Pakistan Journal of Life and Social Sciences
www.pjlss.edu.pk

https://doi.org/10.57239/PJLSS-2025-23.1.00351

RESEARCH ARTICLE

Robust Multiclass Fault Detection in Wireless Sensor Networks Using Feature Engineering and Residual Neural Networks

Ridha Mohammed Alfoudi[1*], Mohsen Nickray[2]

[1,2] Department of Computer Engineering and InformationTechnology, University of Qom Qom , Iran

ARTICLE INFO

Received: Nov 16, 2024

Accepted: Jan 28, 2025

Keywords

Wireless Sensor Networks
Fault Detection
Residual Neural Network
Chatterjee Correlation
Neighborhood Component Analysis ResNet
Horse Herd Optimization Algorithm

***Corresponding Author:**

ridha.alfoudi@gmail.com

ABSTRACT

Wireless Sensor Networks (WSNs) are prone to various types of faults that compromise their reliability and performance, necessitating accurate and efficient fault detection methods. This paper proposes an enhanced WSN fault detection approach leveraging a lightweight Residual Neural Network (ResNet) architecture. The method begins with data preprocessing, where redundant information is eliminated using Chatterjee correlation to reduce input space complexity. Feature selection is then performed using Neighborhood Component Analysis (NCA) optimized by the Horse Herd Optimization Algorithm (HOA), ensuring the selection of the most informative features. To utilize the ResNet's capability of capturing spatial relationships, the selected 1D feature vectors are transformed into 2D images via an inverse zigzag scanning technique. A lightweight ResNet model is specifically designed to process these small 2D inputs efficiently, incorporating three residual layer stacks to extract hierarchical spatial patterns. The proposed method achieves a remarkable accuracy of 99.41% in detecting five fault categories: normal condition, offset faults, gain faults, stuck-at faults, and out-of-bounds faults. Unlike traditional binary classifiers that detect faults separately, leading to potential conflicts, this approach performs holistic multiclass classification, ensuring reliability and consistency in predictions. Additionally, the lightweight architecture addresses the computational constraints of WSN environments. By integrating redundancy reduction, optimized feature selection, and a tailored ResNet design, this method offers a significant advancement in WSN fault detection, providing high accuracy, computational efficiency, and robust multiclass classification.

INTRODUCTION

Wireless sensor networks (WSNs) have emerged as a cornerstone of modern monitoring systems, enabling real-time data collection and communication across diverse environments. These networks play a vital role in applications ranging from environmental monitoring and industrial automation to healthcare and security. However, the efficient functioning of WSNs is frequently challenged by faults arising from hardware failures, environmental interferences, communication disruptions, or software anomalies. Accurate fault detection is essential to ensure data reliability, network security, and the longevity of these resource-constrained systems. In recent years, machine learning approaches have garnered significant attention as robust solutions for fault detection in WSNs. These

methods leverage pattern recognition and predictive capabilities to identify anomalies effectively. Despite their potential, existing techniques often grapple with issues such as high-dimensional data processing, scalability, and computational efficiency, particularly in large-scale deployments or diverse fault scenarios. Addressing these challenges demands innovative methods that integrate effective feature selection, optimized classification techniques, and resource-efficient algorithms.

Fault detection in WSNs is a critical challenge due to resource limitations and diverse deployment environments. It is crucial for maintaining data quality, network security, and longevity (Zhang et al., 2018). Machine learning approaches have emerged as promising solutions for this problem. Zidi et al. (2018) investigated the use of support vector machines (SVMs), demonstrating their effectiveness through experimental comparisons with existing methods. The SVM-based approach offered a lightweight decision function suitable for execution on cluster heads. In a more comprehensive study, Noshad et al. (2019) compared multiple classifiers, including SVM, Convolutional Neural Network, Stochastic Gradient Descent, Multilayer Perceptron, Random Forest (RF), and Probabilistic Neural Network. Their analysis, conducted on real-world datasets, evaluated the classifiers' performance based on detection accuracy, true positive rate, Matthews Correlation Coefficients (MCC), and F1-score. The results indicated that the RF algorithm outperformed other classifiers in fault detection for WSNs. The study by Gupta et al. (2019) introduces an improved fault detection technique for wireless sensor networks (WSNs) called the Improved Fault Detection Crow Search Algorithm (IFDCSA). This algorithm addresses the critical issue of faulty data in WSNs, which can lead to system failures. IFDCSA, an enhanced version of the original Crow Search Algorithm, injects faults into datasets and then classifies them using machine learning classifiers. The researchers evaluated IFDCSA on three real-world datasets: Intel lab data, multihop labeled data, and SensorScope data. The proposed algorithm achieved an impressive average accuracy of 99.94% in fault prediction. Jan et al. (2021) proposed a distributed approach using autoencoders and Support Vector Machines for fault detection on sensors, with a Fuzzy Deep Neural Network for diagnosis at a central node. Gnanavel et al. (2022) compared six classifiers for detecting various fault types in WSNs, finding that Random Forest outperformed others in fault detection. Saeed et al. (2021) introduced an Extremely Randomized Trees (Extra-Trees) based method for fault diagnosis, which demonstrated robustness to noise and reduced bias and variance error. This approach was compared to other machine learning algorithms and showed superior performance in terms of accuracy, precision, and F1-score, as well as lower training time. ArulJothi and Venkatesan (2023) proposed a Feed-forward Autoencoder Neural Network (FANN) model for efficient anomaly detection, achieving improved accuracy and reduced energy consumption. Similarly, Atiga et al. (2021) developed a Recurrent Neural Network NARX model for distributed fault detection in WSNs. Karmarkar et al. (2020) introduced an optimized Support Vector Machine (SVM) based fault diagnosis scheme, utilizing Grey Wolf Optimization (GWO) for classifier optimization and a cluster-based topology for energy conservation. Laiou et al. (2019) proposed a decision tree-based machine learning approach for autonomous fault detection and diagnosis, achieving 96.46% accuracy in identifying common faults like connectivity loss and packet loss. To improve fault detection accuracy, Mohapatra & Khilar (2020) proposed an improved negative selection algorithm (INSA) combined with support vector machine (SVM) for fault classification. The INSA-SVM method successfully classified faults into soft permanent, soft intermittent, and soft transient categories.

Recent research has focused on using deep learning techniques for fault detection in WSNs. Deep neural networks, particularly Recurrent Neural Networks (RNNs) and Bidirectional Long Short-Term Memory (Bi-LSTM) models, have shown promising results in detecting various fault types, including soft permanent, intermittent, and transient faults (Mazibuco et al., 2023; Gupta et al., 2021). These approaches outperform traditional machine learning methods in terms of fault detection accuracy, false alarm rate, and false-positive rate (Gupta et al., 2021). Comparative studies have evaluated the performance of deep learning techniques against other machine learning algorithms, such as Support Vector Machine (SVM) and Random Forest (RF), using metrics like Detection Accuracy and True Positive Rate (Azzouz et al., 2020). The application of deep learning methods in WSN fault diagnosis has gained significant interest in both industry and academia, addressing challenges such as sensor

domain knowledge requirements and the need for neighboring sensor data in distributed approaches (Panda et al., 2020). Swain & Khilar (2017) proposed a soft fault diagnosis model using Particle Swarm Optimization (PSO) for classification. Their approach involved three phases: initialization, fault identification using Analysis of Variance (ANOVA), and fault classification using a feed-forward neural network with PSO learning. Kumar et al. (2022) proposed an Improved Deep Convolutional Neural Network for detecting malicious nodes and an Extended K-Means algorithm with t-Distribution based Satin Bowerbird Optimization for energy-efficient data transmission. Biswas et al. (2019) developed a hybrid approach using Kalman filters and Extreme Learning Machines for fault detection, achieving high prediction accuracy with low communication overhead. Javaid et al. (2019) introduced four enhanced classification techniques (EKNN, EELM, ESVM, and ERELM) to improve belief function-based decision fusion and fault detection in WSNs. They induced four types of faults and evaluated the proposed methods using detection accuracy, true positive rate, and error rate metrics. (Regin et al., 2018) introduce a convex hull algorithm to identify extreme points among neighboring nodes, maintaining message duration efficiency as the network scales. They also employ a Naïve Bayes classifier and a convolutional neural network (CNN) to enhance convergence performance and detect node faults. The study evaluates these algorithms using real-world datasets to identify and categorize faults. Simulation and experimental results demonstrate the feasibility and efficiency of the proposed methods. Notably, the CNN algorithm outperforms the convex hull algorithm in fault identification based on various performance metrics. Prasad & Baghel (2023) introduce a deep belief network-based self-detection algorithm that improves detection accuracy and reduces false alarm rates compared to existing approaches. Their method is scalable for large-scale WSNs and reduces energy overhead and detection latency. Similarly, Prasad & Baghel (2022) present a feedforward neural network-based technique where each sensor node detects its own fault status using only its sensed data. This approach eliminates the need for communication with neighboring nodes or base stations, thereby improving energy efficiency and reducing communication overhead and delay.

Sarangi & Tripathy (2023) developed an outlier detection technique using generative adversarial networks (GANs) with autoencoders, demonstrating improved detection accuracy and increased network lifetime compared to existing methods. Similarly, Arul Jothi & Venkatesan (2023) introduced a Feed-forward Autoencoder Neural Network (FANN) model for anomaly detection, which showed enhanced accuracy and reduced energy consumption while minimizing false alarms. These studies highlight the potential of machine learning and deep learning approaches in addressing the challenges of fault and anomaly detection in WSNs, particularly in improving accuracy, reducing energy consumption, and extending network lifetime.

Despite these advancements, many existing approaches still face significant challenges in processing high-dimensional data, computational inefficiencies, and scalability across diverse fault scenarios. These limitations restrict their performance and applicability, especially in complex environments. To address these gaps, this paper introduces an innovative method for fault detection in WSNs. The proposed method utilizes Chatterjee Correlation, specifically designed for Sample Dimension Reduction, to retain critical information while removing irrelevant or less important features. This process identifies nonlinear relationships among samples and selects an optimized subset of data, enabling efficient processing. Subsequently, Neighborhood Component Analysis (NCA) is employed to identify the most influential features that contribute significantly to fault detection accuracy. To enhance NCA's performance and optimize its key hyperparameters, the Horse Herd Optimization Algorithm (HOA) is utilized. HOA evaluates various hyperparameter settings and their impact on model performance, providing optimal configurations. Additionally, Residual Neural Networks (ResNet), a powerful deep learning architecture, is used for classifying the optimized data. ResNet excels in learning complex patterns and nonlinear relationships within the data, enabling robust and reliable classification across diverse fault scenarios. This integration of advanced techniques ensures that the proposed method not only handles high-dimensional data efficiently but also significantly improves fault detection accuracy and scalability in complex and dynamic environments.

The remainder of this paper is organized as follows: Section 2 discusses the fundamental concepts needed for introducing the proposed method. Section 3 details the proposed methodology, including the integration of feature selection techniques and classification frameworks. Section 4 introduces the datasets used for evaluating the proposed method. Section 5 outlines the evaluation metrics needed for evaluating the proposed method. Section 6 presents the results and a comparative analysis of related methods, while Section 7 provides a comparison of the proposed method with other methods in the literature. Finally, Section 8 concludes the paper with potential future research directions.

Basic Concepts

This section offers a detailed explanation of the fundamental principles and key concepts required to understand the proposed method.

Chatterjee correlation coefficient

The Chatterjee correlation coefficient is a measure of association between two random variables that quantifies their dependence while remaining invariant to strictly monotonic transformations. Unlike traditional correlation measures like Pearson's correlation, which captures linear dependence, or Spearman's rank correlation, which measures monotonicity, the Chatterjee correlation coefficient is particularly well-suited for identifying general forms of dependence (Chatterjee et al., 2021).

This concept was first introduced by Sourav Chatterjee in 2020 in the context of developing robust statistical tools to measure dependence with minimal assumptions about the underlying data structure. Chatterjee's work provided a novel approach by employing ranks of observations, making the coefficient computationally efficient and robust to outliers. To calculate the Chatterjee correlation coefficient, consider (X, Y) represent the input dataset, defined as (x_1, y_1), ..., (x_n, y_n), where Y is not constant. The computation of the Chatterjee correlation coefficient is first described for the case where no duplicate values exist in the dataset (Chatterjee et al., 2021).

In this scenario, the input pairs are ordered as $(x_{(1)}, y_{(2)})$, ..., $(x_{(n)}, y_{(n)})$, such that $x(1) \le x(2) \le \cdots \le x(n)$. Since the values in X are unique, there is a single unique order that satisfies this condition. For each i, the parameter r_i , or the rank, is defined as the count of indices j for which $y_{(j)} \le y_{(i)}$. Under these conditions, the Chatterjee correlation coefficient is computed using the following Equation:

$$\xi_n(X,Y) = 1 - \frac{3\sum_{i=1}^{n-1}|r_{i+1}-r_i|}{n^2-1} \quad (1)$$

When duplicate values exist in X, one random ordering of the x_i values, sorted in non-decreasing order as described above, is selected. Using the previously defined r_i, the parameter l_i is introduced, representing the count of indices j such that $y_{(j)} \ge y_{(i)}$. In this case, the Chatterjee correlation coefficient is given by the following Equation:

$$\xi_n(X,Y) = 1 - \frac{n\sum_{i=1}^{n-1}|r_{i+1}-r_i|}{2\sum_{i=1}^{n} l_i(n-l_i)} \quad (2)$$

Horse Herd Optimization Algorithm

The Horse Herd Optimization Algorithm (HOA) is a nature-inspired meta-heuristic algorithm developed to address high-dimensional optimization problems. Inspired by the natural and social behaviors of horses, the algorithm models six key behavioral traits: grazing, hierarchy, sociability, imitation, defense mechanism, and roaming. These traits allow the algorithm to effectively balance exploration (diversifying the search for optimal solutions) and exploitation (intensifying the search around promising areas). In HOA, the movement of each horse is influenced by its age group, which determines its behavior patterns. Four age categories are defined: young (δ), adolescent (γ), adult (β), and old (α), with each group contributing uniquely to the optimization process (MiarNaeimi et al., 2021). The position of each horse in the search space is updated iteratively based on the velocity vector, computed as:

$$X_m^{Iter,AGE} = V_m^{Iter,AGE} + X_m^{Iter-1,AGE} \quad (3)$$

where $X_m^{Iter,AGE}$ is the position, $V_m^{Iter,AGE}$, AGE is the velocity, and AGE denotes the age group.

Velocity Calculation for Different Age Groups: The velocity vector for each horse is determined by combining the behavioral components. For example, the velocity for adolescent horses (γ) is expressed as:

$$V_m^{Iter,\gamma} = G_m^{Iter,\gamma} + H_m^{Iter,\gamma} + S_m^{Iter,\gamma} + I_m^{Iter,\gamma} + D_m^{Iter,\gamma} + R_m^{Iter,\gamma} \quad (4)$$

Grazing Behavior: The grazing motion models the horse's tendency to explore its surroundings and is calculated as:

$$G_m^{Iter,AGE} = g^{Iter,AGE}.(u + Pl)[X_m^{Iter-1}] \quad (5)$$

where u and l are bounds, P is a random factor, and $g^{Iter,AGE}$ decays over iterations.

Defense Mechanism: Horses avoid unfavorable regions based on their defense mechanism, expressed as:

$$D_m^{Iter,AGE} = -d^{Iter,AGE}[\frac{1}{qN}\sum_{j=1}^{qN} X_j^{Iter-1} - X_m^{Iter-1}] \quad (6)$$

where qN represents the subset of poorly performing solutions, and $d^{Iter,AGE}$ decays over time.

Roaming Behavior: Young horses (δ) perform random movements for exploration:

$$R_m^{Iter,AGE} = r^{Iter,AGE} P\ X_m^{Iter-1} \quad (7)$$

where $r^{Iter,AGE}$ is a decay factor.

Neighborhood Component Analysis

Neighborhood Component Analysis (NCA) is a supervised learning algorithm designed for feature selection and dimensionality reduction in classification tasks. Unlike traditional feature selection methods that rely on predefined assumptions about the data, NCA is non-parametric and learns a feature weighting vector directly from the data by optimizing classification performance. The method works by identifying a subset of features that contribute most significantly to improving classification accuracy while minimizing redundancy and noise. By maximizing the expected leave-one-out (LOO) classification accuracy, NCA enhances the performance of machine learning models, reduces computational complexity, and mitigates the risk of overfitting. This adaptability makes NCA a powerful tool for preprocessing high-dimensional datasets in multi-class classification problems (Yang et al., 2012).

In a dataset $S = \{(x_i, x_j),\ i \in 1,2,\ldots,N\}$, where $x_i \in R^d$ is a d-dimensional feature vector and $y_i \in \{1,2,\ldots C\}$ denotes the corresponding class label, NCA computes the weighted distance between two feature vectors x_i and x_j as:

$$D_w(x_i, x_j) = \sum_{k=1}^{d} w_k^2 |x_{ik} - x_{jk}| \quad (8)$$

where w_k represents the weight of the k^{th} feature. These weights are learned to optimize the classification process.

NCA employs a probabilistic framework for selecting reference points during LOO classification. The probability of selecting x_j as a reference for x_i is given by:

$$p_{ij} = \frac{k(D_w(x_i,x_j))}{\sum_{l \neq i} k(D_w(x_i,x_l))} \text{ if } i \neq j \quad (9)$$

where $k(z) = e^{(-z/\sigma)}$ is a kernel function with σ as the kernel width. If $i=j$, the probability p_{ij} is set to zero.

The probability of correctly classifying x_i is defined as:

$$p_i = \sum_j p_{ij} y_{ij} \quad (10)$$

where y_{ij}=1 if $y_i = y_j$, and y_{ij}= 0 otherwise.

Objective Function: To optimize feature selection, the average LOO classification accuracy across all samples serves as the objective function (OF). To prevent overfitting, a regularization term is added, resulting in the following objective function:

$$OF(w) = \sum_i p_i - \lambda \sum_{k=1}^{d} w_k^2 \quad (11)$$

where λ is a regularization parameter that controls the trade-off between maximizing classification accuracy and minimizing overfitting. The parameter λ can be fine-tuned using cross-validation to achieve optimal performance.

The goal of NCA is to maximize OF(w) with respect to w, thereby learning the most effective weights for the features. This process ensures the selection of a subset of features that are most relevant for classification tasks, leading to improved accuracy and reduced computational complexity.

Residual Neural Network

A Residual Neural Network (ResNet) is a type of artificial neural network introduced by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in 2015 to address performance degradation and gradient vanishing issues in deep architectures. ResNet's key innovation is the use of residual blocks, which maintain information flow across layers through *skip connections*. These connections bypass one or more layers by adding the input directly to the output, enabling the network to learn residual functions instead of approximating the desired mapping (Nagpal, Bhinge, & Shitole, 2022; He, Zhang, Ren, & Sun, 2016) (Figure 1). Mathematically, this is expressed as:

$$H(x) = f(x) + x \quad (12)$$

where f(x) is the transformation learned by the block and x is the input. If f(x)=0, the input is directly passed as the output, preventing gradient issues during backpropagation and facilitating the training of very deep networks.

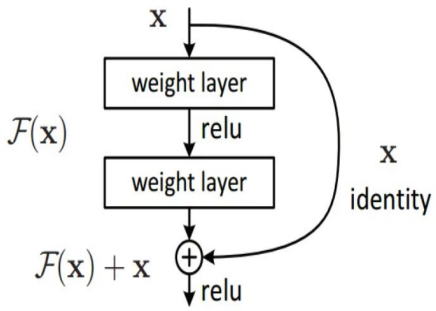

Figure 1. Single Residual block (Nagpal, Bhinge, & Shitole, 2022)

ResNet retains traditional ANN components, such as activation layers and Batch Normalization, but its residual blocks differentiate it by creating shortcut paths for the input. This ensures that deeper networks maintain performance without redundancy or unnecessary transformations, allowing additional layers to enhance learning. To handle dimensional mismatches between inputs and outputs, ResNet uses either zero-padding to align dimensions or a 1x1 convolutional layer within skip connections. These strategies ensure input integrity while enabling deeper layers to learn more abstract features. The skip connections and residual learning mechanism in ResNet have significantly improved the modeling of complex functions and enabled the development of scalable deep architectures like ResNet-50 and ResNet-101. These models are widely used in tasks such as image classification and object detection, overcoming the limitations of traditional deep networks while enhancing efficiency and performance (Nagpal, Bhinge, & Shitole, 2022; He, Zhang, Ren, & Sun, 2016).

METHODOLOGY

In this section, the proposed methodology for enahancing WSN fault detection is comprehensively explained. The first step in the proposed method is data preparation, where input and output data are reorganized for further processing. The input data includes outdoor humidity and temperature measurements collected from a multi-hop wireless sensor network. These measurements encompass various fault types, fault gain values, and fault ratios. Initially, all data are concatenated to form a uniform multiclass input, ensuring that the method operates comprehensively across diverse conditions without requiring algorithmic adjustments. This approach also mitigates potential conflicts that may arise when multiple classes receive similar votes from different classifiers.

Once concatenated, the number of consecutive measurement instances is determined to enable effective decision-making. Unlike traditional WSN fault classifiers that typically use a small number of samples (Zidi, et al, 2018), the proposed method processes five consecutive instances, enabling it to handle more complex datasets. Given the availability of multiple sensors per observation, the inclusion of five successive samples significantly increases the dataset size. However, concatenating observations in this manner can introduce redundant information, increasing computational complexity and potentially biasing the method toward certain classes.

To address this, the proposed method employs a novel sample reduction technique using Chatterjee correlation. This technique reduces the input space dimensionality while retaining critical information. Unlike Pearson correlation, Chatterjee correlation excels at detecting nonlinear relationships, which is crucial for capturing all relevant patterns for fault detection. The reduction process involves calculating the Chatterjee correlation for each pair of samples, followed by summing the absolute correlations for each sample with all others. A subset with the lowest overall Chatterjee correlation is then selected, ensuring reduced redundancy and greater variety in the retained information. The lower correlation values indicate higher sample diversity, enhancing the robustness of the method.

The next step in the proposed method focuses on feature selection, a critical process for enhancing the efficiency and effectiveness of fault detection. This paper introduces a novel feature selection approach that combines the Horse Herd Optimization Algorithm (HOA) with Neighborhood Component Analysis (NCA). Feature selection is essential for reducing data dimensionality, improving model performance, and minimizing the risk of overfitting. NCA is a powerful feature selection technique, but its effectiveness heavily depends on a key parameter: the regularization parameter. This parameter controls the balance between preserving local and global neighborhood information during the feature selection process.

To optimize the regularization parameter for NCA, the HOA optimization algorithm is employed. HOA iteratively evaluates different regularization parameters within its population based on the cross-entropy loss of the trained NCA. To configure HOA for this task, a single decision variable—matching the number of NCA parameters—is defined, with a standard range of [0, 1] for the regularization parameter. Additionally, the algorithm is set to perform a maximum of 30 iterations with a population size of 10. These settings are carefully chosen to balance execution time and the evaluation of a sufficient range of regularization parameters.

Once the optimal regularization parameter is identified, NCA is used to rank features based on their importance. The top 16 features are selected to form the input feature subset. This number is chosen to strike a balance between effective dimensionality reduction and preserving sufficient data for input into the ResNet model.

After feature selection, the data format must be modified to be compatible with the ResNet network, which utilizes convolutional layers that apply 2D filters to input images. To leverage the full potential of ResNet's capabilities in capturing mutual relationships between features, the proposed method transforms the 1D feature vector into a structured 2D array. This transformation enables ResNet’s convolutional layers to identify local patterns and dependencies within the data using spatial filters effectively.

The proposed transformation employs an inverse zigzag scanning technique, ensuring that features are arranged spatially in a meaningful manner. First, the 16-element feature vector is expanded by repeating each feature 16 times, producing a 1×256 vector. This expansion ensures sufficient representation of each feature in the subsequent 2D arrangement. Next, the expanded 1×256 vector is reorganized into a 16×16 2D array using an inverse zigzag scanning pattern. This pattern systematically arranges the features to preserve their local adjacency, which is crucial for the operation of convolutional filters.

The inverse zigzag scanning method mimics a natural traversal of adjacent data, ensuring that neighboring features from the 1D vector are placed in close proximity within the 2D array. This structured arrangement allows ResNet's convolutional filters to explore intricate relationships between features, capturing all potential interactions in small, localized regions. By doing so, ResNet leverages its hierarchical residual blocks to learn complex patterns and dependencies effectively.

This transformation maximizes the utility of ResNet's convolutional operations, enhancing its ability to extract meaningful representations from the input data. Figure 2 illustrates the conversion process from a 1D feature vector to a 2D image format for use in the ResNet network.

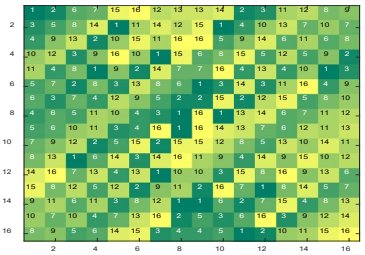

Figure 2. The proposed 1D feature vector to 2D image conversion using inverse zigzag scanning. The numbers in each pixel show the element numbers of the initial 1×16 vector

The final step in the proposed method is the design of the ResNet network for accurate WSN fault detection. ResNet employs skip connections that address the vanishing gradient problem, enabling the successful training of deep networks. This depth is essential for effectively generalizing across various fault types, given the inherent complexity of WSN fault detection. ResNet's strength in capturing precise spatial relationships makes it particularly suited for analyzing the proposed 2D features, which utilize structural details and mutual relationships preserved through the inverse zigzag scanning.

However, standard ResNet architectures like ResNet-50 and ResNet-101 are not suitable for this application due to their high computational demands. These variants are optimized for large inputs, such as 224×224 pixel images, whereas the input feature arrays in this study are significantly smaller, measuring just 16×16 pixels. Scaling up the input images or extracting additional features would result in unnecessary computational overhead. To address this challenge, a lightweight ResNet architecture is specifically tailored for the extracted features, enabling efficient processing of the small inputs while preserving essential spatial hierarchies required for accurate fault classification.

The proposed ResNet architecture comprises three stacks of residual layers. The first stack includes four residual blocks, the second contains three blocks, and the third consists of two blocks, as illustrated in Figure 3. Each stack progressively increases the number of convolutional filters: the first stack uses 16 filters, the second stack uses 32 filters, and the third stack employs 64 filters. Additionally, a preliminary convolutional layer, batch normalization layer, and ReLU activation layer are introduced before the residual stacks. This initial convolutional layer, equipped with 16 filters of size 3×3, is designed to capture fine-grained details and enhance the overall efficiency of the network.

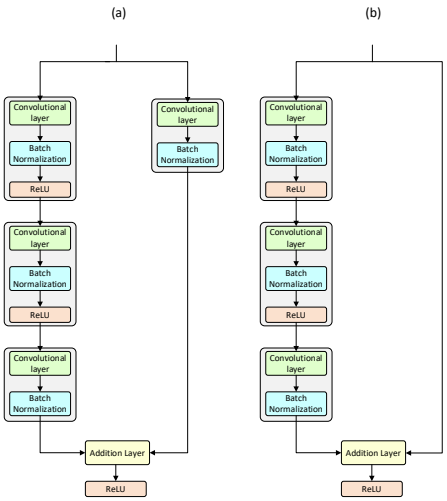

Figure 3. The structure of (a) the initial residual block in the stack and (b) the subsequent residual blocks in the stack

Following the last residual block of the second stack, a global average pooling layer is added to reduce each feature map to a single value, effectively summarizing spatial information across the feature maps. The network then concludes with a fully connected layer consisting of five neurons, followed by a SoftMax activation layer to classify the sensor data into five distinct categories: normal condition, offset faults, gain faults, stuck-at faults, and out-of-bounds faults. The complete architecture of the designed ResNet, with 247.8k learnable parameters, is illustrated in Figure 4.

Figure 4. The structure of the designed ResNet network

Once the network architecture is finalized, the training configuration is established. The ADAM optimizer is selected for training due to its adaptive learning rate, which dynamically adjusts based on the first and second moments of gradients. This makes ADAM particularly effective for deep learning tasks, offering faster and more stable convergence even with noisy gradients and large models. The optimizer's gradient decay factor is set to 0.9, and its squared gradient decay factor is set to 0.999.

The training process is configured with a mini-batch size of 512 and a total of 500 epochs, providing a balance between robust performance and hardware limitations. Additionally, the learning rate is managed using a piecewise decay schedule, starting with an initial rate of 0.1 and applying a decay factor of 0.85 every five epochs. This approach ensures gradual refinement of the learning process, enabling the network to converge effectively. Figure 5 illustrates the learning rate values over the course of training of the network.

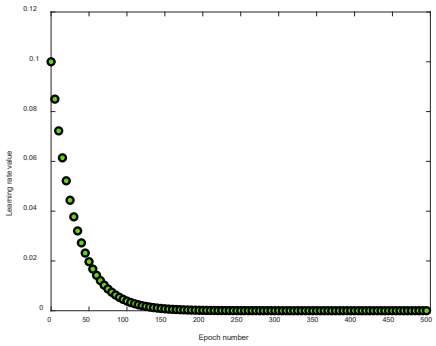

Figure 5. The values of learning rate over the course of epochs during the training procedure

DATASET

The dataset utilized in this study is a labeled wireless sensor network dataset designed for fault detection research. It comprises a collection of sensor measurements with various injected faults and contains 281,280 observations (vectors), each consisting of 12 attributes.

The dataset is based on an original dataset published in 2010 by researchers at the University of North Carolina at Greensboro (Suthaharan et al., 2010). The original data was collected using TelosB motes from both single-hop and multi-hop wireless sensor networks. Measurements include humidity and temperature values recorded every 5 seconds over a 6-hour period. During data collection, an event was introduced by injecting steam from hot water to increase humidity and temperature. This process created two distinct classes in the dataset: *normal data* and *anomalies*.

The prepared dataset has been used in previous fault detection research, with results published by Zidi et al. (2018), and we use this dataset in our study. This dataset only use the outdoor data from the multi-hop wireless sensor network. The prepared dataset contains 4,688 observations, each represented as a 12-dimensional vector. Each vector includes measurements recorded over three successive instances (t_0, t_1, t_2), with each instance comprising two temperature (T_1,T_2) and two humidity (H_1,H_2) readings. To introduce variability, a set of faults was randomly injected into the data. Faults were added at different rates (50%, 40%, 30%, 20%, and 10%) and included various fault types, such as *Offset, Gain, Stuck-at,* and *Out of bounds.* This process resulted in 60 distinct datasets, each containing 4,688 observations. For each dataset, two files were prepared: one containing the observations and the other containing labels (y), where y = 1 indicates a normal observation, and y = −1 represents a fault observation (Zidi et al., 2018).

Evaluation Metrics

In classification tasks, evaluation metrics are essential for assessing the performance of the models. These metrics offer insights into how well a model predicts the class labels, enabling a comprehensive

evaluation of its effectiveness. Among the most commonly used metrics are accuracy, precision, recall, and the F1-score, each providing a distinct perspective on model performance.

Accuracy is defined as the ratio of correctly predicted instances to the total number of instances in the dataset. Mathematically, it is expressed as:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (13)$$

Where:

TP: True Positives

TN: True Negatives

FP: False Positives

FN: False Negatives

While accuracy is a straightforward and widely used metric, it can be misleading in imbalanced datasets. For instance, in a dataset with a significant majority class, a model predicting only the majority class can yield high accuracy without effectively identifying the minority class.

Precision measures the proportion of correctly identified positive predictions relative to the total positive predictions. It is defined as:

$$Precision = \frac{TP}{TP+FP} \quad (14)$$

Precision is particularly useful in scenarios where the cost of false positives is high, such as in medical diagnostics or fraud detection, where an incorrect positive prediction can have severe consequences.

Recall, also known as sensitivity or true positive rate, quantifies the proportion of actual positives correctly identified by the model. It is given by:

$$Recall = \frac{TP}{TP+FN} \quad (15)$$

High recall indicates that the model effectively captures most of the true positive instances, making it a critical metric in applications where missing a positive instance (false negatives) is costly, such as disease detection.

The **F1 score** is the harmonic mean of precision and recall, offering a balanced metric when precision and recall are equally important. It is calculated as:

$$F_1 Score = \frac{2\times(Precision\times Recall)}{Precision+Recall} \quad (16)$$

The F1 score ranges between 0 and 1, with higher values indicating better performance. This metric is particularly beneficial in cases of imbalanced datasets, where a single metric like accuracy might not adequately reflect model performance.

Simulation Results

This section presents the simulation results for the proposed WSN fault detection method, evaluating the effectiveness of data preparation and the overall fault detection performance. The input data is partitioned into two categories: 70% for training and 30% for testing, with data randomly assigned to each group. The simulations are performed using MATLAB 2024a on a system featuring an Intel Core i7 13650HX CPU, 16 GB RAM, and an NVIDIA RTX 4060 GPU with 8GB of GDDR6 memory.

DATA PREPARATION RESULTS

In this section, the outcomes of the data preparation phase are presented, showcasing the impact of redundant information removal, feature selection optimization, and the transformation of 1D feature

vectors into 2D arrays. These results demonstrate the efficiency of the proposed techniques in preparing the data for accurate WSN fault detection.

As the first step in the data preparation process, redundant information is eliminated using Chatterjee correlation. This technique ensures that only the most diverse and relevant samples are retained, reducing the input space while preserving the essential information needed for fault detection. As a result, the number of samples in the dataset is reduced to 22965, minimizing computational overhead and mitigating the risk of bias caused by redundant data.

The next step involves feature selection using Neighborhood Component Analysis (NCA) optimized with the Horse Herd Optimization Algorithm (HOA). Figure 6 shows the convergence curve of the HOA during the optimization of the NCA regularization parameter. The curve illustrates the iterative refinement of the cross-entropy loss over 30 iterations (310 evaluation points from which 300 points are due to having 10 horses in the population and the first 10 points are related to the initialization of the horses' positions), where the loss decreases significantly in the initial iterations and stabilizes as the algorithm converges. The final optimized regularization parameter value is 0.0013, indicating the optimal trade-off between preserving local and global neighborhood information during feature selection. This convergence result confirms the efficacy of HOA in optimizing NCA, enabling it to rank and select features that contribute most to fault classification accuracy.

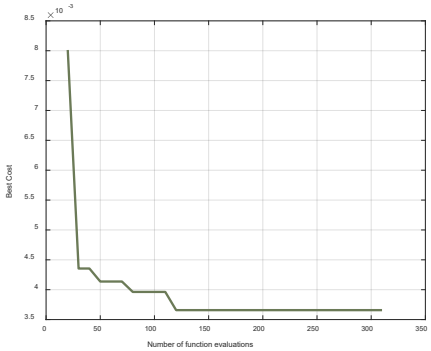

Figure 6. The HOA convergence curve for optimizing the regularization parameter of the NCA

The final stage of data preparation transforms the selected 1D feature vectors into 2D arrays for ResNet input. This transformation leverages the inverse zigzag scanning technique to spatially arrange features in a structured 16×16 array. Figure 7 illustrates six random samples of the resulting 2D images. These visualizations demonstrate how the spatial adjacency of features is preserved, enabling ResNet's convolutional filters to capture intricate local patterns and relationships. By converting the 1D vectors into 2D arrays, the proposed method ensures the effective utilization of ResNet's hierarchical structure for feature extraction and fault detection.

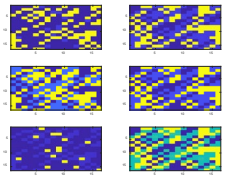

Figure 7. six random samples of the proposed 1D features to 2D images conversion using inverse zigzag scanning

WSN Fault Detection Results

In this section, the results of the proposed WSN fault detection method are presented, including the training convergence curve, evaluation metrics, confusion matrix, and receiver operating characteristic (ROC) curve. These results demonstrate the effectiveness of the designed ResNet architecture in accurately classifying different fault types in wireless sensor networks.

Figure 8 illustrates the training convergence curve of the ResNet model, showing both the training and validation accuracy over the course of 17,500 iterations. The plot indicates that the model successfully learns to generalize, with the training and validation curves following similar trends, suggesting that overfitting is minimized. The convergence of both curves signifies that the network has adequately learned to classify the data, leading to stable performance over time.

Table 1 presents the evaluation metrics for the WSN fault detection task, including accuracy, precision, recall, and F1 score. These metrics offer a comprehensive overview of the model's performance, assessing both the classification accuracy and the balance between precision and recall for the five fault categories. The accuracy of the model is 99.4106%, while the precision, recall, and F1 scores for each fault category are 98.7884%, 99.3796%, and 99.0831%, respectively. These values highlight the model's ability to identify faults with high accuracy and to maintain a balanced performance across all classes.

Next, Figure 9 presents the confusion matrix for the fault detection task. A confusion matrix is a useful tool for evaluating the performance of classification models by displaying the true positive, false positive, true negative, and false negative values for each class. It provides a detailed view of how well the model is distinguishing between different fault categories. The rows of the matrix represent the true classes, while the columns represent the predicted classes.

Upon analyzing the confusion matrix, the model demonstrates a high rate of correct classification for most classes. However, some misclassifications are observed, particularly between the offset and stuck-at faults, which may indicate the need for further refinement in distinguishing these fault types. The diagonal elements of the matrix show that 99.8% of the instances in the gain faults were correctly classified, while 0.7% of the instances in the out-of-bound faults were misclassified as gain faults. This analysis helps identify areas for improvement, such as adjusting the model's sensitivity to certain classes or refining the training data.

Figure 10 shows the ROC curve for the fault detection model. The ROC curve is a graphical representation of the trade-off between sensitivity (true positive rate) and specificity (1 - false positive rate) across different threshold values. A higher area under the curve (AUC) value indicates better model performance. The ROC curve illustrates how well the model can distinguish between the five fault types, with an average AUC value of 0.9994, which suggests that the model performs well in distinguishing between different fault categories. The curve is close to the top-left corner, indicating a low rate of false positives and false negatives.

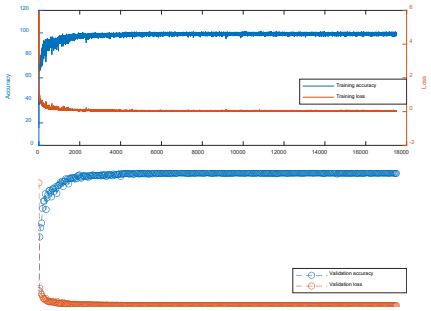

Figure 8. The training accuracy and loss of training and validation data during the training procedure

Table 1. The evaluation metric values for WSN fault detection using the proposed method

Accuracy	Precision	Recall	F1-Score
99.4106%	98.7884%	99.3796%	99.0831%

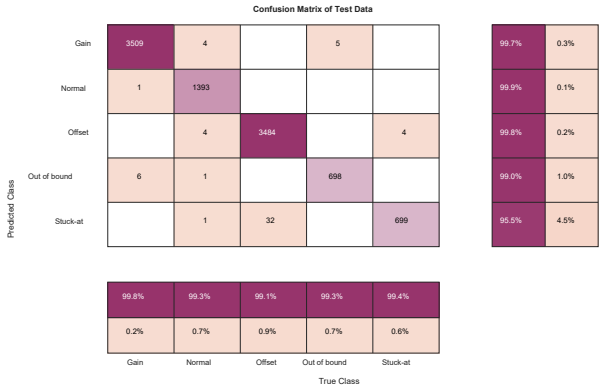

Figure 9. The confusion matrix of the final WSN fault detection using the proposed method

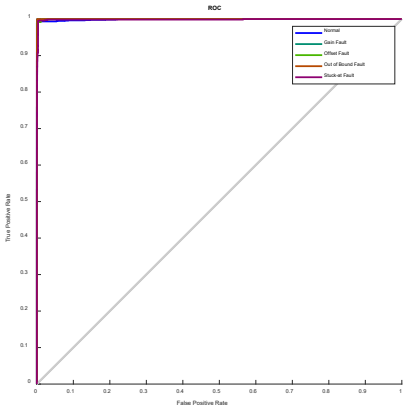

Figure 10. The ROC curve of the final WSN fault detection using the proposed method

Comparison

The proposed method in this study is compared to two prominent approaches for fault detection in WSNs (Table1): one based on Support Vector Machines (Zidi et al., 2018) and the other leveraging decision fusion with enhanced classifiers (Javaid et al., 2019). Unlike the SVM-based approach, which uses a single lightweight classifier to detect anomalies, our method integrates more advanced optimization techniques to enhance classification performance. While the SVM-based approach excels in simplicity and low computational resource requirements, it may struggle with complex fault patterns due to its reliance on a single decision function. In contrast, our method not only achieves comparable efficiency but also offers higher adaptability to diverse and intricate fault scenarios by incorporating additional mechanisms for feature optimization and fault detection.

When compared to the decision fusion method, which utilizes Enhanced K-Nearest Neighbor (EKNN), Enhanced Extreme Learning Machine (EELM), Enhanced Support Vector Machine (ESVM), and Enhanced Recurrent Extreme Learning Machine (ERELM), our approach demonstrates distinct advantages in terms of accuracy and computational efficiency. The decision fusion method, while effective for complex fault detection, requires significant computational resources due to the fusion of multiple classifiers and the processing of large volumes of data. Our method, by optimizing both feature selection and fault classification processes, strikes a balance between accuracy and resource consumption. Additionally, it simplifies implementation while achieving competitive detection rates against advanced techniques like ERELM, which demands more processing power.

Moreover, the most important advantage of the proposed method is that not only obtains higher accuracy than other methods but also performs multiclass classification. The binary classification performed in

other investigated methods is done separately. When each classifier predicts between normal and one type of fault, conflicts can occur if multiple classes receive similar votes in different classifiers.

Table 2. Summary of reviewed references

References	Methodology	Dataset	Accuracy	Multiclass classification
Zidi et al., 2018	Support vector machine	The WSN fault detection dataset	Exceed 99%	×
Javaid et al., 2019	Proposing four enhanced machine learning classifiers (EKNN, EELM, ESVM, ERELM) for the decision fusion approach	The WSN fault detection dataset	The best obtained accuracy is 98.8%	×
Proposed method	Chatterjee correlation, HOA, NCA, ResNet	The WSN fault detection dataset	99.41%	✓

CONCLUSION

In this paper, a novel approach for fault detection in wireless sensor networks (WSNs) was proposed, leveraging a combination of advanced techniques to achieve high accuracy and robust multiclass classification. The methodology includes a series of carefully designed steps, starting with data preprocessing, where redundant information is removed using Chatterjee correlation to reduce the input space and enhance computational efficiency. This step ensures that only the most informative features are retained, minimizing bias while maintaining a diverse representation of the input data.

Next, a feature selection mechanism combining Neighborhood Component Analysis (NCA) and the Horse Herd Optimization Algorithm (HOA) is introduced. NCA ranks features based on their relevance to classification, and HOA optimizes its regularization parameter to achieve optimal feature selection. This results in a compact yet highly informative feature subset that improves model performance while reducing dimensionality. The selected 1D feature vectors are then transformed into 2D images using an inverse zigzag scanning technique. This transformation enables the convolutional filters of the residual neural network (ResNet) to capture spatial patterns and mutual relationships among features, maximizing the network's ability to learn complex dependencies within the data.

The tailored ResNet architecture is lightweight and specifically designed to process the small 16×16 input images efficiently. It incorporates three stacks of residual layers with progressively increasing filter sizes, ensuring that spatial hierarchies are effectively captured. The final network is equipped with a global average pooling layer and a fully connected layer for fault classification, making it both computationally efficient and accurate. The training configuration, including the use of the ADAM optimizer and a piecewise learning rate schedule, ensures stable convergence and optimal performance.

The experimental results demonstrate the robustness of the proposed approach, achieving an impressive final accuracy of 99.41% in detecting five fault categories: normal condition, offset faults,

gain faults, stuck-at faults, and out-of-bounds faults. This result significantly outperforms other existing methods, both in terms of accuracy and classification capability. Unlike most prior works, which rely on separate binary classifiers for each fault type, the proposed method performs comprehensive multiclass classification. This eliminates the potential for conflicts arising from multiple binary classifiers providing similar predictions for different faults, thereby enhancing the reliability and interpretability of the fault detection system.

Moreover, the proposed method's lightweight ResNet architecture makes it suitable for deployment in real-world WSN environments with hardware constraints. By effectively addressing the challenges of redundancy, feature selection, and classification, the proposed approach establishes itself as a significant advancement in WSN fault detection.

REFERENCES

Arul, J. S., & Venkatesan, R. (2023). A deep learning approach for efficient anomaly detection in WSNS. *International Journal of Computers, Communications and Control*, *18*(1).

Atiga, J., Hamdi, M., Ejbali, R., & Zaied, M. (2021). Recurrent neural network NARX for distributed fault detection in wireless sensor networks. *International Journal of Sensor Networks*, *37*(2), 100-111.

Azzouz, I., Boussaid, B., Zouinkhi, A., & Abdelkrim, M. N. (2020, December). Multi-faults classification in WSN: A deep learning approach. In *2020 20th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)* (pp. 343-348). IEEE.

Biswas, P., Charitha, R., Gavel, S., & Raghuvanshi, A. S. (2019, April). Fault detection using hybrid of KF-ELM for wireless sensor networks. In *2019 3rd international conference on trends in electronics and informatics (ICOEI)* (pp. 746-750). IEEE.

Chatterjee, S. (2021). A new coefficient of correlation. *Journal of the American Statistical Association*, *116*(536), 2009-2022.

Gnanavel, S., Sreekrishna, M., Mani, V., Kumaran, G., Amshavalli, R.S., Alharbi, S., Maashi, M.S., Khalaf, O.I., Abdulsahib, G.M., Alghamdi, A.D., & Aldhyani, T.H. (2022). Analysis of Fault Classifiers to Detect the Faults and Node Failures in a Wireless Sensor Network. *Electronics*.

Gupta, S., Kaur, G., & Chanak, P. (2021, November). A deep bi-LSTM based fault detection algorithm for WSNs. In *2021 IEEE Bombay Section Signature Conference (IBSSC)* (pp. 1-5). IEEE.

Gupta, D., Sundaram, S., Rodrigues, J. J., & Khanna, A. (2023). An improved fault detection crow search algorithm for wireless sensor network. *International Journal of Communication Systems*, *36*(12), e4136.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

Jan, S. U., Lee, Y. D., & Koo, I. S. (2021). A distributed sensor-fault detection and diagnosis framework using machine learning. *Information Sciences*, *547*, 777-796.

Javaid, A., Javaid, N., Wadud, Z., Saba, T., Sheta, O. E., Saleem, M. Q., & Alzahrani, M. E. (2019). Machine learning algorithms and fault detection for improved belief function based decision fusion in wireless sensor networks. *Sensors*, *19*(6), 1334.

Karmarkar, A., Chanak, P., & Kumar, N. (2020, February). An optimized svm based fault diagnosis scheme for wireless sensor networks. In *2020 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)* (pp. 1-7). IEEE.

Kumar, M., Mukherjee, P., Verma, K., Verma, S., & Rawat, D. B. (2021). Improved deep convolutional neural network based malicious node detection and energy-efficient data transmission in wireless sensor networks. *IEEE Transactions on Network Science and Engineering*, *9*(5), 3272-3281.

Laiou, A., Malliou, C. M., Lenas, S. A., & Tsaoussidis, V. (2019). Autonomous Fault Detection and Diagnosis in Wireless Sensor Networks Using Decision Trees. *J. Commun.*, *14*(7), 544-552.

Mazibuco, V. A., Nhung, N. P., & Linh, N. T. (2023). Fault detection in wireless sensor networks with deep neural networks. *Journal of Military Science and Technology,(CSCE7)*, 27-36.

MiarNaeimi, F., Azizyan, G., & Rashki, M. (2021). Horse herd optimization algorithm: A nature-inspired algorithm for high-dimensional optimization problems. *Knowledge-Based Systems*, *213*, 106711.

Mohapatra, S., & Khilar, P. M. (2020). Fault diagnosis in wireless sensor network using negative selection algorithm and support vector machine. *Computational Intelligence*, *36*(3), 1374-1393.

Nagpal, P., Bhinge, S. A., & Shitole, A. (2022, December). A comparative analysis of ResNet architectures. *2022 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)* (pp. 1-8). IEEE.

Noshad, Z., Javaid, N., Saba, T., Wadud, Z., Saleem, M. Q., Alzahrani, M. E., & Sheta, O. E. (2019). Fault detection in wireless sensor networks through the random forest classifier. *Sensors*, *19*(7), 1568.

Panda, M., Gouda, B. S., & Panigrahi, T. (2020). Fault diagnosis in wireless sensor networks using a neural network constructed by deep learning technique. In *Nature Inspired Computing for Wireless Sensor Networks* (pp. 77-101). Singapore: Springer Singapore.

Prasad, R., & Baghel, R. K. (2023). Self-detection based fault diagnosis for wireless sensor networks. *Ad Hoc Networks*, *149*, 103245. Prasad, R., & Baghel, R. K. (2023). Self-detection based fault diagnosis for wireless sensor networks. Ad Hoc Networks, 149, 103245.

Prasad, R., & Baghel, R. K. (2021). A novel fault diagnosis technique for wireless sensor network using feedforward neural network. IEEE Sensors Letters, 6(1), 1-4.

Regin, R., Rajest, S., & Singh, B. (2021). Fault detection in wireless sensor network based on deep learning algorithms. *EAI Endorsed Transactions on Scalable Information Systems*, *8*(32).

Sarangi, B., & Tripathy, B. (2023). Outlier detection technique for wireless sensor network using GAN with Autoencoder to increase the network lifetime. *International Journal of Computer Network and Information Security*, *14*(1), 26.

Swain, R. R., & Khilar, P. M. (2017, November). Soft fault diagnosis in wireless sensor networks using PSO based classification. In *TENCON 2017-2017 IEEE Region 10 Conference* (pp. 2456-2461). IEEE.

Saeed, U., Jan, S. U., Lee, Y. D., & Koo, I. (2021). Fault diagnosis based on extremely randomized trees in wireless sensor networks. *Reliability engineering & system safety*, *205*, 107284.

Yang, W., Wang, K., & Zuo, W. (2012). Neighborhood component feature selection for high-dimensional data. *J. Comput.*, *7*(1), 161-168.

Zidi, S., Moulahi, T., & Alaya, B. (2017). Fault detection in wireless sensor networks through SVM classifier. *IEEE Sensors Journal*, *18*(1), 340-347.

Zhang, Z., Mehmood, A., Shu, L., Huo, Z., Zhang, Y., & Mukherjee, M. (2018). A Survey on Fault Diagnosis in Wireless Sensor Networks. *IEEE Access, 6*, 11349-11364.