공학석사 학위논문

# Convolutional Neural Network for Prediction of Two-Dimensional Core Power Distributions in PWRs

# 가압경수로 노심 2차원 출력분포 예측을 위한 합성곱 인공 신경망

2019 년 8 월

서울대학교 대학원

에너지시스템공학부

이 진 영

# Abstract

The purpose of this research is to develop a neural network model that is computationally inexpensive in predicting two-dimensional assembly-wise power distributions along with assembly-wise pin power peaking factors (PPPFs) by taking only a set of beginning of cycle (BOC) macroscopic cross sections. Such a computationally inexpensive and fast prediction model is needed because the conventional prediction model still renders a computational burden in loading pattern(LP) optimization processes.

As the first step of the research, the previously developed state-of-the-art power prediction neural network models are evaluated to select the best one. It is then modified using convolutional neural network architectures. 20,000 Korean Standard Nuclear Power Plant (OPR1000) LPs are randomly generated and used for supervised learning. The reference power distributions are generated by using a three-dimensional core analysis code called ASTRA. The averaged and maximum absolute error(AE) in the assembly power predictions obtained by the trained neural network turns out to be 0.19% and 7.34%, respectively, while those for PPPF are 0.31% and 9.13%, respectively. In order to test the model in the region of interest, 3,000 general design bounded LPs which reside outside of the range of the trained data are separately generated. It appears that the maximum AE for assembly-wise power and PPPF are 3% and 5%, respectively. Those errors are within the acceptable range when an approximate model is used in a LP optimization process. The computing time of the neural network model is around 0.2 second, which is about 1000 time faster than

ASTRA.

The model can greatly reduce the computing time of LP optimization processes. Although it can be a great utility for a nuclear designer as well as in an automatic LP optimization program, it has one limitation. The trained neural network model is only valid within the specified core conditions: a number of total and fresh fuels, initial boron concentration, T/H conditions, and etc. If any of the core condition changes, the model can produce higher than presented errors.

**Keyword:** Convolutional Neural Network(CNN)
Supervised learning
power distribution prediction
Pressurized Water Reactor(PWR)

**Student Number:** 2017-26188

# Table of Contents

# List of Tables

# List of Figures

# 1. Introduction

## 1.1 Background

The loading pattern (LP) optimization is carried out by comparing and evaluating a number of candidates which are generated based on the engineer's design experience. In the LP optimization process shown in Figure 1-1, the "Calculate CORE" has been performed by a three-dimensional (3D) core analysis code like the ASTRA code of the KEPCO Nuclear Fuel Company (KNF) while the "Change the LP" is carried out based on the designer's experience and/or automatic LP optimization code's judgments.

In the history of core analysis that involves the solution of the transport and/or diffusion equations, many trials had been conducted on balancing the solution accuracy and computing time. Even with the recent advances in the computational resources, the direct whole core transport calculations are too computationally expensive for 3D core depletion calculations. Therefore, the most widely used procedure in nuclear core analyses is a two-step system in which the lattice transport and the diffusion core calculations are combined[1]. As computer performance and the calculation methods are improved, the accuracy and computing time of 3D core depletion calculations have been improved over time. The computing time is, however, still quite a burden in an LP optimization process. If the time for 3D core depletion calculations is greatly reduced, the overall optimization process will be faster and easier and will lead to an economic benefit at a reload design company.

## 1.2 Purpose and Scope

In the past, there have been studies to reduce the core calculation[①] time using the artificial neural network(ANN) such as Optimization Layer by Layer method(OLL)[2]. As in the previous study, the objective of this paper is to construct a fast prediction ANN that can deliver high accuracy. The neutronic characteristics of interest for this ANN are two-dimensional (2D) assembly-wise core power distribution and the pin power peaking factor (PPPF) for each assembly. As the recent improvement of ANNs, a new branch has been developed called Deep Neural Network (DNN). Within the sub-field, Convolutional Neural Network (CNN) has been introduced to solve spatial relationship problems such as image classification and regressions[3]. Although the previous networks such as OLL performed well in the trained distribution, it worsens quickly outside the trained distributions. In other words, the network was not generalized to a wider range of problems. It is thought that this is because of the neglect of the spatial relationship. By considering the spatial relationship using a CNN, it is possible to greatly improve the accuracy of the prediction even with the data that are outside the trained distribution. In addition, burnup depletion can be performed using an ANN.

[①] the results of the core calculations include: power distribution, pin power peaking factors, burnup distribution, cycle length, and etc..

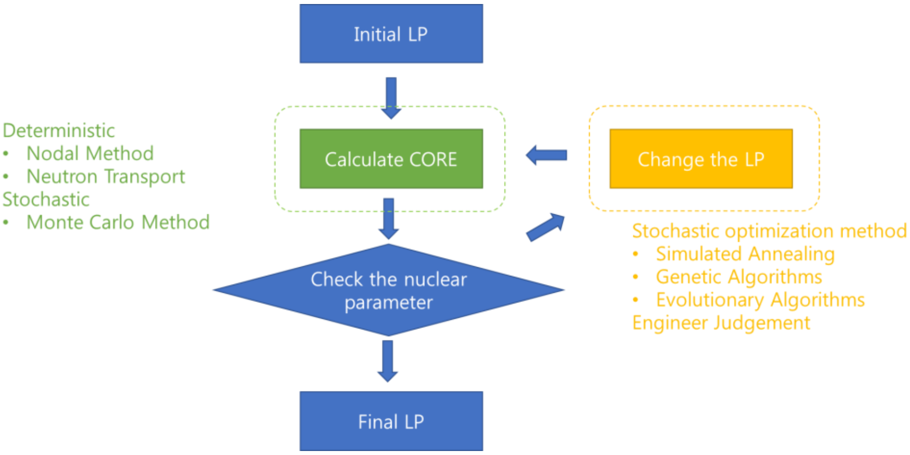


Figure 1−1. Flow chart to loading pattern optimization

# 2. Review of previous researches

## 2.1 Optimization Layer by Layer

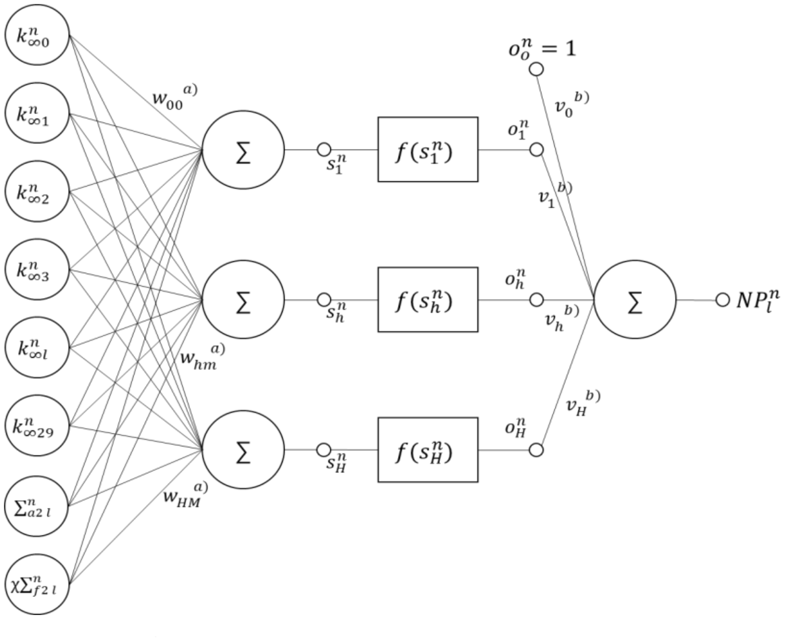
The applicability of the ANN for prediction of PWR core neutronic parameters was first demonstrated by Kim et al. (1993)[14]. From this research, several variants of ANN have been developed but due to their high error, there were no practical applications. Among the researches, most promising in accuracy and computational speed was demonstrated by Jang et al. (2001)[2]

The optimization layer by layer (OLL) learning algorithm is applied as shown in Figure 2-1 to predict the assembly-wise core power and burnup distributions, the critical soluble boron concentration, and the pin power peaking factor (PPPF) for a Pressurized Water Reactor (PWR) based on the given set of k-infinity and Macroscopic Cross Sections (XSs). They utilized the nodal powers and the assembly discontinuity factors of the given fuel assembly(FA) and eight surrounding FAs for training the OLL networks to predict the PPPF of the individual FA as like as the pin power reconstruction. The summary of the validation test is as shown in Table 2-1. As shown in the table, the OLL predict the assembly power and PPPF very well and can show 40 times improvement in computing time when compared to the modern nodal method code.

Table 2-1: Power distribution and pin power peaking factor prediction validation error of OLL

| | $e_{avg}$[a] | $e_{max}$[b] | Frac. with $e$[c] > 5% | Frac. with $e$[c] > 10% |
|---|---|---|---|---|
| Assembly Power | 0.73 | 9.11 | 0.19 | - |
| PPPF | 0.85 | 11.26 | 0.80 | 0.06 |

a= average relative error (%)
b= maximum relative error (%)
c= Fraction of the assemblies with relative error (%)



a) $w_{hm}$ $(m=0,1,\cdots,M;\ h=0,1,\cdots,H)$ = weights for hidden neurons

b) $v_h$ $(h=0,1,\cdots,H)$ = weights for output neuron

c) $f(s)$ = sigmoidal function defined by $(1+e^{-s})^{-1}$

d) l = assembly(1~29)

e) n = burnup

Figure 2-1. Three-layer OLL network for prediction of normalized FA power

## 2.2 Improvements

Although there are accurate ANN models in core predicting, we want to revise the models with the latest development in the field of ANN called deep learning for better accuracy.

The first improvement is that the main concept of ANN is changed from OLL to CNN. The conventional nodal method calculates assembly power with four surrounding surface flux. To reflect on this fact, CNN method calculates its assembly power with four surrounding assembly features. It is thus attempted to improve the speed and accuracy by converting the main neural network into CNN based on past researches.

The second change is on the input type. Instead of using a combination of k−infinity and specific macroscopic cross−sections, we use 5 types of macroscopic cross−sections (fast/thermal nu−fission XS, fast/thermal absorption XS, fast−to−thermal scattering XS) that are used to calculate core eigenvalue. Note that the degree of freedom to calculate assembly power is low when k−inf is used. The reason for this is that neutron leakage is different for each position but k−inf is made without considering the leakage. Therefore, in order to predict the power with high accuracy, it is better to consider all 5 XSs that can consider leakage.

Another notable improvement is predicting the core power distribution (PD) over the entire cycle with only the beginning of cycle (BOC) XSs rather than using different macro cross sections for each burnup step. If we update the XSs to calculate the depleted power, final prediction error will include the error from XSs updates. More importantly, we are taking advantage of the parallel nature of GPU computing by un−linking the depletion process.

And the other one is a reflector and moderator area is included for analyzing periphery area assembly power. The reason for this is that a neutron leakage of the periphery assembly is higher than that of inside assembly. In order to reflect on this phenomena, we have included the reflector and the moderator.

# 3. Method

## 3.1 Deep Learning Models

Deep learning is a class of machine learning algorithms that[4]:

- use a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input.
- learn in supervised (e.g., classification) and/or unsupervised (e.g., pattern analysis) manners.
- learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts.

## 3.2 Supervised learning

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs[5]. It infers a function from labeled training data consisting of a set of training examples[6]. In supervised learning, each example is a pair consisting of an input object (typically a vector) and the desired output value (also called the supervisory signal). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way.

## 3.3 Convolution Neural Network

A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, RELU layer i.e. activation function, pooling layers, fully connected layers, and normalization layers[3].

Description of the process as a convolution in neural networks is by convention. Mathematically it is a cross−correlation rather than a convolution (although cross−correlation is a related operation). This only has significance for the indices in the matrix, and thus which weights are placed at which index. Convolutional layers apply a convolution operation to the input, passing the result to the next layer. The convolution emulates the response of an individual neuron to visual stimuli[7].

Each convolutional neuron processes data only for its receptive field. Although fully connected feedforward neural networks can be used to learn features as well as classify data, it is not practical to apply this architecture to images. A very high number of neurons would be necessary, even in a shallow (opposite of deep) architecture, due to the very large input sizes associated with images, where each pixel is a relevant variable. For instance, a fully connected layer for a (small) image of size 100 x 100 has 10000 weights for each neuron in the second layer. The convolution operation brings a solution to this problem as it reduces the number of free parameters, allowing the network to be deeper with fewer parameters[8]. For instance, regardless of image size, tiling regions of size 5 x 5, each with the same shared weights, requires only 25 learnable parameters.

## 3.4 Residual Neural Network

A residual neural network is an artificial neural network (ANN) of a kind that builds on constructs known from pyramidal cells in the cerebral cortex. Residual neural networks do this by utilizing skip connections or short−cuts to jump over some layers[9].

One motivation for skipping overlayers is to avoid the problem of vanishing gradients by reusing activations from a previous layer until the layer next to the current one learns its weights. During training, the weights adapt to mute the previous layer and amplify the layer next to the current. In the simplest case, only the weights for the connection to the next to the current layer is adapted, with no explicit weights for the upstream previous layer. This usually works properly when a single non−linear layer is stepped over, or when the intermediate layers are all linear. If not, then an explicit weight matrix should be learned for the skipped connection. Skipping initially compresses the network into fewer layers, which speeds learning. The network gradually restores the skipped layers as it learns the feature space. During later learning, when all layers are expanded, it stays closer to the manifold and thus learns faster. A neural network without residual parts explores more of the feature space. This makes it more vulnerable to perturbations that cause it to leave the manifold and necessitates extra training data to recover.

## 3.5 Architecture

The architecture of the network is summarized in Figure 3−1. Five types of the macroscopic cross−section are fast/thermal nu−fission XS, fast/thermal absorption XS, fast to thermal scattering XS. The XSs used is taken from the lattice code (KARMA) calculation which is mainly used for commercial core analysis. It is node−wise (1/4 assembly node). Five XSs is spatially represented as 17 × 17 (Quadrant core, the complete input shape is [17×17×5]). A first convolution (CONV1) is performed using a 1×1 filter, same padding, ReLU activation function and 64 channels (the output shape is [17×17×64]). The second convolution (CONV2) has the same properties of (CONV1) except for the size of filter–growing now to 3×3 filter (shape of [17×17×64]). The third convolution (CONV3) has the same properties as (CONV1) except for the number of channels–growing now to 256 (shape of [17×17×256]). There is shortcut−connection that only passed CONV3 (shape of [17×17×256]). After adding two convolution layers, it is divided into 2 main feature, the first is assembly−wise power and the second is PPPF. The fourth convolution (CONV4) is performed using a 2×2 filter, 2×2 strides, same padding, ReLU activation function and 128 channels (the output shape is [9×9×128]). The fifth convolution (CONV5) is performed using a 1×1 filter, ReLU activation function and 1 channel (the output shape is [9×9×1]). In assembly power prediction, the fifth convolution is the final step to predict. But, in PPPF, the final value is the product of the fifth convolution output and assembly power prediction value[9×9×1]. the resulting data (of shape [9×9×1]) is flattened to a single vector (of [81] elements).

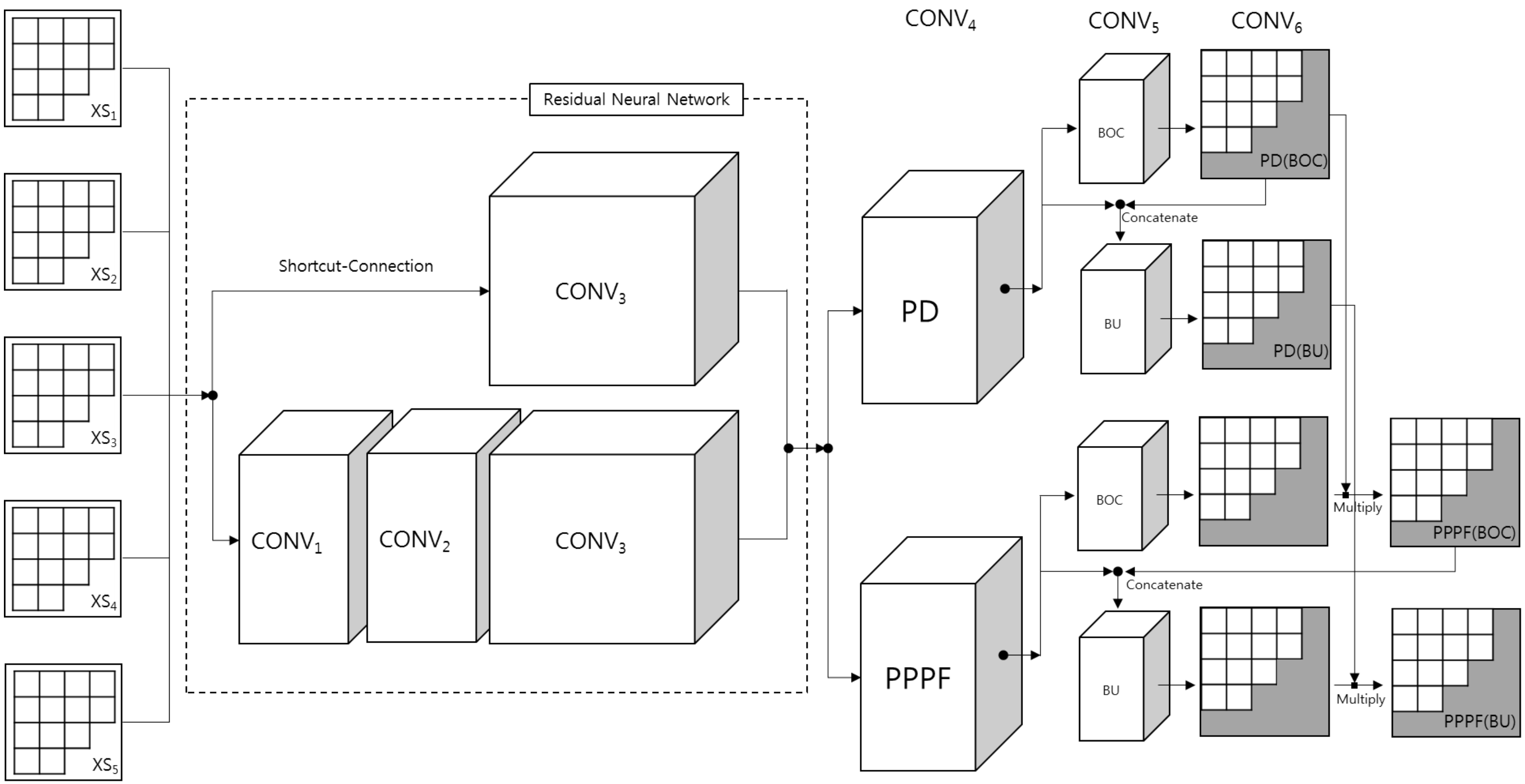
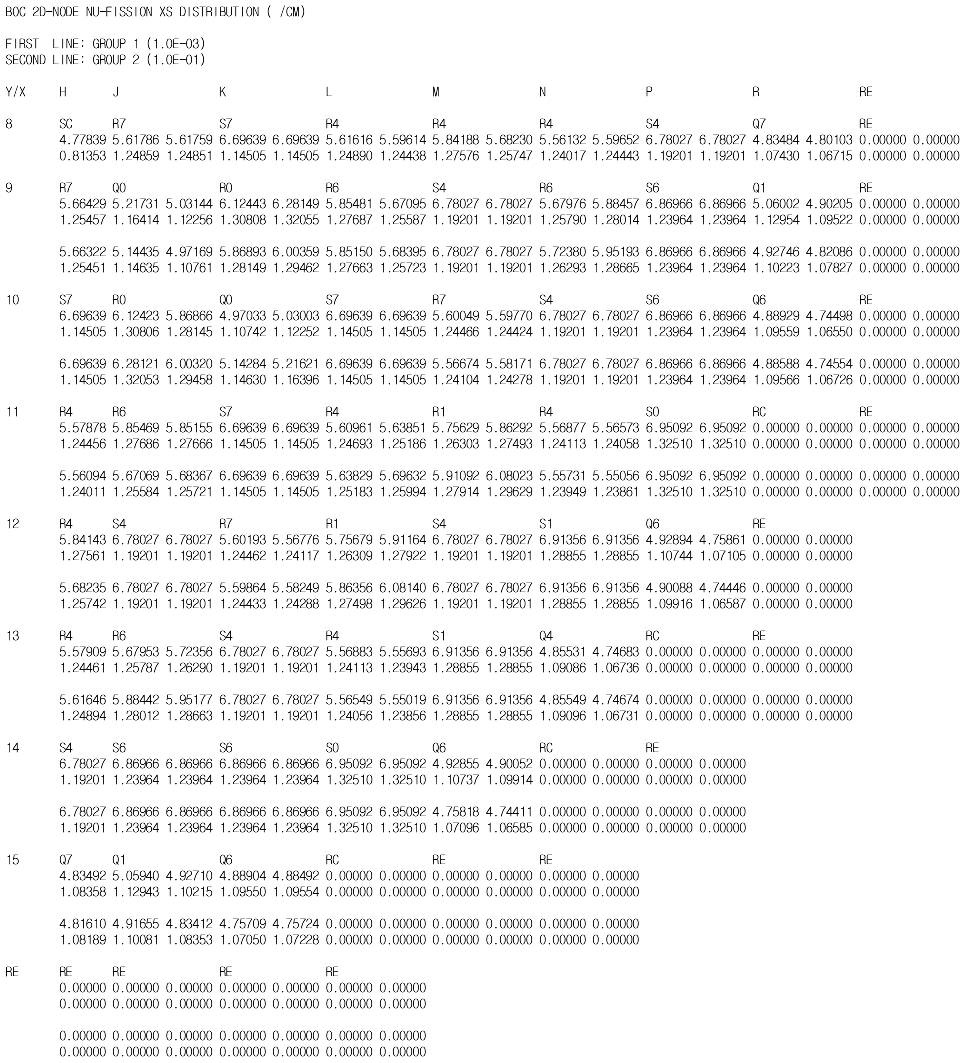


Figure 3-1. Convolution neural network architecture with shortcut -connection for assembly and PPPF prediction.

### 3.5.1 Block: XS INPUT

Like most ANN architecture defining input is the most important part of the architecture optimization. Therefore, we carefully choose our input as a building block of core simulation: beginning of cycle (BOC) macroscopic XSs. As we previously have shown, the input shape was 2D node-wise full core XSs 34x34x5. It was 34x34 nodes because there are radially 15 assemblies with 2 reflectors on both sides. Since the core is rotationally symmetric, by simply taking 4th quadrant of the full core, we can simulate the rest of the core. Therefore, our input shape becomes 17x17x5 and sample input is shown in figure 3-2.



```
BOC 2D-NODE NU-FISSION XS DISTRIBUTION ( /CM)

FIRST  LINE: GROUP 1 (1.0E-03)
SECOND LINE: GROUP 2 (1.0E-01)

Y/X     H       J               K               L               M               N               P               R               RE

8       SC      R7              S7              R4              R4              R4              S4              Q7              RE
        4.77839 5.61786 5.61759 6.69639 6.69639 5.61616 5.59614 5.84188 5.68230 5.56132 5.59652 6.78027 6.78027 4.83484 4.80103 0.00000 0.00000
        0.81353 1.24859 1.24851 1.14505 1.14505 1.24890 1.24438 1.27576 1.25747 1.24017 1.24443 1.19201 1.19201 1.07430 1.06715 0.00000 0.00000

9       R7      Q0              R0              R6              S4              R6              S6              Q1              RE
        5.66429 5.21731 5.03144 6.12443 6.28149 5.85481 5.67095 6.78027 6.78027 5.67976 5.88457 6.86966 6.86966 5.06002 4.90205 0.00000 0.00000
        1.25457 1.16414 1.12256 1.30808 1.32055 1.27687 1.25587 1.19201 1.19201 1.25790 1.28014 1.23964 1.23964 1.12954 1.09522 0.00000 0.00000

        5.66322 5.14435 4.97169 5.86893 6.00359 5.85150 5.68395 6.78027 6.78027 5.72380 5.95193 6.86966 6.86966 4.92746 4.82086 0.00000 0.00000
        1.25451 1.14635 1.10761 1.28149 1.29462 1.27663 1.25723 1.19201 1.19201 1.26293 1.28665 1.23964 1.23964 1.10223 1.07827 0.00000 0.00000

10      S7      R0              Q0              S7              R7              S4              S6              Q6              RE
        6.69639 6.12423 5.86866 4.97033 5.03003 6.69639 6.69639 5.60049 5.59770 6.78027 6.78027 6.86966 6.86966 4.88929 4.74498 0.00000 0.00000
        1.14505 1.30806 1.28145 1.10742 1.12252 1.14505 1.14505 1.24466 1.24424 1.19201 1.19201 1.23964 1.23964 1.09559 1.06550 0.00000 0.00000

        6.69639 6.28121 6.00320 5.14284 5.21621 6.69639 6.69639 5.56674 5.58171 6.78027 6.78027 6.86966 6.86966 4.88588 4.74554 0.00000 0.00000
        1.14505 1.32053 1.29458 1.14630 1.16396 1.14505 1.14505 1.24104 1.24278 1.19201 1.19201 1.23964 1.23964 1.09566 1.06726 0.00000 0.00000

11      R4      R6              S7              R4              R1              R4              S0              RC              RE
        5.57878 5.85469 5.85155 6.69639 6.69639 5.60961 5.63851 5.75629 5.86292 5.56877 5.56573 6.95092 6.95092 0.00000 0.00000 0.00000 0.00000
        1.24456 1.27686 1.27666 1.14505 1.14505 1.24693 1.25186 1.26303 1.27493 1.24113 1.24058 1.32510 1.32510 0.00000 0.00000 0.00000 0.00000

        5.56094 5.67069 5.68367 6.69639 6.69639 5.63829 5.69632 5.91092 6.08023 5.55731 5.55056 6.95092 6.95092 0.00000 0.00000 0.00000 0.00000
        1.24011 1.25584 1.25721 1.14505 1.14505 1.25183 1.25994 1.27914 1.29629 1.23949 1.23861 1.32510 1.32510 0.00000 0.00000 0.00000 0.00000

12      R4      S4              R7              R1              S4              S1              Q6              RE
        5.84143 6.78027 6.78027 5.60193 5.56776 5.75679 5.91164 6.78027 6.78027 6.91356 6.91356 4.92894 4.75861 0.00000 0.00000
        1.27561 1.19201 1.19201 1.24462 1.24117 1.26309 1.27922 1.19201 1.19201 1.28855 1.28855 1.10744 1.07105 0.00000 0.00000

        5.68235 6.78027 6.78027 5.59864 5.58249 5.86356 6.08140 6.78027 6.78027 6.91356 6.91356 4.90088 4.74446 0.00000 0.00000
        1.25742 1.19201 1.19201 1.24433 1.24288 1.27498 1.29626 1.19201 1.19201 1.28855 1.28855 1.09916 1.06587 0.00000 0.00000

13      R4      R6              S4              R4              S1              Q4              RC              RE
        5.57909 5.67953 5.72356 6.78027 6.78027 5.56883 5.55693 6.91356 6.91356 4.85531 4.74683 0.00000 0.00000 0.00000 0.00000
        1.24461 1.25787 1.26290 1.19201 1.19201 1.24113 1.23943 1.28855 1.28855 1.09086 1.06736 0.00000 0.00000 0.00000 0.00000

        5.61646 5.88442 5.95177 6.78027 6.78027 5.56549 5.55019 6.91356 6.91356 4.85549 4.74674 0.00000 0.00000 0.00000 0.00000
        1.24894 1.28012 1.28663 1.19201 1.19201 1.24056 1.23856 1.28855 1.28855 1.09096 1.06731 0.00000 0.00000 0.00000 0.00000

14      S4      S6              S6              S0              Q6              RC              RE
        6.78027 6.86966 6.86966 6.86966 6.86966 6.95092 6.95092 4.92855 4.90052 0.00000 0.00000 0.00000 0.00000
        1.19201 1.23964 1.23964 1.23964 1.23964 1.32510 1.32510 1.10737 1.09914 0.00000 0.00000 0.00000 0.00000

        6.78027 6.86966 6.86966 6.86966 6.86966 6.95092 6.95092 4.75818 4.74411 0.00000 0.00000 0.00000 0.00000
        1.19201 1.23964 1.23964 1.23964 1.23964 1.32510 1.32510 1.07096 1.06585 0.00000 0.00000 0.00000 0.00000

15      Q7      Q1              Q6              RC              RE              RE
        4.83492 5.05940 4.92710 4.88904 4.88492 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
        1.08358 1.12943 1.10215 1.09550 1.09554 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000

        4.81610 4.91655 4.83412 4.75709 4.75724 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
        1.08189 1.10081 1.08353 1.07050 1.07228 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000

RE      RE      RE              RE              RE
        0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
        0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000

        0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
        0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
```

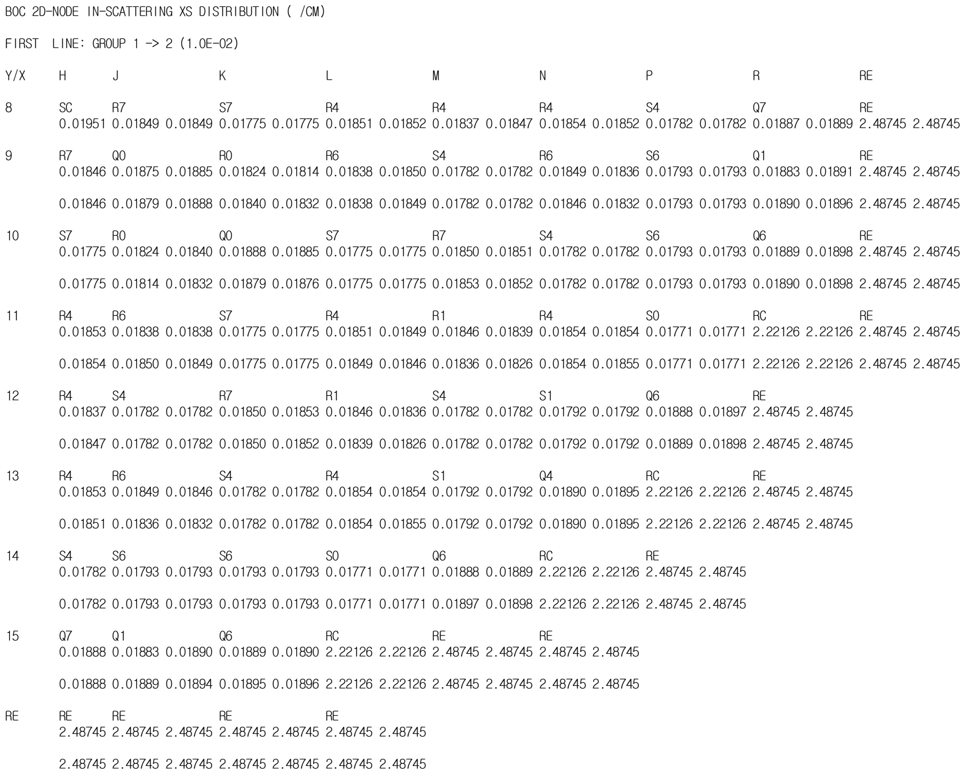
Figure 3-2. Sample input of the 5 types of the BOC macro cross-section: fast/thermal nu-fission XS, fast/thermal absorption XS, fast to thermal scattering XS

(1/3)

```
BOC 2D-NODE ABSORPTION XS DISTRIBUTION ( /CM)

FIRST  LINE: GROUP 1 (1.0E-03)
SECOND LINE: GROUP 2 (1.0E-02)

Y/X     H       J               K               L               M               N               P               R               RE

8       SC      R7              S7              R4              R4              R4              S4              Q7              RE
        8.01580 9.14219 9.14218 9.33165 9.33165 9.15271 9.15400 9.11081 9.13901 9.16237 9.15392 9.31903 9.31903 9.24578 9.25647 1.24007 1.24007
        4.74785 8.15172 8.15148 7.64284 7.64284 8.16940 8.16107 8.07232 8.14474 8.17620 8.16094 7.60627 7.60627 7.80842 7.79959 8.88229 8.88229

9       R7      Q0              R0              R6              S4              R6              S6              Q1              RE
        9.13270 9.21002 9.23826 9.10429 9.08931 9.11806 9.15170 9.31903 9.31903 9.15144 9.11424 9.31254 9.31254 9.22870 9.26432 1.24007 1.24007
        8.13426 8.06055 7.97677 7.99441 7.87720 8.07879 8.16504 7.60627 7.60627 8.16946 8.06595 7.58630 7.58630 7.97787 7.92128 8.88229 8.88229

        9.13306 9.21774 9.24926 9.13545 9.11520 9.11859 9.14857 9.31903 9.31903 9.14232 9.10704 9.31254 9.31254 9.26025 9.28828 1.24007 1.24007
        8.13539 8.01504 7.94057 8.13298 8.05246 8.08109 8.15765 7.60627 7.60627 8.14831 8.03635 7.58630 7.58630 7.94208 7.90037 8.88229 8.88229

10      S7      R0              Q0              S7              R7              S4              S6              Q6              RE
        9.33165 9.10431 9.13550 9.24975 9.23908 9.33165 9.33165 9.14234 9.14269 9.31903 9.31903 9.31254 9.31254 9.26897 9.31465 1.24007 1.24007
        7.64284 7.99455 8.13310 7.94096 7.97858 7.64284 7.64284 8.14237 8.14255 7.60627 7.60627 7.58630 7.58630 7.93459 7.89830 8.88229 8.88229

        9.33165 9.08933 9.11524 9.21854 9.21029 9.33165 9.33165 9.15144 9.14771 9.31903 9.31903 9.31254 9.31254 9.27111 9.31694 1.24007 1.24007
        7.64284 7.87741 8.05265 8.01692 8.06060 7.64284 7.64284 8.16158 8.15435 7.60627 7.60627 7.58630 7.58630 7.93976 7.91120 8.88229 8.88229

11      R4      R6              S7              R4              R1              R4              S0              RC              RE
        9.16184 9.11807 9.11862 9.33165 9.33165 9.15270 9.14848 9.13800 9.11951 9.16116 9.16147 9.04297 9.04297 1.24007 1.24007 1.24007 1.24007
        8.18734 8.07883 8.08127 7.64284 7.64284 8.16371 8.16286 8.11873 8.06531 8.17542 8.17472 7.09388 7.09388 8.88229 8.88229 8.88229 8.88229

        9.16246 9.15175 9.14864 9.33165 9.33165 9.14852 9.13804 9.11125 9.09177 9.16328 9.16485 9.04297 9.04297 1.24007 1.24007 1.24007 1.24007
        8.17632 8.16513 8.15785 7.64284 7.64284 8.16293 8.14661 8.03268 7.93989 8.17757 8.17971 7.09388 7.09388 8.88229 8.88229 8.88229 8.88229

12      R4      S4              R7              R1              S4              S1              Q6              RE
        9.11070 9.31903 9.31903 9.14157 9.15120 9.13792 9.11115 9.31903 9.31903 9.21099 9.21099 9.26449 9.31508 1.24007 1.24007
        8.07163 7.60627 7.60627 8.13980 8.16119 8.11854 8.03233 7.60627 7.60627 7.51606 7.51606 7.97273 7.92276 8.88229 8.88229

        9.13891 9.31903 9.31903 9.14242 9.14754 9.11938 9.09148 9.31903 9.31903 9.21099 9.21099 9.26736 9.31615 1.24007 1.24007
        8.14427 7.60627 7.60627 8.14189 8.15409 8.06485 7.93810 7.60627 7.60627 7.51606 7.51606 7.94493 7.90177 8.88229 8.88229

13      R4      R6              S4              R4              S1              Q4              RC              RE
        9.16177 9.15149 9.14237 9.31903 9.31903 9.16113 9.16337 9.21099 9.21099 9.27179 9.30530 1.24007 1.24007 1.24007 1.24007
        8.18725 8.16954 8.14840 7.60627 7.60627 8.17532 8.17767 7.51606 7.51606 7.93076 7.89654 8.88229 8.88229 8.88229 8.88229

        9.15265 9.11426 9.10705 9.31903 9.31903 9.16154 9.16494 9.21099 9.21099 9.27197 9.30531 1.24007 1.24007 1.24007 1.24007
        8.16931 8.06600 8.03641 7.60627 7.60627 8.17487 8.17983 7.51606 7.51606 7.93145 7.89632 8.88229 8.88229 8.88229 8.88229

14      S4      S6              S6              S0              Q6              RC              RE
        9.31903 9.31254 9.31254 9.31254 9.31254 9.04297 9.04297 9.26461 9.26759 1.24007 1.24007 1.24007 1.24007
        7.60627 7.58630 7.58630 7.58630 7.58630 7.09388 7.09388 7.97269 7.94535 8.88229 8.88229 8.88229 8.88229

        9.31903 9.31254 9.31254 9.31254 9.31254 9.04297 9.04297 9.31524 9.31639 1.24007 1.24007 1.24007 1.24007
        7.60627 7.58630 7.58630 7.58630 7.58630 7.09388 7.09388 7.92270 7.90216 8.88229 8.88229 8.88229 8.88229

15      Q7      Q1              Q6              RC              RE              RE
        9.26665 9.22897 9.26033 9.26895 9.27155 1.24007 1.24007 1.24007 1.24007 1.24007 1.24007
        7.89237 7.97796 7.94192 7.93413 7.94016 8.88229 8.88229 8.88229 8.88229 8.88229 8.88229

        9.30153 9.28996 9.31397 9.34042 9.34305 1.24007 1.24007 1.24007 1.24007 1.24007 1.24007
        7.92831 7.96969 7.94739 7.94424 7.95788 8.88229 8.88229 8.88229 8.88229 8.88229 8.88229

RE      RE      RE              RE              RE
        1.24007 1.24007 1.24007 1.24007 1.24007 1.24007 1.24007
        8.88229 8.88229 8.88229 8.88229 8.88229 8.88229 8.88229

        1.24007 1.24007 1.24007 1.24007 1.24007 1.24007 1.24007
        8.88229 8.88229 8.88229 8.88229 8.88229 8.88229 8.88229
```

Figure 3-2. Sample input of the 5 types of the BOC macro cross-section: fast/thermal nu-fission XS, fast/thermal absorption XS, fast to thermal scattering XS

(2/3)



```
BOC 2D-NODE IN-SCATTERING XS DISTRIBUTION ( /CM)

FIRST  LINE: GROUP 1 -> 2 (1.0E-02)

Y/X     H       J               K               L               M               N               P               R               RE

8       SC      R7              S7              R4              R4              R4              S4              Q7              RE
        0.01951 0.01849 0.01849 0.01775 0.01775 0.01851 0.01852 0.01837 0.01847 0.01854 0.01852 0.01782 0.01782 0.01887 0.01889 2.48745 2.48745

9       R7      Q0              R0              R6              S4              R6              S6              Q1              RE
        0.01846 0.01875 0.01885 0.01824 0.01814 0.01838 0.01850 0.01782 0.01782 0.01849 0.01836 0.01793 0.01793 0.01883 0.01891 2.48745 2.48745

        0.01846 0.01879 0.01888 0.01840 0.01832 0.01838 0.01849 0.01782 0.01782 0.01846 0.01832 0.01793 0.01793 0.01890 0.01896 2.48745 2.48745

10      S7      R0              Q0              S7              R7              S4              S6              Q6              RE
        0.01775 0.01824 0.01840 0.01888 0.01885 0.01775 0.01775 0.01850 0.01851 0.01782 0.01782 0.01793 0.01793 0.01889 0.01898 2.48745 2.48745

        0.01775 0.01814 0.01832 0.01879 0.01876 0.01775 0.01775 0.01853 0.01852 0.01782 0.01782 0.01793 0.01793 0.01890 0.01898 2.48745 2.48745

11      R4      R6              S7              R4              R1              R4              S0              RC              RE
        0.01853 0.01838 0.01838 0.01775 0.01775 0.01851 0.01849 0.01846 0.01839 0.01854 0.01854 0.01771 0.01771 2.22126 2.22126 2.48745 2.48745

        0.01854 0.01850 0.01849 0.01775 0.01775 0.01849 0.01846 0.01836 0.01826 0.01854 0.01855 0.01771 0.01771 2.22126 2.22126 2.48745 2.48745

12      R4      S4              R7              R1              S4              S1              Q6              RE
        0.01837 0.01782 0.01782 0.01850 0.01853 0.01846 0.01836 0.01782 0.01782 0.01792 0.01792 0.01888 0.01897 2.48745 2.48745

        0.01847 0.01782 0.01782 0.01850 0.01852 0.01839 0.01826 0.01782 0.01782 0.01792 0.01792 0.01889 0.01898 2.48745 2.48745

13      R4      R6              S4              R4              S1              Q4              RC              RE
        0.01853 0.01849 0.01846 0.01782 0.01782 0.01854 0.01854 0.01792 0.01792 0.01890 0.01895 2.22126 2.22126 2.48745 2.48745

        0.01851 0.01836 0.01832 0.01782 0.01782 0.01854 0.01855 0.01792 0.01792 0.01890 0.01895 2.22126 2.22126 2.48745 2.48745

14      S4      S6              S6              S0              Q6              RC              RE
        0.01782 0.01793 0.01793 0.01793 0.01793 0.01771 0.01771 0.01888 0.01889 2.22126 2.22126 2.48745 2.48745

        0.01782 0.01793 0.01793 0.01793 0.01793 0.01771 0.01771 0.01897 0.01898 2.22126 2.22126 2.48745 2.48745

15      Q7      Q1              Q6              RC              RE              RE
        0.01888 0.01883 0.01890 0.01889 0.01890 2.22126 2.22126 2.48745 2.48745 2.48745 2.48745

        0.01888 0.01889 0.01894 0.01895 0.01896 2.22126 2.22126 2.48745 2.48745 2.48745 2.48745

RE      RE      RE              RE              RE
        2.48745 2.48745 2.48745 2.48745 2.48745 2.48745 2.48745

        2.48745 2.48745 2.48745 2.48745 2.48745 2.48745 2.48745
```

Figure 3-2. Sample input of the 5 types of the BOC macro cross-section: fast/thermal nu-fission XS, fast/thermal absorption XS, fast to thermal scattering XS

(3/3)

### 3.5.2 Block: SE Residual CNN

This ANN module is the main neural network for this prediction. Since most of the current convolutional neural network works with a residual neural network the main concept should be first explained. The key concept of the residual neural network is the skip connection as shown in Figure 3-3.

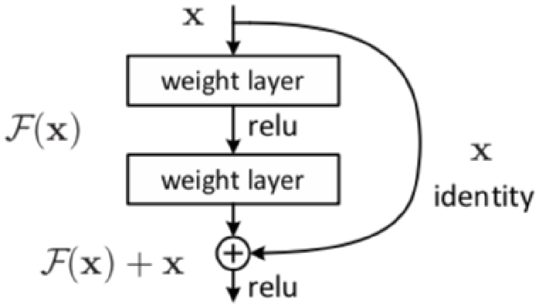

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 3-3. residual neural network

Without the skip connection, F(x) is the output from convolutional layers. Because the original information x is may still be relevant information, x identity was added to F(x) which is H(x). Only learning that we need to do is everything except identity x. Therefore, it is easier for ANN to learn more complex features[9].

Our model is a combination of three networks, ResNet, Inception-v3, and SE Network. The summary of the Modified-ResNet network is as shown in Figure 3-4. Each layer is explained in detail in the following sections.

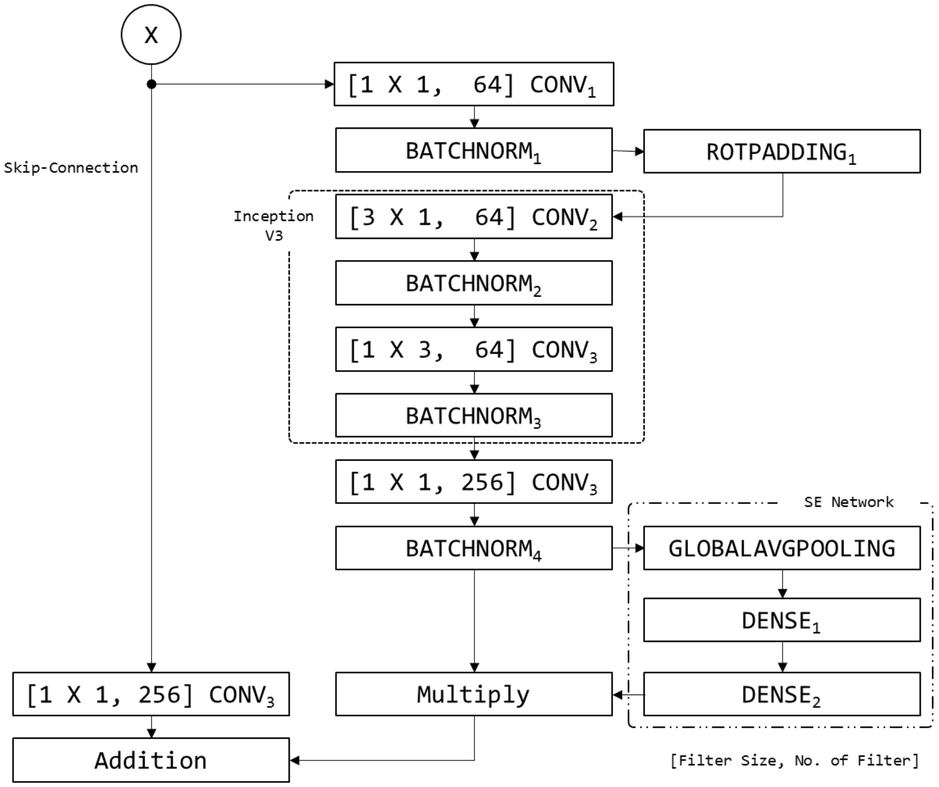


Figure 3-4. Modified SE Residual CNN Layout

### 3.5.3 Layer: CONV1

CONV1 layer is to reduce the dimension for CONV2 for faster calculation. This specific residual neural network is called bottleneck design in the original paper[9]. CONV1 layer can take two inputs: 5 features from the input layer or 256 features from the output of the previous residual neural network layer. The input from the previous layer is applied with 1x1x64 filters with 1x1 strides which means that it will produce 17x17x64 as its output. Disregarding the input from the input layer, the number of features is reduced from 256 to 64.

### 3.5.4 Layer: BATCHNORM

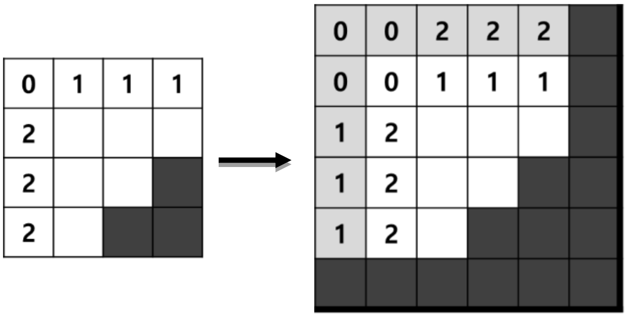
Batch–Normalization (BN) layer is presented to solve the vanishing and exploding gradient problem and internal covariate shift. Since each layer is normalized, any small value will be increased and any large value will be decreased. This results in mitigating the effects of vanishing and exploding gradient descent. Moreover, BN layer helps NN to train faster, because of the whitening effect where means are set to zero and only the variants within the layers will be meaningful. “The covariate shift is the change in the distribution of network activations due to the change in network parameters during training.” [10]

## 3.5.5 Layer: CONV2 and 3

CONV2 and 3 layers are presented to find a spatial relationship between surrounding node−wise assemblies. Our prototype model is based on ResNet which is a 3x3 convolution filter. However, later papers such as inception v3 show that it is better to implement two 1x3 and 3x1 layers for a deeper network. [11]
The general knowledge is that it is efficient to build a deeper model with skipping connections. Therefore, we repeated 15 times.

## 3.5.6 Layer: ROTPADDING1

ROTPADDING1 layer is needed for quadrant inputs. Because our input only has 4th quadrant, the rotational symmetry padding must be applied beginning of each CONV2 and 3 layers because the gray area is not reflectors (denoted as black boxes) but another node as shown in the Figure 3−5[12]. Therefore, before each CONV2 and 3 layers, this layer needs to copy the nodes from white spaces to gray spaces. This will greatly reduce memory issues and computational time.



| | | | |
|---|---|---|---|
| 0 | 1 | 1 | 1 |
| 2 | | | |
| 2 | | | ■ |
| 2 | | ■ | ■ |

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 2 | 2 | 2 | ■ |
| 0 | 0 | 1 | 1 | 1 | ■ |
| 1 | 2 | | | | ■ |
| 1 | 2 | | | ■ | ■ |
| 1 | 2 | | ■ | ■ | ■ |
| ■ | ■ | ■ | ■ | ■ | ■ |

Figure 3−5. Rotationally symmetric padding

### 3.5.7 Layer: Squeeze and Excitation (SE) Network

SE layer is for faster training and by learning the importance of each layer’ s features. This layer achieves the importance in features with GLOBAL−AVERAGE −POOLING1 layer. There are 256 features from CONV3 with batch normalization. The features are globally averaged. From the features of the average, DENSE1 and DENSE2 can learn the importance of each feature by finding the relationship with each other. Finally, important features are multiplied to the output of CONV5 layers for faster learning[13].

### 3.5.8 Block: CNN for Depletion

This ANN module is sub neural network for depletion. The summary of the CNN for depletion is as shown in Figure 3−6. In this network, ROTPADDING and BATCHNORM layer is used as same as SE−Residual CNN, but SE and Residual network concept are not used. Each layer is explained in detail in the following sections.

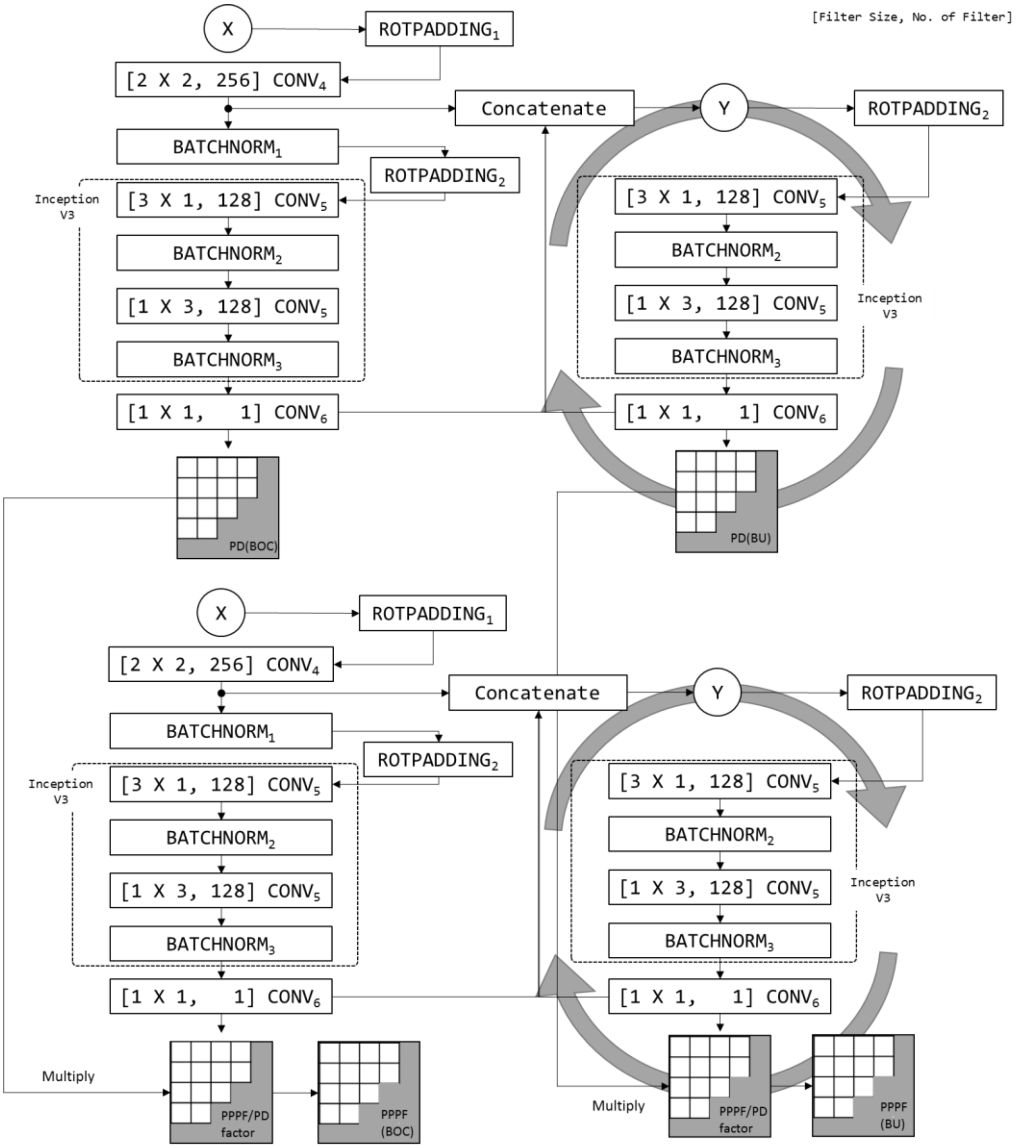


Figure 3-6. CNN for depletion layout

### 3.5.9 Layer: CONV4

CONV4 layer is to change the dimension to assembly-wise[9x9] from node-wise[17x17]. For that, it is applied with 2x2x256 filters with 2x2 strides. It will produce 9x9x256 that is the core feature for depletion each assembly power and pin power peaking factor.

### 3.5.10 Layer: CONV5

CONV5 layer is depletion series CNN. To reflect the unique properties of each depletion step, CNN that looks at immediate surrounding (3x3) assemblies is introduced. Moreover, features from previous depletion step are concatenated to facilitate the prediction of the current depletion step. As same as CONV 2 and 3, inception v3 is used.

### 3.5.11 Layer: CONV6

CONV6 is applied with a 1x1 filter with 1x1 stride 1 feature. So, it will produce 9X9X1 same as assembly power and pin peaking assembly-wise quadrant distribution. In the two-step method, the pin-wise power distribution is determined by employing the pin power reconstruction method. Similarly, Fr/PD factor will be produced by CONV6 and multiplied by PD from current burnup depletion step.

# 4. Result

## 4.1 Loading Pattern Random Generation

### 4.1.1 Train & Validation Data

The target plant for test is OPR1000 (177 fuel assemblies). The feed assembly uses gadolinia as a burnable absorption rod. Four different types of burnable absorbers differing in the number and position were randomly selected for each location. In addition, the following are assumed:

- No. of Feed assembly is fixed (69 Feed)
- Octant Symmetry

18827 loading patterns (16659 LPs for train data and 2168 LPs for validation data) were produced using the assumed conditions and 3-D core calculation code (ASTRA).

Train data

- Assembly Relative Power Range: 0.07 ~ 3.80
- Assembly Maximum PPPF Range: 0.15 ~ 4.33
- Cycle Maximum PPPF Range: 1.54 ~ 4.33

Validation data

- Assembly Relative Power Range: 0.09 ~ 2.90
- Assembly Maximum PPPF Range: 0.17 ~ 3.54
- Cycle Maximum PPPF Range: 1.53 ~ 3.54

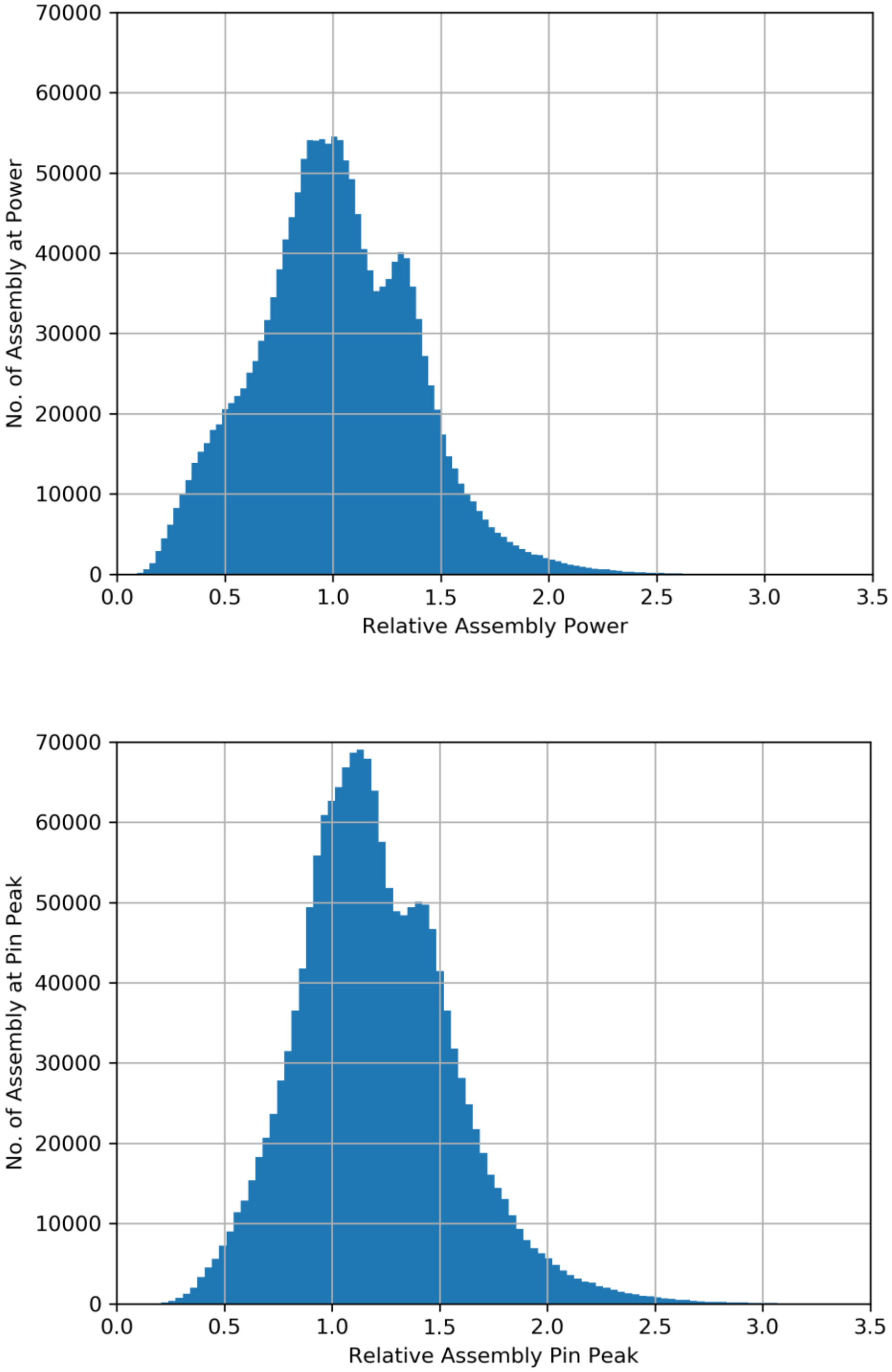


Figure 4−1. Assembly power and PPPF distribution of the train data

### 4.1.2 Test Data

2692 loading patterns are generated for test. Based on the optimized loading pattern (equilibrium cycle) that used to a recent cycle of OPR1000, only the loading pattern with near a maximum pin power peaking factor(PPPF) of '1.60' was selected. '1.60' is a boundary value for accident analysis and is also a reference value when the designer finds the loading pattern. Additional LP's were generated independently from the train data.

- Assembly Relative Power Range: 0.23 ~ 1.50
- Assembly Maximum PPPF Range: 0.46 ~ 1.77
- Cycle Maximum PPPF Range: 1.46 ~ 1.77

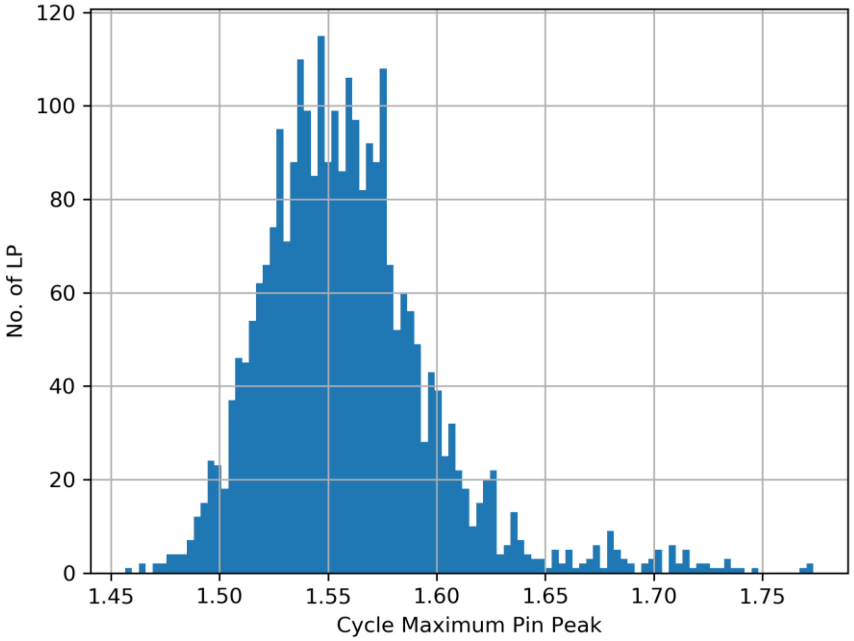


Figure 4-2. Cycle maximum PPPF distribution of the test data

## 4.2 Supervised Learning

### 4.2.1 Fully Connected (FC) vs. Convolutional Neural Network (CNN)

The OLL is a class of fully connected(FC) neural network. So, we compared the FC method and the CNN method instead of OLL and CNN. In order to compare only the differences according to the network, the number of parameters used in the network is made equal. In the case of FC, k−inf and two macro cross−sections are used as in the previous study, and in CNN, five macro cross−sections are used as described above. The mean squared error is set to loss and learned to minimize it. The learning time is the same as the one hour, and at the completion of the learning, the loss can confirm that CNN is lower than FC.

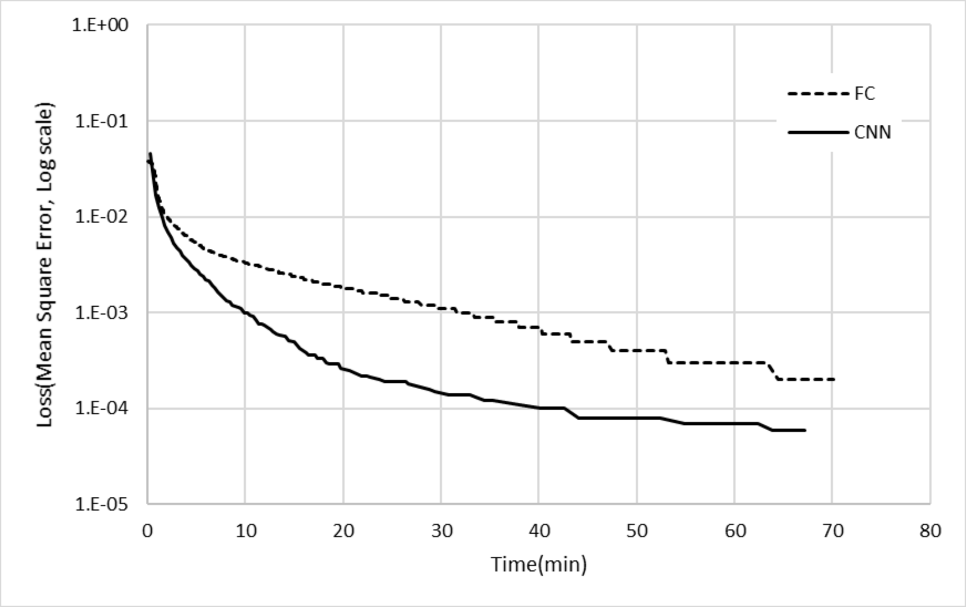


Figure 4−3. Loss according to learning time about FC vs. CNN for BOC only

The predicted results are shown in the following table. There is no significant difference in the mean error, but the maximum error is almost twice the difference. If we calculate the fraction of the assemblies with over than specific absolute error, we can see that CNN accurately predicts the assembly-wise power distribution rather than FC.

Table 4-1: Power distribution prediction error of FC and CNN

| Network Type | $e_{avg}$[a] | $e_{max}$[b] | Frac. With $e^c > 3\%$ | Frac. With $e^c > 5\%$ |
|---|---|---|---|---|
| FC | 1.05 | 11.92 | 4.8 | 0.4 |
| CNN | 0.44 | 4.23 | 0.0 | 0.0 |

a= average absolute error (%)
b= maximum absolute error (%)
c= Fraction of the assemblies with absolute error(%)

Three loading patterns are selected to check the range of prediction. The first is the optimized loading pattern (equilibrium cycle) that used to a recent cycle for OPR1000 and meets the assumed conditions, the second is that add feed assembly inside, and last is that increased neutron leakages by loading feed assembly at the periphery(outermost) location. The results using the learned CNN are as follows.

Table 4-2: Power distribution prediction error of the three models

| Network Type | Model | $e_{avg}$[a] | $e_{max}$[b] |
|---|---|---|---|
| CNN | Eq. | 0.48 | 1.96 |
| | 73Feed | 0.62 | 2.43 |
| | **73Feed+ H.L.**[d] | **3.27** | **17.35** |
| FC | Eq. | 1.00 | 2.60 |
| | 73Feed | 0.83 | 2.62 |
| | **73Feed+ H.L.**[d] | **24.35** | **69.14** |

a= Average absolute error (%)
b= Maximum absolute error (%)
c= Fraction of the assemblies with absolute error(%)
d= High neutron leakage rather than assumed condition

For the first model, the mean and maximum errors were larger than the validation model but well predicted. The error of the model which added feed assembly inside also increased, but CNN predicts the power distribution using the relation between the assemblies, considering that the absolute error exceeding 3% does not occur. Because we never saw the 73 feed H.L. model in the training process, the high error was found. However, when compared to FC, CNN error is comparably low. This demonstrates that CNN predicts the power distribution using the relationships and the trained model is well generalized to the problem.

### 4.2.2 Convolutional Neural Network for the whole cycle

16659 loading patterns and architecture (Figure 2-1) are used for training. The mean squared error is also used as the loss for training. The loss according to the time is as follows. It takes about two minutes to learn once (epoch). So, the time of training is almost 3 days and it learned over 2000 epoch to reach the desired loss.

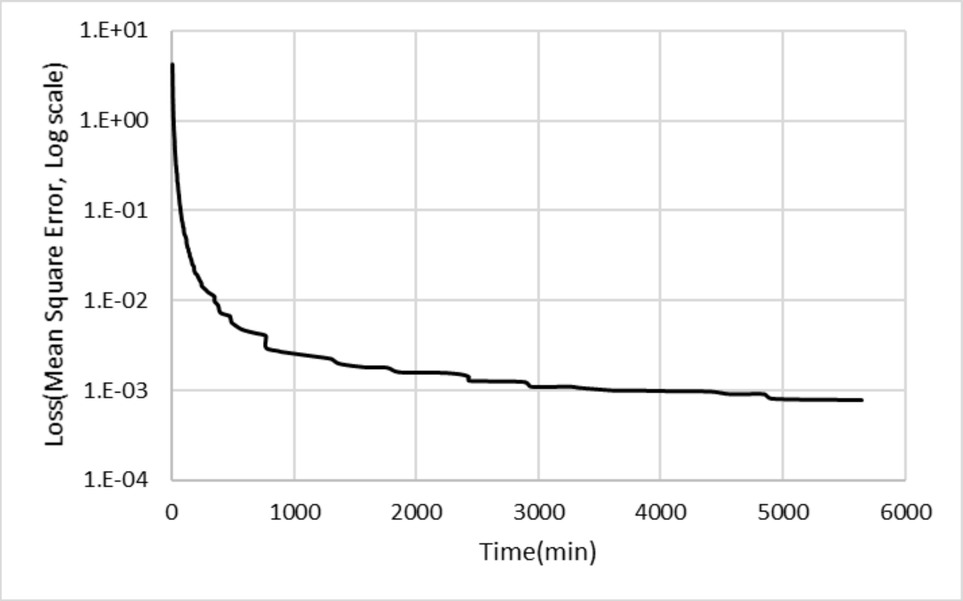


Figure 4-4. Loss according to learning time about CNN for the whole cycle

The graph (Figure 4-4) shows enough learning, and the maximum errors by each location of 2168 loading patterns for validation are as follows.

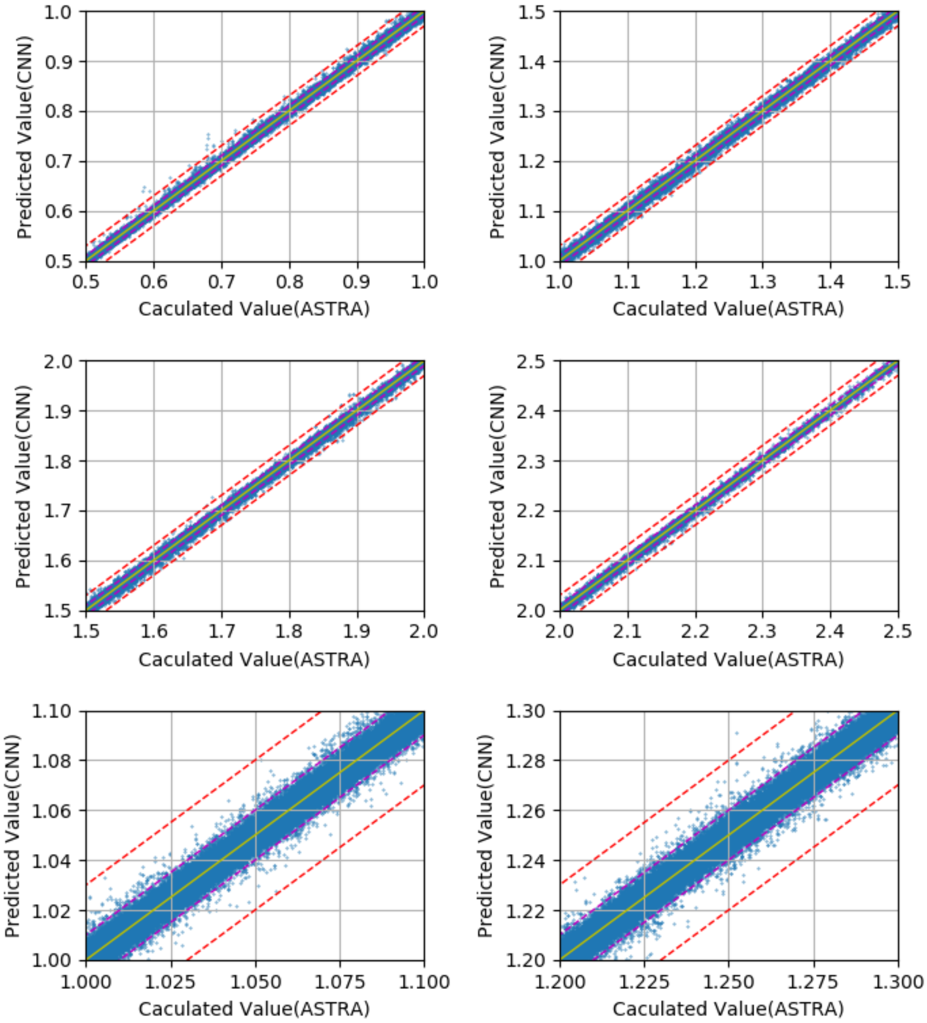
Table 4-3: Power distribution and pin power peaking factor prediction validation error of CNN for the whole cycle

| | $e_{avg}$[a] | $e_{max}$[b] | Frac. with $e^c > 1\%$ | Frac. with $e^c > 3\%$ |
|---|---|---|---|---|
| Assembly Power | 0.19 | 7.33 | 0.457 | 0.002 |
| PPPF | 0.31 | 9.13 | 2.836 | 0.013 |

a= average absolute error (%)
b= maximum absolute error (%)
c= Fraction of the assemblies with absolute error (%)

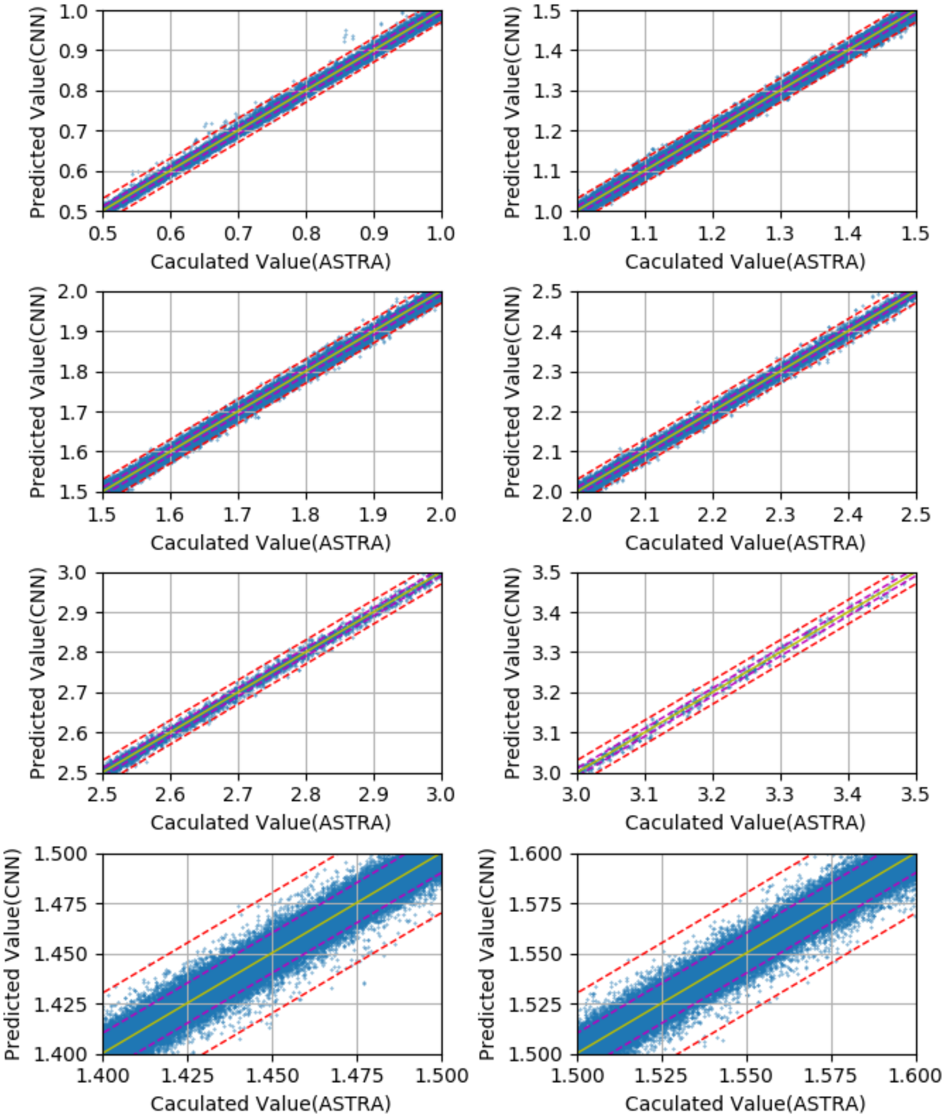
As a result of the validation, it was confirmed that the assembly and PPPF with the absolute error exceeding 1% is only 0.47% and 2.83% of the total assembly, respectively. And that it is in good agreement with the 3-D core calculation. The supervised learning method predicts the results through nonlinear regression analysis. The fact that the average is close to zero suggests that supervised learning is well done. In order to make it easier to understand, the scattering plot of assembly power and PPPF is shown in figure 4-5 and 4-6, respectively. The red line in the figure means absolute error ± 3% and the violet line means absolute error ± 1%.

The prediction results can be summarized as shown in figure 4-7, which is based on the application program used by the actual designer.



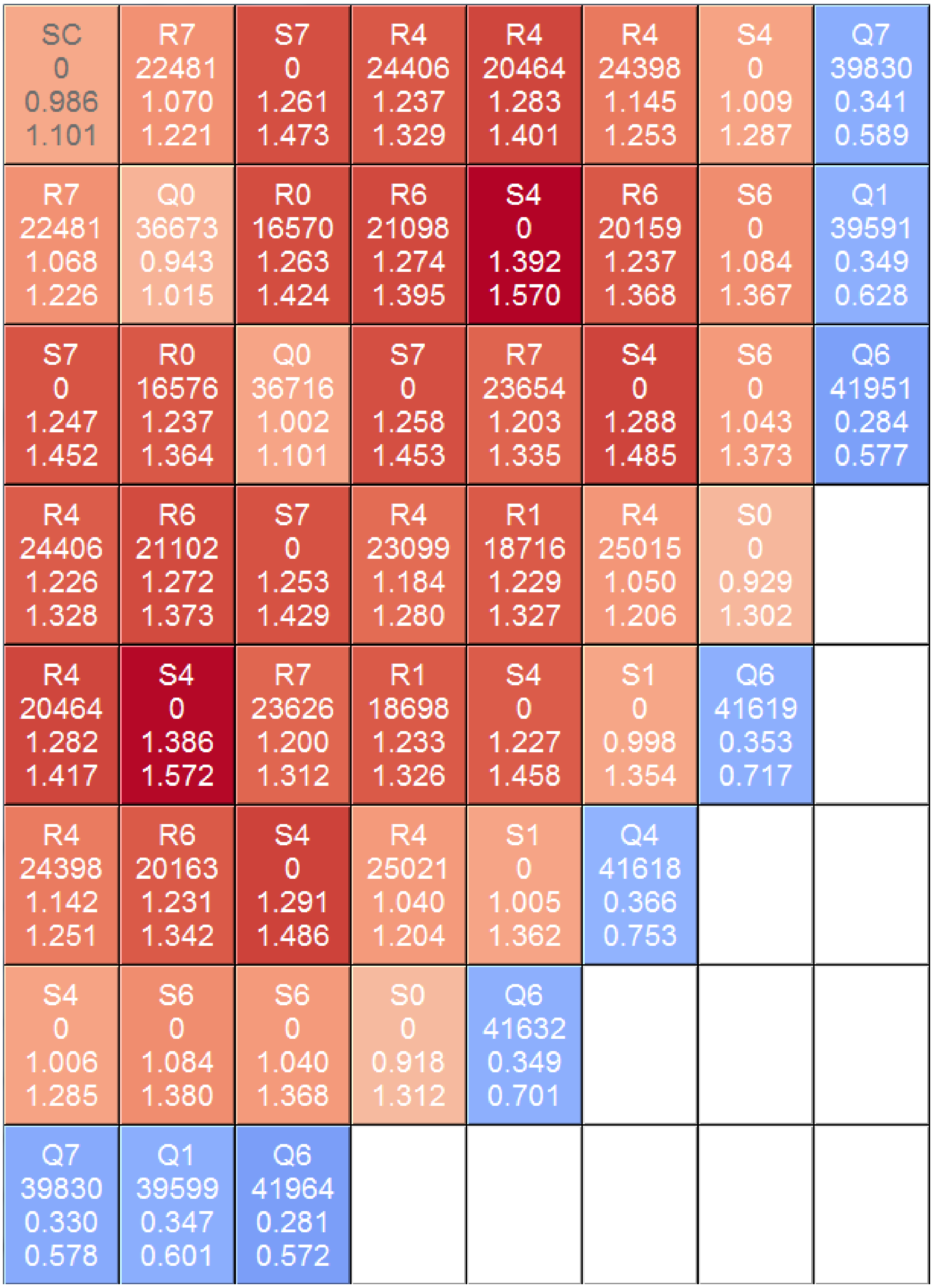
Purple line: ±1%
Red line: ±3%

Figure 4-5. Predicted(CNN) vs. Calculated(ASTRA): Assembly-wise assembly power



Purple line: ±1%
Red    line: ±3%

Figure 4-6. Predicted(CNN) vs. Calculated(ASTRA): Assembly-wise pin power peaking factor



| | | | | | | | |
|---|---|---|---|---|---|---|---|
| SC<br>0<br>0.986<br>1.101 | R7<br>22481<br>1.070<br>1.221 | S7<br>0<br>1.261<br>1.473 | R4<br>24406<br>1.237<br>1.329 | R4<br>20464<br>1.283<br>1.401 | R4<br>24398<br>1.145<br>1.253 | S4<br>0<br>1.009<br>1.287 | Q7<br>39830<br>0.341<br>0.589 |
| R7<br>22481<br>1.068<br>1.226 | Q0<br>36673<br>0.943<br>1.015 | R0<br>16570<br>1.263<br>1.424 | R6<br>21098<br>1.274<br>1.395 | S4<br>0<br>1.392<br>1.570 | R6<br>20159<br>1.237<br>1.368 | S6<br>0<br>1.084<br>1.367 | Q1<br>39591<br>0.349<br>0.628 |
| S7<br>0<br>1.247<br>1.452 | R0<br>16576<br>1.237<br>1.364 | Q0<br>36716<br>1.002<br>1.101 | S7<br>0<br>1.258<br>1.453 | R7<br>23654<br>1.203<br>1.335 | S4<br>0<br>1.288<br>1.485 | S6<br>0<br>1.043<br>1.373 | Q6<br>41951<br>0.284<br>0.577 |
| R4<br>24406<br>1.226<br>1.328 | R6<br>21102<br>1.272<br>1.373 | S7<br>0<br>1.253<br>1.429 | R4<br>23099<br>1.184<br>1.280 | R1<br>18716<br>1.229<br>1.327 | R4<br>25015<br>1.050<br>1.206 | S0<br>0<br>0.929<br>1.302 | |
| R4<br>20464<br>1.282<br>1.417 | S4<br>0<br>1.386<br>1.572 | R7<br>23626<br>1.200<br>1.312 | R1<br>18698<br>1.233<br>1.326 | S4<br>0<br>1.227<br>1.458 | S1<br>0<br>0.998<br>1.354 | Q6<br>41619<br>0.353<br>0.717 | |
| R4<br>24398<br>1.142<br>1.251 | R6<br>20163<br>1.231<br>1.342 | S4<br>0<br>1.291<br>1.486 | R4<br>25021<br>1.040<br>1.204 | S1<br>0<br>1.005<br>1.362 | Q4<br>41618<br>0.366<br>0.753 | | |
| S4<br>0<br>1.006<br>1.285 | S6<br>0<br>1.084<br>1.380 | S6<br>0<br>1.040<br>1.368 | S0<br>0<br>0.918<br>1.312 | Q6<br>41632<br>0.349<br>0.701 | | | |
| Q7<br>39830<br>0.330<br>0.578 | Q1<br>39599<br>0.347<br>0.601 | Q6<br>41964<br>0.281<br>0.572 | | | | | |

First line: Fuel Assembly Name
Second line: BOC burnup(MWD/MTU)
Third line: Power
Fourth line: Pin Power Peaking Factor(PPPF)

Figure 4-7. Sample prediction result

## 4.2.3 Test

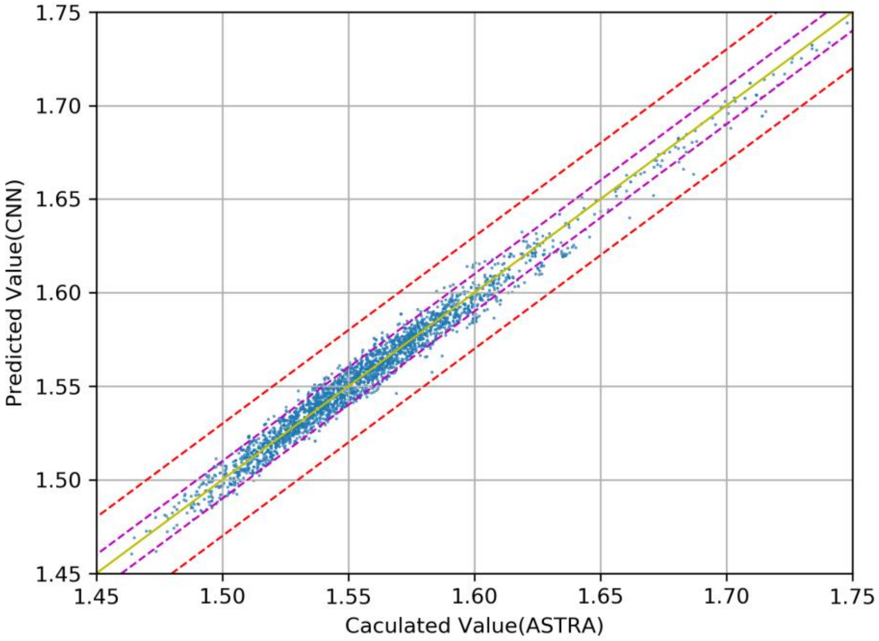
The reason for performing "Test" in addition to "Validation" is to ensure that it works well in the area of interest. As shown in Section 4.1.2, the loading patterns used in "Test" is somewhat optimized from the cycle maximum pin power peaking factor perspective. The results using the supervised learned convolution neural network are as follows.

Table 4-4: Power distribution and pin power peaking factor prediction error of test models

| | $e_{avg}$[a] | $e_{max}$[b] | Frac. with $e$[c] > 1% | Frac. with $e$[c] > 3% |
|---|---|---|---|---|
| Assembly Power | 0.14 | 2.47 | 0.141 | 0.000 |
| PPPF | 0.28 | 4.71 | 2.053 | 0.008 |

a= Average absolute error (%)
b= Maximum absolute error (%)
c= Fraction of the assemblies with absolute error (%)

As a result, it was confirmed that the artificial neural network has better prediction below a boundary value area. The maximum absolute error is less than 3% for the assembly power and the maximum error is 4.71% for the PPPF. Since the fraction of the error exceeding 3% is close to 0, it can be sufficiently connected with the automatic loading pattern searching code. In the case of the cycle maximum pin power peaking factor, it is predicted very well as shown in Figure 4-8 and the maximum absolute error is 2.56%. It is lower than that of assembly-wise value.



Purple line: ±1%
Red    line: ±3%

Figure 4-8. Predicted(CNN) vs. Calculated(ASTRA): Cycle maximum pin power peaking factor

### 4.2.4 Convolution Neural Network for the cycle maximum PPPF

In this section, we will look at why we should look at all of the assembly-wise power in order to better predict the cycle maximum pin peaking factor which is the main value of optimization, rather than seeing the one value. For this evaluation, we modified the existing artificial neural network. The first model predicts the cycle maximum pin peaking factor only(Figure 4-9) and the second predicts the maximum pin peaking factor at each burnup(Figure 4-10).

First, the assembly-wise maximum pin power peaking factor is predicted by using the artificial neural network evaluated above, and then the cycle maximum value is classified and predicted well as shown in the following Figure 4-11.

Table 4-5: No. of trained data by the 3 types of ANN

| | First[a] | Second[b] | Original[c] |
|---|---|---|---|
| No. of the trained data | 16,659 | 399,816 | 20,790,432 |

a= the cycle maximum pin peaking factor only
b= the maximum pin peaking factor at each burnup
c= the assembly-wise maximum pin peaking factor

In contrast, if the first model is not predicted as shown in Figure 4-12. The largest reason is that the number of trained data(Table 4-5). In the case of the original artificial neural network, the number of trained data is the number of LPs(16,659) * the number of burnup steps(24) * the number of quadrant assembly(52), whereas the first model is that the number of LPs is all of the trained data. The second model has better predictions than the first model but still does not predict the level of existing neural network model(Figure 4-13). In other words, the original artificial neural networks can predict more accurate with less the number of loading patterns data. In an aspect of prediction using a deep neural network with supervised learning, showing various output distributions makes better prediction possible.

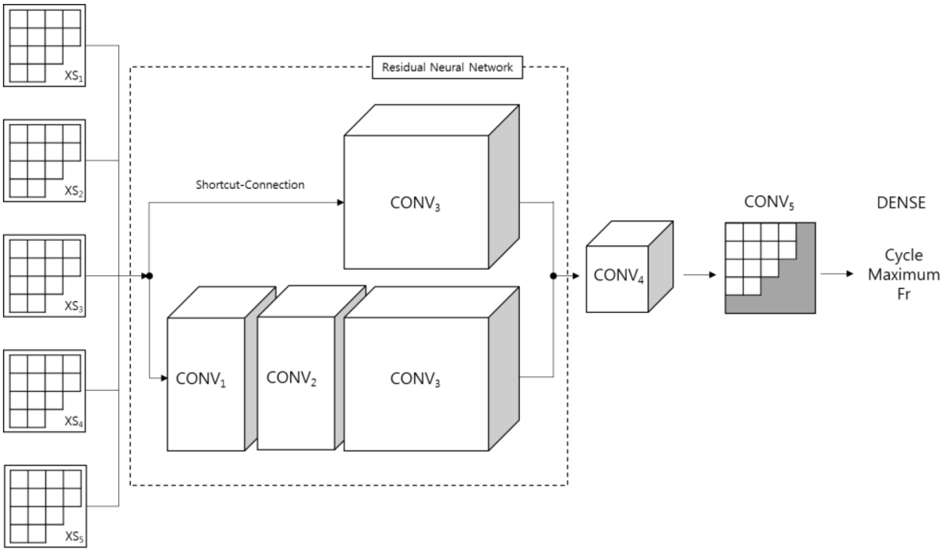


Figure 4-9. Modified CNN for the cycle maximum layout

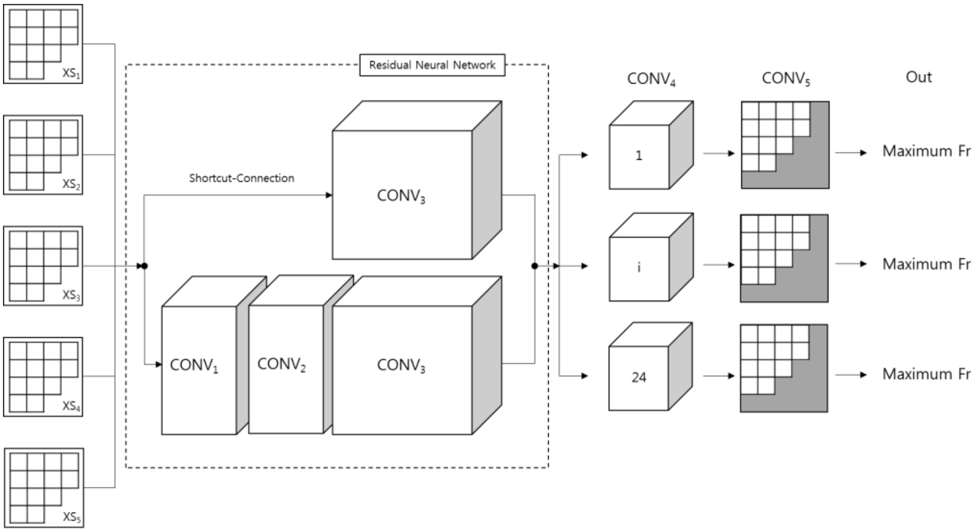
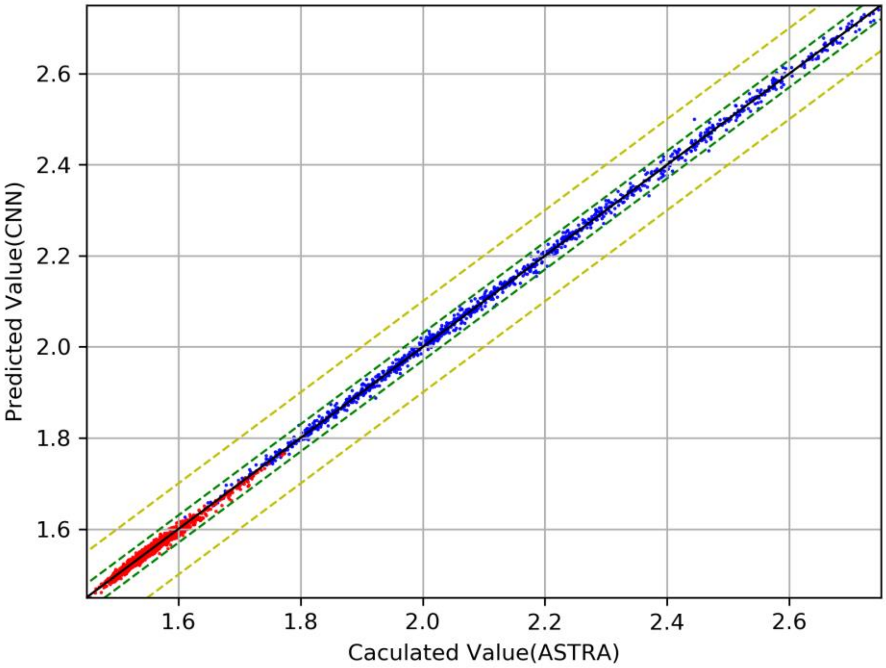


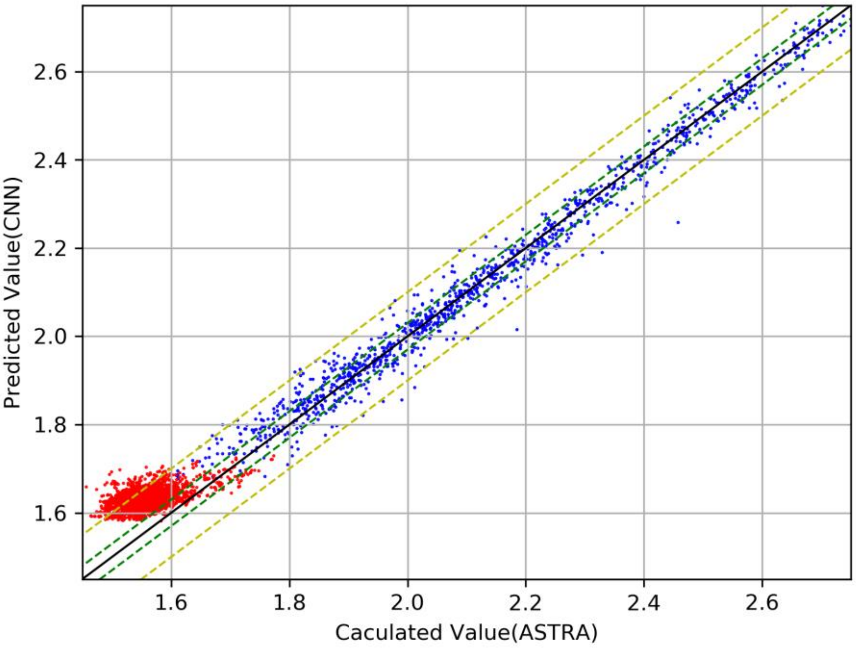
Figure 4-10. Modified CNN for the maximum Fr at each burnup step layout



Green  line: ±3%
Yellow line: ±10%

Blue   dot: “Validation”
Red    dot: “Test”

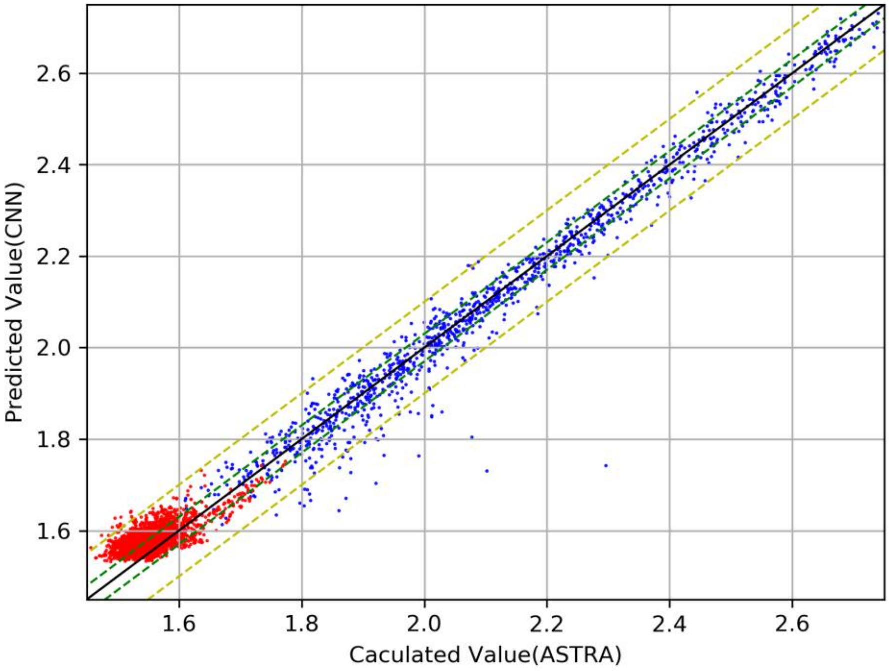
Figure 4-11. Predicted(CNN) vs. Calculated(ASTRA): Cycle maximum pin power peaking factor, Original model



Green line: ±3%
Yellow line: ±10%

Blue dot: “Validation”
Red dot: “Test”

Figure 4-12. Predicted(CNN) vs. Calculated(ASTRA): Cycle maximum pin power peaking factor, the cycle maximum model



Green line: ±3%
Yellow line: ±10%

Blue dot: “Validation”
Red dot: “Test”

Figure 4-13. Predicted(CNN) vs. Calculated(ASTRA): Cycle maximum pin power peaking factor, the maximum at each burnup model

# 5. Conclusions

Convolutional neural networks were applied in the prediction of the 2D assembly−wise core power and pin power peaking factor distributions for a whole cycle. It turned out that accurate values were obtained in a very short time. The computational time is only around 0.2 second on a personal computer equipped with a CPU of Intel i7−3770 (3.40GHz, DDR3 16GB) and the error in the cycle maximum pin power peaking factor at the region of interest was less than 3%. The model can greatly reduce the computing time of the LP optimization process. It will thus be of great help to those who design the fuel loading patterns as well as to an automatic LP searching program. Nonetheless, this model has one limitation. The trained neural network model is valid only within the specified core conditions: the number of the total and fresh fuels, initial boron concentration, T/H conditions, etc.  In order to improve this, it is necessary to perform reinforcement learning and to develop a module that creates a loading pattern that is equivalent to the actual design at different conditions.

# Reference

1. Yoon, J.I., Joo, H.G., "Two-level coarse mesh finite difference formulation with multigroup source expansion nodal kernels", J. Nucl. Sci. Technol. 45, 668-682, (2008).

2. C. S. Jang, H. J. Shim, C. H. Kim, Optimization layer by layer networks for in-core fuel management optimization computations in PWRs, Annals of Nuclear Energy Volume 28, Issue 11, July 2001, Pages 1115-1132.

3. "CS231n Convolutional Neural Networks for Visual Recognition", cs231n.github.io, Retrieved December 13, 2018.

4. Deng, L., Yu, D., Deep Learning: Methods and Applications, Foundations and Trends in Signal Processing, 2014, pages 1-199.

5. Stuart J. Russell, Peter Norvig, Artificial Intelligence: A Modern Approach, Third Edition, Prentice Hall, 2010.

6. Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar Foundations of Machine Learning, The MIT Press, 2012.

7. "Convolutional Neural Networks (LeNet) - DeepLearning 0.1 documentation". Theano Development Team, Retrieved August 31, 2013.

8. Hamed Habibi Aghdam, Elnaz Jahani Heravi, “Guide to convolutional neural networks: a practical application to traffic-sign detection and classification” , Springer, May 2017.

9. He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2015-12-10). "Deep Residual Learning for Image Recognition". arXiv:1512.03385 [cs.CV].

10. Sergey loffe, Christian Szegedy (2015-02-11). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". arXiv:1502.03167 [cs.LG].

11. Christian Szegedy, Vincent Vanhoucke, Sergey loffe, Jonathon Shlens, Zbigniew Wojna (2015-12-02). "Rethinking the inception Architecture for Computer Vision". arXiv:1512.00567 [cs.CV].

12. Y. D. Nam, J. Y. Lee, H. J. Shim, "Convolutional Neural Network for BOC Pin Power Prediction", KNF, KNS Spring Meeting 2019.

13. Jie Hu, Li Shen, Samuel Albanie, Gang Sun, Enhua Wu (2017-09-05). "Squeeze-and-Excitation Networks". arXiv:1709.01507 [cs.CV].

14. Kim, H.G., Chang, S.H., Lee, B.H., “Pressurized water reactor core parameter prediction using an artificial neural network”, Nucl. Sci. Eng. 113, pp70-76, 1993.

# 초 록

## 경수로 노심 2차원 출력분포 예측을 위한 합성곱 인공 신경망

이진영
원자핵공학과
The Graduate School
Seoul National University

이 논문의 주 목표는 2차원 형태의 거시단면적을 이용하여 한국의 대표적인 가압경수로인 OPR1000의 장전모형을 빠르게 분석한 후 집합체 단위의 출력과 최대 핀 출력을 예측하는 인공신경망을 개발하는 것이다. 이와 같은 인공신경망 개발이 필요한 이유는 장전모형 최적화 단계에서 보다 빠른 계산 능력이 필요하기 때문이다.

이번 논문과 비슷한 목표를 가진 선행연구를 바탕으로 다른 형태의 인공신경망을 적용 평가하였다. 최근 이미지 분석을 위해 널리 쓰이고 있는 합성곱 인공신경망을 적용하였으며, 학습을 위한 데이터는 KNF사의 ASTRA 코드를 통해 만들었다. 약 2만개의 랜덤한 장전모형을 생산하였으며, 이를 인공신경망으로 지도학습하였다. 집합체 단위의 출력의 평균 오차와 최대 오차는 각각 0.19%와 7.34%이며, 최대 핀 출력의 평균 오차와 최대 오차는 각각 0.31%와 9.13%이다. 기존의 설계된 최적 장전모형과 비슷한 분포를 가진 약 3천개의 랜덤한 장전모형을 이용하여 검증한 결과 집합체 출력과 최대 핀 출력의 최대 오차는 각각 3%와 5% 수준으로 장전모형 최적화 작업 시 ASTRA 코드 대신 간이 평가 코드로 활용 가능한 수준이다. 계산 시간은 CPU(Intel i7-3770 3.40GHz, DDR3 16GB)기준 ASTRA 코드는 200초 정도 소요되지만, 개발된 인공신경망은 이보다 1000배 빠른 0.2초이다.

개발된 인공신경망은 장전모형 최적화 코드와의 연계를 통해 장전모형 최적화 시간을 크게 단축시킬 수 있을 뿐 아니라 설계자에게도 많은 도움을 줄 수 있다. 지도학습된 인공신경망은 동일한 노심 조건(연료 총 다발 수, 초기 붕소농도, 열수력 조건, 신연료 다발 수 등)에서만 앞선 오차율을 보장하며, 조건이 바뀔 경우 추가로 데이터 생산 후 추가 지도학습이 필요하다.

**주요어 :** 합성곱 신경망
지도학습
출력 분포 예측
가압경수로
**학　번 :** 2017-26188