A major issue with using MAE as a loss function is that its gradient remains large, potentially causing gradient descent to overshoot the minimum during model training. In contrast, MSE's gradient decreases as the loss approaches its minimum, enabling more accurate convergence. This problem can be mitigated using Huber Loss, a regression loss function that combines characteristics of both MSE and MAE. Huber Loss reduces gradients near the minimum to prevent overshooting while being more robust to outliers than MSE. This study uses Huber Loss as the loss function, expressed in Equation (4):

$$L_\delta(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{if } |y - f(x)| \le \delta \\ \delta|y - f(x)| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases}$$

Huber Loss synthesizes MSE and MAE, containing a hyperparameter δ that determines its emphasis on MSE versus MAE. As $\delta \to 0$, Huber Loss approaches MAE; as $\delta \to \infty$, it approaches MSE. In Table 1, δ corresponds to loss_{delta}.

Since these parameter settings critically determine FCNN performance, they are designated as hyperparameters for Bayesian optimization, with their search spaces provided in Table 1.

1.3 Dataset Acquisition

The datasets used in this study were generated through VITAS calculations of the TAKEDA1 and TAKEDA2 benchmark problems, comprising 10,000 and 20,000 data groups, respectively. Each data sample consists of a core configuration and two core parameters: effective multiplication factor (k_{eff}) and regional integral flux (ϕ). Input dimensions are determined by the reactor core arrangement, while output dimensions correspond to core parameters. TAKEDA1 represents a 1/8 symmetric light water reactor model, while TAKEDA2 represents a 1/4 fast breeder reactor model.

TAKEDA1 contains three component types: Control Rod (CR), Reflector, and Core. Its three-dimensional schematic and x-y cross-section core arrangement are shown in Figure 3 [Figure 3: see original paper]. To ensure physical meaningfulness, the outermost reflector layer in the 1/4 x-y cross-section is fixed, while remaining components are shuffled and then mirrored to form a complete core, as illustrated in Figure 4 [Figure 4: see original paper]. For FCNN processing, the three components (CR, Reflector, Core) are mapped to -1, 0, and 1, respectively. This generates 10,000 distinct random configurations, with each input vector's sequence of -1, 0, 1 corresponding to the shuffled component arrangement. The input dimension is 16. Two independent FCNNs are trained to predict two physical quantities: k_{eff} with output dimension 1, and ϕ with output dimension 6.

TAKEDA2 contains three component types: Control Rod (CR), Axial Blanket, and Radial Blanket, as shown in Figure 5 [Figure 5: see original paper]. To ensure physical meaningfulness, the outermost three layers of Radial Blanket in the 1/4 x-y cross-section are fixed, while remaining components are shuffled and mirrored to form a complete core, as shown in Figure 6 [Figure 6: see original paper]. For FCNN processing, the three components (CR, Axial Blanket, Radial Blanket) are mapped to -1, 0, and 1, respectively, generating 20,000 distinct random configurations. Each input vector's sequence of -1, 0, 1 corresponds to the shuffled component arrangement, with input dimension 121. Two independent FCNNs predict two physical quantities: k_{eff} with output dimension 1, and ϕ with output dimension 20.

For FCNN training, raw data must be transformed into a standard dataset. Assuming N training samples, each training sample should be: $T_k = (X_k, Y_k)$, where X_k and Y_k represent the k-th input and output variables, respectively, with i and j denoting input and output variable dimensions. For TAKEDA1, $k = 10,000$, input vector X is a sequence of -1, 0, 1 with dimension $i = 16$. Two independent FCNNs predict two physical quantities: k_{eff} with dimension $j = 1$, and ϕ with dimension $j = 6$. For TAKEDA2, $k = 20,000$, input vector X has dimension $i = 121$. Two FCNNs predict k_{eff} ($j = 1$) and ϕ ($j = 20$). The k_{eff} values range between 0-2 and require no normalization, while ϕ values are on the order of 10^{-2} to 10^{-4} and must be normalized to the 0-1 range before neural network training.

1.4 Model Validation

This study utilizes 10,000 TAKEDA1 and 20,000 TAKEDA2 data groups, with inputs as reactor core configurations and outputs as k_{eff} and ϕ. The data are partitioned into training and validation sets at a 6:4 ratio. The algorithm flowchart is shown in Figure 7 [Figure 7: see original paper].

Using TAKEDA1 as an example, Table 2 presents a manually configured set of hyperparameters alongside a set output by Bayesian optimization:

Table 2 Hyperparameter selection for TAKEDA1

Hyperparameters	Manual Setting	Bayesian Optimization
Batch_{size}	5.5e-5	4.5e-5
Hidden_{layers}	800, 900, 1600, 800	400, 800, 1800, 600, 50

Hyperparameters	Manual Setting	Bayesian Optimization
Min_{delta}	1.248e-5	785, 1683, 1042
Factor	8.927e-05	1033, 1741, 237, 1937, 579
Num_{neurons}	-	-
Loss_{delta}	-	-

The training process curves for the two prediction parameter models in TAKEDA1 are presented below:

Figure 8 [**Figure 8: see original paper**] Comparison of loss function curves between Bayesian optimization (left) and manual tuning (right) of k_{eff}

Figure 8 demonstrates that as the learning rate decreases stepwise, the training set loss function gradually declines. The test set loss function in the Bayesian optimization group (left) stabilizes after 400 epochs, ceasing to decrease after finding the optimal solution. In contrast, the manually tuned group (right) exhibits test set loss that initially decreases then increases, failing to find the optimal solution and showing severe oscillation and overfitting.

Figure 9 [**Figure 9: see original paper**] Comparison of loss function curves between Bayesian optimization (left) and manual tuning (right) of ϕ

For ϕ, the Bayesian optimization group (left) shows gradual training set loss decline with stepwise learning rate reduction, with test set loss stabilizing after 250 epochs upon finding the optimal solution. The manual tuning group (right) exhibits increasing test set loss despite decreasing training loss, diverging from the optimal solution with severe overfitting and oscillations.

2 Results and Discussion

Evaluation Metrics: Definitions of MAE and MSE are provided in Equations (2) and (3). Mean Absolute Percentage Error (MAPE) and R^2 (coefficient of determination) are defined as:

$$\text{MAPE} = \frac{1}{m}\sum_{i=1}^{m}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\%$$

$$R^2 = 1 - \frac{\sum_{i=1}^{m}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{m}(y_i - \bar{y})^2}$$

where y_i represents true values, $\bar{y}$ denotes sample mean, and $\hat{y}_i$ represents predicted values. R^2 measures the proportion of variance in the dependent variable explained by independent variables, ranging from 0 to 1. An R^2 value closer to 1 indicates better regression fit.

Tables 3 and 4 compare prediction errors between FCNN outputs using Bayesian-optimized versus manually set hyperparameters for k_{eff} and ϕ on TAKEDA1 and TAKEDA2 datasets:

Table 3 Comparison of errors between Bayesian optimization and manual hyperparameter tuning for k_{eff}

Metric	TAKEDA1	TAKEDA2
	Manual	Bayesian
MAE ($\times 10^{-5}$)	1525	118
MSE ($\times 10^{-5}$)	-	-

Table 3 reveals that Bayesian-optimized hyperparameters significantly outperform manual settings in both error magnitude and goodness-of-fit. The Bayesian-optimized k_{eff} achieves average absolute errors of 118×10^{-5} for TAKEDA1 and 132×10^{-5} for TAKEDA2, compared to 1525×10^{-5} and 1625×10^{-5} from manual tuning. This represents a 13-fold error reduction for TAKEDA1 and a 12-fold reduction for TAKEDA2, demonstrating substantial accuracy improvement. The R^2 values from Bayesian optimization are approximately 0.999, surpassing manual tuning results and indicating superior FCNN fitting.

Table 4 Comparison of errors between Bayesian optimization and manual hyperparameter tuning for ϕ

Metric	TAKEDA1	TAKEDA2
	Manual	Bayesian
MAPE (%)	-	1.7187
Max Percentage Error (%)	-	7.5585

Bayesian optimization achieves TAKEDA1 ϕ average percentage error of 1.7187% and maximum percentage error of 7.5585%, while TAKEDA2 shows average error of 0.8213% and maximum error of 11.130%, meeting acceptable accuracy requirements. Compared to manual tuning, Bayesian optimization reduces average percentage error by 9-fold for TAKEDA1 and 5-fold for TAKEDA2, with maximum percentage error reductions of 50-fold and 8-fold, respectively. Bayesian-optimized R^2 values are approximately 0.99, again outperforming manual tuning.

Figure 10 [Figure 10: see original paper] Error distribution histograms of k_{eff}: (a) TAKEDA1 (b) TAKEDA2

Figure 10 shows that k_{eff} errors follow normal distributions with concentrated dispersion. For TAKEDA1, 90% of errors fall within 300×10^{-5} and 50% within

100×10^{-5}. For TAKEDA2, 90% of errors are within 500×10^{-5} and 50% within 150×10^{-5}.

Figure 11 [**Figure 11: see original paper**] Percentage error distribution histograms of ϕ: (a) TAKEDA1 (b) TAKEDA2

Figure 11 demonstrates that ϕ percentage errors also follow normal distributions. TAKEDA1 shows more dispersed errors, while TAKEDA2 exhibits a "tall and narrow" distribution indicating concentrated error dispersion. For TAKEDA1, 90% of ϕ percentage errors are within 3.5%, while for TAKEDA2, 90% are within 1%, achieving excellent accuracy.

To efficiently solve core reloading optimization problems, this study proposes a Bayesian optimization algorithm for neural network hyperparameters as a surrogate model for core transport calculations. Training on datasets from TAKEDA1 and TAKEDA2 benchmarks validates model accuracy through comparison with manual tuning. The research demonstrates:

1. FCNN with Bayesian-optimized hyperparameters effectively fits VITAS calculation results, achieving k_{eff} average errors within 150×10^{-5}, TAKEDA1 ϕ average percentage error of 1.7187%, and TAKEDA2 ϕ average percentage error of 0.8213%, all within acceptable limits.
2. The FCNN model established with Bayesian-optimized hyperparameters significantly outperforms manually tuned models in both training error and goodness-of-fit metrics.
3. The hyperparameters and corresponding FCNN architecture are specifically applicable to TAKEDA1 and TAKEDA2 datasets generated by VITAS. For predicting other core parameters, the corresponding dataset can be substituted and the Figure 7 flowchart followed to obtain new hyperparameter combinations and train FCNN, replacing manual tuning with universal applicability.
4. This work validates the feasibility and advantages of neural network-based parameter prediction methods in reactor physics calculations, though further exploration of different neural network architectures for reactor physics applications is warranted.

Author Contributions

ZHANG Fan performed simulations and wrote the manuscript; ZHANG Junda, SUN Qizheng, and XIAO Wei provided technical support; LIU Xiaojing provided funding support; ZHANG Tengfei provided funding support and revised the manuscript.

References

1. Kazuteru S, Toshikazu T. An improvement of the transverse leakage treatment for the nodal SN transport calculation method in hexagonal-z geometry[J]. Journal of Nuclear Science and Technology, 1996(8). DOI:10.3327/jnst.33.620.

2. Palmiotti G, Lewis E E, Carrico C B. VARIANT: VARIational Anisotropic Nodal Transport for multidimensional Cartesian and hexagonal geometry calculation: ANL-95/40[R]. Argonne, IL, USA: Argonne National Laboratory, 1995.

3. Zhang T F, Wu H C, Cao L Z, et al. An improved variational nodal method for the solution of the three dimensional steady-state multi-group neutron transport equation[J]. Nuclear Engineering Design, 2018, 419-427. DOI:10.1016/j.nucengdes.2018.07.009.

4. Zhang Tengfei, Zhu Lei, Wu Hongchun, et al. Development and validation of high flux engineering test reactor fuel management platform HEFT[J]. Atomic Energy Science Technology, 2013, 47(S1): 467-471. DOI: 10.7538/yzk.2013.47.S1.0467.

5. Ji Nan, Yi Jinhao, Zhao Pengcheng, et al. Research on adaptive RBF neural network prediction method for core thermal-hydraulic parameters reactor[J]. Nuclear Techniques, 2022, 45(09): 65-74. DOI: 10.11889/j.0253-3219.2022.hjs.45.090601.

6. Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors[J]. Nature, 1986, 323: 533-536. DOI:10.1038/323533a0.

7. Huang Yu, Liu Junqiang, Liu Le. Application of BP neural network in DNBR prediction[J]. Nuclear Techniques, 2015, 38(7): 070606. DOI: 10.11889/j.0253-3219.2015.hjs.38.070606.

8. Lyu Y Q, Zhou Q W, Li Y F, et al. A predictive maintenance system for multi-granularity faults based on AdaBelief-BP neural network and fuzzy decision making[J]. Advanced Engineering Informatics, 2021, 49: 101318. DOI: 10.1016/j.aei.2021.101318.

9. He Can, Qiu Xiaoping, Sun Zheng, et al. Radiation shielding optimization of space reactor based on intelligent decision support system[J]. Nuclear Techniques, 2022, 45(05): 65-75. DOI: 10.11889/j.0253-3219.2022.hjs.45.050601.

10. Akkoyun S. Estimation of fusion reaction cross-sections by artificial neural networks[J]. Nuclear Instruments and Methods in Physics Research B, 2020, 462: 51-54. DOI:10.1016/j.nimb.2019.11.014.

11. Guo Z, Uhrig R E. Use of artificial neural networks to analyze nuclear power plant performance[J]. Nuclear Technology, 1992, 99(1): 36-42.

DOI:10.13182/NT92-A34701.

12. Hou S H, Fang S L, Zeng Q Y, et al. Element grouping faceted fully connected network based on RIS[J]. Journal of System Simulation, 2024, 36(4): 1017-1027.
13. Zhang T F, Xiao W, Yin H, et al. VITAS: A multi-purpose simulation code for the solution of neutron transport problems based on variational nodal methods[J]. Annals of Nuclear Energy, 2022. DOI:10.1016/j.anucene.2022.109335.
14. Zhang Tengfei, Yin Han, Sun Qizheng, et al. Application research on VITAS — a general-purpose neutron transport code[J]. Nuclear Power Engineering, 2023, 44(02): 15-23. DOI:10.13832/j.jnpe.2023.02.0015.
15. Tang Lin, Li Yong, Tang Yufeng, et al. Application of an LSTM model based on deep learning through X-ray fluorescence spectroscopy[J]. Nuclear Techniques, 2023, 46(07): 070502. DOI: 10.11889/j.0253-3219.2023.hjs.46.070502.
16. Hosain M T, Jim J R, Mridha M F, et al. Explainable AI approaches in deep learning: Advancements, applications and challenges[J]. Computers and Electrical Engineering, 2024, 117. DOI:10.1016/j.compeleceng.2024.109246.
17. Cui Jiaxu, Yang Bo. Survey on Bayesian optimization methodology applications[J]. Journal of Software, 2018, 29(10): 3068-3090. DOI:10.13328/j.cnki.jos.005607.
18. Ghahramani. Probabilistic machine learning artificial intelligence[J]. Nature, 2015, 521: 452-459. DOI:10.1038/nature14541.
19. Martin A, Paul B, Jianmin C, et al. TensorFlow: A system for large-scale machine learning[J]. USENIX Association, 2016. DOI:10.48550/arXiv.1605.08695.

Note: Figure translations are in progress. See original paper for figures.