Transactions of the Korean Nuclear Society Spring Meeting
Jeju, Korea, May 22-24, 2019

Development of Reactor Accident Classification Model using Long Short-Term Memory

Seongmin Son [a], ChoHwan Oh[a], Jeong Ik Lee [a]*
[a]*Department of Nuclear and Quantum Engineering, Korea Advanced Institute of Science and Technology (KAIST)*
373-1 Guseong-dong, Yuseong-gu, Daejeon, 305-701, Korea
[*]*Corresponding author: jeongiklee@kaist.ac.kr*

1. Introduction

Although the nuclear reactors are machines with numerous safety systems, they still require heavy responsibility from the operator. When a reactor state changes, it is up to the operator to determine the cause of the change. If there are misjudgments by an operator, serious problems can occur even involving severe reactor damage. Because of this, machine support is a topic that has attracted much attention in this field. Of course, the machine should not infringe the human right to make decisions, but it can apply a supplementary pre-diagnostic system to help operators to make a decision. Thus, through machine learning, the technology to determine the current situation from the sensed information can be a good information provider to the operator working under high stress.

In this research, an Artificial Neural Network (ANN) model is proposed to classifying nuclear reactor accident. ANN is one of the most popular algorithms in the field of machine learning. Also, its derivative type, such as Probability Neural Network (PNN), has already been applied in similar studies [1-2]. In this study, a recurrent type of neural network called Long Short-Term Memory (LSTM) is applied. [3] The accident data is generated through MARS-KS v 1.4 and aims to identify the Small Break (SB), Middle Break (MB) and Large Break (LB) loss of coolant accidents at the Cold Leg / Hot Leg based on the APR 1400 reactor.

2. Methodology

2.1. Accident Data Generation

For machine learning, a large amount of data is required first. Of course, it would be best if the data could be obtained through experimentation, but it is practically impossible to accumulate LOCA data through experiments. Therefore, data simulating accident situations are used which was generated from MARS-KS v 1.4, a verified system code. APR-1400 is a Korean designed 1400MWe class large pressurized water reactor (PWR). The nodalization of 1D, 2 channel model of APR-1400 is shown in Figure 1. For each leg, 3000 accident data sets were generated, which are 1,000 data sets per accident class (e.g. SB, MB, and LB).

.

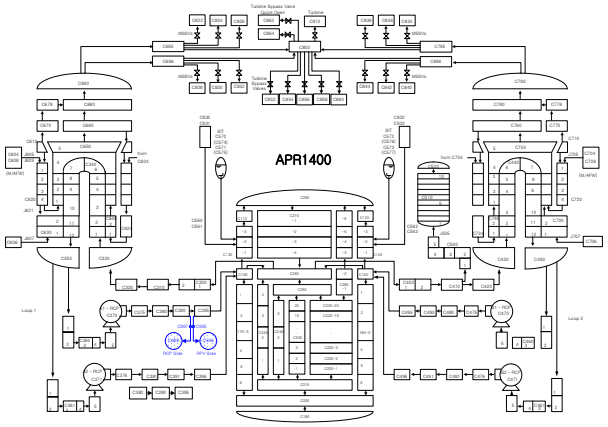

Figure 1. MARS-KS 1.4 nodalization model of APR-1400 (2 channel model)

2.2. Training LSTM model

Time series datasets, such as reactor accident state sensing, are not suitable for applying a general feed forward network. This is because for this type of dataset, the future depends not only on the present but also on the past. Therefore, a neural network with recursive is needed. However, a simple recurrent neural network (RNN) has a problem of gradient vanishing [4] that does not properly reflect the weight of old data. LSTM is developed for solving the problems of RNN structure. [3] The LSTM is one of the theoretically deepest neural network and has shown high performance in time series processing. The structure of LSTM is in Figure 2. The results from the previous time step affects the next time.

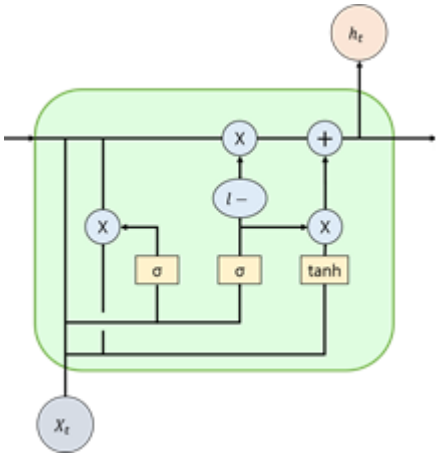

Figure 2. LSTM structure

The learning is carried out using PyTorch v 0.4.0. [5] in Python 3.6 environment [6]. Two LSTM layers consist of 64 Perceptron and they are used as a hidden layer for the model. Optimizer is selected as the Adam optimizer [7] which is a type of stochastic gradient descent method optimizer. Loss function is MSE loss and maximum epoch is 110. The number of training set is 17,808. 70% of data are used for training, and 30% data are used for validation. Batch size is 256.

3. Result and Conclusion

The loss and accuracy during the training process are shown in Figure 3. In the last epoch, Train accuracy is 95% and Validation accuracy is 95.5%.

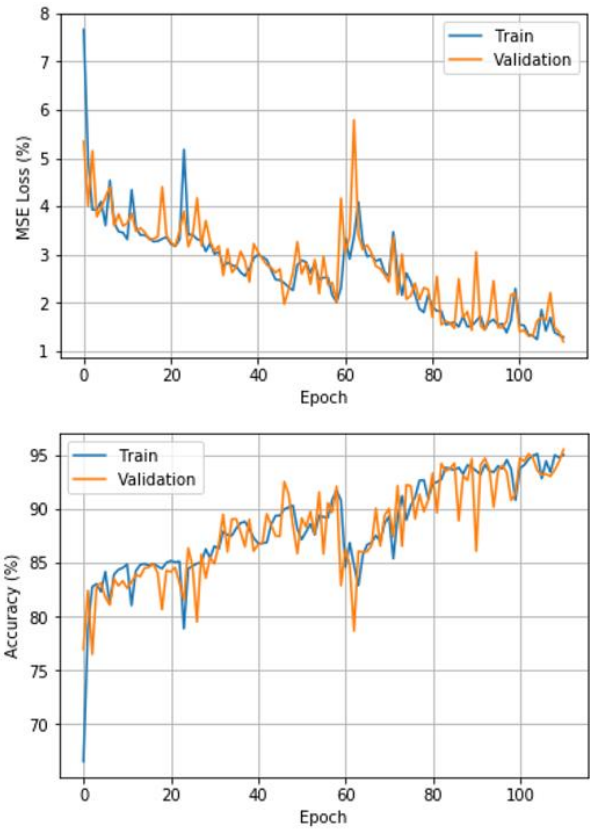

Figure 3. Training result of LSTM model

The performance of the LSTM model is evaluated using the new 360 data not used for training and validation. The results are shown in Table 1. Wrong judged five cases are misjudged by Cold Leg Small Break except for one case that judged Hot Leg Large Break by Cold Leg Middle Break.

From this result, the accuracy of test set is 98.6%, which is similar as the result of train and validation set. This is not a perfect result, but it shows that it can play a good pre-diagnosis role.

Currently, only six types of accidents have been analyzed, but it is expected that similar LSTM based methods will be possible to diagnose more accidents or complex accidents by expanding and applying them in the future.

Table 1. Performance of LSTM model

Accident Kind	Wrong Judgement / Total
Cold Leg Small Break	0 / 80
Hot Leg Small Break	3 / 40
Cold Leg Middle Break	1 / 80
Hot Leg Middle Break	0 / 40
Cold Leg Large Break	0 / 80
Hot Leg Large Break	1 / 40

ACKNOWLEDGEMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2016R1A5A1013919).

REFERENCE

[1] Bartal, Yair, Jie Lin, and Robert E. Uhrig. "Nuclear power plant transient diagnostics using artificial neural networks that allow “don’t-know” classifications." *Nuclear Technology* 110.3 (1995): 436-449.

[2] Na, Man Gyun, et al. "Prediction of major transient scenarios for severe accidents of nuclear power plants." *IEEE transactions on nuclear science* 51.2 (2004): 313-321.

[3] Gers, Felix A., Jürgen Schmidhuber, and Fred Cummins. "Learning to forget: Continual prediction with LSTM." (1999): 850-855.

[4] Hochreiter, Sepp. "The vanishing gradient problem during learning recurrent neural nets and problem solutions." *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6.02 (1998): 107-116.

[5] Paszke, Adam, et al. "Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration." *PyTorch: Tensors and dynamic neural networks in Python with strong GPU acceleration* (2017).

[6] Rossum, Guido. "Python reference manual." (1995).

[7] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).