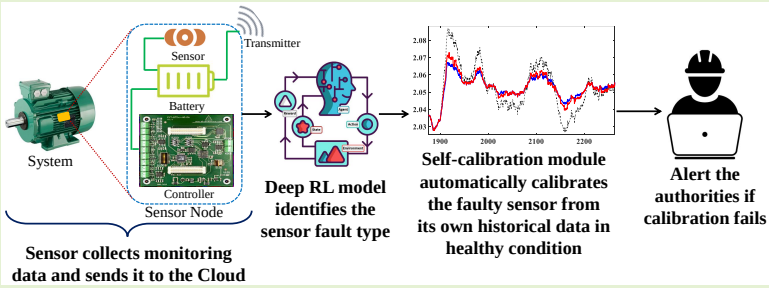
SNRepair: Systematically Addressing Sensor Faults and Self-Calibration in IoT Networks

Aparna Sinha *Student Member, IEEE* and Debanjan Das *Senior Member, IEEE*

***Abstract*— An accurate and robust technique for sensor fault diagnosis proves useful for an uninterrupted supply of correct monitoring data across the Internet of Things (IoT) network. The manual checking and calibration of thousands of sensors deployed in IoT network is a challenging task. Further, most calibration techniques require additional hardware support for calibration. To address these issues, a unique IoT-based framework, *SNRepair*, has been proposed that uses modified Deep Reinforcement Learning (DRL) for detecting different types of sensor faults, such as bias, drift, complete failure and precision degradation. This technique is also capable of self-calibrating the faulty sensors automatically using the historical healthy data of the same sensor. The fault detection module identifies four types of sensor faults with 96.17% accuracy. The self-calibration module can calibrate the faulty sensors within a few seconds, thereby ensuring the uninterrupted availability of accurate monitoring information. The proposed model is robust as it can work efficiently even in noisy environments with accurate results.**



***Index Terms*— Internet of Things (IoT), Sensor fault, Deep Reinforcement Learning (DRL), Self-calibration**

I. Introduction

The Internet of Things (IoT) networks are becoming increasingly important precursors to Industry 4.0, real-time monitoring, Cyber-physical Systems (CPS) and cloud computing [1]. For the continuous monitoring of the different machines and their environments, the Sensor Nodes (SNs), consisting of sensors, controllers, transducers and batteries, are installed across many aspects of the modern world [2]. In recent years, various compact and affordable IoT-compatible low-cost sensors (LCSs) have emerged in market, which have the potential to enhance the spatio-temporal resolution of data capturing in densely populated environments. However, these LCSs are often error-prone due to manufacturing defects, aging, hardware malfunctions or adverse environmental situations [3]. The most common types of sensor faults are bias, drift, complete failure (CF) and precision degradation (PD). Such sensor faults are very common, especially for those used in harsh environments [4], such as industries, smart cities, etc. The efficiency of the IoT-based smart monitoring of the industries largely depends on the quality of data collected by the LCSs. In addition, the measured data using LCS is less reliable. Hence, for uninterrupted and accurate IoT services, it is very important for the early detection of different types of sensor faults and try to calibrate them automatically.

Traditionally, the scheduled maintenance is performed for the health monitoring and calibration of the sensors. However, the deployment of thousands of sensors across the IoT network makes the manual procedure costly and time-consuming. The advent of Machine Learning (ML) and Artificial Intelligence (AI) has made real-time monitoring of sensor health feasible. But, these algorithms are trained for particular sensors deployed in specific working conditions, and extension of the models for the hundreds of heterogeneous sensors in the factory floor are costly short-term solutions [5]. Further, the faulty process data is limited and contains environmental noise that increases the challenge of training the fault detection models. After fault detection, the calibration of the faulty sensors is essential. The traditional calibration process is cumbersome and requires additional hardware support to be used as the standard. The various challenges encountered in identification and calibration of the sensor faults in IoT networks in industrial harsh environments (as a use case) are shown in Fig. 1.

In order to overcome the challenges, a unique IoT-based solution, *SNRepair*, has been proposed that can detect and calibrate the different sensor faults. A Deep Reinforcement Learning (DRL)-based self-learning algorithm is used for identifying all types of sensor faults in any working condition. The DRL model can adapt and mature according to its unseen working environment without manual intervention [6]. Further, the algorithm utilizes its own historical healthy data to self-calibrate the faulty sensors. For this purpose, it has been assumed that the sensor performs in healthy condition for the initial seven days after it is newly installed. An automatic self-

Aparna Sinha and Debanjan Das are with the Department of Electronics & Communication Engineering, IIIT Naya Raipur, Email: aparna.sinha@ieee.org; debanjan@ieee.org.

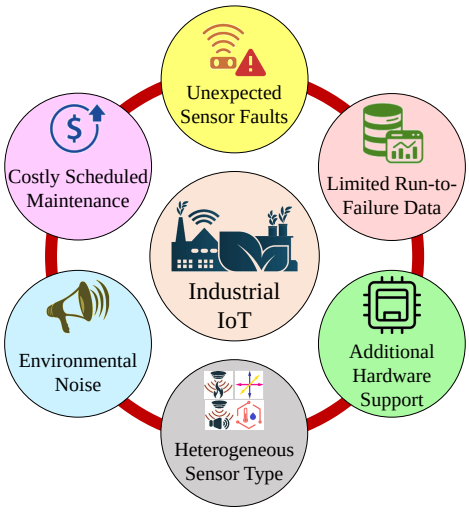


Fig. 1: Challenges in identification and calibration of sensor faults in factory-floor level.

calibration algorithm has been proposed, in which the faulty sensor uses its own historical healthy data for a successful calibration. It is to be noted that complete failure of a sensor cannot be rectified, and the sensor must be replaced. The novel contributions of the paper are as follows:

- A DRL-based fault detection technique, *SNRepair*, is proposed that can detect the different types of sensor faults. The entire process is automatic, thereby making it suitable to deploy in an IoT sensor node.
- The proposed self-learning mechanism improves the generalization of the model and is also capable of handling imbalanced data.
- We propose sensor self-calibration solution in IoT networks which can automatically calibrate a faulty sensor in few seconds, using historical data of the same sensor in healthy condition. This eliminates the necessity of additional redundant sensors in the vicinity for calibration.
- The proposed solution is robust as it can work efficiently even in a noisy environment.

The rest of the paper is organized as follows. A brief review of the literature survey has been stated in Sec. II, the problem formulation is given in Sec. III, the method used for fault identification and self-calibration are discussed in Section IV and the results and discussion are explained in Sec. V. Finally, the entire paper is summed up in Sec. VI.

II. Related Works and Research Gap

The detection and calibration of sensor faults in IoT sensor nodes is essential to ensure proper health monitoring of the different associated systems in an industry. Generally, the methods for sensor fault detection are either model-based or data-driven. Model-based algorithms such as Kalman Filter (KF) [7], hidden Markov model [8], Canonical Correlation Analysis (CCA) [9] etc. have been used to detect the various sensor faults. But, these complex methods require detailed system knowledge that are not possible to implement for sensor fault detection in the SNs.

To mitigate the pitfalls of model-based techniques, data-driven-based methods have gained popularity in recent years for fault detection. An unsupervised feature selection method [10] was proposed for automatic sensor fault detection. A Hybrid Ensemble Learning-based fault diagnosis method for high-g accelerometers used in the aircraft was proposed in [11], which gave 100% accuracy. [12] used Multi-layer perceptrons (MLPs) and Digital twins for sensor fault detection, isolation, and accommodation. But, all of these works did not identify the different types of sensor faults. Random Forest (RF)-based sensor drift fault detection was performed using Discrete Cosine Transform (DCT) for feature extraction [13], with a very high accuracy of 99%. These methods give high accuracy but require additional computation for feature extraction. Aircraft sensor fault detection was done using Transfer Learning, giving 86.3% accuracy [14]. Deep belief network (DBN) has been used for sensor fault classification in thermocouples of nuclear power plants [15]. However, in most of these methods, the classifier directly maps the input with the fault type. Hence, these methods are static, with reduced generalization and intelligence. This problem can be resolved by using DRL, which resembles human learning by interacting with the environment [16] [17]. Additionally, none of the discussed methods consider the automatic calibration of faulty sensors.

TABLE I: Comparison of existing solutions with SNRepair

Method	DD	FT	Calib	Auto	Robust
Unsupervised Learning [10]	B	NS	×	✓	×
Hybrid Ensemble Learning [11]	B	NS	×	✓	×
RF [13]	Imb	Drift	×	✓	×
Transfer Learning [14]	B	Drift & Bias	×	×	×
Multisensor Data Fusion [18]	B	None	✓	×	×
Long Short-term Memory [19]	B	Drift & Bias	✓	×	×
Category-Based Calibration [20]	B	None	✓	✓	×
UKF [21]	B	Drift	✓	×	×
DRL (Current paper)	**Imb**	**Drift, Bias, PD, CF**	✓	✓	✓

DD: Data Distribution; **FT:** Fault Types; **Calib:** Calibration; **Auto:** Automatic; **B:** Balanced; **Imb:** Imbalanced; **NS:** Not Specified

The traditional process of sensor calibration is based on the requirement of a standard instrument. A multi-sensor data fusion-based calibration that needs specialized sensor hardware was discussed in [18]. The calibration of faulty sensors using the correct readings from other sensors measuring the correlated parameters has been performed in [19] and [22]. Finite State Machine (FSM) can also be used for sensor calibration [23], but the calibration process can begin only when the standard sensor is inserted manually. Calibration of low cost ozone sensors using a standard mobile *buddy* has been performed by Miskell *et al.* [24]. But, this method requires additional hardware and manual intervention for successful calibration. A data-driven architecture was also proposed for sensor calibration using the virtual sensors [1], but this method is very complex to implement. The use of redundant sensors has also been considered for calibration [3], but there is the possibility of simultaneous failures in all the nearby

sensors due to the same harsh environmental conditions. A calibration method using Unscented KF (UKF) was proposed to eliminate the requirement of redundant sensors [21], but this method is difficult to implement in real-time. Hence, a DRL-based technique, *SNRepair*, has been proposed to identify the different sensor faults and automatically calibrate the faulty sensor. The comparative summary of the related works with the proposed *SNRepair* method is given in Table I.

A. Problems Addressed in the Current Paper

The health of a sensor in a SN plays a vital role in the health monitoring of any systems. Most existing sensor fault detection models are pre-trained and do not allow adaptive and generalized learning that can be deployed for any sensor in any working condition. Further, these faults need to be rectified automatically for an uninterrupted supply of accurate sensor data. Most existing methods require additional hardware or redundant sensors. Hence, this paper has attempted to address the challenge of detecting and correcting all types of sensor faults in IoT SNs.

III. Problem Formulation

The proposed architecture considers a scenario where multiple sensor nodes are deployed across a factory to monitor the health of the different systems and subsystems. Consider a sensor node with a sensor S. The data recorded by the sensor S is represented as $y_S(i), i = 1, 2, ..., N$, where N represents the number of collected sensor readings. The measured sensor values may or may not follow a Gaussian distribution, as per the process characteristics. However, in this paper, the Gaussian distribution has been considered with mean μ_S and covariance matrix Σ_S. For a healthy sensor, the data measured $y_{healthy}(i)$ can be expressed as:

$$y_{healthy}(i) = y_S(i) + e(i), \tag{1}$$

where $e(i)$ is the noise caused by external factors. Sensors are susceptible to faults due to harsh working conditions. From a statistical point of view, the sensor faults drift, bias and CF affect the mean of the recorded signal but have no effect on the variance/covariance. However, the PD fault influences the variance/covariance of the measured signal [9].

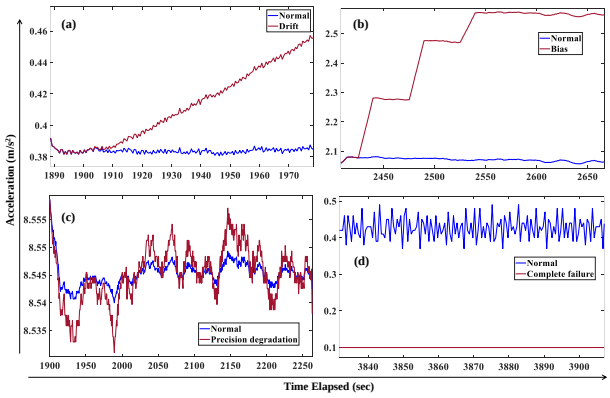


Fig. 2: The different faults occurring in a sensor: (a) Drift (b) Bias (c) PD (d) CF.

The sensor drift fault can be defined as a time-varying offset, that is, the sensor measurement deviates from the actual calibration with time [25] [26]. Mathematically, the drift fault $f_d(t)$ is represented as [27]:

$$f_d(t) = \begin{cases} 0 & \text{for} \quad t < T_f \\ \alpha(t - T_f) & \text{for} \quad t \geq T_f \end{cases}, \tag{2}$$

where T_f is the time when fault occurrence starts and α is the drift rate. The sensor bias fault occurs due to a constant offset from the actual value, causing an abrupt alteration in measurement. The sensor bias fault $f_b(t)$ is given by [27]:

$$f_b(t) = \begin{cases} 0 & \text{for} \quad t < T_f \\ \beta & \text{for} \quad t \geq T_f \end{cases}, \tag{3}$$

where β represents the bias constant. It is to be noted that f_d and f_b are the absolute values of the additive components to the signal $y_{healthy}$. The values of α and β generally vary with time [26]. However, in this paper, we have considered a simplified fault model [27], where the values of α and β are kept constant. Complete failure or freeze fault of the sensor occurs when the sensor stops responding, and a constant value is obtained as the output [28]. When a sensor undergoes CF, the data collected y_{cf} can be represented as:

$$y_{cf}(i) = c, \tag{4}$$

where c is a constant. The precision degradation of a sensor may occur due to aging and wear, and it is generated from a probabilistic distribution [29].

$$y_{pd}(i) = y_{healthy}(i) + f(i)\xi_j, \tag{5}$$

where $f(i)$ is random denoting precision degradation with zero expectation and δ^2 variance, i.e., $f(i) \sim \mathcal{N}(0, \delta^2)$. ξ_j is a unit norm vector. It has been assumed that $f(i)$ is independent of $y_S(i)$. Fig. 2 depicts the signals for four fault conditions.

IV. Methodology

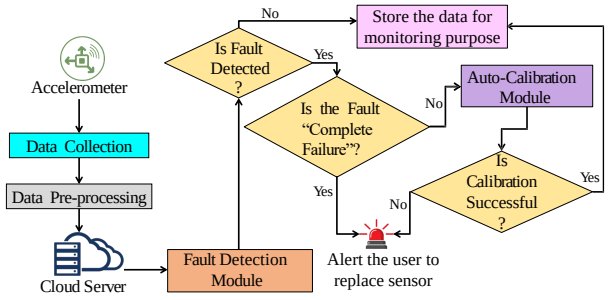


Fig. 3: The process-flow for *SNRepair*.

A. Data Acquisition and Pre-processing

The vibration data is collected by mounting an MPU-6050 sensor and NodeMCU on a motor, as shown in Fig. 4. The motor is newly installed, and hence it can be assumed that it is running in healthy condition. The MPU-6050 is a 6-axis motion tracking device, combining 3-axis gyroscope, 3-axis accelerometer and embedded temperature sensor. It operates on a supply voltage 2.3–3.4 V. The user-programmable accelerometer can operate at a full-scale range of ±2g, ±4g, ±8g, and ±16g, with a calibration tolerance of ±3%.

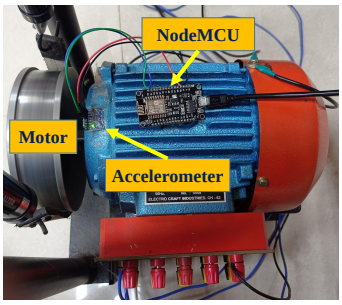


Fig. 4: The experimental setup for data acquisition.

The recorded data is then injected with different faults using the Eq. (2) - (5) to create a dataset with imbalanced distribution of the faults. In order to achieve better results from the reinforcement learning model, the dataset is normalized using the Min-max scaling. The normalized value corresponding to $y(i)$ is y_{norm}, as shown in the equation:

$$y_{norm}(i) = \frac{y(i) - y_{min}}{y_{max} - y_{min}}, \tag{6}$$

where y_{min} and y_{max} refer to the minimum and maximum values of the feature y, respectively.

B. Imbalanced classification using DRL

The fault detection for the sensors deployed in an IoT sensor node can be performed using DRL. The classification problem is taken as a guessing game, in which the agent classifies a received sample to the category it guessed. If the category is correct, then the agent gets an immediate positive reward, otherwise it receives a negative reward, and then receives the next sample. The agent becomes capable of detecting the faults after getting maximum accumulated rewards by interacting with the environment [6]. In real-life scenarios, the faulty data is found to be much less compared to any healthy data, giving rise to the data imbalance issue. If this issue is not resolved, the fault detection model can produce biased results. For this purpose, the Imbalanced Classification Markov Decision Process (ICMDP) framework is used to convert the imbalanced classification problem into a sequential decision-making problem. For better identification of the minority classes, the agent receives a larger positive or negative reward when encountering any minority sample. The reward value for each class is based on the imbalance ratio. If the ratio of the number of samples in a particular class l_p to the total number of samples in the given dataset is λ_p, then the value of the reward Λ_p is taken as:

$$\Lambda_p = 1 - \lambda_p, \tag{7}$$

where λ_p, Λ_p ϵ [0, 1]. Let us consider the state of the training environment as s_t, corresponding to the sample y_t. The agent takes action a_t to determine the sensor health condition correctly. The set of actions taken by the agent is represented as $\mathcal{A}$ = {Normal, Drift, Bias, PD, CF}. Here, 'Normal' is the majority class, whereas the rest are minority classes. If the class label is predicted as l_t, then the reward function R can be defined as:

$$R(s_t, a_t, l_t) = \begin{cases} +\Lambda, & a_t = l_t \quad \text{and} \quad s_t \in D \\ -\Lambda, & a_t \neq l_t \quad \text{and} \quad s_t \in D \end{cases} \tag{8}$$

Here, D represents the entire training dataset. The number of training samples for normal, drift, bias, PD and CF comprises of 62%, 8%, 16%, 12% and 2% of the entire dataset, respectively. Hence, the value of Λ for these classes become 0.38, 0.92, 0.84, 0.88 and 0.98, respectively, that is, the reward value is much higher for the minority classes compared to the majority class.

In ICMDP, the transition probability $p(s_{t+1}|s_t, a_t)$ is deterministic, i.e., the agent goes from state s_t to s_{t+1} as per the sample sequence in the training dataset. The classification policy (π) receives a sample and returns the probability of occurrence of all class labels, as given in:

$$\pi(a|s) = p(a_t = a|s_t = s). \tag{9}$$

Since the agent gets a positive reward when it classifies the sensor health condition correctly, hence it can achieve the maximum cumulative reward g_t as [30]:

$$g_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}, \tag{10}$$

where $\gamma : \gamma \in$ [0,1] is the discount factor, which maintains the balance between the immediate and the future rewards. γ = 0 indicates that the agent takes only immediate reward into consideration. Higher values of γ denote that the agent is more concerned about the future rewards. The Q-function in RL helps in calculating the state-action combination. It follows the Bellman equation, as given by:

$$Q^{\pi}(s, a) = E_{\pi}[r_t + \gamma Q^{\pi}(s_{t+1}, a_{t+1})|s_t = s, a_t = a]. \tag{11}$$

The agent achieves its goal of maximizing the cumulative rewards by solving the optimal Q^* function. The optimal classification policy π^* obtained from the greedy policy under the optimal Q^* function is given as:

$$\pi(a|s) = \begin{cases} 1, & \text{if } a = \arg\max_a Q^*(s, a) \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

From the Eq. (11) and (12), the optimized Q^* function can be expressed as:

$$Q^*(s, a) = E_{\pi}[r_t + \gamma \max_a Q^*(s_{t+1}, a_{t+1})|s_t = s, a_t = a]. \tag{13}$$

For high-dimensional continuous state space, the Q function is integrated with neural networks to form the deep Q-learning function, where the agent performs random sampling to form mini-batches. Then, the gradient descent is done to minimize the losses of the deep Q-function, thereby maximizing the cumulative rewards obtained. The proposed DRL network has two FullyConnected (FC) layers and one Softmax layer for effective sensor fault classification. Both FC layers have 128 nodes with activation function *ReLU*. The final Softmax layer has five hidden nodes.

C. Automated self-calibration of faulty sensors

In *SNRepair*, we created a method for the automatic calibration of a faulty sensor. This can be implemented by associating each sensor deployed in the factory floor with a Calibration Table. So, the 3-axis accelerometer S_a is associated the

Calibration Table T_a. At the beginning, S_a does not need calibration and T_a is empty; hence samples a_x, a_y and a_z from S_a is filled into T_a. When a fault is identified in the Sensor S_a, the *SNRepair* automatically calibrates the S_a, using the healthy historical data stored in T_a. The Calibration Table T_a is then modified by the historical data to be used in the Calibration method.

The self-calibration method depends on the error calculated between the samples acquired from S_a and T_a. To calibrate the data obtained from x-axis direction, we use the Eq. (14)

$$\delta_x(t_x) = \left| \frac{a_x - t_x}{t_x} \right|, \tag{14}$$

where the simultaneous samples from S_a and T_a are denoted by a_x and t_x respectively. A maximum error of 2% is allowed between the correct and incorrect sensor readings [31]. For error values less than or equal to 2%, the *Steady* state is achieved. If it is greater than 2%, then the *Transient* state is attained and the Calibration Table T_a gets modified by t_x. This process is detailed in the Algorithm 1.

Algorithm 1 : *Self-Calibration Algorithm*

1: Accept the Calibration Request for S_a with Sensor ID
2: Retrieve historical data of S_a from T_a
3: **for** ($\forall$ a in T_a) **do**
4: Set State = Initial
5: Acquire sample t_x from T_a
6: Calculate **Error** using Eq. (14)
7: **if** **Error** $\leqslant 2\%$ **then**
8: Set State = Steady.
9: **else**
10: $a_x \leftarrow t_x$.
11: Set State = Transient.
12: **end if**
13: **if** State = Steady **then**
14: Send the Message: Calibration Successful
15: **else**
16: Send the Message: Calibration Failed
17: **end if**
18: **end for**

For a particular sample, the multiple iterations of the statements from Step 6 - 12 can be done for greater efficiency. However, the computation time rises with increased number of iterations. Hence, it has been experimentally determined that the repetition is to be allowed only twice for the performance optimization of the Self-Calibration module [23]. The sensor is successfully calibrated if the final state is *Steady*; otherwise, the calibration fails.

V. Results and Discussion

A. Fault Detection

The effectiveness of the proposed sensor fault detection method can be analysed using the various performance metrics. Since the dataset is imbalanced, the G-mean and the F-measure metrics are utilized for better evaluation of the proposed algorithm. The G-mean is the Geometric mean of sensitivity and precision, that measures the balance between classification performances on both the majority and minority classes. F-measure is the harmonic mean of precision and recall, that evaluates the model's performance in a single metric. The G-mean and the F-measure metrics are mathematically denoted as:

$$\text{G} - \text{mean} = \sqrt{\frac{TP}{TP+FN} \times \frac{TN}{TN+FP}} \tag{15}$$

$$\text{F} - \text{measure} = \sqrt{\frac{TP}{TP+FN} \times \frac{TP}{TP+FP}}, \tag{16}$$

where TP, TN, FP, and FN refer to True Positive, True Negative, False Positive, and False Negative, respectively. Higher values of Overall Accuracy (OA), G-mean and F-measure indicate better performance of the proposed algorithm. The performance metrics of the proposed fault identification method are given in Table II. Apart from the metrics for the entire dataset, Table II also shows the evaluation metrics for the individual fault types.

TABLE II: Model performance for individual fault classes

Fault Type	Accuracy	G-mean	F-measure
Overall	96.17%	94.89%	90.67%
Drift	98.96%	98.98%	99.34%
Bias	100%	100%	100%
Precision degradation	86.12%	81.97%	63.58%
Complete failure	100%	100%	100%

One important metric for performance diagnosis is the OA for all the classes of sensor faults, as given by:

$$\text{OA} = \frac{\sum_{i=1}^{M} x_{correct}^{(i)}}{N}, \tag{17}$$

where $x_{correct}^{(i)}$ denotes the correctly classified sample size of the category i and N is the total sample size.

The proposed DRL-based sensor fault detection approach takes 7.908 seconds to execute on NVIDIA GTX 1660-ti 6GB Dedicated V-RAM and Intel Core-i5 Quad Core processor. The energy consumption is 0.000014 kWh with CO_2 equivalent of 0.008737 grams. This is equal to approximately 0.00061 Km traveled by car. These parameters have been calculated using the python package Carbon Tracker. The model performs self-learning by maximizing the cumulative rewards. The effect of tuning three hyper-parameters on the model performance is discussed in this paper.

1) Tuning the training episodes: One of the important hyper-parameters of RL is *Episodes*, which shows the trajectory adopted by the agent from the initial to the final state. To understand the effect of this parameter on the proposed model performance, the method is executed multiple times on the same dataset, using different *episodes*. The change in OA for different *episodes* is shown in Fig. 5.

The values of OA increase with the increased *episodes*, i.e., the classification accuracy and the diagnosis stability of the proposed model improves with increased *episodes*. It obtains the OA of 95.63% when *episodes* is 700. After that, the OA values become almost stable, thereby indicating that *episodes* 700 is the optimized value of this hyper-parameter. The change in accuracy for the different fault classes is also indicated in

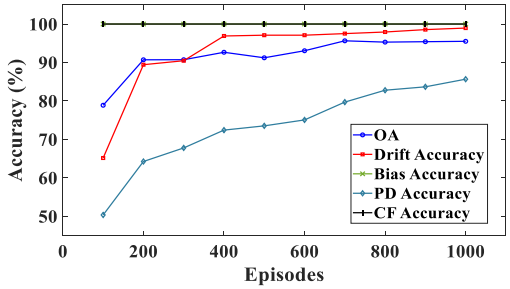


Fig. 5: Effect of different episode size on the accuracy of the proposed model.

Fig. 5. The accuracy of these subsets also increase with the increment of *episodes*. In general, a larger *episode* improves the agent's interaction with the environment and raises the accuracy of the model. However, larger *episode* increases the computational complexity and the latency of the model.

2) Tuning the γ parameter: The discount factor γ is another important hyper-parameter of the DRL model. The effect of γ on the performance of the proposed model is verified by repeatedly executing the proposed model, by changing the γ parameter from 0 to 0.9. The effect of different γ values on the OA and the individual fault classes is shown in Fig. 6.

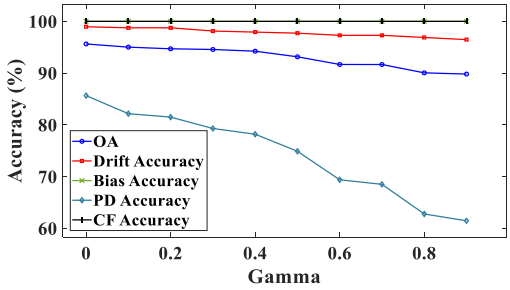


Fig. 6: Effect of different γ values on the accuracy of the proposed model.

From Fig. 6 it is clear that the accuracy of the model decreases with the increase in the discount factor γ. The OA of the method is found to be 95.63%, when $\gamma = 0$, and it is reduced by 5.82% when $\gamma = 0.9$. The accuracy of the other fault classes also follow similar trends. The PD suffers from the sharpest fall in accuracy value. The accuracy for PD is reduced by 24.2% when the value of γ changes from 0 to 0.9. This occurs because there is very less correlation between two adjacent states for a classification problem. Hence, the agent is more concerned about the immediate rewards compared to the future rewards.

3) Tuning the nodes in FC layers: It is important that the fault diagnosis model must have low complexity along with high accuracy. The model complexity can be reduced by using less number of hidden nodes in the FC1 and FC2 layers. The effect of the node numbers in the FCs on the performance of the sensor fault detection model is determined by executing the proposed model by taking the number of nodes as 32, 64, 128, 256, 512 and 1024. It must be noted that the number of nodes is taken the same for both the FC layers. The γ value is taken as 0.0, and the number of episodes is 700. The effect of different node numbers on the OA and the individual fault classes is depicted in the Fig. 7.

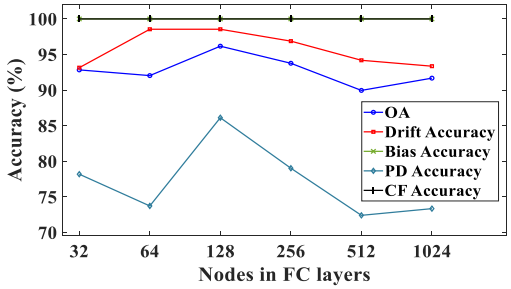


Fig. 7: Effect of different FC1 and FC2 nodes on the accuracy of the proposed model.

From Fig. 7, it is understood that the OA of the proposed model is 96.17% when the number of hidden nodes in both FC1 and FC2 are 128. Moreover, due the self-learning process, the model consumes more time for interacting with the environment. The average time taken for training the model is 6 seconds. The accuracy of the individual fault classes also achieve high values when the number of nodes is 128 for the FC layers.

B. Self-Calibration

In this work, four different sensor faults are injected in different sections of the healthy sensor data to test the efficiency of the self-calibration module. The faults have been injected in all three axis of the accelerometer data, with varying fault rate. After the fault detection module detects a fault, the self-calibration of the faulty sensor starts automatically. If the fault detected is complete failure, then the self-calibration module is not executed and the sensor needs to be replaced. For the other three sensor faults, the sensor provides the correct data readings. Fig. 8 (a), (c) and (e) shows how the calibration has been performed efficiently for three different faults in three different axis, within a very short time. The faulty sensor data when no calibration is done is shown by the dotted lines. The Fig. 8 (b), (d) and (f) depicts the absolute error (AE) between the healthy sensor data and the sensor data after calibration. It can be observed from the graphs that the values of AE are very low, thereby establishing the closeness of the actual and calibrated data.

Table III depicts that the time needed by the proposed self-calibration module for different fault types and fault rates. The algorithm takes few seconds to successfully calibrate the faulty sensors, and the time taken decreases with rise in the fault rate.

TABLE III: Time required by the self-calibration module for successful calibration of the faulty sensors

Fault	Fault Rate	Time taken for Calibration (s)
Drift (x-axis)	0.0001	19
Drift (y-axis)	0.0005	15
Drift (z-axis)	0.0003	16
Bias (x-axis)	0.2	15
Bias (y-axis)	0.11	9
Bias (z-axis)	0.17	14
PD (x-axis)	1.3	32
PD (y-axis)	1.1	24
PD (z-axis)	1.5	41

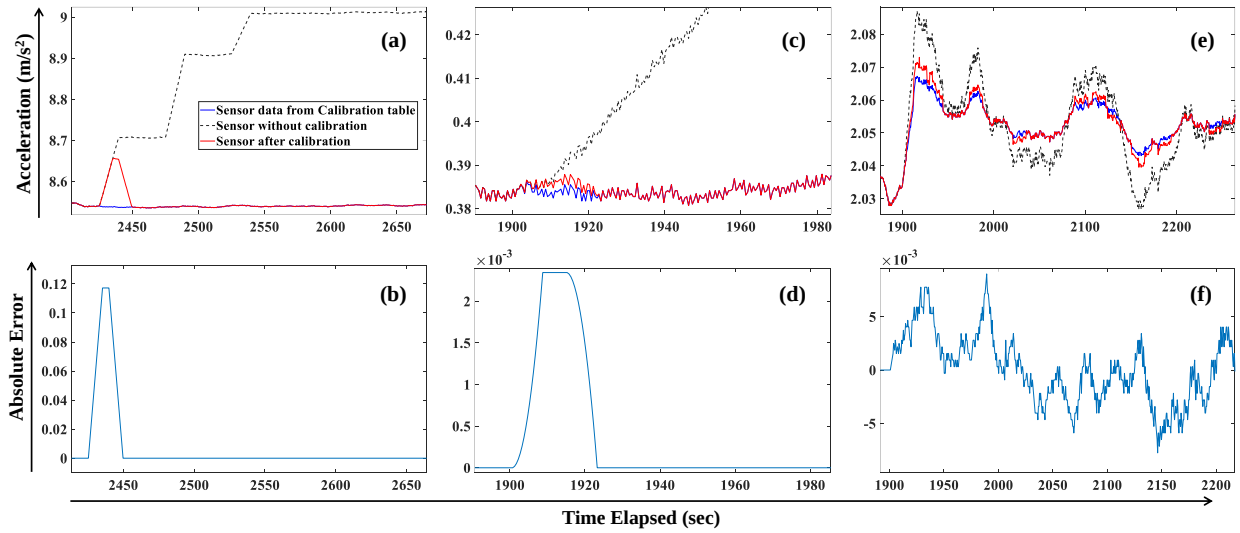


Fig. 8: Successful Calibration and AE for (a)-(b) bias fault in z-axis, (c)-(d) drift fault in y-axis, and (e)-(f) PD in x-axis of the 3-axis accelerometer, respectively.

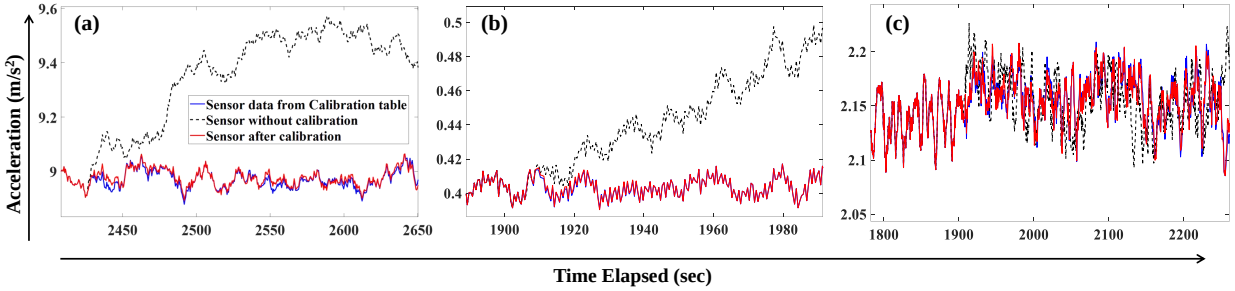


Fig. 9: Calibration of noisy data for (a) bias in z-axis (b) drift in y-axis, and (c) PD in x-axis of the 3-axis accelerometer.

TABLE IV: Model performance for individual fault classes considering the noisy data

Fault Type	Accuracy	G-mean	F-measure
Overall	92.27%	91.35%	84.74%
Drift	93.14%	90.56%	91.24%
Bias	98.62%	99.22%	99.21%
Precision degradation	76.35%	76.78%	58.96%
Complete failure	100%	100%	100%

C. Effect of noise on the proposed SNRepair

The health monitoring of the sensors is made more difficult due to the presence of environmental noise. In the factory floor, noise may generate due to the running of a nearby machine or any maintenance or transportation work occurring in the vicinity. To test the effectiveness of the proposed *SNRepair* technique, the Additive White Gaussian Noise (AWGN) having zero mean and 0.2 standard deviation is applied to the data collected from all three axis of the accelerometer. The proposed DRL model is again applied to the entire noisy dataset. The performance metrics for the proposed fault detection model are given in Table IV. Although the OA is less for noisy data compared to that of the noise-less data, the model gives good performance with acceptable accuracy. The reduced accuracy occurs because noisy data is often mistaken as the degraded one.

The self-calibration algorithm is executed using the noisy data. Examples of successful calibration of three sensor faults are depicted in Fig. 9. The closeness between actual and calibrated data establishes the robustness of the proposed method.

D. Comparison with the Existing Literature

The proposed *SNRepair* technique utilizes its uniqueness to overcome the drawbacks of the existing works, which are generalized model for all sensor faults identification and the requirement of additional hardware for calibration. Although some literature deal with the sensor fault identification [9] [13] [14], they do not consider all types of sensor faults. However, our proposed *SNRepair* uses self-learning using DRL model to identify all four types of sensor faults with 96.17% accuracy, which is an acceptable outcome. Moreover, many of the existing works do not consider calibration of the faulty sensors. The few existing studies on sensor calibration require complex calculations [19], specialized hardware [18], or redundant sensors [3] for a successful calibration. These challenges are overcome using the historical healthy data of the same sensor for automatic self-calibration. Thus, the *SNRepair* is well-suited for efficient sensor fault identification and self-calibration in the SNs deployed in industries.

VI. Conclusion and Future Scope

This paper proposes a unique *SNRepair* technique with automatic sensor health monitoring and self-calibration capability of the LCSs placed in the IoT-SNs for accurate monitoring

of machine health in smart industry. The major conclusions of this paper are listed as follows: (1) A generalized self-learning-based model has been proposed that can detect all types of sensor faults in any working condition with 96.17% accuracy. (2) A simple self-calibration algorithm has been presented in this paper that uses its own historical healthy data for calibration without any additional hardware requirement. (3) The proposed method works fine even in noisy conditions, establishing its robustness. Thus, it can be confirmed that the model is accurate and efficient enough to be implemented for sensor health diagnosis in the SNs deployed in smart industries. However, there are some areas of improvement for the proposed model. For example, the model can be optimized so that it can be light-weight enough to be deployed on the edge to reduce latency. Moreover, complex fault models with time-varying drift and bias constants can also be taken into account as the future scope. Further, the scenario where both system and sensor faults occur at the same time can be considered for future work.

REFERENCES

[1] H. Darvishi, D. Ciuonzo, E. R. Eide, and P. S. Rossi, "Sensor-Fault Detection, Isolation and Accommodation for Digital Twins via Modular Data-Driven Architecture," *IEEE Sensors J.*, vol. 21, no. 4, pp. 4827–4838, 2021.

[2] G. Kaur and P. Chanak, "An Energy Aware Intelligent Fault Detection Scheme for IoT-Enabled WSNs," *IEEE Sensors J.*, vol. 22, no. 5, pp. 4722–4731, 2022.

[3] A. Sinha and D. Das, "sCalib: A Warehouse Sensor Fault Detection and Self-Calibration Technique for Sustainable IoT," in *2021 IEEE 18th India Council International Conference (INDICON)*, 2021, pp. 1–6.

[4] B. Maag, Z. Zhou, and L. Thiele, "A Survey on Sensor Calibration in Air Pollution Monitoring Deployments," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4857–4870, 2018.

[5] K. S. H. Ong, D. Niyato, and C. Yuen, "Predictive Maintenance for Edge-Based Sensor Networks: A Deep Reinforcement Learning Approach," in *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*. IEEE, 2020, pp. 1–6.

[6] Y. Ding, L. Ma, J. Ma, M. Suo, L. Tao, Y. Cheng, and C. Lu, "Intelligent fault diagnosis for rotating machinery using deep Q-network based health state classification: A deep reinforcement learning approach," *Advanced Engineering Informatics*, vol. 42, p. 100977, 2019.

[7] A. Vafamand, B. Moshiri, and N. Vafamand, "Fusing unscented kalman filter to detect and isolate sensor faults in dc microgrids with cpls," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–8, 2021.

[8] C. Alippi, S. Ntalampiras, and M. Roveri, "Model-free fault detection and isolation in large-scale cyber-physical systems," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 1, no. 1, pp. 61–71, 2016.

[9] Z. Chen, C. Yang, T. Peng, H. Dan, C. Li, and W. Gui, "A cumulative canonical correlation analysis-based sensor precision degradation detection method," *IEEE Trans. Ind. Electron.*, vol. 66, no. 8, pp. 6321–6330, 2018.

[10] H. Y. Teh, K. I.-K. Wang, and A. W. Kempa-Liehr, "Expect the Unexpected: Unsupervised Feature Selection for Automated Sensor Anomaly Detection," *IEEE Sensors J.*, vol. 21, no. 16, pp. 18 033–18 046, 2021.

[11] J. Wen, H. Yao, Z. Ji, B. Wu, and M. Xia, "On Fault Diagnosis for High-g Accelerometers via Data-Driven Models," *IEEE Sensors J.*, vol. 21, no. 2, pp. 1359–1368, 2020.

[12] H. Darvishi, D. Ciuonzo, and P. S. Rossi, "Real-Time Sensor Fault Detection, Isolation and Accommodation for Industrial Digital Twins," in *2021 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, 2021, pp. 1–6.

[13] M. N. Hasan and I. Koo, "Machine Learning-Based Sensor Drift Fault Classification using Discrete Cosine Transform," in *2021 International Conference on Electronics, Communications and Information Technology (ICECIT)*, 2021, pp. 1–4.

[14] X. Li and X. Kong, "Aircraft sensor Fault Diagnosis Method Based on Residual Antagonism Transfer Learning," in *2021 IEEE International Conference on Artificial Intelligence and Industrial Design (AIID)*, 2021, pp. 469–472.

[15] S. Mandal, B. Santhi, S. Sridhar, K. Vinolia, and P. Swaminathan, "Nuclear power plant thermocouple sensor-fault detection and classification using deep learning and generalized likelihood ratio test," *IEEE Trans. Nucl. Sci.*, vol. 64, no. 6, pp. 1526–1534, 2017.

[16] H. Wang, J. Xu, C. Sun, R. Yan, and X. Chen, "Intelligent fault diagnosis for planetary gearbox using time-frequency representation and deep reinforcement learning," *IEEE/ASME Trans. Mechatronics*, vol. 27, no. 2, pp. 985–998, 2021.

[17] Y. Chang, J. Chen, W. Wu, T. Pan, Z. Zhou, and S. He, "Intelligent Fault Quantitative Identification for Industrial Internet of Things (IIoT) via a Novel Deep Dual Reinforcement Learning Model Accompanied with Insufficient Samples," *IEEE Internet Things J.*, 2022.

[18] P. Ferrer-Cid, J. M. Barcelo-Ordinas, J. Garcia-Vidal, A. Ripoll, and M. Viana, "Multisensor Data Fusion Calibration in IoT Air Pollution Platforms," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3124–3132, 2020.

[19] P. Han, H. Mei, D. Liu, N. Zeng, X. Tang, Y. Wang, and Y. Pan, "Calibrations of Low-Cost Air Pollution Monitoring Sensors for CO, $NO_2, O_3, and SO_2,''$ *Sensors*, $vol.\ 21, no.\ 1, p.256, 2021.$

[20] R. Wang, Q. Li, H. Yu, Z. Chen, Y. Zhang, L. Zhang, H. Cui, and K. Zhang, "A Category-Based Calibration Approach With Fault Tolerance for Air Monitoring Sensors," *IEEE Sensors J.*, vol. 20, no. 18, pp. 10 756–10 765, 2020.

[21] D. Chowdhury, A. Melin, and K. Villez, "Automatic Drift Correction through Nonlinear Sensing," in *2021 Resilience Week (RWS)*. IEEE, 2021, pp. 1–6.

[22] K. Yamamoto, T. Togami, N. Yamaguchi, and S. Ninomiya, "Machine learning-based Calibration of Low-cost Air Temperature Sensors using Environmental Data," *Sensors*, vol. 17, no. 6, p. 1290, 2017.

[23] Y.-W. Lin, Y.-B. Lin, and H.-N. Hung, "CalibrationTalk: A Farming Sensor Failure Detection and Calibration Technique," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6893–6903, 2020.

[24] G. Miskell, K. Alberti, B. Feenstra, G. S. Henshaw, V. Papapostolou, H. Patel, A. Polidori, J. A. Salmond, L. Weissert, and D. E. Williams, "Reliable data from low cost ozone sensors in a hierarchical network," *Atmospheric Environment*, vol. 214, p. 116870, 2019.

[25] D. Li, Y. Wang, J. Wang, C. Wang, and Y. Duan, "Recent advances in sensor fault diagnosis: A review," *Sensors and Actuators A: Physical*, vol. 309, p. 111990, 2020.

[26] G. Li, H. Hu, J. Gao, and X. Fang, "Dynamic Calibration Method of Sensor Drift Fault in HVAC System Based on Bayesian Inference," *Sensors*, vol. 22, no. 14, p. 5348, 2022.

[27] X. Dai, F. Qin, Z. Gao, K. Pan, and K. Busawon, "Model-based on-line sensor fault detection in Wireless Sensor Actuator Networks," in *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, 2015, pp. 556–561.

[28] Z. Yang, K. B. Rasmussen, A. T. Kieu, and R. Izadi-Zamanabadi, "Fault detection and isolation for a supermarket refrigeration system–part one: Kalman-filter-based methods," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 13 233–13 238, 2011.

[29] R. Dunia, S. J. Qin, T. F. Edgar, and T. J. McAvoy, "Identification of faulty sensors using principal component analysis," *AIChE Journal*, vol. 42, no. 10, pp. 2797–2812, 1996.

[30] E. Lin, Q. Chen, and X. Qi, "Deep Reinforcement Learning for Imbalanced Classification," *Applied Intelligence*, vol. 50, no. 8, pp. 2488–2502, 2020.

[31] Y. Wang, C. J. Simonson, R. W. Besant, and W. Shang, "Transient Humidity Measurements—Part I: Sensor Calibration and Characteristics," *IEEE Trans. Instrum. Meas.*, vol. 56, no. 3, pp. 1074–1079, 2007.