energies

MDPI

Article

Development and Validation of a Nuclear Power Plant Fault Diagnosis System Based on Deep Learning

Bing Liu, Jichong Lei, Jinsen Xie * and Jianliang Zhou *

School of Nuclear Science and Technology, University of South China, Hengyang 421000, China
* Correspondence: jinsen_xie@usc.edu.cn (J.X.); zjl@usc.edu.cn (J.Z.)

Abstract: As artificial intelligence technology has progressed, numerous businesses have used intelligent diagnostic technology. This study developed a deep LSTM neural network for a nuclear power plant to defect diagnostics. PCTRAN is used to accomplish data extraction for distinct faults and varied fault degrees of the PCTRAN code, and some essential nuclear parameters are chosen as feature quantities. The training, validation, and test sets are collected using random sampling at a ratio of 7:1:2, and the proper hyperparameters are selected to construct the deep LSTM neural network. The test findings indicate that the fault identification rate of the nuclear power plant fault diagnostic model based on a deep LSTM neural network is more than 99 percent, first validating the applicability of a deep LSTM neural network for a nuclear power plant fault-diagnosis model.

Keywords: nuclear power plant; PCTRAN; deep learning; fault diagnosis; deep LSTM

check for updates

Citation: Liu, B.; Lei, J.; Xie, J.; Zhou, J. Development and Validation of a Nuclear Power Plant Fault Diagnosis System Based on Deep Learning. *Energies* **2022**, *15*, 8629. https://doi.org/10.3390/en15228629

Academic Editor: Ali Ahmadian

Received: 31 October 2022
Accepted: 13 November 2022
Published: 17 November 2022

1. Introduction

The nuclear power plant is a complex and extensive system comprised of many subsystems, which in turn contain many different devices; to obtain a comprehensive picture of the operating status of the equipment in each system, a large number of sensors are distributed throughout the system equipment to measure parameters such as temperature, pressure, and water level. Therefore, it is complicated for operators to obtain information directly from the large amount of measurement data generated by the monitoring system at any given time. This situation is seen primarily when an abnormal condition occurs in the plant, and an alarm signal is generated; even a well-trained operator may make a mistaken judgment under tremendous mental pressure and in the presence of numerous signs. In the early phases of the Three Mile Island catastrophe, the operator failed to discern the condition of the pressure release valve from a significant volume of data, resulting in a misdirection of the plant's status and a severe accident [1]. Suppose a diagnostic algorithm could be utilized to give information on the operational state of the system's equipment. In that case, it would significantly minimize the mental stress of the operators and the likelihood of error, which is crucial for ensuring the system's safe operation.

Research on fault diagnostic technology began in the United States, and the tragedy caused by equipment failure during the Apollo program was precipitated in 1967 when the U.S. Office of Naval Research formed a mechanical failure prevention division. In the late 1960s, the establishment of the British Machine Health and Condition Monitoring Association furthered the development of fault diagnosis technology. Subsequently, European countries conducted relevant research on condition monitoring and fault diagnosis technology and developed their distinctive diagnosis technology system. Japan's fault diagnosis technology began in the mid-1970s, and by learning from the world's research and with continuous development, it has become one of the most advanced in the world. Since the early 1980s, China's fault diagnostic technology has developed into a generally flawless theoretical framework [2–5]. In the nuclear field, Tsinghua University has researched and developed a fault diagnosis system for a 200 MW nuclear heating station [6].

Harbin Engineering University has designed and developed a nuclear power plant operation support system, which includes functions such as condition monitoring, alarm analysis, fault diagnosis, and emergency operation guidance [7]. The Korean Academy of Science and Technology (KAIST) has developed a fault diagnosis advisory system (ADAS) for nuclear power plant fault diagnosis [8]. In recent years, under the influence of the "fourth industrial revolution"—the artificial intelligence wave, the development of artificial intelligence-based mechanical fault diagnosis has been very rapid [9–14].

In summary, this paper proposes to apply deep LSTM neural networks to nuclear power plant fault diagnosis, using the self-developed autoPCTRAN code to achieve automatic data extraction for different faults of the PCTRAN code as well as different fault levels, selecting some important nuclear parameters (nuclear power, regulator pressure, regulator water level, coolant flow rate, average coolant temperature, and steam generator water level) as feature quantities. The training set, validation set, and test set are obtained using random sampling at a ratio of 7:1:2, and a deep LSTM neural network is constructed to train on and learn the accident data training set, using the validation set to correct the model to avoid model overfitting. The test set is used to test the model.

2. Introduction to PCTRAN

PCTRAN is a tiny software code created by Microsimulation Technology (MST) in the United States that may be used for nuclear power plant simulation and severe accident analysis [15]. PCTRAN is a PC-based simulation software code designed specifically for nuclear power plant operation and disaster response training. Severe accidents, such as core meltdown, containment failure, and radioactive leakage, are also within its purview. PCTRAN has been the most effective training simulation code implemented globally in nuclear power plants and research institutions since 1985. The International Atomic Energy Agency (IAEA) has chosen PCTRAN as the training software for its biennial Advanced Reactor Simulation Symposium [16] since 1996. The NPP models currently included in PCTRAN include ACP100, ABWR, BWR5 MARK II, AP1000, AREVA EPR, TRIGA, RadPuff, ESBWR, VVER 1200, Korean APR1400, Korean KSNP, HTGR, SFP, MHI APWR NuScale, PWR 3-loop, BWR4, and SMART [17]. Their operating interfaces, shown in Figure 1, are simple and can be used with direct control through the operator interface and provide instantaneous feedback on different operating condition values such as temperature, pressure, flow, and dose. Figure 2 depicts the overall flow diagram of the application [18].

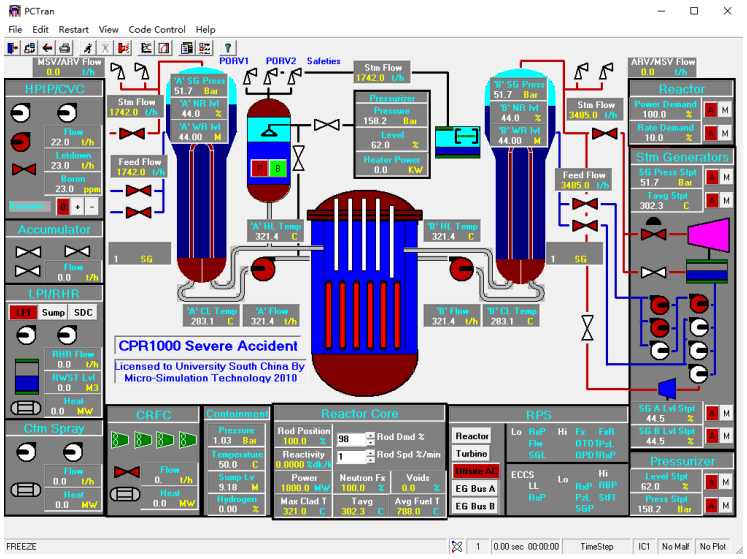


Figure 1. PCTRAN main control interface.

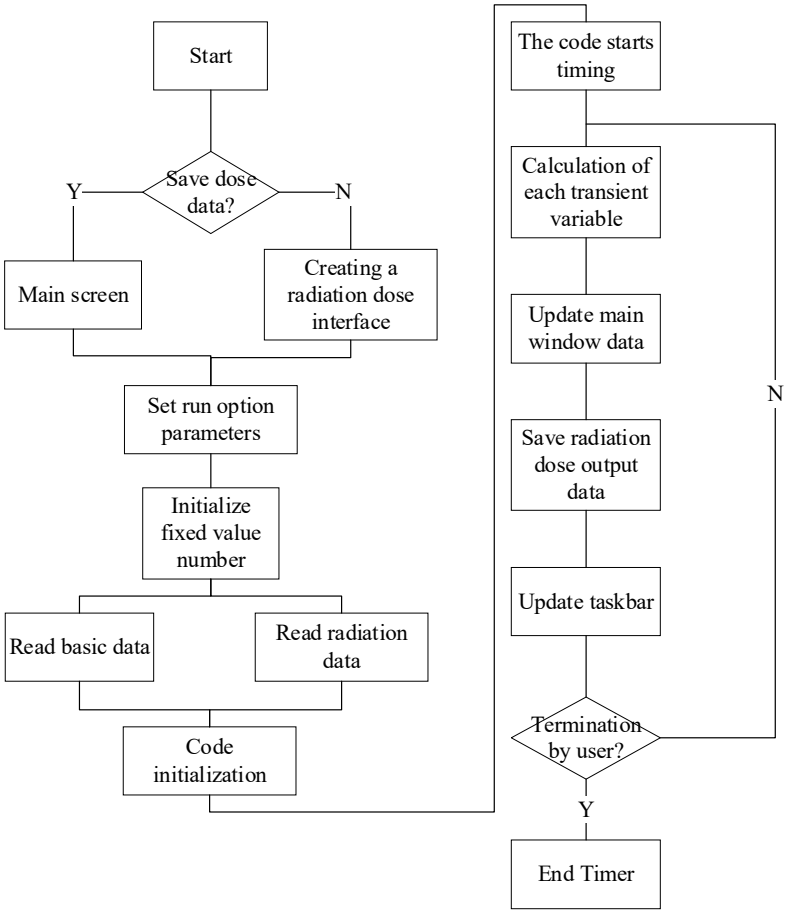


Figure 2. PCTRAN general flow block diagram.

In this research, a CPR1000 type (i.e., PWR 3-loop type) nuclear power plant is utilized to simulate the first operating circumstances at various operational points of the NPP. More than ten starting conditions are employed to model the beginning operating conditions. In addition, PCTRAN covers 20 distinct kinds of NPP operational failures, such as feed-water loss, primary pump failure, ATWT, coolant loss, and steam generator pipe rupture, representing the majority of NPP, as well as inevitable severe design benchmark failures. Meanwhile, PCTRAN follows the "no intervention for 30 min" policy for nuclear power plants in accident mode [19] to prevent failures caused by personnel error.

3. Deep Learning

3.1. LSTM Neural Network

A Recurrent Neural Network (RNN) [20] is a class of neural networks dedicated to processing temporal data samples, in which each layer not only outputs to the next layer but also outputs a hidden state for the current layer to use when processing the next sample. Just as convolutional neural networks can easily scale to images with large widths and heights, and some convolutional neural networks can also handle images of different sizes, recurrent neural networks can scale to longer sequential data, and most of them can handle data with different sequence lengths. It can be regarded as a fully connected neural network with self-loop feedback. Its network structure is shown in Figure 3, where W is the self-looping parameter matrix from the hidden layer to the hidden layer, U is the parameter matrix from the input layer to the hidden layer, and V is the parameter matrix from the hidden layer to the output layer. However, the general recurrent neural network suffers from a long-term dependence problem, which leads to gradient disappearance and gradient explosion in RNN. To solve this problem, Sepp Hochreiter proposed a long and short-term memory network in 1997 [21]. The LSTM neural network cell unit consists of

a forgetting gate (ft), an input gate (it), and an output gate (ot). The input gate is used to update the structural state value of the cell to be added to the cell. The forgetting gate is used to determine the proportion of cell values retained from the previous moment, and the output gate generates a hidden layer state value (ht) as an additional input for the next moment. According to the moment t signal, they generate the structural state value (ct) of this cell and the hidden layer state ht at moment t, and an additional input at time t + 1. Thus the update of the open and closed cell values of each link can be controlled internally and spontaneously based on the data in the network training, giving the network a variable-length "memory". The cell structure of the LSTM model is shown in Figure 4, and its calculation formula is shown in Equations (1)–(5) [22].

$$i_t = \sigma(\sum W_{xi}x_t + \sum W_{xi}x_{t-1} + \sum W_{xi}x_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(\sum W_{xf}x_t + \sum W_{xf}x_{t-1} + \sum W_{xf}x_{t-1} + b_f) \quad (2)$$

$$o_t = \sigma(\sum W_{xo}x_t + \sum W_{xo}x_{t-1} + \sum W_{xo}x_{t-1} + b_o) \quad (3)$$

$$\widetilde{c}_t = f_t c_{t-1} + i_t \tanh(\sum W_{xc}x_t + \sum W_{hc}x_{t-1} + b_c) \quad (4)$$

$$h_t = o_t \tanh(\widetilde{c}_t) \quad (5)$$

x_t is the input vector at time t; W is the weight matrix; b is the weight partiality term; σ is the activation function; $\widetilde{c}_t$, c_{t-1} are the cell structure state values at times t and $t-1$, respectively; *tanh* is the hyperbolic tangent activation function; tanh is the input gate; f_t is the forgetting gate; o_t is the output gate, and h_t is the output value of the cell at time t.

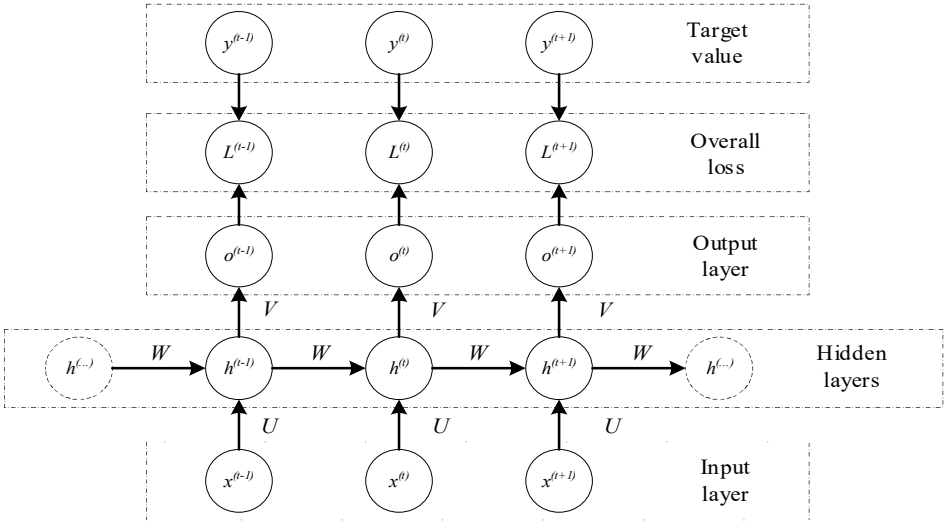


Figure 3. RNN network structure diagram.

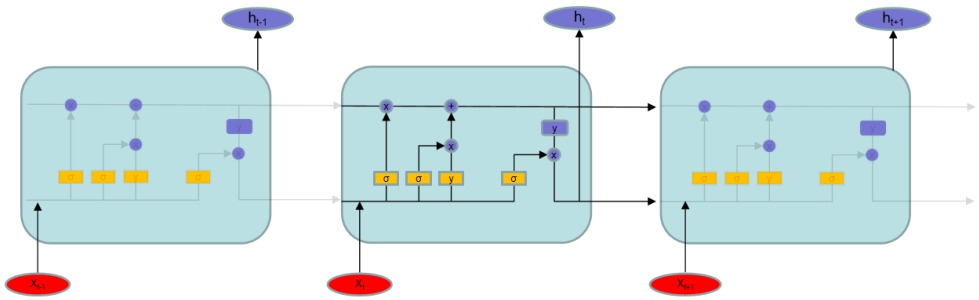


Figure 4. LSTM cell structure diagram.

3.2. Deep Neural Network

Deep Neural Networks (DNNs) are the foundation of deep learning, and to understand DNNs, we first need to understand the DNN model. The neural network is based on the extension of the perceptron, and a DNN can be understood as a neural network with many hidden layers. The terms "multi-layer neural network" and "deep neural network" (DNN) refer to the same thing; DNN is sometimes called Multi-Layer perceptron (MLP). From the DNN's use of the location of different layers, the neural network layers inside the DNN can be divided into three categories: input layer, hidden layer, and output layer, as shown in Figure 5. Generally, the first layer is the input layer, the last layer is the output layer, and the middle layers are all hidden layers. The layers are fully connected, i.e., any neuron in layer I must be connected to any neuron in layer I + 1. Although the DNN looks complex, it is still the same as a perceptron in terms of a small local model, i.e., a linear relationship. The so-called DNN forward propagation algorithm uses several weight coefficient matrices *W*, a bias vector b to perform a series of linear operations, and activation operations with the input value vector x. Starting from the input layer, layer by layer, the backward computation is carried out until the operation reaches the output layer, and the final output result is obtained. Usually, MLPs with more than three hidden layers are called DNNs. Deep LSTM neural networks are a combination of DNNs and LSTMs, and their neurons are not independent of each other but are the same as LSTMs, with interconnections between neurons in each layer and weight transfer between each neuron and the number of hidden layers greater than three.

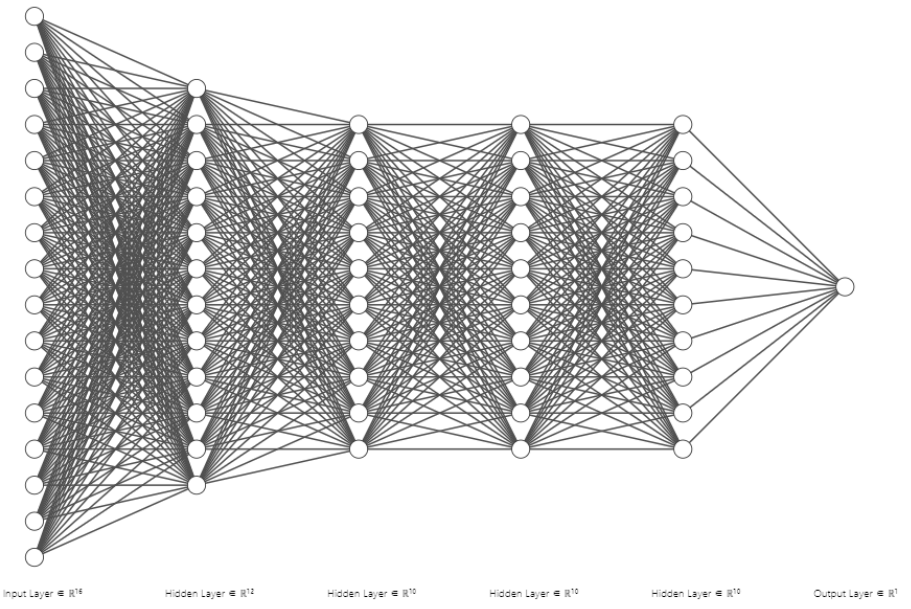


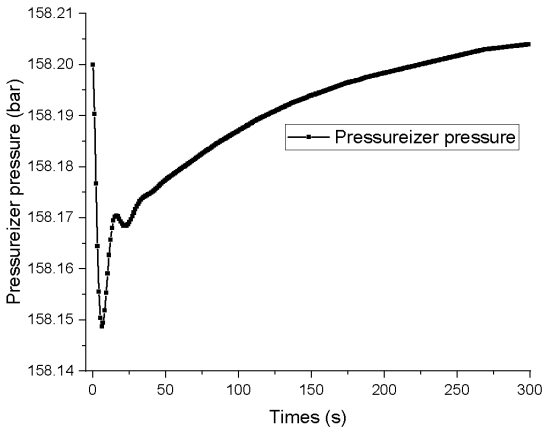
Figure 5. DNN structure diagram.

4. Results

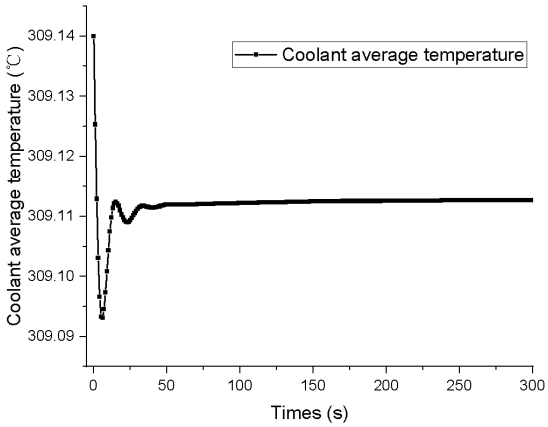
4.1. Data Access

The safety of a nuclear power plant is contingent upon the capacity to rapidly and precisely monitor operating trends in critical operating parameters. Typically, experienced nuclear plant operators monitor the plant's status by tracking data changes over time. In this work, the PCTRAN code simulated four distinct operating modes of a nuclear power station (normal operation, loss of coolant accident, steam generator tube rupture, containment steam pipe rupture). In order to generate appropriate data sets for various failure types, PCTRAN simulations were conducted at varying simulation levels. Based on the data set produced from PCTRAN simulations, six data quantities crucial for the operating states of nuclear power plants were selected as feature quantities (pressurizer pressure, coolant average temperature, coolant flow rate, pressurizer water level, steam

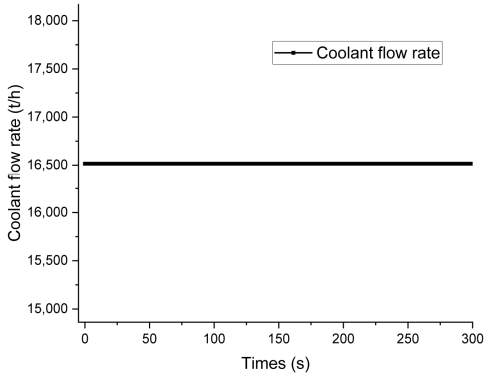
generator water level, nuclear power). As illustrated in Figures 6–9, for each operating condition, a collection of data was picked for each description.



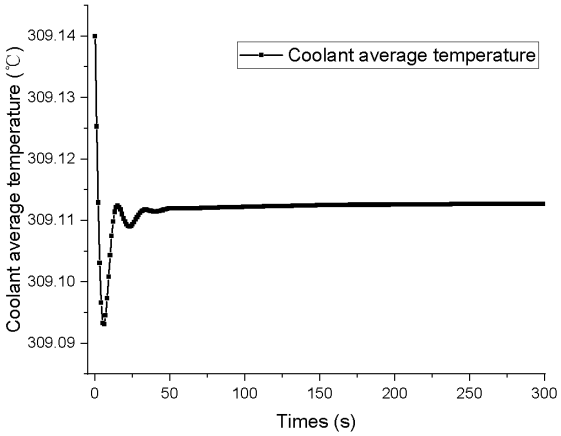
(**a**) pressurizer pressure



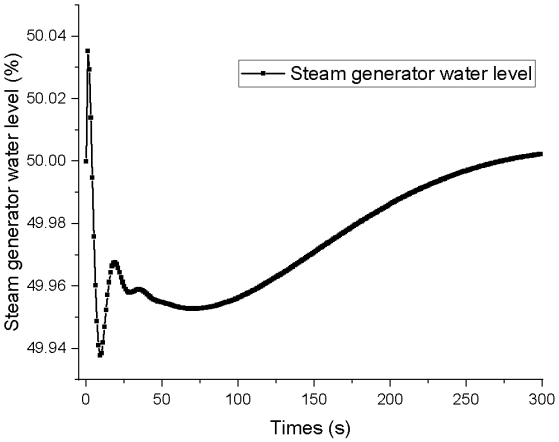
(**b**) coolant average temperature



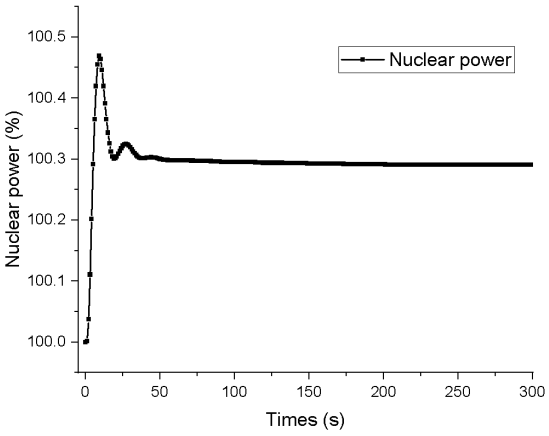
(**c**) coolant flow rate



(**d**) pressurizer water level

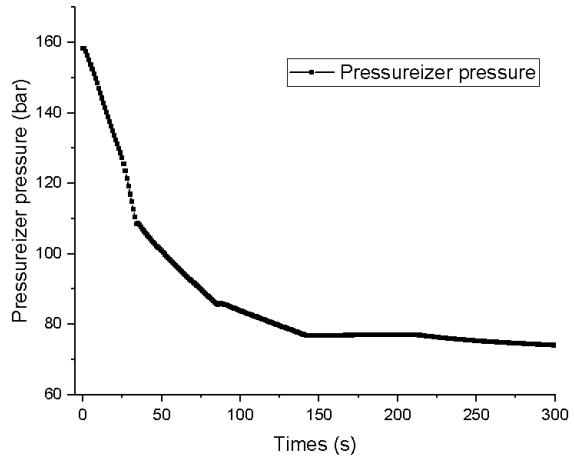


(**e**) steam generator water level

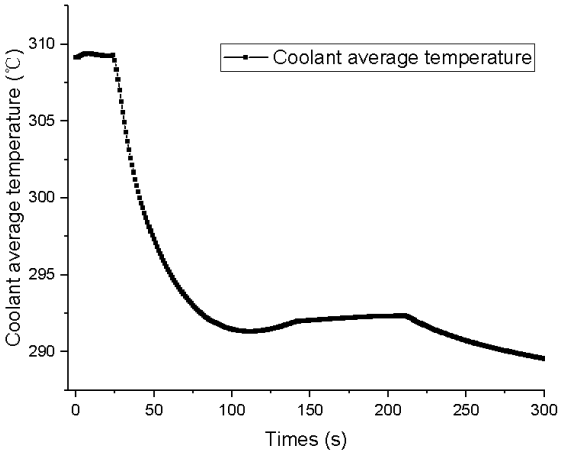


(**f**) nuclear power

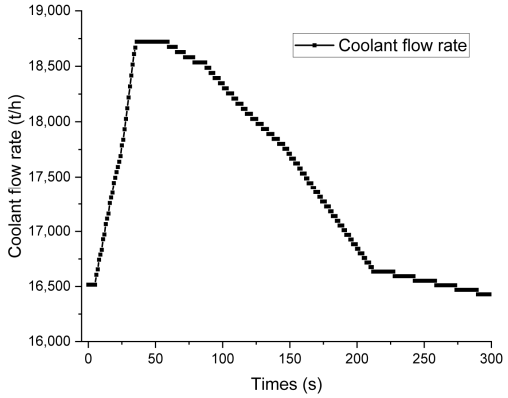
Figure 6. Operating trend of normal operation.



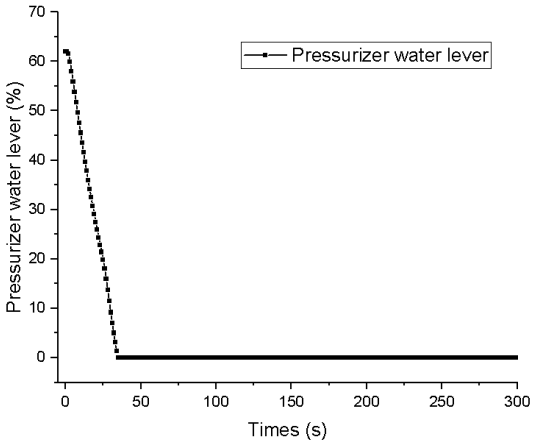
(**a**) pressurizer pressure



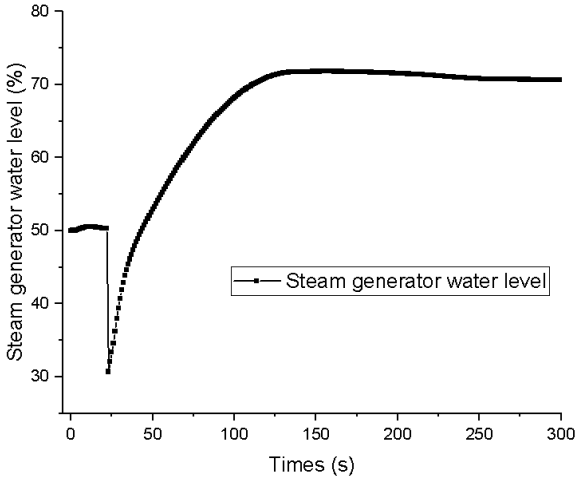
(**b**) coolant average temperature



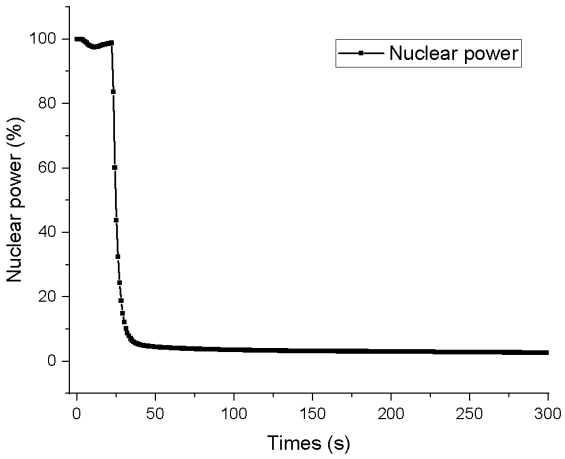
(**c**) coolant flow rate



(**d**) pressurizer water level

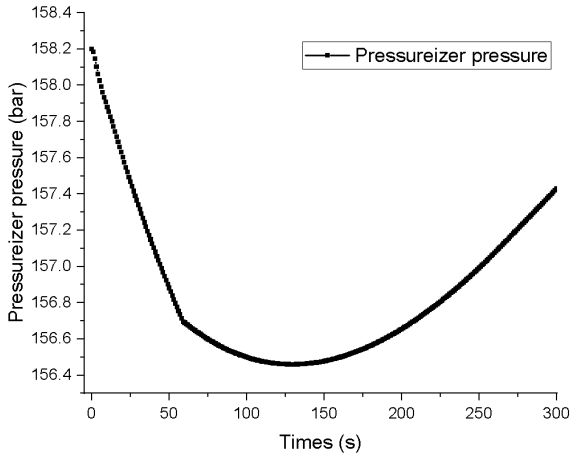


(**e**) steam generator water level

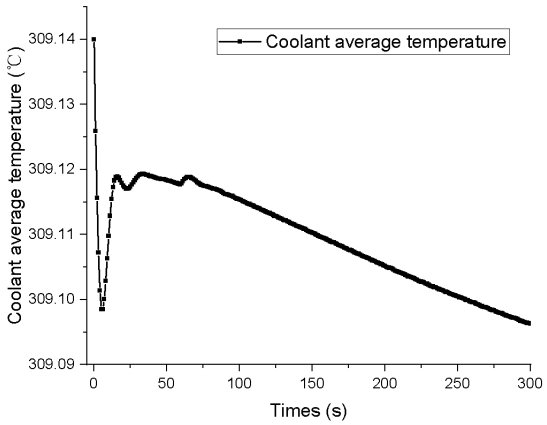


(**f**) nuclear power

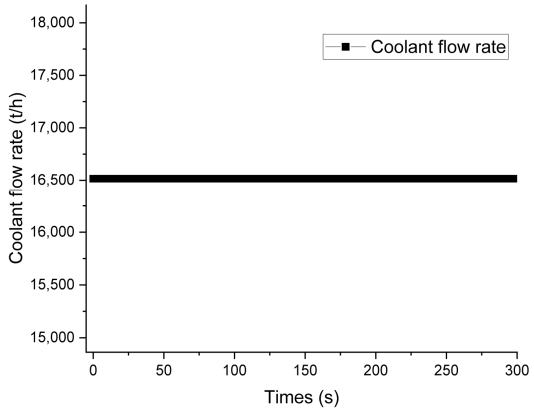
Figure 7. Operating trend of loss of coolant accident.



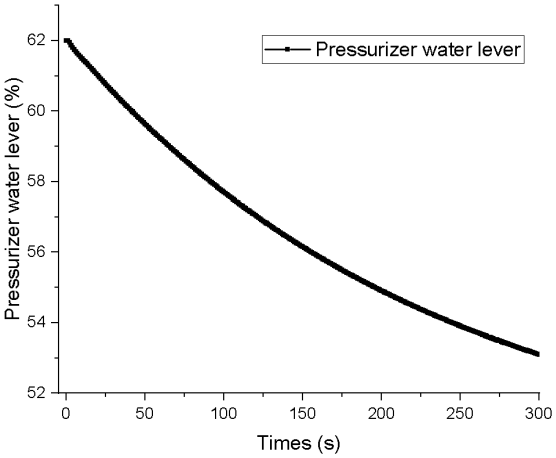
(**a**) pressurizer pressure



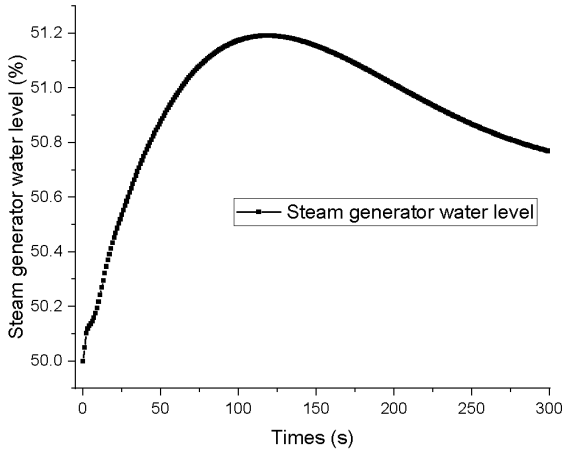
(**b**) coolant average temperature



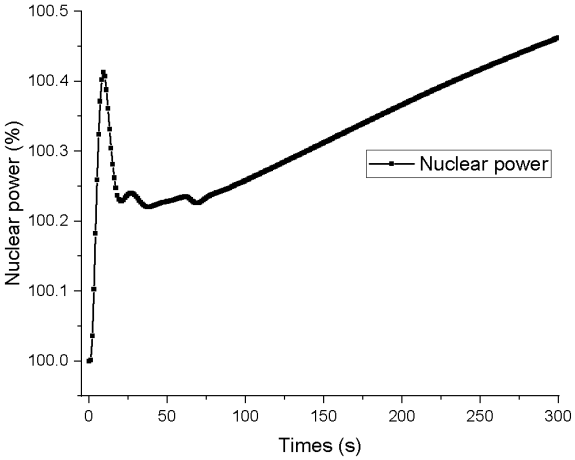
(**c**) coolant flow rate



(**d**) pressurizer water level

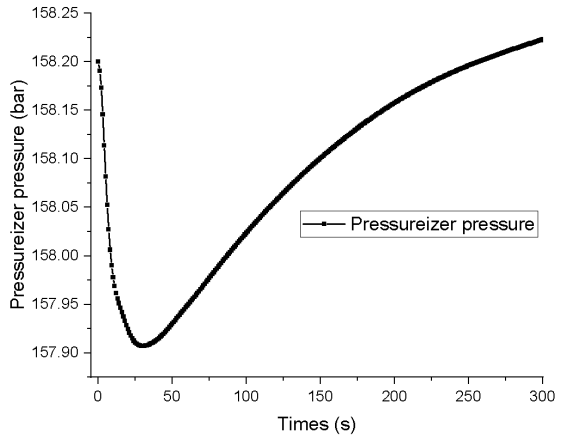


(**e**) steam generator water level

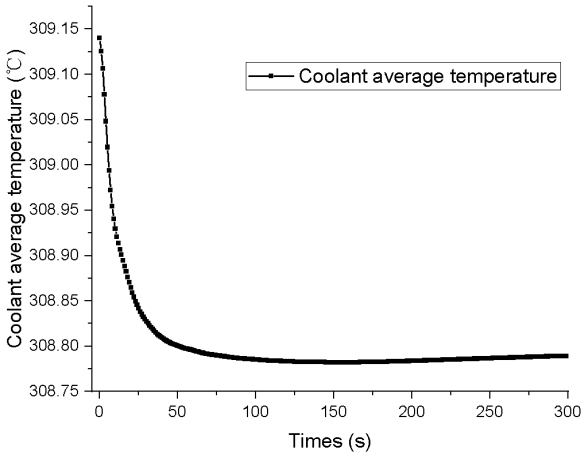


(**f**) nuclear power

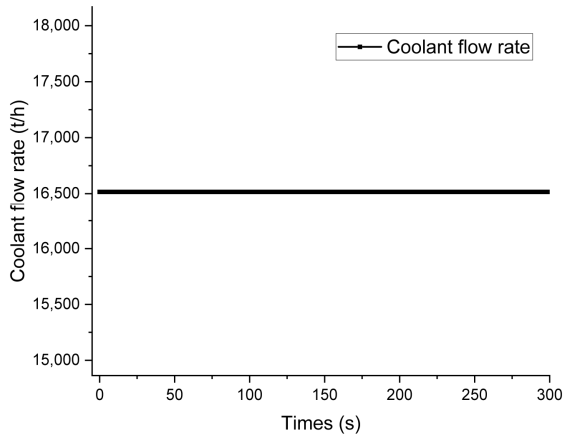
Figure 8. Operating trend of steam generator tube rupture.



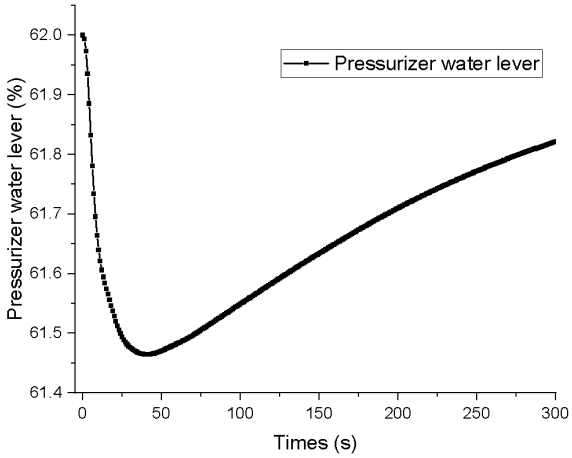
(**a**) pressurizer pressure



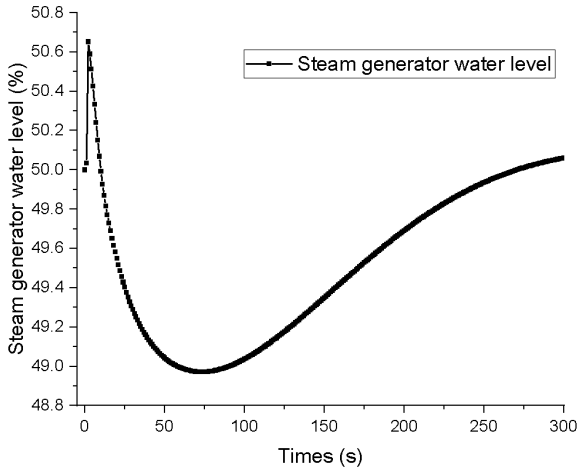
(**b**) coolant average temperature



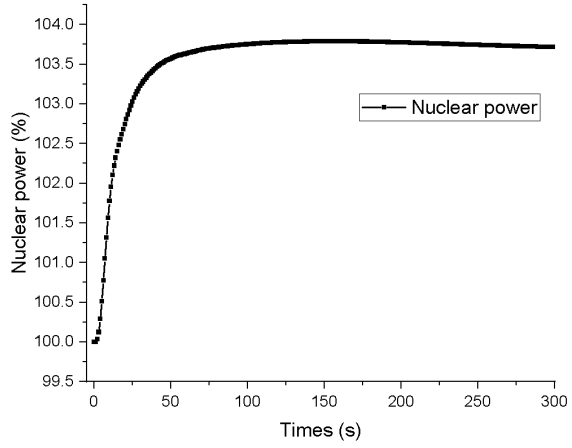
(**c**) coolant flow rate



(**d**) pressurizer water level



(**e**) steam generator water level



(**f**) nuclear power

Figure 9. Operating trend of containment steam pipe rupture.

As seen in Figures 6–9, the pattern of data set changes in various states is entirely distinct, which provides the nuclear plant operator with a foundation for establishing state determinations and theoretical support for problem detection. Due to limited human attention, it is impossible to concentrate on numerous data volumes simultaneously. If just

one or two changes in data volume are considered, multiple outcomes may be obtained. In states 1 and 2, for instance, the coolant flow rate is always constant, but pressurizer pressure first exhibits a downward trend and later an upward trend. If just the pressurizer pressure and coolant flow rate trends are considered, the operational state of the nuclear facility may be overestimated, which might have severe implications.

4.2. Data Pre-Processing

Data sets were simulated using PCTRAN, with each data set including 300 s of data, and 114,000 6-dimensional data sets were created after separating the data. The linear normalization [23] method (i.e., the minimum-maximum normalization method) was used to normalize the characteristic quantities in order to improve the model accuracy. The formula is shown in Equation (6), where x_{min} is the feature minimum, x_{max} is the feature maximum, x is the initial feature value, and x^* is the processed feature value. Using random sampling under normal operation settings, steam generator heat transfer tube rupture, containment steam pipe rupture, loss of coolant feed water, and moderator dilution, the data sets were retrieved and split into a training set, validation set, and test set at a ratio of 7:1:2 [24].

$$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{6}$$

4.3. Model Training

In this paper, accuracy (accuracy) and the cross-entropy loss function (cross-entropy) are used to evaluate the accuracy of the model [25]. Accuracy is the ratio of the number of correct classifications to the total number of classifications. The cross-entropy loss function is used to evaluate the difference between the probability distribution obtained from the current training and the true distribution. With the total number of samples k and the number of correctly classified samples k_1, the formula of accuracy is shown in (7), and the formula of the cross-entropy loss function is shown in (8)(where n denotes the total sample size, c is the number of accident types, $y_{i,t}$ denotes the predicted value, and $\hat{y}_{i,t}$ denotes the true value). It can be concluded from the formula that the closer acc is to 1, the closer Loss is to 1, and the better the prediction.

$$acc = \frac{k_1}{k} \tag{7}$$

$$Loss = -\sum_{i=1}^{n}\sum_{t=1}^{c} y_{i,t} \log \hat{y}_{i,t} \tag{8}$$

The parameters of the LSTM neural network model are a continuous debugging process in which some parameters are defined for the user to change in order to apply the corresponding engineering model. As a result, some hyperparameters need to be adjusted in the process of use. Based on the processed data, the prediction model is formed in the training set by establishing hyperparameters, which are provided in Table 1. The trained model is employed for validation using the validation set and the dropout layer to prevent overfitting of the model. Using Occam's razor [26], if there are two explanations for anything, the most probable proper explanation is the simplest one, i.e., the one with the fewest assumptions. Given specific training data and network design, multiple weight values (i.e., many models) may describe the data. Simple models are less likely to be overfitted than complicated ones. Dropout [27] is a deep learning training procedure in which neural network training units are eliminated from the network according to a given probability. Each mini-batch is training a new network for stochastic gradient descent since it is dropped randomly. The mechanism of action, shown in Figure 10, prevents overfitting of the model by randomly removing some training units of the neural network from the network and constructing a new network using stochastic gradient descent.

Table 1. The Model hyperparameter setting.

Parameters	Parameter Description	Value
time_step	Time step	1–10
num	Number of hidden layers	5
num_units	Number of hidden neurons	32, 32, 16, 8, 4
activation	Activation function	Sigmoid, Relu, tanh
optimizer	Optimizer	adam, RMSProp, Adagrad, Adadelta
epoch	Number of iterations	100–500
batch	Batch Size	16, 32, 64, 128
dropout	Dropout	0.1–0.5

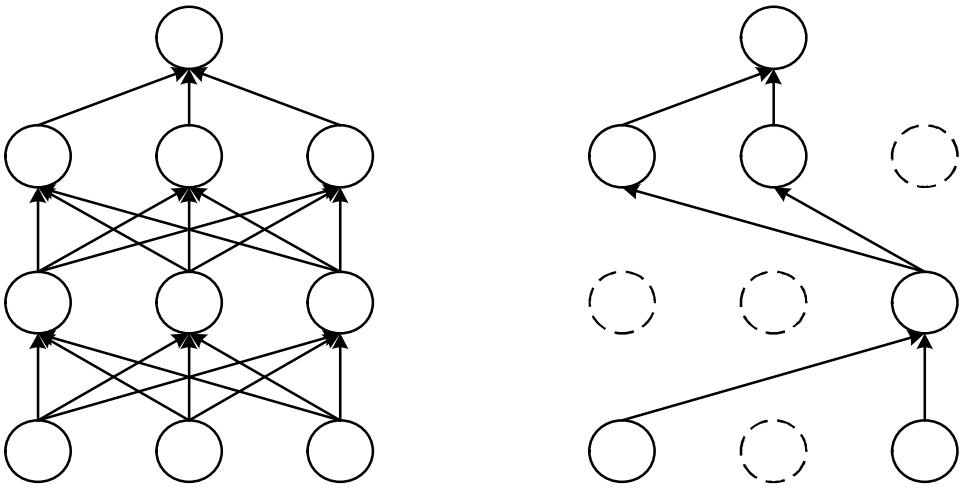


Figure 10. Dropout mechanism diagram.

4.4. Analysis of Results

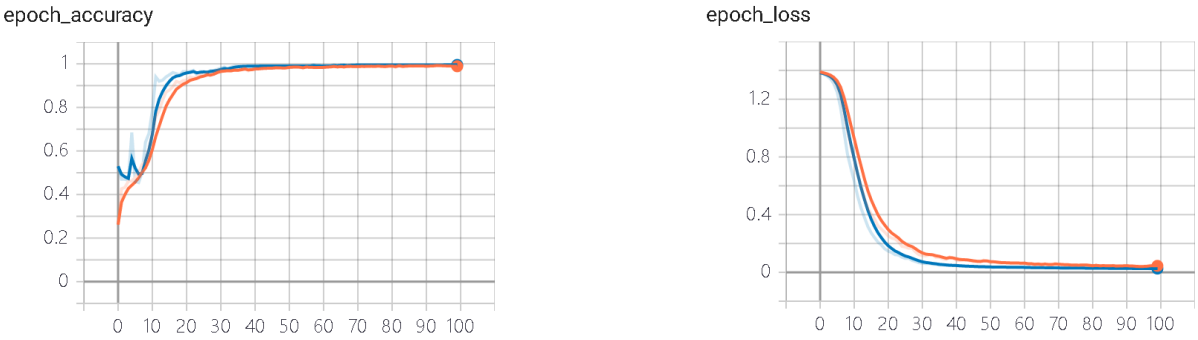
Some 5-layer deep LSTM neural networks were constructed based on the hyperparameters in Table 1. Using one-hot coding, our method was to encode N states using N-bit status registers. Each state had its independent register bits, and only one was valid at any time as this experiment selected four operating conditions (normal operation, loss of coolant accident, steam generator tube rupture, containment steam pipe rupture). The one-hot code had four states, and the one-hot codes are shown in Table 2. PCTRAN was used in each of the four conditions to obtain 28,500 sets of data, for a total of 114,000 sets of data. Using random sampling, 300 sets of data (each set of data was a six-dimensional array of time steps in length) were obtained for each condition, totaling 500× time step sets of data. The data were divided into the training set, validation set, and test set, which are 350*time step, 50*time step, and 100*time step sets, respectively. The model was trained using the training set and validated using the validation set. The experimental model found the optimal hyperparameters, as shown in Table 3. The training process of the LSTM neural network model was not an iterative process. The parameter epoch represents the number of iterations of the entire training set; epoch accuracy and epoch loss were the accuracy and loss function values for the corresponding number of iterations. The results obtained from the training are shown in Figure 11, where the orange curve represents the training results, and the blue curve represents the validation results. The lighter-colored lines represent the true calculated values, and the darker-colored lines represent the smoothed values.

Table 2. Fault and label correspondence table.

Operation Status	Tag
Loss of Coolant Accident	0
Steam Generator Tube Rupture	1
Steam Line Break Inside Containment	2
Normal Operation	3

Table 3. The optimal Model hyperparameter setting.

Parameters	Parameter Description	Value
time_step	Time step	1–10
num	Number of hidden layers	5
num_units	Number of hidden neurons	32, 32, 16, 8, 4
activation	Activation function	Relu
optimizer	Optimizer	adam
epoch	Number of iterations	100
batch	Batch Size	16
dropout	Dropout	0.5



(**a**) Deep LSTM training accuracy (**b**) Deep LSTM loss

Figure 11. Deep LSTM training parameters diagram.

In the confusion matrix constructed based on the deep LSTM neural network, the matrix rows represent the predicted fault classes, and the columns represent the actual fault classes. If the predicted and actual results agree, the data are on the diagonal of the confusion matrix; if the prediction is wrong, the data are outside the diagonal. The test accuracy is obtained using the test set test, and the analysis of its confusion matrix (Figure 12) shows that it only has one classification error in tag 3 (Normal Operation), which shows that the deep LSTM neural network-based nuclear power plant fault diagnosis model can accurately determine the operating group condition of nuclear power plants. In case of nuclear plant accidents, it can effectively help operation operators to quickly identify fault types and improve the overall safety of nuclear plants.

Meanwhile, we developed a simple LSTM model using the same parameters. The sole difference between the simple LSTM and deep LSTM models is that the simple LSTM has just one hidden layer, whereas the deep LSTM model has five. Figures 13 and 14 illustrate the outcomes of using the same dataset for training and validation.

The accuracy of the simple LSTM model is only 0.915, while the accuracy of the deep LSTM model is above 0.996. The accuracy of the simple LSTM model is lower than that of the deep LSTM model, and the loss value is higher than that of the deep LSTM model. The test results indicate that the deep LSTM model makes up for the shortcomings of the traditional simple LSTM model to a certain extent and further improves the applicability of the LSTM model.

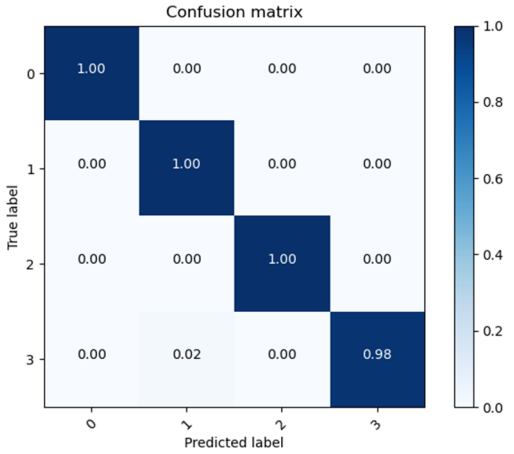
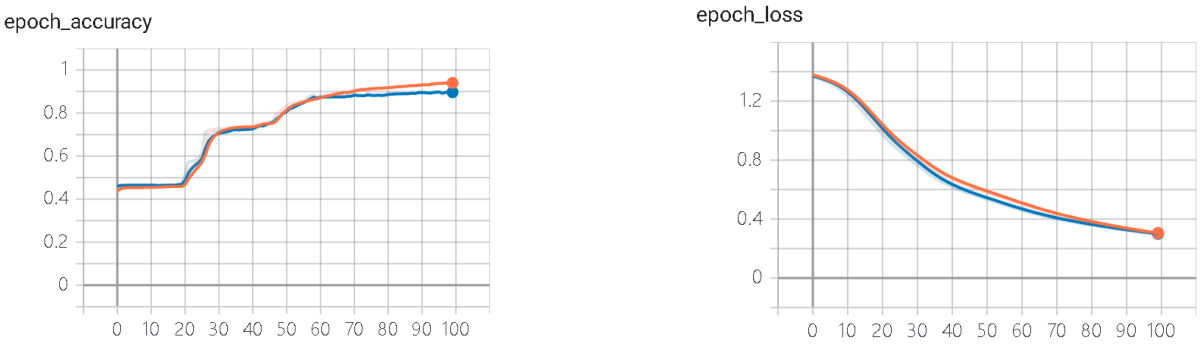


Figure 12. Deep LSTM model confusion matrix diagram.



(**a**) Simple LSTM training accuracy (**b**) Simple LSTM loss

Figure 13. Simple LSTM training parameters diagram.

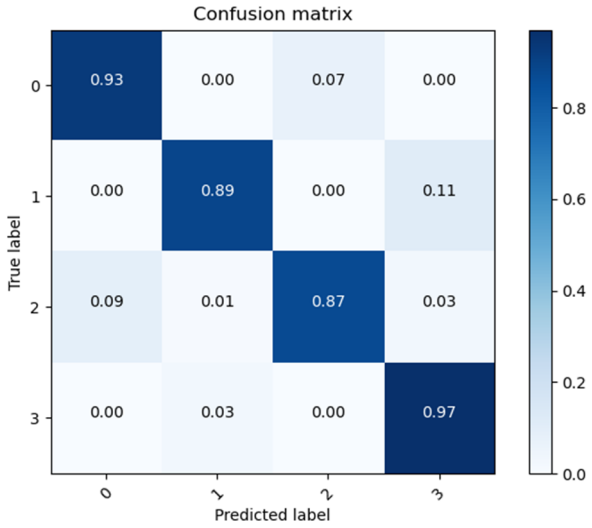


Figure 14. Simple LSTM model confusion matrix diagram.

5. Conclusions

In order to solve the nuclear power plant fault diagnosis issue, a nuclear power plant fault diagnostic system was developed utilizing deep LSTM neural network modeling and nuclear power plant accident critical parameter data supplied by PCTRAN. After comparing it to the simple LSTM model, we found that the deep LSTM model greatly improved the accuracy of fault prediction. Based on the training and test performance, it is evident that the system performed better in nuclear power plant fault diagnosis and could

satisfy the standards for nuclear power plant fault diagnosis. It can better assist nuclear power plant operators in controlling unit status in the event of nuclear power plant faults, ensuring the safe and reliable operation of nuclear power plants, reducing the likelihood of operator error in the event of nuclear power plant accidents, and enhancing the safety of nuclear power plant operation.

Author Contributions: Conceptualization, B.L. and J.L.; methodology, B.L.; software, J.L.; validation, J.X.; formal analysis, B.L.; investigation, B.L.; resources, B.L.; data curation, J.L.; writing—original draft preparation, B.L.; writing—review and editing, J.L.; visualization, J.Z.; supervision, J.X.; project administration, J.X.; funding acquisition, J.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (Grant No.71974091 and No.12175101), Scientific Research Fund of Hunan Provincial Education Department (Grant No.18C0414), Natural Science Foundation of Hunan province (Grant No.2022JJ30481), and the Postgraduate Scientific Research Innovation Project of Hunan Province (CX20220970).

Data Availability Statement: Data available on request.

Acknowledgments: We thank the NEAL group of the University of South China for their help.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yong-Kuo, L.; Min-Jun, P.; Chun-Li, X.; Ya-Xin, D. Research and design of distributed fault diagnosis system in nuclear power plant. *Prog. Nucl. Energy* **2013**, *68*, 97–110. [CrossRef]
2. Lee, S.L.; Seong, P.H. A Dynamic Neural Network Based Accident Diagnosis Advisory Systemfor Nuclear Power. *Prog. Nucl. Energy* **2005**, *46*, 268–281. [CrossRef]
3. Yu, H. Application of Artificial Neural Network for NHR Fault Diagnosis. *Nucl. Power Eng.* **1999**, *20*, 434–439.
4. Jichong, L.; Zhenping, C.; Jinsen, X.; Yu, T. Validation of Doppler temperature coefficients and assembly power distribution for the lattice code KYLIN V2.0. *Front. Energy Res.* **2021**, *10*, 898887. [CrossRef]
5. Lei, J.; Yang, C.; Ren, C.; Li, W.; Liu, C.; Sun, A.; Li, Y.; Chen, Z.; Yu, T. Development and validation of a deep learning-based model for predicting burnup nuclide density. *Int. J. Energy Res.* **2022**, 1–9. [CrossRef]
6. Jaime, G.D.G. Integration of computerized operation support system on a nuclear power plant environment. *Inst. Eng. Nucl. Prog. Rep.* **2018**, *3*, 94.
7. Sheng, Z.; Yin, Q. *Equipment Condition Monitoring and Fault Diagnosis Technology and Application*; Press of Chemical Industry: Beijing, China, 2003.
8. Kim, K.; Aljundi, T.L.; Bartlett, E.B. *Nuclear Power Plant Fault-Diagnosis Using Artificial Neural Networks*; Dept. of Mechanical Engineering, Iowa State Univ. of Science and Technology: Ames, IA, USA, 1992.
9. Wu, J.; Li, Q.; Chen, Q.; Peng, G.; Wang, J.; Fu, Q.; Yang, B. Evaluation, Analysis and Diagnosis for HVDC Transmission System Faults via Knowledge Graph under New Energy Systems Construction: A Critical Review. *Energies* **2022**, *15*, 8031. [CrossRef]
10. Tran, Q.T.; Nguyen, S.D. Bearing Fault Diagnosis Based on Measured Data Online Processing, Domain Fusion, and ANFIS. *Computation* **2022**, *10*, 157. [CrossRef]
11. Esakimuthu Pandarakone, S.; Mizuno, Y.; Nakamura, H. A comparative study between machine learning algorithm and artificial intelligence neural network in detecting minor bearing fault of induction motors. *Energies* **2019**, *12*, 2105. [CrossRef]
12. Nsaif, Y.M.; Hossain Lipu, M.S.; Hussain, A.; Ayob, A.; Yusof, Y.; Zainuri, M.A. A New Voltage Based Fault Detection Technique for Distribution Network Connected to Photovoltaic Sources Using Variational Mode Decomposition Integrated Ensemble Bagged Trees Approach. *Energies* **2022**, *15*, 7762. [CrossRef]
13. Wang, Z.; Zhang, Z.; Zhang, X.; Du, M.; Zhang, H.; Liu, B. Power System Fault Diagnosis Method Based on Deep Reinforcement Learning. *Energies* **2022**, *15*, 7639. [CrossRef]
14. Katreddi, S.; Kasani, S.; Thiruvengadam, A. A Review of Applications of Artificial Intelligence in Heavy Duty Trucks. *Energies* **2022**, *15*, 7457. [CrossRef]
15. Liu, Y.-K.; Xie, F.; Xie, C.-L.; Peng, M.-J.; Wu, G.-H.; Xia, H. Prediction of time series of NPP operating parameters using dynamic model based on BP neural network. *Ann. Nucl. Energy* **2015**, *85*, 566–575. [CrossRef]
16. Saha, A.; Fyza, N.; Hossain, A.; Sarkar, M.R. Simulation of tube rupture in steam generator and transient analysis of VVER-1200 using PCTRAN. *Energy Procedia* **2019**, *160*, 162–169. [CrossRef]
17. Horikoshi, N.; Maeda, M.; Iwasa, H.; Momoi, M.; Oikawa, Y.; Ueda, Y.; Kashiwazaki, Y.; Onji, M.; Harigane, M.; Yabe, H.; et al. The usefulness of brief telephonic intervention after a nuclear crisis: Long-term community-based support for Fukushima evacuees. *Disaster Med. Public Health Prep.* **2022**, *16*, 123–131. [CrossRef]

18. Lei, J.; Ren, C.; Li, W.; Fu, L.; Li, Z.; Ni, Z.; Li, Y.; Liu, C.; Zhang, H.; Chen, Z.; et al. Prediction of crucial nuclear power plant parameters using long short-term memory neural networks. *Int. J. Energy Res.* **2022**. [CrossRef]
19. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef]
20. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
21. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Phys. D Nonlinear Phenom.* **2020**, *404*, 132306. [CrossRef]
22. Boring, R.L.; Thomas, K.D.; Ulrich, T.A.; Lew, R.T. Computerized operator support systems to aid decision making in nuclear power plants. *Procedia Manuf.* **2015**, *3*, 5261–5268. [CrossRef]
23. Shi, X.; Chen, Z.; Wang, H.; Yeung, D.Y.; Wong, W.K.; Woo, W.C. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 802–810.
24. Lei, J.; Zhou, J.; Zhao, Y.; Chen, Z.; Zhao, P.; Xie, C.; Ni, Z.; Yu, T.; Xie, J. Prediction of burn-up nucleus density based on machine learning. *Int. J. Energy Res.* **2021**, *45*, 14052–14061. [CrossRef]
25. Lei, J.; Zhou, J.; Zhao, Y.; Chen, Z.; Zhao, P.; Xie, C.; Ni, Z.; Yu, T.; Xie, J. Research on the preliminary prediction of nuclear core design based on machine learning. *Nucl. Technol.* **2022**, *208*, 1223–1232. [CrossRef]
26. Quinlan, J.R. *C4. 5: Programs for Machine Learning*; Elsevier: Amsterdam, The Netherlands, 2014.
27. Gal, Y.; Ghahramani, Z. A theoretically grounded application of dropout in recurrent neural networks. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 1019–1027.