Prediction of LOCA Break Size Using CFNN

Geon Pil Choi, Kwae Hwan Yoo, Ju Hyun Back, Dong Yeong Kim, and Man Gyun Na*
Department of Nuclear Engineering, Chosun University, 309 Pilmun-daero, Dong-gu, Gwangju, Korea
**Corresponding author: magyna@chosun.ac.kr*

1. Introduction

The many nuclear power plants (NPPs) have been operated globally because the demand of energy increases continuously. However, the most of the NPPs in operation were operated for a long time. The long-term aged NPPs can induce an accident such as loss of coolant accident (LOCA), because the pipes in plants are weak. The NPPs have the emergency core cooling system (ECCS) such as a safety injection system. The ECCS may not function properly in case of the small break size due to a slight change of pressure in the pipe. If the coolant is not supplied by ECCS, the reactor core will melt. Therefore, the meltdown of reactor core have to be prevented by appropriate accident management through the prediction of LOCA break size in advance. This study presents the prediction of LOCA break size using cascaded fuzzy neural network (CFNN). The CFNN model repeatedly applies FNN modules that are serially connected. The CFNN model is a data-based method that requires data for its development and verification. The data were obtained by numerically simulating severe accident scenarios of the optimized power reactor (OPR1000) using MAAP code, because real severe accident data cannot be obtained from actual NPP accidents [1].

2. CFNN method

2.1 CFNN Model

The CFNN model predicts the target value through the process of repeatedly adding FNN modules. The FNN module is a combination of a fuzzy inference system (FIS) and neuronal training. The conditional rule of FIS is applied by a fuzzy *if–then* rule that consists of an antecedent and a consequence. A general drawing of the CFNN model is shown in Fig. 1 [2]. The present study uses the Takagi-Sugeno-type FIS because it does not need defuzzifier in the output terminal, which is a real value [3].

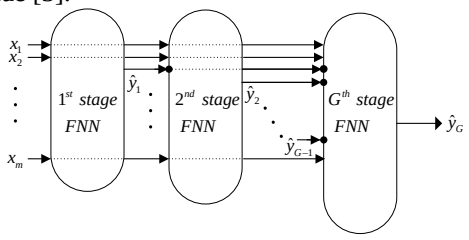


Fig. 1. CFNN model.

$$\begin{aligned}
&\text{Stage 1}\left[\begin{array}{l} If\ x_1(k)\ is\ A_{11}^1(k)\ AND\cdots AND\ x_m(k)\ is\ A_{1m}^1(k), \\ then\ \hat{y}_1^i(k)\ is\ f_1^i(x_1(k),\cdots,x_m(k)) \end{array}\right] \\
&\text{Stage 2}\left[\begin{array}{l} If\ x_1(k)\ is\ A_{11}^2(k)\ AND\cdots AND\ x_m(k)\ is\ A_{1m}^2(k) \\ \quad AND\ \hat{y}_1(k)\ is\ A_{1(m+1)}^2(k), \\ then\ \hat{y}_2^i(k)\ is\ f_2^i(x_1(k),\cdots,x_m(k),\hat{y}_1(k)) \end{array}\right] \\
&\qquad\vdots \\
&\text{Stage } G\left[\begin{array}{l} If\ x_1(k)\ is\ A_{11}^G(k)\ AND\cdots AND\ x_m(k)\ is\ A_{1m}^G(k), \\ \quad AND\ \hat{y}_1(k)\ is\ A_{1(m+1)}^G(k)\ AND\cdots AND\ \hat{y}_{G-1}(k)\ is\ A_{1(m+G-1)}^G(k), \\ then\ \hat{y}_G^i(k)\ is\ f_G^i(x_1(k),\cdots,x_m(k),\hat{y}_1(k),\cdots,\hat{y}_{G-1}(k)) \end{array}\right] \\
&\frac{\text{Fact}: x_1(k)\ is\ A_{11}^1(k)\ AND\cdots AND\ x_m(k)\ is\ A_{1m}^1(k)}{Consequent: \hat{y}_G^i(k)\ is\ f_G^i(x_1(k),\cdots,x_m(k),\hat{y}_1(k),\cdots,\hat{y}_{G-1}(k))}
\end{aligned} \tag{1}$$

where $x_j(k)$ is FIS input value, A_{ij} is fuzzy set for the i^{th} fuzzy rule and the j^{th} input variable, $\hat{y}^i(k)$ is the i^{th} fuzzy rule output, m is the number of input variables, n is the number of fuzzy rules, and $f^i(x_1(k),\cdots,x_m(k)) = \sum_{j=1}^{m} q_{ij}x_j(k) + r_i$.

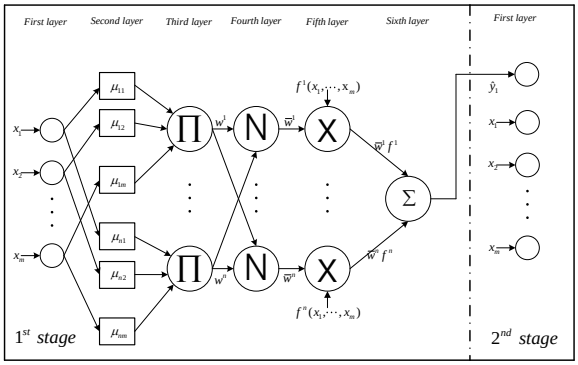


Fig. 2. First stage of the FNN module

The first stage of the FNN module is shown in Fig. 2. The first layer indicates the input nodes that transmit the input values to the next layer. Each output from the first layer is transmitted to the input of the membership function. The second layer indicates the fuzzification layer that calculates the membership function using the Gaussian function using Eq. (2). The third layer indicates a product operator on the membership functions that is expressed as Eq. (3). The fourth layer indicates normalization using Eq. (4). The fifth layer generates the output of each fuzzy *if-then* rule. Finally, the sixth layer indicates an aggregation of all the fuzzy *if-then* rules and is expressed as Eq. (5). The second-stage FNN module uses the initial input variables and the output of the first-stage FNN module as the input

variables. This process is repeated G-times to find the optimum output value.

$$\mu_{ij}(x_j(k)) = e^{-\frac{(x_j(k)-c_{ij})^2}{2\sigma_{ij}^2}} \quad (2)$$

$$w^i(k) = \prod_{j=1}^{m} \mu_{ij}(x_j(k)) \quad (3)$$

$$\bar{w}^i(k) = \frac{w^i(x(k))}{\sum_{i=1}^{n} w^i(x(k))} \quad (4)$$

$$\hat{y}(k) = \sum_{i=1}^{n} \bar{w}^i(k) y^i(k) = \sum_{i=1}^{n} \bar{w}^i(k) f^i(\mathbf{x}(k)) = \mathbf{w}^T(k)\mathbf{q} \quad (5)$$

where

c_{ij} : center position of the peak

σ_{ij} : width of the bell shape

$$\mathbf{w}(k) = [\bar{w}^i(k)x_1(k)\cdots\bar{w}^n(k)x_1(k)\cdots\cdots \bar{w}^1(k)x_m(k)\cdots\bar{w}^n(k)x_m(k)\ \bar{w}^1(k)\cdots\bar{w}^n(k)]^T$$

$$\mathbf{q} = [q_{11}\cdots q_{n1}\cdots\cdots q_{1m}\cdots q_{nm} r_1\cdots r_n]^T$$

2.2 Data Optimization

To predict the LOCA break size using the CFNN model, the CFNN model needs the numerical simulation data for severe accidents. The data were obtained by simulating the MAAP code for the LOCA scenarios of OPR1000. The simulation comprised 600 cases of severe accident scenarios. The data consisted of 200 hot-leg LOCAs, 200 cold-leg LOCAs, and 200 steam generator tube ruptures (SGTRs). The break positions were divided into hot-leg, cold-leg and SGT. Table I shows the input variables for prediction of the LOCA break size.

Table I: Input variables

	Input variables
Hot-leg	pressurizer pressure, pressurizer water level, pressure in the pipe, temperature of gas in the pipe
Cold-leg	
SGTR	water level in the broken side S/G, temperature in the unbroken side S/G, pressurizer pressure, pressurizer water level

The CFNN model is optimized by a combined method using the specified training data. The antecedent parameters in the membership function are optimized by a genetic algorithm. The consequent parameters are optimized by the least square method. In the genetic algorithm, the following fitness function is proposed to minimize the maximum and root-mean-square (RMS) errors:

$$F = \exp(-\lambda_1 E_1 - \lambda_2 E_2) \quad (6)$$

where

$$E_1 = \sqrt{\frac{1}{N_t}\sum_{k=1}^{N_t}\left(y(k)-\hat{y}(k)\right)^2}$$

$$E_2 = \max_k \left(y(k)-\hat{y}(k)\right)^2,\ k = 1, 2, \cdots, N_t$$

λ_1 : weighting value of the RMS error

λ_2 : weighting value of the maximum error

N_t : number of training data

$y(k)$: actual output value

$\hat{y}(k)$: predicted value by FNN

The consequent parameter $\mathbf{q}$ in Eq. (5) is optimized by the least square method and is computed to minimize the objective function represented by the squared error between the measured value *y(k)* and the predicted value $\hat{y}(k)$.

$$J = \sum_{k=1}^{N_t}\left(y(k)-\hat{y}(k)\right)^2 = \sum_{k=1}^{N_t}\left(y(k)-\mathbf{w}^T(k)\mathbf{q}\right)^2 = \frac{1}{2}(\mathbf{y}_t - \hat{\mathbf{y}}_t)^2 \quad (7)$$

where

$$\mathbf{y}_t = [y(1)\ \ y(2)\cdots y(N_t)]^T$$

$$\hat{\mathbf{y}}_t = [\hat{y}(1)\ \hat{y}(2)\cdots\hat{y}(N_t)]^T$$

The solution to minimize the objective function in Eq. (7) is expressed as follows:

$$\mathbf{y}_t = \mathbf{W}_t\mathbf{q} \quad (8)$$

where

$$\mathbf{W}_t = [\mathbf{w}(1)\ \ \mathbf{w}(2)\cdots\mathbf{w}(N_t)]^T$$

The parameter vector $\mathbf{q}$ in Eq. (8) is solved from the pseudo-inverse function as follows:

$$\mathbf{q} = (\mathbf{W}_t^T\mathbf{W}_t)^{-1}\mathbf{W}_t^T\mathbf{y}_t \quad (9)$$

The parameter vector $\mathbf{q}$ is computed from a series of input data, output data, and their membership function values because the matrix $\mathbf{W}_t$ is composed of the input data and membership function values and $\mathbf{y}_t$ is the output data.

The CFNN model is sequentially trained at each FNN module. However, the CFNN model may suffer from an overfitting problem. When overfitting occurs, the

process of adding FNN modules will stop. The overfitting problem can be resolved through cross checking using the verification data. A criterion used to evaluate whether or not an overfitting problem occurs at stage g is the sum of fractional errors for the checking data, which is expressed as follows:

$$E_f(g) = \frac{\sum_{k=N_t+1}^{N_c} \left(y_g(k) - \hat{y}_g(k)\right)^2}{\sum_{k=N_t+1}^{N_c} \left(y_g(k)\right)^2} \quad (10)$$

The fractional errors in Eq. (10) are different from those we generally know. The training and checking processes stop if $E_f(g+1) > E_f(g)$, which means that the fractional error of the checking data increases according to the increase in the number of stages. When the condition $(E_f(g+1) > E_f(g))$ is satisfied, the CFNN model may begin to overfit if the process of adding an FNN module is continued. If the condition is not satisfied, the algorithm moves to the next stage, and an FNN module will be added.

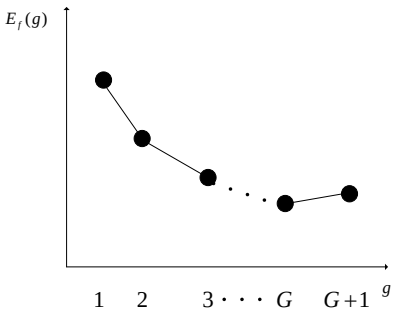


Fig. 3. The fraction error $E_f(g)$ according to stage number.

According as the number of FNN modules increases, the structure of the CFNN model will be complex. The complexity of the CFNN model is expected to be proportional to the element number of parameter vector $\mathbf{q}$ in Eq. (8). Thus, in this study, the complexity is defined as the element number of parameter vector $\mathbf{q}$ of all FNN modules contained in the CFNN and is calculated as follows:

$$complexity = G \times (G + 2m) \times n / 2$$

where G is the number of the FNN modules, m is the number of the input variables, and n the is number of fuzzy rules.

The complexity of the CFNN model radically increases as the number of FNN modules increases, and it linearly increases according to the number of input variables and number of fuzzy rules.

3. Result of simulation

Table II lists the performance results obtained by the CFNN model for the hot-leg LOCA, cold-leg LOCA, and SGTR, respectively. As listed in Table II-(a), for the hot-leg LOCA, the average RMS error were 0.4% and 0.6% for the development data and test data, respectively. As listed in Table II-(b), for the cold-leg LOCA, the average RMS error were 0.3% and 1.2% for the development data and test data, respectively. In addition, in Table II-(c) for the SGTR, the average RMS error were 3.2% and 3.8% for the development data and test data, respectively.

Fig. 4 shows the RMS errors of the development and test data, respectively. The RMS error gradually decrease as the number of stages in the CFNN model is increased by the repetitive process. Even if the RMS error differs according to the LOCA break positions and is relatively large for SGTR, the RMS error level is below about 4%. The CFNN model is confirmed to accurately predict the LOCA break size.

Table III lists the RMS errors of development data for the CFNN model and the support vector regression (SVR) model. The basic concept of the SVR is to nonlinearly map the original data into a higher dimensional space.

$$y = f(\mathrm{x}, \mathrm{w}) = \sum_{i=1}^{N} w_i \phi_i(\mathrm{x}) + b = \mathrm{w}^T \varphi(x) + b$$

The input variables of SVR and CFNN are the same. The CFNN model is much better than the SVR model.

Table II: Performance of the CFNN model

(a) Hot-leg

Number of fuzzy rules	Development data		Test data	
	RMS error(%)	Max error(%)	RMS error(%)	Max error(%)
2	0.27	0.91	0.55	0.93
3	0.65	1.96	0.72	1.43

(b) Cold-leg

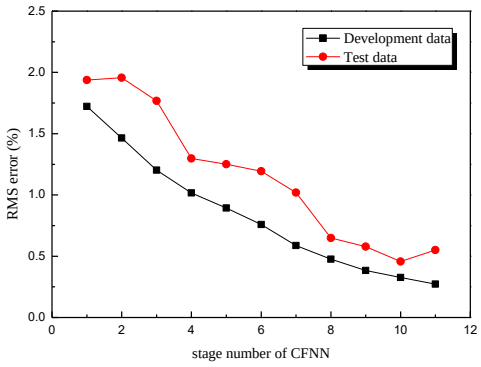
Number of fuzzy rules	Development data		Test data	
	RMS error(%)	Max error(%)	RMS error(%)	Max error(%)
2	0.40	1.14	0.66	1.74
3	0.15	0.40	1.84	5.77

(c) SGTR

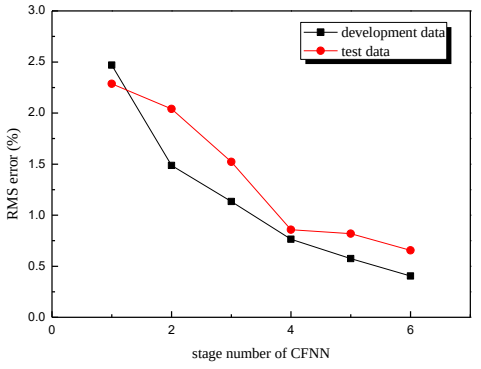
Number of fuzzy rules	Development data		Test data	
	RMS error(%)	Max error(%)	RMS error(%)	Max error(%)
2	3.86	15.29	4.04	7.40
3	2.60	10.45	3.62	8.43

Table III: Comparison of the CFNN and SVR models for development data

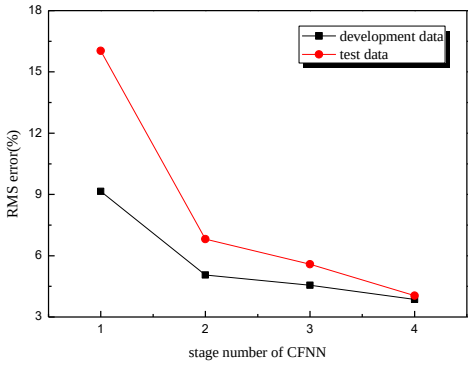
LOCA position	CFNN	SVR
Hot-leg	0.27	3.16
Cold-leg	0.40	2.52
SGTR	3.86	5.27



(a) Hot-leg LOCA



(b) Cold-leg LOCA



(c) SGTR

Fig. 4. Stage number of CFNN versus RMS error

4. Conclusions

The CFNN model has been designed to rapidly predict the LOCA break size in LOCA situations. The CFNN model was trained by using the training data set and checked by using test data set. These data sets were obtained using MAAP code for OPR1000 reactor. The performance results of the CFNN model show that the RMS error decreases as the stage number of the CFNN model increases. In addition, the performance result of the CFNN model presents that the RMS error level is below 4%. Therefore, it is confirmed that the CFNN model can accurately predict the LOCA break size. If the operators can predict the break size in the LOCA, they can response quickly and properly to LOCA circumstances to prevent the meltdown of reactor core.

REFERENCES

[1] R. E. Henry, et al., MAAP4 – Modular Accident Analysis Program for the LWR Power Plants, User's Manual, Fauske and Associates, Inc., Vol. 1, 2, 3, and 4, 1990.
[2] D. Y. Kim, K. H. Yoo, and M. G. Na, Estimation of minimum DNBR using Cascaded Fuzzy Neural Networks, IEEE Trans. Nucl. Sci. 62, pp.1849-1856, July 15, 2015.
[3] T. Takagi and M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Trans. Systems, Man, Cybern., vol. SMC-15, no. 1, pp. 116-132, Jan./Feb. 1985.