A CNN-Based PINNs Framework for Solving the 2D Neutron Diffusion Equation

Jaeguk Lee [a] and Hyung Jin Shim [a*]
[a]*Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul 08826, Republic of Korea*
[*]*Corresponding author: shimhj@snu.ac.kr*

Keywords: Physics-informed neural network (PINNs), Convolutional neural network (CNN), Neutron diffusion equation

1. Introduction

Deep learning has recently emerged as a promising tool across diverse engineering domains [1, 2], offering significant potential for supplementing or advancing traditional methods in reactor physics [3]. By leveraging large-scale datasets for pattern recognition and predictive modeling, deep learning can enable more comprehensive understandings of complex problems and provide effective solutions to high-dimensional challenges. Among the many deep learning frameworks, Physics-Informed Neural Networks (PINNs) [4] have gained significant attention. PINNs embed fundamental physical principles—often expressed as partial differential equations (PDEs)—directly into the neural network structure, diminishing reliance on large labeled datasets and guiding models toward physically consistent solutions. They have demonstrated success in various fields such as fluid mechanics, structural analysis, and electromagnetics [5, 6, 7]. They also have been applied to core reactor physics problems, including point kinetics [8] and neutron diffusion eigenvalue analyses [9, 10].

Deep learning models—including PINNs—often serve as surrogate models in engineering contexts involving repetitive calculations or high-dimensional design spaces, thus reducing computational costs and simplifying analysis. From a direct solver perspective, researchers have also explored PINNs for directly solving PDEs or enhancing traditional numerical methods. Nonetheless, many published studies remain confined to single or fixed problem setups, leading to a need for retraining when different physical systems or design conditions arise. Overcoming this limitation requires more generalized representations of the governing physics and learning strategies capable of handling diverse parameters. Consequently, while PINNs show considerable promise for reactor physics, further work should focus on combining physics-based constraints with deep learning and improving generalization—key steps toward establishing these methods as more robust and widely applicable tools in future reactor analysis.

Building on this concept, this study introduces a convolutional neural network (CNN) -based PINNs framework for solving the neutron diffusion equation in PWR cores. This model predicts both flux distributions and the effective multiplication factor (k_{eff}) for multiple core configurations using a single, unified network. Rather than relying on explicit spatial coordinates or requiring separate training for each configuration, our approach employs node-wise material properties and is trained solely through first-principles equations—namely, the governing physics and boundary conditions—without any pre-existing labeled datasets. As a result, the model can be extended to novel core loading pattern sharing the same cross-section dataset without retraining, offering considerable scalability and computational savings. This flexibility underscores both the robustness and efficiency of the proposed CNN-based PINNs approach, paving the way for broader deployment of deep learning in practical nuclear engineering applications.

2. Methods

The two important considerations in designing a PINNs approach are (1) the formulation of the loss function and (2) the selection of inputs and the neural network architecture. This section describes how these elements are configured in our method.

2.1 Physics-Informed Neural Networks

PINNs integrate known physical laws, often expressed as differential equations, directly into their loss functions [4]. Unlike purely data-driven models—which rely on large volumes of labeled input-output pairs—PINNs combine limited or even no labeled data with physics-based constraints. The loss function typically has two components: one comparing network predictions to available data, and another enforcing the governing equations at collocation and boundary points.

At these points, the model checks its predictions against the physics; if the residual—defined as the discrepancy from the governing equation—is large, the network adjusts its weights to reduce it. This process guides PINNs to solutions consistent with established physical laws, even with relatively small datasets. Consequently, PINNs are especially valuable in

scientific and engineering applications where adherence to physical principles is critical.

2.2 Application of a CNN-based PINNs to the Neutron Diffusion Equation

We now detail how our CNN-based PINNs framework is adapted for the two-group neutron diffusion equations in PWR cores. Below, we first describe the inputs and neural network architecture on Section 2.2.1, followed by the physics-informed loss function on Section 2.2.2.

2.2.1. Inputs and Neural Network Architecture

Classical PINNs use explicit spatial coordinates (e.g., x, y, z) as inputs, enabling direct calculation of spatial derivatives via automatic differentiation. Here, we employ a CNN-based approach that uses node-wise material properties, which is expressed as cross sections for inputs. This method enables the model to generalize across different reactor core loading patterns, provided they share the same cross-section dataset, thereby avoiding retraining for each new configuration.

Inputs are node-wise two-group cross sections (XSs) for a discretized M×M lattice, organized into a 4D tensor (N, 7, M, M), where N is the batch size. These XSs include the diffusion coefficients, D_1^i and D_2^i, the absorption XSs, Σ_{a1}^i and Σ_{a2}^i, the fission yield XSs, $\nu\Sigma_{f1}^i$ and $\nu\Sigma_{f2}^i$ for both fast and thermal group and the fast-to-thermal scattering XSs, Σ_{s12}^i for each node *i*.

The network architecture employs a modified UNet [11], which is based on convolutional neural networks (CNN). It features a contracting (encoder) path to capture global patterns and an expansive (decoder) path to produce high-resolution outputs, with skip connections between corresponding levels.

A key aspect of the proposed model is its dual-output scheme, which predicts both the flux distribution and k_{eff} within a single model. After the decoder stage of UNet, the final convolutional layer outputs the flux distribution in the shape of (N, 2, M, M), where N and M match the input dimensions, and *2* corresponds to the two energy groups in the neutron diffusion equation. This flux outputs are then flattened and passed through fully connected layers to produce a scalar value for k_{eff}. By training these two outputs simultaneously, the network exploits the intrinsic relationship between flux and k_{eff}, enhancing overall consistency and reducing the need for separate models. Finally, both outputs are then fed into the loss function to measure how well the predicted values satisfy the governing equations, which is explained in the next section.

2.2.2. Physics-Informed Loss Function

To train the CNN-based PINNs, we define a physics-based loss term derived from the two-group neutron diffusion equations, which can be written as:

$$f_g(x,y) = -\nabla\cdot(D_g\nabla\phi_g(x,y)) + \Sigma_{rg}\phi_g(x,y) - \frac{1}{k}\chi_g\sum_{g'=1}^{2}\nu\Sigma_{fg'}\phi_{g'}(x,y) - \sum_{g'=1}^{2}\Sigma_{g'g}\phi_{g'}(x,y)\ (g=1,\ 2), \tag{1.1}$$

$$f = f_1 + f_2. \tag{1.2}$$

Because our inputs are grid-based node properties rather than continuous spatial coordinates, we cannot apply automatic differentiation for second-order terms. Instead, we use the FDM box scheme to compute spatial derivatives. The residual at each node measures how closely the predicted flux satisfies the neutron diffusion equations. We then take the mean squared residual across all nodes as the physics-based loss:

$$L = \frac{1}{N_i\cdot N_j}\sum_{j=1}^{N_j}\sum_{i=1}^{N_i}\left|f^e(i,j)\right|^2, \tag{1.3}$$

where $f^e(i,j)$ is the residual at node (*i*, *j*).

Boundary nodes are treated with the same FDM box scheme. By incorporating these constraints into the FDM equations at the boundary nodes, we ensure that the governing neutron diffusion principles hold over the entire domain, including its edges. Consequently, the boundary conditions become an integral part of the physics-informed residual, eliminating the need for additional data or post-processing steps. This combined approach ensures that the CNN-based PINNs satisfy both the FDM-based spatial discretization and the fundamental neutron diffusion equations throughout the entire reactor core lattice.

3. Experimental Setup

To evaluate the proposed CNN-based PINNs framework, we employ the 2D IAEA PWR benchmark problem [12], which represents a typical PWR core comprising 177 fuel assemblies arranged in quarter-core symmetry. By varying the assembly loading patterns while preserving the geometry and cross-section data, we generate 20,000 distinct core configurations. These are split into 18,000 training, 1,000 validation, and 1,000 test datasets, ensuring comprehensive coverage of possible arrangements. Notably, no precomputed flux distributions or k_{eff} values are provided while training; instead, the model learns solely from the physics-informed loss function derived from the two-group

neutron diffusion equation. Only the test set includes labeled data—namely, flux distributions and k_{eff} values for each configuration—calculated using an in-house FDM code for final model evaluation.

The training process is implemented in PyTorch and executed on an NVIDIA RTX 3090 Ti. The Adam optimizer is utilized and the learning rate is scheduled via cosine annealing with the maximum temperature of 50 and minimum learning rate of 10^{-6}. Early stopping is enforced with a patience of 30 epochs to prevent overfitting, and a maximum of 500 epochs is allowed. Each configuration is discretized at a 1 cm mesh, leading to input tensors of size (N, 7, 170, 170) for batch size N. The boundary conditions are treated in accordance with the IAEA benchmark specifications. This setup ensures our dataset covers a broad range of plausible core states while relying solely on physics-based constraints for model training.

4. Numerical Results

The overall training concludes in about 3 hours on an NVIDIA RTX 3090 Ti. Once trained, the model requires under 0.002 seconds to infer flux distributions and k_{eff} for a single core configuration. All accuracy assessments focus on the fuel region, where flux precision is of primary operational interest.

Table I. PINNs results for 1000 test data

	Top 1% error		Average error		Bottom 1% error	
	Flux (%)	k_{eff} (pcm)	Flux (%)	k_{eff} (pcm)	Flux (%)	k_{eff} (pcm)
Total	1.67	2.7	3.65	333.7	7.12	1184.9
Fast	1.47		3.43		7.28	
Thermal	1.88		3.88		7.20	

Table I summarizes the model's performance over 1,000 unseen test configurations, each discretized at a 1cm mesh. The CNN-based PINNs model achieves an average RMS flux error of 3.65% and a corresponding k_{eff} error of 333.7 pcm for the test set. Table I provides a more granular view of the results, where the top-performing 1% of test cases shows a flux error of 1.67% and k_{eff} error of 2.7 pcm, while the bottom-performing 1% exhibits flux and k_{eff} errors of 7.12% and 1184.9 pcm, respectively. Fast and thermal flux errors display similar trends in the average, with RMS errors of 3.43% for fast group and 3.88% for thermal group, underscoring the model's fairly balanced performance across energy groups.

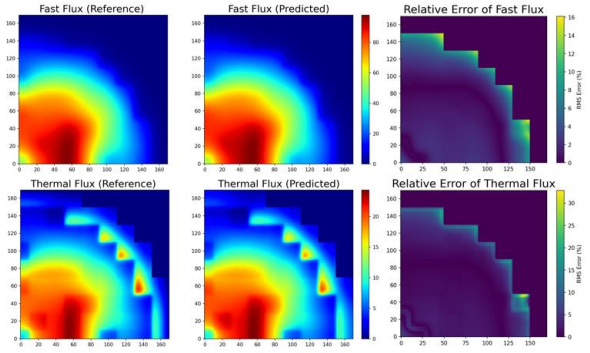

Fig 1. Comparison of predicted and reference flux distribution for a representative case

Figure 1 illustrates a representative test configuration whose RMS error is close to the overall average, thus reflecting the proposed model's typical performance. Each row corresponds to either the fast (top) or thermal (bottom) flux group, and the three columns show the reference solution (left), the predicted solution (middle), and the RMS error in percentage (right). In the inner fuel region, more than 90 % of flux errors remain below 5% for both fast and thermal group, indicating that a CNN-based cross-section input approach effectively captures the essential diffusion behavior across multiple configurations.

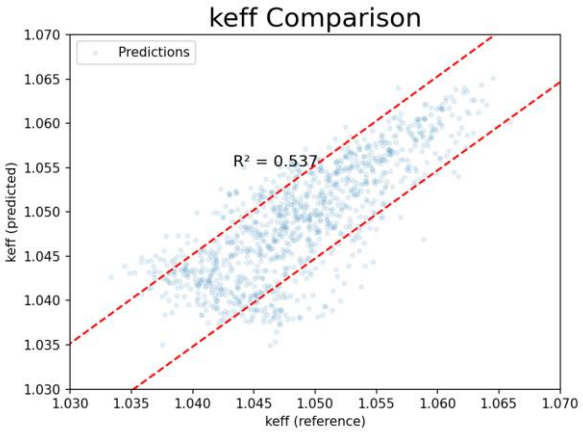

Fig 2. Scatter plot comparing predicted and reference values for all test cases.

Figure 2 depicts the model's distributions of a k_{eff} comparison. The predicted k_{eff} values are compared to the reference solutions, with red dashed lines indicating the ±500 pcm deviation. About 79.6% of the predictions stay within these bounds—suggesting that although the baseline model can capture the overall trend of k_{eff}, its performance remains more constrained than for flux predictions.

Overall, the proposed CNN-based PINNs framework yields reasonable predictions of flux distributions and

k_{eff} across a wide range of PWR core configurations. While flux predictions generally achieve higher accuracy, the model also provides a reasonable foundation for k_{eff}.

5. Conclusion

This study presents a CNN-based Physics-Informed Neural Networks (PINNs) framework for solving the 2D two-group neutron diffusion equation in Pressurized Water Reactor (PWR) cores under multiple distinct loading patterns. By relying solely on node-wise cross-section data and the governing physics—rather than labeled flux or k_{eff} values—the proposed model can infer solutions for unseen configurations (provided they share the same cross-section set) without retraining.

Across 1,000 test cases, the model achieves an overall average root-mean-square (RMS) flux error near 3.65%, with top-performing 1% of cases showing flux errors around 1.67% and the bottom 1% reaching up to 7.12%. Similarly, the best 1% of k_{eff} predictions have errors of about 2.7 pcm, while the worst 1% climb as high as 1184.9 pcm, yielding an average of roughly 333.7 pcm. Fast and thermal flux predictions exhibit parallel performance trends, with respective RMS errors of 3.43% and 3.88%. Once trained, the model requires under 0.002 seconds on a single GPU to predict both flux distributions and k_{eff}, showing promise for near real-time or repeated calculations.

Despite these promising results, two major limitations remain. First, training PINNs without labeled data can introduce convergence challenges and often demands careful hyperparameter tuning. Second, although the model generally captures the correct physical trends, its accuracy is not yet sufficient for practical deployment, particularly given there is no inherent guarantee that the solution corresponds to the fundamental eigenmode rather than a higher-order mode. Further refinements are thus necessary to ensure consistent identification of the fundamental mode and to meet the accuracy requirements demanded by real reactor physics analyses. Looking ahead, future efforts will focus on refining the network architecture, exploring more advanced PINNs strategies, and extending the approach to three-dimensional reactor models. These developments aim to further improve predictive accuracy, reduce the risk of converging to non-fundamental modes, and strengthen the method's overall applicability for practical reactor physics problems.

REFERENCES

[1] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," Science, vol. 350, pp. 1332–1338, 2015.
[2] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning," Nat. Biotechnol., vol. 33, pp. 831–838, 2015.
[3] Y. Nam and H. J. Shim, "Development of deep convolutional neural network for prediction of cycle maximum pin power peaking factor in pressurized water reactor," Annals of Nuclear Energy, vol. 194, p. 110083, 2023, doi: 10.1016/j.anucene.2023.110083.
[4] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," J. Comput. Phys., vol. 378, pp. 686-707, 2019.
[5] Z. Mao, A. D. Jagtap, and G. E. Karniadakis, "Physics-informed neural networks for high-speed flows," Comput. Methods Appl. Mech. Eng., vol. 360, 112789, 2020.
[6] E. Zhang, M. Dao, G. E. Karniadakis, and S. Suresh, "Analyses of internal structures and defects in materials using physics-informed neural networks," Science Advances, vol. 8, no. 40, p. eabk0644, Feb. 2022.
[7] X. Zhao, Z. Gong, Y. Zhang, W. Yao, and X. Chen, "Physics-informed convolutional neural networks for temperature field prediction of heat source layout without labeled data," Engineering Applications of Artificial Intelligence, vol. 118, p. 105653, 2023.
[8] E. Schiassi, M. De Florio, B. D. Ganapol, P. Picca, and R. Furfaro, "Physics-informed neural networks for the point kinetics equations for nuclear reactor dynamics," Ann. Nucl. Energy, vol. 167, 108833, 2022.
[9] M. H. Elhareef and Z. Wu, "Physics-informed neural network method and application to nuclear reactor calculations: A pilot study," Nucl. Sci. Eng., vol. 197, no. 4, pp. 601-622, 2023.
[10] Y. Yang et al., "A data-enabled physics-informed neural network with comprehensive numerical study on solving neutron diffusion eigenvalue problems," Ann. Nucl. Energy, vol. 183, 109656, 2023.
[11] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in Proc. MICCAI, vol. 9351, pp. 234–241, 2015.
[12] Argonne Code Center, "ANL Benchmark Book-Report ANL-7416," Argonne National Laboratory, Argonne, IL, 1977.