RESEARCH ARTICLE

Machine Learning-Based Fault Diagnosis for a PWR Nuclear Power Plant

AMINE NAIMI [1], **(Associate Member, IEEE), JIAMEI DENG** [1], **(Senior Member, IEEE),**
PAUL DONEY [1], **AKBAR SHEIKH-AKBARI** [1], **S. R. SHIMJITH** [2], **AND A. JOHN ARUL** [3]
[1] School of Built Environment, Engineering and Computing, Leeds Beckett University, LS1 3HE Leeds, U.K.
[2] Bhabha Atomic Research Centre, Homi Bhabha National Institute, Mumbai 400094, India
[3] Indira Gandhi Centre for Atomic Research, Kalpakkam 603102, India

Corresponding author: Jiamei Deng (j.deng@leedsbeckett.ac.uk)

ABSTRACT In the nuclear power industry, safety and reliability are of the utmost importance. Sensors and actuators are integral components in such systems, and potential faults may adversely impact system performance. It is therefore imperative to design a fault detection and diagnosis (FDD) system that achieves the highest standards of safety. This paper presents a machine learning-based fault detection and diagnosis (FDD) technique for actuators and sensors in a pressurized water reactor (PWR). In the proposed FDD framework, faults are first detected using a shallow neural network. Second, fault diagnosis is performed using 15 different classifiers provided in the MATLAB Classification Learner toolbox, including support vector machine (SVM), K-nearest neighbor (KNN), and ensemble. Several classifiers were found to provide superior classification performance, including medium KNN, cubic KNN, cosine KNN, weighted KNN, fine Gaussian SVM, quadratic SVM, medium Gaussian SVM, coarse Gaussian, bagged trees, and subspace KNN. The accuracy of the FDD approach was demonstrated using a set of simulation results.

INDEX TERMS Fault detection and diagnosis (FDD), nuclear reactor, machine learning, neural network.

I. INTRODUCTION

In recent years, increased awareness of global climate change has led to a renewed interest in nuclear energy. Nearly 10% of the world's electricity comes from nuclear power, and this number is set to increase as more power reactors are being constructed [1]. Nuclear power plants (NPPs) are complex systems with an enormous number of sensors and actuators. As actuators and sensors play a vital role in NPPs, any degradation or damage to them could adversely impact the power plant's performance and seriously affect its reliability and safety. Therefore, it is crucial to develop effective fault detection and diagnosis techniques for actuators and sensors in NPPs to prevent accidents, which will, in turn, guarantee safety and reliability of the NPPs.

Fault diagnosis methods are normally classified into model-, signal- or data-driven-based approaches. Model-based FDD relies on a well-established model that represents the normal behavior of practical processes. Fault diagnosis can be performed by comparing the model's output with the real-time process output [2], [3]. Signal-based FDD is based on an analysis of the output signal. Faults in the process are visible in either or both the frequency and time domains [4], [5]. An approach based on data-driven analysis relies on implicit knowledge derived from historic plant data, which can be obtained either by intelligent training or machine learning [6], [7].

Machine learning techniques play an important role in data-driven fault-diagnosis methods. Principal component analysis (PCA) models were employed to detect and diagnose sensor faults in the pressurizer of a pressurized water reactor (PWR) [8]. Li et al. [9] proposed an improved PCA-based method by combining a conventional PCA algorithm with a false-alarm reduction method, which was able to successfully detect and isolate sensor faults even when a minor failure occurred. In [10], PCA and Fisher discriminant analysis (FCA) were used to diagnose two types of faults in the actuator of a research reactor: the

The associate editor coordinating the review of this manuscript and approving it for publication was Essam A. Rashed [ORCID].

control rod withdrawal fault and the external reactivity insertion fault. Artificial neural networks (ANNs) have also been used for system identification and FDD in NPPs because they provide outstanding performance in estimating nonlinear systems [11], [12], [13], [14], [15]. An ANN-based FDD method using a K-nearest neighbor (KNN) for the sensor and actuator FDD of an NPP was proposed in [15]. Shallow neural networks were used to detect the faults, and a KNN algorithm was employed to classify the faults. They demonstrated the efficiency of their proposed FDD technique in terms of the detection and its classification accuracy.

Most of the methods discussed above focus on either sensor or actuator faults, but not on both. Specifically, the literature refers more to sensor faults than actuator faults [8], [9], [11], [12]. To the best of the authors' knowledge, only a few FDD approaches can detect and isolate faults in sensors and actuators in NPPs. The FDD approach proposed in [15] can detect and diagnose the presence of sensor and actuator faults; however, faults only concentrate on the reactor subsystem. As an extension of [15], this study evaluates actuator faults and sensor faults throughout the entire PWR NPP using a practical machine learning-based approach. The proposed FDD approach was tested on faults that affect the sensors and actuators of the reactor core, steam generator, pressurizer pressure and level, and turbine-speed system. The main contributions of this paper are as follows:

- A machine-learning-based FDD is proposed to address sensor's and actuator's faults of an entire PWR.
- The FDD method is used to diagnose failures in the reactor core, steam generators, pressurizer pressure and level systems, and turbine-speed system.
- An evaluation and comparison of SVM, KNN, and ensemble classifiers is conducted.

The remainder of this paper is organized as follows: Section II provides a description of the PWR process. Section III discusses a fault-detection approch using neural networks. The fault- diagnosis approach is explained in Section IV, and Section V describes the simulation results of the proposed technique. The final section of this paper presents the conclusions.

II. PRESSURIZED WATER REACTOR

A classic PWR plant includes two main loops: primary and secondary loops. The primary loop comprises the reactor core, pressurizer, steam generator (SG), and reactor coolant pump. The reactor core can be described using a point kinetics model. The actuator, also called the control-rod system, is located inside the core. The basic purpose is to control the power of the core by adjusting the number of neutrons in it. The pressurizer is represented by the pressure and level equations. In the primary loop, the pressurizer maintains pressure such that boiling does not occur. Heat is transferred from the reactor core to the SG, where steam is produced and supplied to the secondary system. The turbine, moisture separators, and steam reheaters comprise the secondary

TABLE 1. Definition of PWR model variables.

Variable	Definition
P_n	Reactor core power
ρ_{rd}	Reactivity
P_p	Pressurizer pressure
Q_h	Rate of heat addition by heater
L_w	Pressurizer level
$\dot{m}_{sur}$	Mass surge flow rate
P_{sg}	Steam generator pressure
C_{tg}	turbine-governor valve coefficient
w_{tur}	Turbine speed

loop. A brief summary of the reactor power core model is presented here for brevity. For a detailed and comprehensive description of the NPP mathematical model, sensors, and actuators, the reader is kindly referred to [16]. A summary of the essential inputs and outputs of the plant is presented in Table 1.

A. REACTOR CORE MODEL

The reactor model can be described using a point kinetics equation with six groups of delayed neutron precursors coupled with thermal hydraulics. The reactor core model is represented as follows [16]:

$$\frac{dP_n}{dt} = \frac{\rho_t - \sum_{i=1}^{6} \beta_i}{\Lambda} P_n + \sum_{i=1}^{6} \lambda_i C_{in}, \tag{1}$$

$$\frac{dC_{in}}{dt} = \lambda_i (P_n - C_{in}), \qquad i = 1, 2, \ldots 6, \tag{2}$$

$$\frac{dT_f}{dt} = H_f P_n - \frac{1}{\tau_f} \left(T_f - T_{c1} \right), \tag{3}$$

$$\frac{dT_{c1}}{dt} = H_c P_n + \frac{1}{\tau_c} \left(T_f - T_{c1} \right) - \frac{2}{\tau_r} \left(T_{c1} - T_{cin} \right), \tag{4}$$

$$\frac{dT_{c2}}{dt} = H_c P_n + \frac{1}{\tau_c} \left(T_f - T_{c1} \right) - \frac{2}{\tau_r} \left(T_{c2} - T_{c1} \right), \tag{5}$$

$$\rho_t = \rho_{rd} + \alpha_f T_f + \alpha_c \left(T_{c1} + T_{c2} \right), \tag{6}$$

$$\frac{d\rho_{rd}}{dt} = G v_{rd}. \tag{7}$$

In the above equations, P_n is the normalized neutronic power; C_{in} is the normalized delayed neutron precursor's concentration; β_i and λ_i are the fraction of delayed neutrons and decay constant, respectively; Λ is the prompt neutron lifetime; ρ_t and ρ_{rd} represent the total reactivity and the reactivity due to the control rod, respectively; and T_f, T_{c1}, and T_{c2} are the temperatures at fuel, coolant node 1 and node 2, respectively. H_f and H_c are constant values; τ_c, and τ_r represent time constants; α_f and α_c denote the temperature coefficients of reactivity due to fuel and coolant, respectively; G and v_{rd} represent the reactivity worth and the rod speed, respectively.

B. TYPES OF FAULTS

This study considers bias and drift faults in sensors, and offset and saturation faults in actuators in five different loops of a

PWR plant. In total, there are 20 faults, as listed in Table 2. A description of the faults is provided below.

TABLE 2. Descriptions of faults.

Fault ID	Process variable	Type
F1	$\dot{m}_{sur}$	Offset in the pressurizer level actuator.
F2	$\dot{m}_{sur}$	Saturation in the pressurizer level actuator.
F3	l_w	Bias in the pressurizer level sensor.
F4	l_w	Drift in the pressurizer level sensor.
F5	C_{tg}	Offset added to the turbine-governor-valve (steam pressure loop) signal.
F6	C_{tg}	Saturation on the turbine-governor-valve (steam pressure loop) .
F7	p_{sg}	Bias on the steam pressure sensor.
F8	p_{sg}	Drift on the steam pressure sensor.
F9	Q_h	Offset on the pressurizer heater.
F10	Q_h	Saturation on the pressurizer heater.
F11	p_p	Bias on the pressurizer pressure sensor.
F12	p_p	Drift on the pressurizer pressure sensor.
F13	C_{tg}	Offset on the turbine-governor-valve (turbine speed control).
F14	C_{tg}	Saturation on the turbine-governor-valve (turbine speed control) .
F15	w_{tur}	Bias on the turbine speed sensor.
F16	w_{tur}	Drift on the turbine speed sensor.
F17	ρ_{rd}	Offset on the control rod actuator.
F18	ρ_{rd}	Saturation on the control rod actuator.
F19	P_n	Bias on the power sensor.
F20	P_n	Drift on the power sensor.

1) BIAS FAULT

A fault of this type corresponds to a constant offset in the sensor's output, which could be due to incorrect calibration or physical changes in the sensors [17]. In this study, a bias fault was added to the reactor power, pressurizer level, pressurizer pressure, steam pressure and turbine speed sensors. The bias fault is mathematically modeled as follows:

$$s(t) = s^*(t) + \zeta + b, \tag{8}$$

where $s(t)$ denotes the output of the sensor, s^* is the ideal measurement of the sensor, ζ is the measurement noise, and b is the bias added to the signal at a certain time.

2) DRIFT FAULT

This consists of an offset that changes over time. Drift faults are considered to be one of the most common faults in sensors. A number of factors contribute to this, including the corrosion of sensors over time and physical damage caused by extreme environmental conditions [18]. In the same way as the bias fault, the drift fault was injected into the reactor power, pressurizer level, pressurizer pressure, steam pressure and turbine speed sensors at a certain point in time. The drift fault can be mathematically represented as follows:

$$s(t) = s^*(t) + \zeta + d(t), \quad d(t) = a_0 + a_1 e^{\beta t}, \tag{9}$$

where a_0 is a constant, a_1 denotes the scaling parameter and β is the decay rate.

3) ACTUATOR SATURATION FAULT

occurs when the actuator exceeds a set saturation value [19]. In this study, saturation was applied to the output of the control rod system, turbine governor valve, pressurizer heater, and pressurizer level actuator. Owing to the physical limitations of the actuator, its output can be described as follows:

$$z(t) = \begin{cases} Z_{max}, & \text{if } Z_c > Z_{max}, \\ Z_c, & \text{if } Z_{min} \leq Z_c \leq Z_{max}, \\ Z_{min}, & \text{if } Z_c < Z_{min}, \end{cases} \tag{10}$$

where Z_{max} and Z_{min} denote the maximum and the minimum value of the actuator, respectively. Z_c denotes the required actuator output without taking actuator saturation into account.

4) ACTUATOR OFFSET FAULT

This corresponds to an offset added to the different actuator outputs at a certain time. Often, this failure occurs because of manufacturing defects or design flaws [20]. Similar to the actuator saturation fault, the actuator offset was injected into the control rod system, turbine governor valve, pressurizer heater, and pressurizer-level actuator outputs. The actuator offset fault can be modeled as follows:

$$z(t) = z^*(t) + \omega, \tag{11}$$

where z^* represents the actuator output without considering the faults and ω is the offset injected at a certain time.

III. FAULT DETECTION BASED ON NEURAL NETWORKS

This section examines the use of neural networks for fault detection in PWR. First, a brief description of the neural network structure, specifically shallow neural networks, is provided. Following this, the fault-detection approach based on neural network is formulated.

A. NEURAL NETWORK ARCHITECTURE

This study uses feedforward networks consisting of three layers, as illustrated in Fig. 1 [15]. The first layer (input
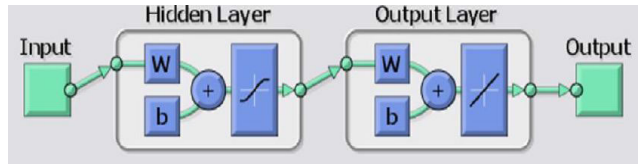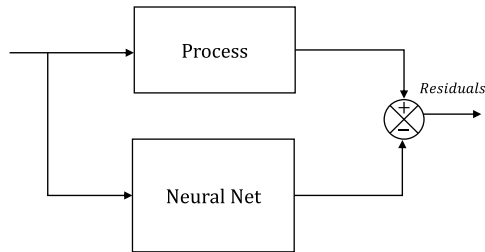
FIGURE 1. Structure of the neural network.

FIGURE 2. On-line residual generation.

FIGURE 3. Testing result of the trained neural network for the pressurizer pressure.

layer) represents the network interface with external inputs. The second layer is a hidden layer, where shallow neural networks typically have one hidden layer. Finally, the output layer generates estimations. The hidden layer $h(h_1, \ldots, h_N)$, and the output layer $y(y_1, \ldots, y_N)$ can be represented using the following equations:

$$h_k = f^1(w_{xh}x + b_h), \tag{12}$$
$$y_k = f^2(w_{hy}h_k + b_y), \quad k = [1, \ldots, N], \tag{13}$$

where x represents the input vector, w denotes the weight matrix (for instance, w_{xh} is the input-to-hidden layer weight matrix), b represents the vector bias (for instance, b_h represents the hidden bias vector, $f^1(.)$ denotes a sigmoid activation function while $f^2(.)$ denotes a linear activation function.

B. FAULT DETECTION METHODOLOGY

The previously described neural network is employed to detect the presence of faults in a PWR plant. The fault-detection approach consists of one neural network for each of the five loops of the plant. To accomplish this, each neural network is trained to learn one of the five input-output dynamics of the PWR plant. Each trained neural network is then used as a reference model to compare the output of the PWR plant (corresponding to a possibly faulty system) with the output of the neural network representing the non-faulty system, resulting in residual signals. A schematic of the neural-network-based fault-detection approach is depicted in Fig. 2. Essentially, the residual signal serves a signal for the detection of a fault in a power plant. If a fault occurs, its value will increase. Two phases exist in the proposed fault detection scheme, which are (1) the phase of system identification, and (2) the phase of fault detection. The system identification exercise aims to represent the nonlinear dynamics of the power plant that is indicative of a healthy operation. The trained neural networks then represent the
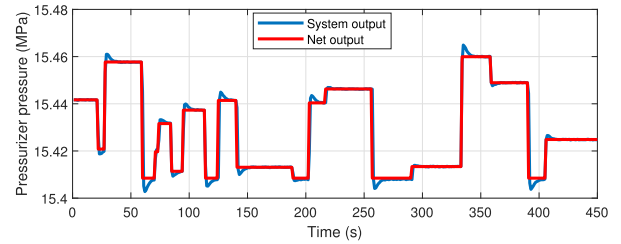
healthy behavior and operation of the power plant. The fault detection phase detects the presence of a fault in the power plant by analyzing the residual signals, which are the difference between the outputs of the plant and those of the trained neural networks, to determine whether any residual error exceeds a threshold that flags the presence of the fault.

1) SYSTEM IDENTIFICATION PHASE

To ensure a successful FDD system, the system identification phase must be executed appropriately. As a result of this phase, the five neural networks learn about the dynamics of the non-faulty power plant. There are three steps to achieve this goal: training, validation and testing. The training exercise is performed using the Levenberg-Marquardt (LM) algorithm. The testing result of one of the trained neural networks corresponding to the pressurizer pressure output is shown in Fig. 3.

2) FAULT DETECTION PHASE

Once the neural networks have been trained, residual signals can be generated based on the trained networks. The residual signal represents the difference between the output of the neural network and corresponding actual output of the power plant. The presence of a fault in the power plant will increase the difference between these two outputs; therefore, the fault is detected by monitoring the residual signal by considering an appropriate threshold.

IV. FAULT CLASSIFICATION METHODS
A. DATA COLLECTION

To solve the fault classification problem, the data of pressurizer pressure, pressurizer level, reactor power, steam pressure, turbine speed, and the residuals generated with the help of the neural networks are used. To make sense of the data, the six output variables were collected for each type of the fault (see Table 2) for 50 simulation runs and a data sampling rate of 1000 samples/s. As a result of this sampled data, the range of each sensor was preprocessed to [0, 1] to avoid polarization during training. Hence, the measurements of each sensor are normalized using min-max scaling, which is commonly used in machine learning

applications, as follows:

$$x'_s = \frac{x_s - x_{min}}{x_{max} - x_{min}}, \tag{14}$$

where x'_s is the normalized measurements of the sth sensor, and x_{max} and x_{min} are the minimum and maximum of the training dataset for measurements of the given sensor. Because the datasets are too large for a personal computer to handle, they are then downsampled to 100 samples per second. Finally, the data are reshaped into a vector matrix.

B. CLASSIFICATION ALGORITHM
In this study, three families of classification algorithms from the MATLAB Classification Learning Toolbox were selected to evaluate their suitability for the fault classification of PWR-NPP actuators and sensors. KNN, SVM (support vector machine)-, and ensemble-based classifiers are the algorithms used for classification.

The kNN approach, which is commonly used for pattern classification, determines the class of a new object by examining its distance from its nearest neighboring training samples [21]. The KNN method classifies an unlabeled sample based on its similarity to training set samples. Numerous applications of the KNN method have been developed in data mining, statistical pattern recognition, and image processing. Recently, the KNN method was applied for fault detection in the process industry [22]. However, KNN has rarely been applied to fault classification because of the difficulty in identifying possible faults and including them in the training data [22], [23], [24]. A review of the basic concepts of the KNN approach for identifying patterns is presented in this section. For example, for a given unlabeled sample x, KNN methods based on distance metrics find the k-nearest labeled samples in the training data set. Distance metrics are used to define how far individuals are from each other. It is common for kNN algorithms to use Euclidean distance, even though several other distance metrics have also been proposed. For example, if p and q are Euclidean points and it is assumed that $p = (p_1, p_2, p_3, p_4, \ldots, p_n)$ and $q = (q_1, q_2, q_3, q_4, \ldots, q_n)$, then, the Euclidean distance can be expressed as follows [25]:

$$Dist(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2},$$
$$= \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}. \tag{15}$$

SVM is a machine-learning algorithm that uses structural risk minimization and statistical learning for classification and regression [26]. Many applications have successfully employed the SVM method to separate the data into two or more classes. The purpose of using an SVM is to determine the optimal hyperplane that separates data points into two classes. Optimal hyperplanes maximize the separation margin between the two classes. Therefore, the SVM is mainly utilized for classifying linearly separable datasets.

The optimal hyperplane can be expressed mathematically as follows [2]:

$$w^T x + b = 0, \tag{16}$$

where x is the input vector, w is the weight vector, and b is the bias vector. The support vector equation for each class is as follows:

$$w^T x + b = +1, \quad \text{for } d_i = +1, \tag{17}$$
$$w^T x + b = -1, \quad \text{for } d_i = -1, \tag{18}$$

where d_i corresponds to the respective class, i.e., $d_i = +1$ for class A, and $d_i = -1$ for class B. The optimization problem for the training sample $\{(x_i, d_i)\}_{i=1}^{k}$ to find the optimal hyperplan is given as

$$\min F(w) = \frac{1}{2} w^T w, \tag{19}$$

such that $d_i(w^T x_i + b) \geq 1$, for $i = 1, 2 \ldots, k$.
Following is the final decision function:

$$f(x) = sign(\sum_{i=1}^{N} \alpha_{0,i}(x^T x_i + b), \tag{20}$$

where x represents the input vector to be classified, and N represents the number of support vectors obtained during training. Support vectors are defined based on non-negative parameters alphas $\alpha_{0,i}$. In order to categorize the linearly nonseparable pattern, a mapping function $\phi(x)$ transforms the data into a higher dimensional feature space. A modification of (20) can be made as follows:

$$f(x) = sign(\sum_{i=1}^{N} \alpha_{0,i}(\phi(x)\phi(x_i)) + b). \tag{21}$$

Kernel functions computed over high-dimensional spaces are used to reduce their complexity. These are defined as $K(x, y) = \phi(x)\phi(y)$. The following updates can be made for the decision function:

$$f(x) = sign(\sum_{i=1}^{N} \alpha_{0,i} K(x, x_i) + b). \tag{22}$$

As far as nonlinear cases are concerned, they cannot be adapted. Therefore, nonlinear transformations are performed using the kernel functions. By mapping a nonlinearly separable object into a higher-dimensional feature space, the kernel functions make it linearly separable. Linear kernels, polynomial kernels, and Gaussian radial basis function kernels are examples of common kernel functions used in the literature, as shown in Table 3 [27], [28].

An ensemble classifier enhances the accuracy of predictions by combining numerous diverse classes. Each classifier is trained separately, and the resulting trained classifiers are combined. The classifiers contain a set of learning algorithms that classify the new data points based on the weight of their predictions. Combined ensembles may be trained later as single hypotheses, and not necessarily constrained by the

TABLE 3. Common SVM kernel functions [27], [28].

Kernel name	Kernel function formula $k(x, y) =$
Linear	$x^T y + c$
Polynomial	$(ax^T y + c)^d$
Gaussian or Radial Basis Function	$exp(-\gamma||x - y||^2)$ where, γ is an ajustable parameter and $||x-y||$ is the euclidean distance between the two feature vectors

hypotheses from which they are derived. Overfitting may occur as a result of this flexibility; bagged trees overcome this problem by training classifiers in different partitions and combining them through a majority vote. An improved prediction accuracy is achieved when the correlation of the errors of a single classifier is weaker [29], [30].

C. CLASSIFIERS FROM THE TOOLBOX
The MATLAB Classification Learner toolbox was used to train the models to determine the classification of data using supervised machine learning. In this study, fault diagnosis was performed using SVM-, KNN-, and ensemble-based classification methods provided by the toolbox. A list of the 15 selected classifiers is provided in Table 6. 1) SVM: linear SVM, quadratic SVM, cubic SVM, fine Gaussian SVM, and medium Gaussian SVM. 2) KNN: medium KNN, coarse KNN, cosine KNN, cubic KNN, and weighted KNN. 3) Ensemble: boosted trees, bagged trees, subspace discriminant, subspace KNN, and RUS-boosted trees.

V. SIMULATION RESULTS AND ANALYSIS
Towards achieving fault diagnosis, two tasks must be completed. We first present the neural network learning phase, in which neural networks learn the input-output dynamics of the power plant. Second, we present the results of the fault diagnosis analysis, which includes fault detection and fault isolation.

A. TRAINING THE NEURAL NETWORKS
The neural network described in Section III was used to identify the different subsystems: pressurizer pressure loop (NN$_{pp}$), pressurizer level loop (NN$_{pl}$), reactor power loop (NN$_{re}$), steam generator loop (NN$_{sg}$), and turbine-speed loop (NN$_{ts}$). For brevity, only the identification of the pressurizer pressure loop is discussed herein. Pressurizer loop identification was performed using 3000 input data samples. The pressurizer demand was considered as a random step input signal and the pressurizer pressure was considered as a target variable. There are three partitions for the identification data: training data, validation data, and testing data, where 70%, 15%, and 15% of the data were used for each of them. The neural network was trained using the

TABLE 4. Inputs and outputs that are used for the neural network training.

Case	Input	Output	Neurons in hidden layer	Output layer	Size of training
NN$_{re}$	P_n demand	P_n	9	1	2000
NN$_{sg}$	p_{sg} demand	p_{sg}	5	1	2000
NN$_{pp}$	P_p demand	P_p	7	1	3000
NN$_{pl}$	L_w demand	L_w	9	1	10000
NN$_{ts}$	w_{tur} demand	w_{tur}	7	1	10000
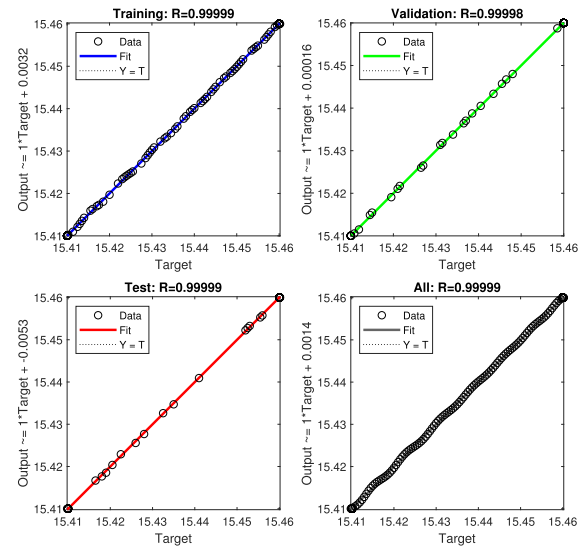
FIGURE 4. Regression curve for N_{pp}.
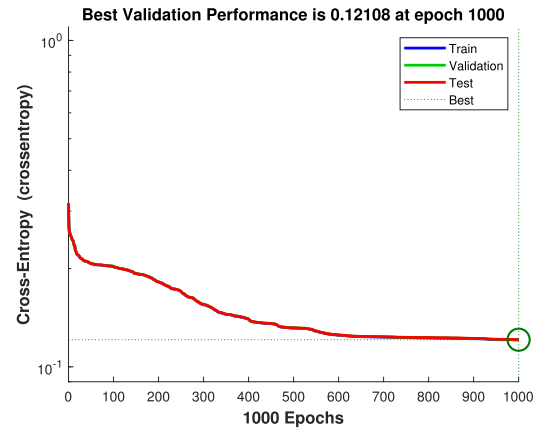
FIGURE 5. Performance for N_{pp}.

LM algorithm, and the largest improvement was obtained for seven neurons in the hidden layers. This procedure was applied to all other loops, and Table 4 summarizes the inputs and outputs of all networks. Fig. 4 shows the performance regression between the predicted and target values of the NN_{pp} model. The regression values were seen to be close to 1, which confirmed that the NN_{pp} model can successfully reproduce the PWR transient. The performance of the identified NN_{pp} model is illustrated in Fig. 5. It can be observed that the mean square error value decreased with epochs. This confirmed the good performance of the identified NN_{pp}.
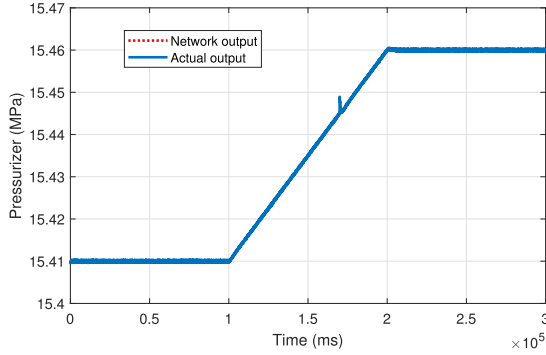
FIGURE 6. Variation of pressurizer pressure in the presence of Bias fault (F11).
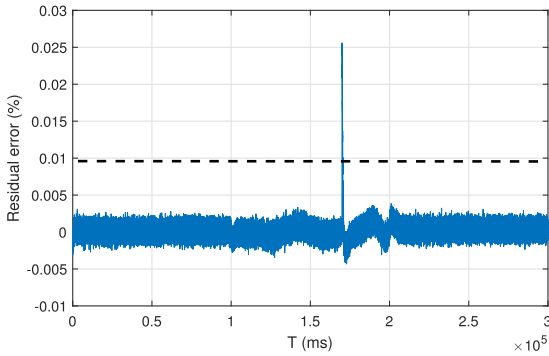
FIGURE 7. Residual generated corresponding to N_{pp} (F11).

B. FAULT DETECTION ANALYSIS

The faults were injected in the transient state of mode of the PWR plant. Different faults were injected into the power plant to determine the severity of the fault that must be detected according to the chosen threshold. Here, one fault case is considered as an example: the bias fault in the pressurizer pressure sensor (F11).

1) FAULT SCENARIO F11

A bias fault of +0.04 bar was injected into the pressurizer pressure sensor at t = 170 s. Fig. 6 shows the variation in the pressurizer pressure during the fault. Fig. 7 shows the residual signal corresponding to N_{pp}. After the fault occurred at $t = 170 s$, the output transient appeared to change, but the error was quickly corrected, as the controller tracked back to the set-point after 10ms. The fault detection time was t = 170.002 s.

C. FAULT CLASSIFICATION ANALYSIS

In this study, the data collected in the previous section were used to train the robust-machine learning classifiers such as SVM (linear, quadratic, cubic, medium Gaussian, fine Gaussian), KNN (medium, coarse, cosine, cubic, weighted); and Ensemble (boosted trees, bagged trees, RUS-boosted 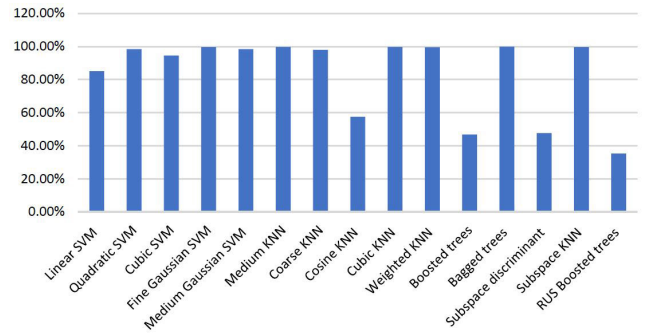tree, subspace discriminant, subspace KNN). For training and validation purposes, the fivefold cross-validation was employed. The fault-diagnosis accuracies of the classifiers

FIGURE 8. Classification accuracy for all faults implemented on PWR-NPP.

are shown in Fig. 8. It was found that ten classifiers (quadratic SVM, fine Gaussian SVM, medium Gaussian SVM, medium KNN, coarse KNN, cosine KNN, cubic KNN, weighted KNN, bagged trees, and subspace KNN) offered outstanding performance in terms of accuracy, with most of them generating 100% classification accuracy. However, not all the selected classifiers are suitable for fault diagnosis. The accuracies of the boosted trees and subspace discriminant classifiers were below 50%. In the worst case, the RUS-boosted trees had a classification accuracy of only 35.2%.

In this study, confusion matrix and receiver operating characteristic (ROC) curve were used to assess the performance of the classifiers. A confusion matrix is a table that depicts and summarizes the performance of a classification algorithm to determine its efficiency. For brevity, only one example of the confusion matrix for the coarse KNN is provided in Fig. 9, where the rows are the true class, columns show the predicted class, and diagonal cells indicate where the true class and predicted class coincide. It is possible to use the confusion matrix to calculate accuracy, precision, and recall, which can be represented as follows [30]:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \quad (23)$$

$$Precision = \frac{TP}{TP + FP}, \quad (24)$$

$$Recall = \frac{TP}{TP + FN}, \quad (25)$$

where TP and TN are true positive and true negative, respectively, and FP and FN are, false positive and negative, respectively. As shown in Fig. 9, all the 20 faults were all classified correctly. With an accuracy of 94.3%, the faults belonging to F7 were the least accurately classified, whereas the F2 and F3 faults were the most accurately classified at 99.8%. The overall classification performance was found to be 98%.

The ROC curves present a confusion matrix in graphic form. The ROC curve is created by plotting the true positive rate (TPR), or sensitivity, against the false positive rate (FPR), where the TPR and FPR are represented as used in [25]: The ROC graph presented in Fig. 10 illustrates the performance

Model 1.3 (Coarse KNN)

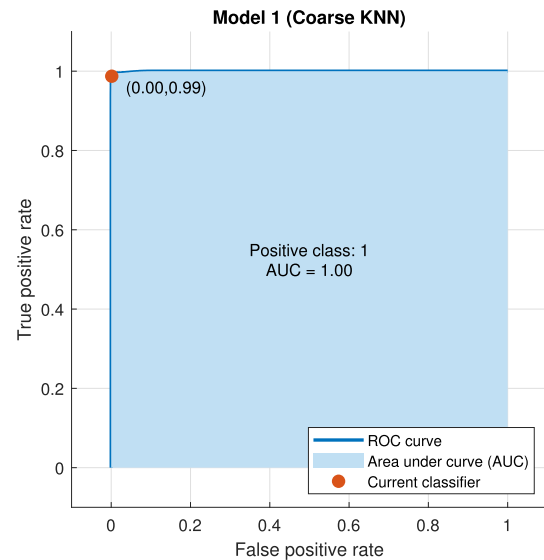| True Class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	TPR	FNR
1	98.7%	0.0%			0.1%	0.0%			0.0%		0.2%				0.0%	0.1%	0.3%	0.0%	0.6%		98.7%	1.3%
2		99.8%				0.0%					0.1%						0.1%				99.8%	0.2%
3	0.0%		99.8%		0.0%	0.0%		0.0%	0.0%		0.0%	0.0%		0.0%				0.1%	0.1%		99.8%	0.2%
4	0.2%			99.6%	0.0%				0.0%		0.0%							0.0%	0.1%		99.6%	0.4%
5	0.2%				94.9%	1.0%	1.0%	2.9%			0.0%					0.0%		0.0%			94.9%	5.1%
6	0.0%	0.0%			0.2%	96.2%	2.6%	0.0%			0.5%	0.5%					0.0%	0.0%			96.2%	3.8%
7	0.2%	0.0%			1.9%	0.9%	94.2%	0.9%			0.0%	0.4%		0.0%	0.2%	0.3%	0.1%	0.8%	0.1%		94.2%	5.8%
8		0.0%			0.0%	0.3%		99.6%			0.0%	0.0%									99.6%	0.4%
9	0.0%	0.1%	0.0%			0.0%	0.0%	0.0%	98.2%	0.7%	0.4%	0.0%	0.0%		0.1%		0.1%			0.2%	98.2%	1.8%
10	0.1%					0.0%	0.0%			99.3%		0.1%		0.1%		0.2%	0.0%	0.1%	0.1%		99.3%	0.7%
11						0.0%	0.0%		0.0%		99.5%	0.4%		0.1%	0.0%			0.0%			99.5%	0.5%
12	0.1%	0.0%		0.1%		0.0%			0.6%	0.0%		98.6%	0.0%			0.0%		0.0%	0.5%		98.6%	1.4%
13	0.0%	0.0%	0.1%		0.1%	2.3%	0.8%	0.1%		0.2%			94.9%	0.9%	0.1%	0.4%		0.1%	0.1%		94.9%	5.1%
14		0.0%	0.0%		0.6%	1.0%	0.3%	0.7%		0.1%			0.4%	96.4%		0.4%	0.0%				96.4%	3.6%
15	0.0%					0.0%	0.3%	0.2%			0.0%	0.0%	0.0%		99.1%	0.2%					99.1%	0.9%
16		0.2%			0.1%	0.3%	0.5%	0.0%	0.1%	0.1%			0.1%	0.0%	0.0%	98.4%		0.1%	0.0%		98.4%	1.6%
17	0.0%	0.1%				0.0%		0.0%			0.0%				0.0%		99.2%	0.6%	0.0%	0.0%	99.2%	0.8%
18	0.7%	0.4%	0.0%	0.0%			0.0%										2.2%	96.6%		0.1%	96.6%	3.4%
19	0.2%			0.0%	0.0%	0.2%	0.2%		0.1%	0.0%		0.0%	0.0%		0.0%		0.0%		98.8%	0.3%	98.8%	1.2%
20	0.8%		0.1%		0.1%				0.2%		0.1%						0.8%	0.7%		97.2%	97.2%	2.8%

Predicted Class

FIGURE 9. Confusion matrix using coarse KNN.

TABLE 5. Performance comparison of the classifiers.

Methods	Average AUC	Average accuracy	Execution time
Linear SVM	0.99	85.1%	4.49×10^4 s
Quadratic SVM	0.93	98.4%	4.69×10^4 s
Cubic SVM	0.99	94.5%	4.74×10^4 s
Fine gaussian SVM	0.99	99.6%	7.15×10^3 s
Medium SVM	0.99	98.5%	3.15×10^4 s
Medium KNN	0.99	99.7%	3.16×10^1 s
Coarse KNN	0.99	98%	8.88×10^1 s
Cosine KNN	0.87	57.5%	1.15×10^1 s
Cubic KNN	0.99	99.8%	7.14×10^1 s
Weighted KNN	0.99	99.5%	3.16×10^1 s
Boosted trees	0.85	46.8%	3.50×10^2 s
Bagged trees	0.99	99.9%	8.87×10^2 s
Subspace discriminant	0.82	47.6%	3.46×10^2 s
Subspace KNN	0.99	99.6%	4.18×10^2 s
RUS Boosted trees	0.75	35.2%	1.16×10^3 s

Model 1 (Coarse KNN)

(0.00,0.99)

Positive class: 1
AUC = 1.00

True positive rate / False positive rate

— ROC curve
▨ Area under curve (AUC)
● Current classifier

FIGURE 10. ROC curve using coarse KNN.

of the coarse KNN to determine the classification of F1. Based on the ROC curve shapes, the coarse KNN algorithm successfully classified F1 based on the area under the curve (AUC) of 1. Table 5 summarizes the AUCs of different classifiers. For most classifiers, the AUC average was high, except for the subspace discriminant, RUSBoosted trees, and Boosted trees algorithms, although the latter had a moderate level of accuracy. The KNN-based algorithms were found to be the most accurate methods, where they also required the least computational cost (measured in terms of execution time) compared with the SVM and ensemble algorithms. The computational time requirements of the SVM algorithms were significantly higher than those of the other algorithms.

VI. CONCLUSION

A machine-learning algorithm was proposed for the FDD of sensors and actuators in the entire PWR-NPP. First, neural networks were used to detect different faulty scenarios. Second, KNN, SVM and ensemble-based fault diagnosis methods were presented and their performances in fault classification were compared. The conclusion can be drawn that, while 10 classifiers were able to classify the 20 faults,

the KNN algorithms were found to have the most accurate classification of sensor and actuator faults. In addition, the KNN algorithms had the lowest computational cost among the compared techniques.

APPENDIX

Table A1 presents the description of the different classifier types.

Table A1. Classifiers from MATLAB machine learning toolbox [31].

Classification algorithms	Classifier types	Description
SVM	Linear SVM	Proceeds to a linear separation between classes, using the linear kernel.
	Quadratic SVM	Uses the quadratic kernel.
	Cubic SVM	Uses the cubic kernel.
	Fine Gaussian SVM	Makes finely-detailed distinctions between classes using the Gaussian kernel with the kernel scale set to $\sqrt{P}/4$ where P is the number of predictors.
	Medium Gaussian SVM	Makes fewer distinctions than a Fine Gaussian SVM, using the Gaussian kernel with the kernel scale set to $\sqrt{P}$.
KNN	Medium KNN	Makes fewer distinctions than a Fine KNN, with K=10.
	Coarse KNN	Makes coarse distinctions between classes, with K=100.
	Cosine KNN	Uses a cosine distance metric, with K=10.
	Cubic KNN	Uses a cubic distance metric, with K=10.
	Weighted KNN	Uses a weighted distance metric, with K=10.
Ensemble	Boosted trees	This model creates an ensemble of medium decision trees using the AdaBoost algorithm.
	Bagged trees	It is a bootstrap-aggregated ensemble of fine decision trees.
	Subspace discriminant	This model creates an ensemble of discriminant classifiers using the random subspace algorithm.
	Subspace KNN	This model creates an ensemble of nearest neighbor classifiers using Random Subspace algorithm.
	RUSBoosted trees	Used for skewed data with many more observation of one class.

REFERENCES

[1] *International Energy Agency, Key World Energy Statistics*, IEA, Paris, France, 2022.

[2] J. Ma and J. Jiang, "Applications of fault detection and diagnosis methods in nuclear power plants: A review," *Prog. Nucl. Energy*, vol. 53, no. 3, pp. 255–266, Apr. 2011.

[3] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis: Part I: Quantitative model-based methods," *Comput. Chem. Eng.*, vol. 27, no. 3, pp. 293–311, 2003.

[4] E. Sejdić, I. Djurović, and J. Jiang, "Time–frequency feature representation using energy concentration: An overview of recent advances," *Digit. Signal Process.*, vol. 19, no. 1, pp. 153–183, Jan. 2009.

[5] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3757–3767, Jun. 2015.

[6] H. Hashemian, *Maintenance of Process Instrumentation in Nuclear Power Plant*. Berlin, Germany: Springer-Verlag, 2006.

[7] G. Zhiwei, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—Part II: Fault diagnosis with knowledge-based and hybrid/active approaches," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3757–3767, Mar. 2015.

[8] P. Baraldi, A. Cammi, F. Mangili, and E. Zio, "An ensemble approach to sensor fault detection and signal reconstruction for nuclear system control," *Ann. Nucl. Energy*, vol. 37, no. 6, pp. 778–790, Jun. 2010.

[9] Y.-K. Liu, C.-L. Xie, M.-J. Peng, and S.-H. Ling, "Improvement of fault diagnosis efficiency in nuclear power plants using hybrid intelligence approach," *Prog. Nucl. Energy*, vol. 76, pp. 122–136, Sep. 2014.

[10] F. Jamil, M. Abid, I. Haq, A. Q. Khan, and M. Iqbal, "Fault diagnosis of Pakistan research reactor-2 with data-driven techniques," *Ann. Nucl. Energy*, vol. 90, pp. 433–440, Apr. 2016.

[11] B. C. Hwang, M. Saif, and M. Jamshidi, "Neural based fault detection and identification for a nuclear reactor," *IFAC Proc. Volumes*, vol. 26, no. 2, pp. 547–550, Jul. 1993.

[12] O. Elnokity, I. I. Mahmoud, M. K. Refai, and H. M. Farahat, "ANN based sensor faults detection, isolation, and reading estimates–SFDIRE: Applied in a nuclear process," *Ann. Nucl. Energy*, vol. 49, pp. 131–142, Nov. 2012.

[13] A. Naimi, J. Deng, A. Abdulrahman, V. Vajpayee, V. Becerra, and N. Bausch, "Dynamic neural network-based system identification of a pressurized water reactor," in *Proc. 8th Int. Conf. Control, Mechatronics Autom. (ICCMA)*, Nov. 2020, pp. 100–104.

[14] A. Naimi, J. Deng, A. Sheikh-Akbari, S. R. Shimjith, and A. J. Arul, "Input-output feedback linearization control for a PWR nuclear power plant," in *Proc. Eur. Control Conf. (ECC)*, Jul. 2022, pp. 1415–1420.

[15] A. Naimi, J. Deng, S. R. Shimjith, and A. J. Arul, "Fault detection and isolation of a pressurized water reactor based on neural network and K-nearest neighbor," *IEEE Access*, vol. 10, pp. 17113–17121, 2022.

[16] V. Vajpayee, V. Becerra, N. Bausch, J. Deng, S. R. Shimjith, and A. J. Arul, "Dynamic modelling, simulation, and control design of a pressurized water-type nuclear power plant," *Nucl. Eng. Des.*, vol. 370, Dec. 2020, Art. no. 110901.

[17] Y. Zhong, W. Zhang, Y. Zhang, J. Zuo, and H. Zhan, "Sensor fault detection and diagnosis for an unmanned quadrotor helicopter," *J. Intell. Robotic Syst.*, vol. 96, nos. 3–4, pp. 555–572, Dec. 2019.

[18] S. U. Jan, Y. D. Lee, and I. S. Koo, "A distributed sensor-fault detection and diagnosis framework using machine learning," *Inf. Sci.*, vol. 547, pp. 777–796, Feb. 2021.

[19] J. Hui and J. Yuan, "Disturbance observer-based robust backstepping load-following control for MHTGRs with actuator saturation and disturbances," *Nucl. Eng. Technol.*, vol. 53, no. 11, pp. 3685–3693, Nov. 2021.

[20] J. Chandrasekar and D. S. Bernstein, "Setpoint tracking with actuator offset and sensor bias-probing the limits of integral control," *IEEE Control Syst. Mag.*, vol. 27, no. 1, pp. 61–68, Feb. 2007.

[21] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Hoboken, NJ, USA: Wiley, 2001.

[22] Q. P. He and J. Wang, "Fault detection using the K-nearest neighbor rule for semiconductor manufacturing processes," *IEEE Trans. Semicond. Manuf.*, vol. 20, no. 4, pp. 345–354, Nov. 2007.

[23] Q. P. He and J. Wang, "Principal component based K-nearest-neighbor rule for semiconductor process fault detection," in *Proc. Amer. Control Conf.*, Jun. 2008, pp. 1606–1611.

[24] G. Wang, J. Liu, and Y. Li, "Fault diagnosis using KNN reconstruction on MRI variables," *J. Chemometrics*, vol. 29, no. 7, pp. 399–410, Jul. 2015.

[25] Z. Chen, L. J. Zhou, X. D. Li, J. N. Zhang, and W. J. Huo, "The lao text classification method based on KNN," *Proc. Comput. Sci.*, vol. 166, pp. 523–528, Jan. 2020.

[26] R. Gholami and N. Fakhari, "Support vector machine: Principles, parameters, and applications," in *Handbook of Neural Computation*. Amsterdam, The Netherlands: Elsevier, 2017, pp. 515–535.

[27] K.-R. Müller, S. Mika, K. Tsuda, and K. Schölkopf, "An introduction to kernel-based learning algorithms," in *Handbook of Neural Network Signal Processing*. Boca Raton, FL, USA: CRC Press, 2018, pp. 1–4.

[28] X. Liang, Y. He, M. Mitolo, and W. Li, "Support vector machine based dynamic load model using synchrophasor data," in *Proc. IEEE/IAS 54th Ind. Commercial Power Syst. Tech. Conf.*, May 2018, pp. 1–11.

[29] M. A. Wiering and H. V. Hasselt, "Ensemble algorithms in reinforcement learning," *IEEE Trans. Syst., Man, Cybern., B*, vol. 38, no. 4, pp. 930–936, Aug. 2008.

[30] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[31] *Statistics and Machine Learning Toolbox User's Guide*, Mathwork, Natick, MA, USA, 2017.

AMINE NAIMI (Associate Member, IEEE) is currently a Researcher with the School of Built Environment, Engineering and Computing, Leeds Beckett University, Leeds. His research interests include control systems, system identification, and artificial intelligence.

JIAMEI DENG (Senior Member, IEEE) received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 1988, the M.S. degree from the Shanghai Institute of Mechanical Engineering, Shanghai, China, and the Ph.D. degree from Reading University, U.K., in 2005. She is currently a Professor of artificial intelligence and energy with Leeds Beckett University with emphasis on control systems design, predictive models, data analytics, and power plant safety monitoring.

PAUL DONEY received the degree in electronic and electrical from Brighton University. He is currently the Head of Subject for Computing with the School of Built Environment, Engineering, and Computing, Leeds Beckett University. His research interests include computing and persuasive technology.

AKBAR SHEIKH-AKBARI received the B.Sc. (Hons.), M.Sc. (Hons.), and Ph.D. degrees in electronic and electrical engineering. He is currently an Associate Professor (Reader) with the School of Built Environment, Engineering and Computing, Leeds Beckett University. After completing his Ph.D. degree at Strathclyde University, he joined Bristol University to work on an EPSRC Project in stereo/multi-view video processing. He continued his career in the industry, working on real-time embedded video analytics systems. His main research interests include biometric identification techniques, color constancy techniques, image source camera identification, hyperspectral image processing, standard and nonstandard image/video codecs, e.g., H.264 and HEVC, multiview image/video processing, image resolution enhancement methods, assisted living technologies, compressive sensing, camera tracking using retro-reflective materials, and signal processing.

S. R. SHIMJITH received the B.E. degree from Mangalore University, India, in 1995, and the Ph.D. degree from the Indian Institute of Technology, Mumbai, India, in 2012. He is currently with the Bhabha Atomic Research Centre, India, and also a Professor with the Homi Bhabha National Institute, India. His research interests include mathematical modeling, analysis, simulation, and control of nuclear reactors and power plants.

A. JOHN ARUL received the M.Sc. degree in physics from Bharathidasan University, in 1988, the M.S. degree in software systems from the Birla Institute of Technology, in 1999, and the Ph.D. degree from the Indian Institute of Technology, Bombay, Mumbai, India, in 2015. He is currently a Researcher in nuclear engineering, risk and reliability analysis with the Indira Gandhi Centre for Atomic Research, Kalpakkam, India. His current research interests include application of artificial intelligence techniques to reactor safety and resilience.

• • •