A Novel Framework for Sensor Node Fault Diagnosis in Agricultural Internet of Things Using Federated Learning

Parthasarthi Aggarwal, Jubin Banerjee, Naveen Kumar Gupta, and Om Jee Pandey, *Senior Member, IEEE*

***Abstract*—In the realm of precision agriculture driven by the Agricultural Internet of Things (AgIoT), the reliability of sensor node data is often compromised by faults, posing challenges to decision-making processes. Addressing aforementioned issue, this work introduces a novel framework for sensor node fault diagnosis using Federated Learning (FL) in AgIoT. Leveraging FL, the framework promotes collaborative model training across edge devices, enhancing accuracy through decentralized learning. Advanced machine learning algorithms enable the framework to discern abnormal sensor node behavior with precision, enabling prompt isolation of faulty sensor nodes. Notably, the framework is adaptable to diverse agricultural scenarios, accommodating variations in sensor node types and deployment configurations. Experimental validation demonstrates the framework's efficacy in accurately diagnosing sensor node faults, achieving a high number of correct fault detection, while maintaining overall accuracy. Hence, this research work significantly advances fault-tolerant AgIoT systems, offering a robust solution that maximizes fault detection accuracy even in scenarios with sparse data, thereby supporting reliable, data-driven decision-making in precision agriculture.**

***Index Terms*—Federated Learning (FL), Agricultural Internet of Things (AgIoT), precision agriculture, fault-tolerant systems, and Machine Learning (ML).**

I. Introduction

In the rapidly evolving agricultural landscape, modern farming is employing the salient features of the Agricultural Internet of Things (AgIoT) to empower precision farming to usher in a transformative era in the agriculture sector. AgIoT deploys a network of intelligent devices with sensors, actuators and communication capabilities. This network enables the collection and transmission of data in real-time. AgIoT facilitates the acquisition of crucial information that provides vital support for precision agriculture by optimising resource utilisation, reducing environmental impact, and heightening agricultural productivity. This combination of innovative practices addresses the pressing challenges of global population growth, ecological sustainability, and resource management. The reliability and security of the data generated by these smart devices are paramount as they are ubiquitous in the AgIot domain. It is increasingly essential to ensure the real-time reliability of data at the minimum cost to sustain the success of precision agriculture as it strives to meet the ever-expanding demands of the growing global population. Hence, addressing the intricacies of data reliability is crucial in realising the full potential of the symbiotic relationship between AgIoT and precision agriculture [1], [2].

Sensor fault diagnosis is a crucial aspect of the upkeep of AgIoT networks that focuses on detecting and identifying issues that compromise the reliability and accuracy of sensor readings [3]. These faults can be caused by a plethora of factors, such as environmental conditions, technical malfunctions, or unforeseen events. Systemic malfunctions or degradation in performance can give rise to symptoms indicating the presence of sensor faults. These symptoms, characterized as observable or non-observable phenomena, serve as indicators for identifying and diagnosing faults within the system. A step-by-step analysis of sensor fault diagnosis is depicted in Fig. 1. It is essential to implement robust systems capable of promptly identifying and diagnosing these faults in real time to address this problem. This ensures the precision of the sensor readings for decision-making processes. Hence, by enhancing sensor fault diagnosis, AgIoT networks can offer precise real-time data crucial for effective resource management, optimized farming practices, and peak productivity. Effective fault diagnosis capabilities enhance the performance of AgIoT by facilitating constant improvement and efficient performance. Traditional sensor node fault diagnosis methodologies have limited adaptability to diverse and dynamic agricultural environments. These approaches are typically centralized and restricted in their scope of accurately detecting sensor failures or anomalies due to variations in environmental conditions such as temperature, weather patterns, air and water quality and soil composition. As a result, there is a heightened risk of inaccurate fault diagnosis, false detections, and, more crucially, missed detection, leading to suboptimal farming practices and reduced efficiency.

Federated learning (FL) is a decentralized machine learning approach that enables model training across multiple devices or nodes while keeping data localized [4]. Numerous agriculture nodes collect and process data from sensors. In the context of AgIoT, FL prevents the need to share all the readings over the global network. Instead of transmitting bulky and sensitive data to a centralized server for model training, FL allows the training to occur locally on each device. The aggregated model is then updated collaboratively, ensuring insights are gained from diverse nodes. This approach is particularly beneficial in AgIoT fault diagnosis as it leverages the diverse data collected from different agricultural environments. At the same time, the local nodes can retain specific individual adaptations to their local environments after their local updates. This advantage of FL, which combines a global model's performance with the ability to capture specific patterns, results in more robust and accurate fault detection models. Since FL reduces the need for constant data transfers, it reduces bandwidth usage and latency, which is crucial for real-time monitoring and response in agricultural operations.

The primary goal of this research is to introduce the FL-based robust machine learning framework as a comprehensive solution to mitigate the impact of faulty sensors in precision agriculture [3], [5], [6]. The major contributions of this re-

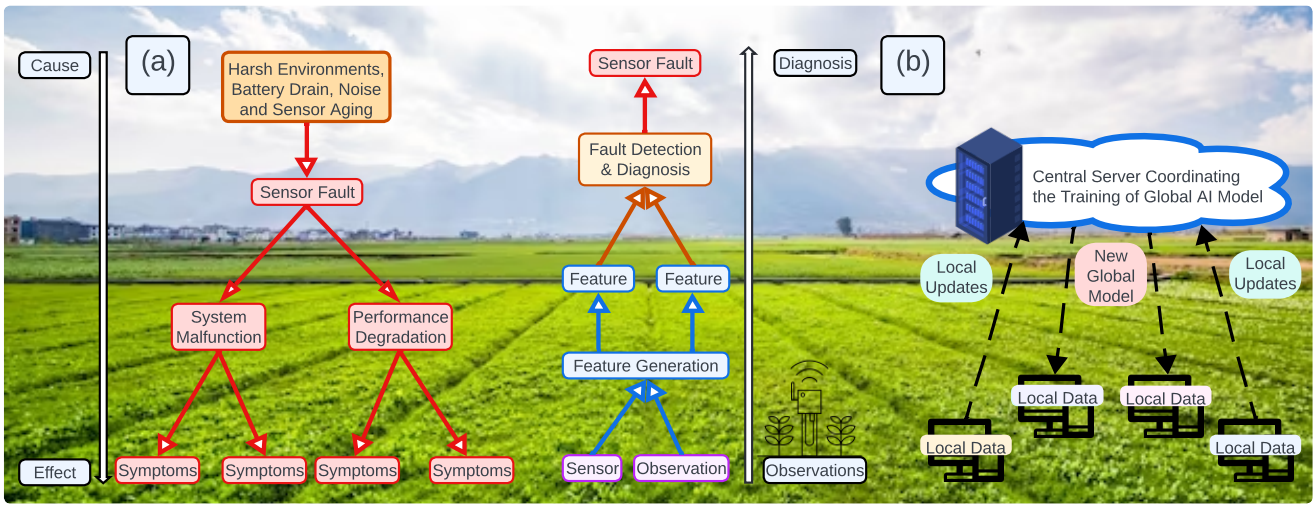


Fig. 1: Visual depiction of an AgIoT system: (a) Depicts sensor diagnosis and fault detection at individual nodes employing a cause− >effect− >observations− >diagnosis framework. (b) Illustrates Federated training of a global AI model through integration of updates gathered from local nodes. Background Image: https://www.freepik.com/free-photo/farmland˙1145885.htmSource

search are as follows:

1) This work presents a novel FL-based framework designed to facilitate communication within a decentralized network of sensors, enabling the generalization of fault detection trends across the network. The research work explores the development and implementation of this framework, showcasing its effectiveness in improving fault detection capabilities in distributed sensor networks.
2) We have evaluated four distinct federated learning methods: Fed-Avg, FedSGD, Weighted Fed-Avg, and Selective Fed-Delta. Our comprehensive analysis focused on their accuracy, processing time, overhead, and CPU usage to compare their performance in decentralized sensor networks.
3) We also proposed a novel ensemble method for fault classification in AgIoT networks. This method utilizes Long Short-Term Memory (LSTM) as the base model and extends classification models to detect specific types of faults.

The rest of the paper is organized as follows: In Section II, we elucidate the terminology necessary for grasping the paper's context. Section III delves into a comprehensive analysis of prior efforts aimed at sensor fault diagnosis. The modeling assumptions pertinent to our proposed method, along with an overview of the network architecture and problem formulation, are detailed in Section IV. Our proposed FL based framework receives thorough exposition in Section V. Subsequently, Section VI offers an empirical account of our experimental endeavors, accompanied by their meticulous analysis. These experiments encompass simulations conducted across multiple sensor nodes, coupled with a detailed study of their resource utilization patterns. Finally, Section VII encapsulates the concluding remarks drawn from our research findings.

II. Background

In AgIoT, ensuring reliable sensor data is crucial for effective agricultural management. Sensors face faults during data acquisition and transmission. Addressing these faults is essential for AgIoT efficacy. Specialized models tailored to specific fault scenarios are developed to enhance fault diagnosis and improve system reliability. In this section, we have first discussed sensor faults. Subsequently, specialized models for addressing specific fault scenarios have been described.

A. Sensor Faults

Errant deviation from the expected patterns can be a symptom of the sensor encountering certain faults. These symptoms can be sporadic incorrect values or matching known faulty patterns. Sensor nodes are often deployed in harsh environments and are circumscribed by limitations of power, memory and computation power. Prolonged use exacerbates the rundown of sensors and other node components, resulting in erroneous readings that limit AgIoT systems' efficacy in agriculture. These faults can result in irreparable damage to the AgIoT nodes and critical loss in operations if not detected promptly. TABLE I describes various sensor faults that may manifest within the agricultural context.

B. Specialized Models Addressing Specific Fault Scenarios

In this work, we strive to develop and scrutinize specialized models crafted to address distinct fault scenarios in the AgIoT landscape. Fault diagnosis encompasses not only fault detection but also the classification of distinct fault patterns to enable precise and efficient corrective responses. Our research endeavours to augment fault diagnosis precision and efficacy by tailoring models to the unique characteristics of specific fault types. The ensuing performance analysis scrutinizes these specialized models [3], [15], [16], evaluating their effectiveness in accurately predicting specific faults. This scrutiny advances our comprehension of fault diagnosis in AgIoT and

Table I: Description of different fault types [3].

Fault Type	Description
Hard Fault [7]	Gradual deviation of sensor readings over time, often caused by aging components or environmental changes.
Drift Fault [8]	A gradual change in sensor values over time, requires timely diagnosis to prevent measurement errors and damage.
Bias Fault [9]	A constant added to the sensor measurement, causing deviation; impacting control system stability and decision-making.
Stuck Fault [10]	A sudden, constant sensor measurement error, with persistent characteristics that are evident and simple to identify.
Accuracy Decline Fault [11]	An accuracy decline fault reflects a stable average sensor measurement but increasing output variance, leading to reduced measurement accuracy.
Spike Fault [12], [13]	A spike fault manifests as a sudden amplitude spike in sensor measurements, often caused by loose connections, leading to incorrect system decisions.
Random Fault [14]	A random fault is an irregular occurrence of faulty readings, appearing with a frequency below a specified threshold, challenging detection and diagnosis.

furnishes valuable insights for developing specialized models adept at addressing nuanced fault scenarios.

We have employed a robust validation methodology involving extensive simulations and empirical assessments, which ensures the reliability and generalizability of the specialized models' performance across different fault occurrences. Furthermore, our research addresses optimization considerations, investigating opportunities to enhance the computational efficiency of the specialized models by reducing the number of communication rounds [5], [17], [18], aligning them with the resource constraints typical in AgIoT deployments. Table II provides an overview of the conventional approaches for fault classification.

Simulating and testing the Bernoulli Restricted Boltzmann Machine (BRBM) model and the Deep Belief Network (DBN) [3], [15] for analyzing drift faults has revealed a significant limitation. When using a sparse dataset which contains very few fault occurrences, the models exhibit a strong bias towards negative predictions; that is, the models consistently predict the absence of faults even when faults are present. This critical shortcoming results in frequent false negatives—instances where the models fail to detect actual faults. In essence, the models tend to overlook the occurrence of faults that warrant detection, severely crippling the efficacy of fault diagnosis in the AgIoT network. This discrepancy undermines the reliability of the fault analysis process, as the emphasis on true negatives, while essential in certain contexts, inadvertently diminishes the models' sensitivity to accurately identifying true faults. Addressing this flaw is imperative for ensuring the robustness and scalability of fault diagnosis systems, necessitating the testing of alternative methodologies or fine-tuning strategies to rectify the skewed biases exhibited by some of the conventional fault classification models.

III. Related Work

In the realm of sensor fault diagnosis, numerous researchers have explored the development of lightweight models tailored for deployment on IoT devices. For instance, Shangjun et. al [19] introduced a methodology that employs a rectified linear unit lightweight convolution approach to extract features through a deeper network with a fast learning mechanism. Similarly, Xing et. al [20] proposed a lightweight 1D-CNN-based sensor fault diagnosis system specifically designed for solar insecticidal lamp (SIL) IoT nodes, termed separable and attention 1D-CNN (SA1D CNN). This approach uses a time and channel attention module (TCAM) to extract important input features to improve diagnostic accuracy. It uses parameter reduction in the depthwise separable convolution layer and feature recalibration of the TCAM. Results of their work have indicated that TCAM can improve the feature-extracting ability of SA1D-CNN, and a small increase in the number of parameters is acceptable. The drawback of this method is that the lack of priority lies in the lack of priority for a variety of faults, and thus, it cannot detect many severe faults at the early stage.

Table II: Conventional Fault Classification Methodologies [3].

Model Description	Applicable Fault Type
Linear AutoEncoder-based Classification Model [6]	Drift Fault
Three-Layered Logistic Regression model [17]	Bias Fault
An SVR model using a linear kernel and a regularization of 1.0 [5]	Malfunction (Hard) Fault
Bernoulli Restricted Boltzmann Machine+Logistic Regression [15]	Random Fault
Three Stacked SimpleRNN Layers, with a dense layer and a sigmoid classification output layer [3]	Drift Fault
An Autoencoder based binary classifier [3]	Bias Fault
BackPropagation Neural Network utilising wavelet denoising techniques [3]	Malfunction (Hard) Fault
A Deep Belief Network combined with a Convolutional Neural Network [3]	Random Fault

In the recent decades, different sensor fault diagnosis methods have been implemented for the AgIoT system, broadly being classified into into centralized and distributed methods. A centralized approach uses a central node, which diagnoses each sensor's condition, leveraging its ample resources [21]. This setup extends the lifespan of AgIoT Sensor Networks by storing a fault diagnosis model and periodically analyzing sensor data [7], [22]. While minimizing hardware requirements and prolonging sensor life, challenges like network congestion, slower detection speed, and difficulty meeting real-time needs arise. Additionally, transmitting data to the central node increases pressure on sink nodes, consuming their resources. However, in agricultural settings, the limitations of centralized fault diagnosis due to sparse base stations and weak signals prompt the adoption of distributed strategies. [23] proposed a distributed system using machine learning for fault detection within sensors and diagnosis at the central node, ensuring real-time performance while minimizing sensor computations [24], [25]. However, this approach requires high hardware resources for fault diagnosis. To address the requirement of high hardware, lightweight fault detection algorithms must be explored to reduce sensor resource consumption during fault diagnosis.

The applied detection techniques can be widely classified into model-based methods, signal-based methods, and data-driven methods. Model-based methods [26]–[28] construct sophisticated white-box models by using the underlying physical

principles, have low diagnostic latency; however, it is difficult to set the appropriate threshold and their performance highly relies on parameter estimation and expert knowledge. Signal-based methods do not rely on system parameters but use the envelope demodulation analysis method to extract faulty features [29] and apply variational mode decomposition to decompose vibration signals into a series of intrinsic mode functions, which can be used to enhance periodic fault features of reconstructing signals [30]. Data-driven methods that mainly employ machine-Learning and Deep-learning based techniques have also been widely adopted nowadays for fault analysis. SqueezeNet [31], Xception [32], MobileNet [33], and ShuffleNet [34] are the most typical lightweight CNN architectures employed for the task of fault detection.

IV. System Model and Problem Formulation

This section presents the system model used for the development of fault detection and classification in the AgIoT network. Subsequently, problem formulation for fault detection and fault classification is discussed.

A. System Model

In this work, we have considered an FL Network that consists of one central server and N sensor nodes. All the devices are connected by means of channels with a channel capacity of 180 KBps. The considered FL Network is illustrated in Fig. 2. The central server is responsible for managing the global network, coordinating the training of the FL fault detection model, and classifying the type of fault occurrence when a fault is encountered. A detailed description of the steps used in Fig. 2 is discussed in the following sections.

1) FL model: The system utilizes a decentralized FL approach to train the global fault detection model across diverse fog nodes collaboratively. The fault detection model gets trained on multiple fog nodes, each making local updates to the global model. The central server aggregates only these model updates made by the individual devices between the duration t to $t+1$ on the central server and redistributes the updated global model to the AgIoT nodes for the following round. The weight parameters ($\boldsymbol{\theta}$) corresponding to the global fault detection model play a crucial role in identifying the efficacy of fault detection.

Additionally, the performance of the updates for each round is analyzed over the entire network to ensure comprehensive evaluation and improvement.

If any node detects a fault, the data point is transmitted to the central server to determine the type of fault. This classification accuracy of fault type is independent of the initial detection accuracy.

2) Ensemble Classifier: As shown in Fig. 3, an ensemble fault classifier is used at the central server to find the type of fault. The ensemble classifier consists of n number of fault classification models denoted by $M = \{m_1, m_2, ..., m_n\}$, which predict specific fault types. The set $X = \{x_1, x_2, x_3, ...\}$, represents the instances of data where we know a fault has occurred. On applying the fauly data instances, each prediction model produces a predicted value p_i,

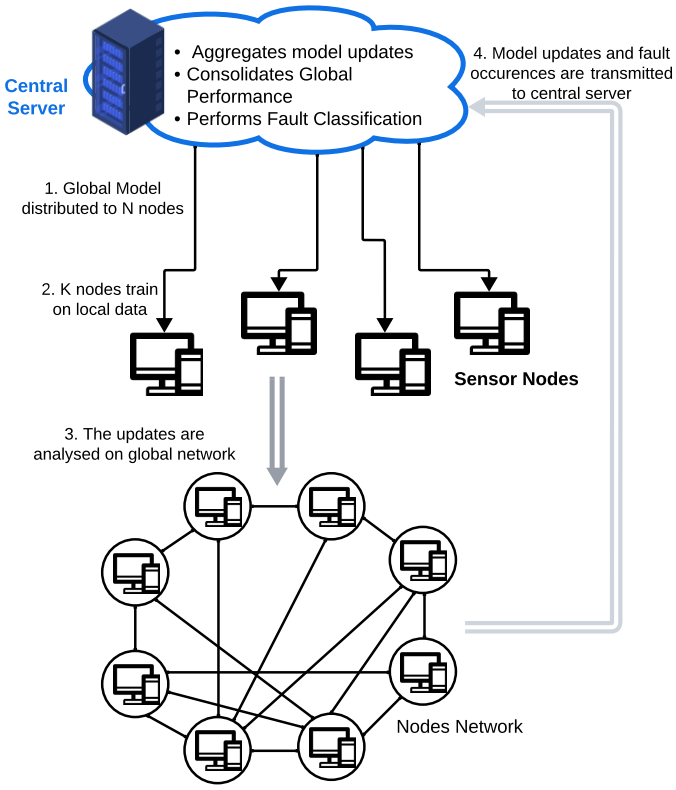


Fig. 2: Working Overview of the FL network

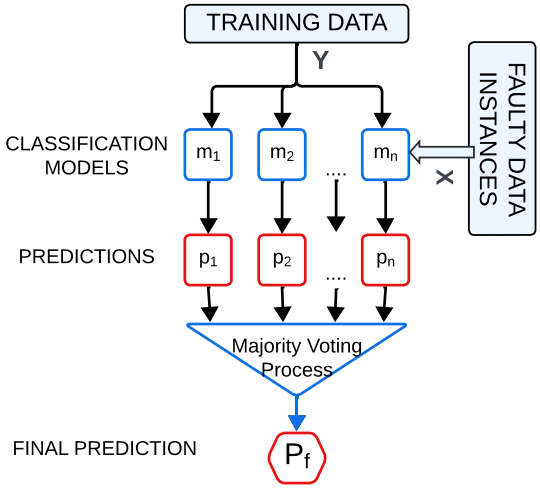


Fig. 3: Ensemble Classifier at the central server

which is limited by $0 \leq p_i \leq 1$, which represents the model m_i's confidence of it being the f_i fault type. The majority voting algorithm provides the most likely fault type P_f for a given testing instance. It can be represented as (1).

$$P_f = max\{m_1(x), m_2(x), ..., m_n(x)\}, \tag{1}$$

where $p_i = m_i(x)$ and $P_f \in F$, F denotes the set encompassing all possible fault types that can be detected.

The problem formulation considered for fault detection and classification is discussed in the next section.

B. Problem Formulation

Each sensor node collects environmental data, but external factors can lead to sensor degradation and failure. In order to correct the execution of the system, an accurate fault diagnosis system is desirable where nodes may encounter n type of faults $F = \{f_1, f_2, f_3, ..., f_n\}$.

The principal objective of this work is to maximize the number of correct fault detections while maintaining high accuracy in the FL framework. The parameter ($\boldsymbol{\theta}$) is optimized to enhance the performance of the global model. Thus, we seek to minimize the loss function $\boldsymbol{F(\theta)}$ while maximizing the true positive parameter $\boldsymbol{TP(\theta)}$. In each iteration, the AgIoT network aims to learn optimal global weight parameters $\boldsymbol{\theta}$ by minimizing the empirical loss function maximizing the number of global correct fault detections. Accordingly, the problem formulation can be written as follows:

$$\min_{\boldsymbol{\theta}} \left\{ \boldsymbol{F(\theta)} = \Psi \left(\{ f^j(\boldsymbol{\theta}) \}_{1 \leq j \leq K} \right) \right\}, \tag{2}$$

$$\max_{\boldsymbol{\theta}} \left\{ \boldsymbol{TP(\theta)} = \sum_{j=1}^{N} \text{tp}^j(\boldsymbol{\theta}) \right\}, \tag{3}$$

$$\sum_{j=1}^{T} \sum_{i=1}^{K} D_i^j \leq \mathcal{T}, \tag{4a}$$

$$c \cdot \sum_{I=0}^{I_z} U_I \cdot \gamma \leq \mathcal{C}, \tag{4b}$$

$$\sum_{j=1}^{T} \sum_{i=1}^{N} \left(PT_i^j + Qt_i^j \right) \leq \mathcal{O}. \tag{4c}$$

Here, $\Psi(\cdot)$ is the aggregate function, j represents a node in the network, and f^j is the local objective function. We define 'true positives' (tp^j) as the number of correctly identified faults for node j. Moreover, variable K represents the constituent nodes actively participating in the FL algorithm. It's important to note that not all network nodes are necessarily involved in the training process; hence, $K \subseteq N$ contributes to the collaborative learning procedure.

The research objectives include improving fault diagnosis accuracy within resource-constrained environments. The constraints include upper limits of total training time $\mathcal{T}$, computation cost $\mathcal{C}$, and communication overhead $\mathcal{O}$ across proposed FL approaches. Constraint (4a) guarantees that total training time duration does not exceed threshold $\mathcal{T}$. This duration is the summation of individual training times D_i^j across K nodes over the T global communication rounds. The constraint (4b) limits over total computational cost that cannot exceed the upper limit $\mathcal{C}$. The computation cost is the product of a cost constant c, interval duration γ and the summation of processor utilization U_I at interval I over all intervals. I_z is the total number of intervals determined by the monitoring time divided by interval duration γ. Moreover, The constraint (4c) signifies the maximum tolerable communication overhead time parameter $\mathcal{C}$. The communication overhead time comprises communication time for updating the global model, which includes transmission time PT_i^j and queuing time Qt_i^j over the T global communication rounds.

Apart from fault detection, another problem is fault classification to identify the type of fault at the central server. We optimized the specialized model m_i to maximize their accuracy in classifying specific fault types, and integrate their predictions based on the highest confidence score.

$$\max_{\{m_i\}} \sum_{i=1}^{n} A_i \left(\left\{ \hat{y}_i^j \left(m_i \right) \right\}, \left\{ y_j \right\} \right) \tag{5}$$

where:

- A_i: Function that evaluates the accuracy of predictions.
- $\{\hat{y}_i^j(\boldsymbol{m}_i)\}$: Predicted values of the i-th model using parameters $\boldsymbol{m}_i$.
- $\{y_j\}$: Actual values or true labels associated with instances j.

V. Proposed Framework

We have utilized an LSTM-based CNN architecture for our fault detection model in the FL setup. This architecture is meticulously designed to optimize performance. Its lightweight and compact nature makes it well-suited for deployment on resource-constrained ad-hoc sensor nodes. The architecture unfolds as follows:

1) Conv1D (Convolutional Layer)
2) MaxPooling1D layer
3) LSTM layer
4) Dense output layer for binary classification

The Conv1D layer [35] uses one-dimensional convolutional filters to extract temporal features, crucial for detecting sensor faults in AgIoT systems. The MaxPooling1D layer [36] downsamples feature maps to retain relevant temporal information, minimizing noise and highlighting fault patterns. The LSTM layer [37] processes sequential sensor data, capturing long-term dependencies essential for identifying complex fault signatures. Finally, the dense layer [38] maps the extracted features to output classes, facilitating precise fault detection and efficient troubleshooting in AgIoT systems. This architecture effectively captures both spatial and temporal dependencies in the data, leading to superior fault detection accuracy and reliability compared to traditional methods.

In this study, we explored several federated learning methodologies to enhance fault detection. We focused on two key methodologies: Federated Averaging (FedAvg) and Federated Stochastic Gradient Descent (FedSGD) in the form of minibatch FedSGD. Additionally, we introduced weighted FedAvg and selective Fed-Delta methods to further optimize performance [39].

FedAvg is a prominent algorithm where clients train local models and transmit new weights for central aggregation. Minibatch FedSGD performs stochastic gradient descent on minibatches of data and transmits the gradients to a central node for aggregation. Our introduced methods, weighted FedAvg and selective Fed-Delta, adopt an imbalanced approach favoring best-performing nodes. This is crucial for accurately identifying sensor faults, especially in datasets with limited positive instances where traditional methods might inflate accuracy through false negatives.

1) Local updates at nodes: In our testing, we utilized the LSTM + CNN model, chosen for its compact architecture, making it well-suited for deployment on fog nodes. We have prioritised models that demonstrate proficiency in accurately identifying faults. We used the Adam optimizer for FedAvg, Weighted FedAvg, and Selective Fed-Delta due to its adaptive learning rates and momentum, which facilitate faster convergence [40].

FedAvg, Weighted FedAvg, and Selective Fed-Delta have been trained for a single epoch on the entire dataset, with batch sizes of 128. Conversely, minibatch FedSGD trained on randomly selected minibatches of size 32 for a single epoch. Notably, while FedAvg, Weighted FedAvg, and Selective Fed-Delta have trained on the entire dataset for the epoch, minibatch FedSGD trained on only one single minibatch during every epoch.

In models utilizing the Adam optimizer, at iteration t, given weights $\boldsymbol{\theta_{[t]}}$, gradients g_t and the parameters α (learning rate), β_1 (decay rate for momentum), β_2 (decay rate for squared gradients), and ϵ (numerical stability) at iteration t, the local update for node j is given by:

$$\boldsymbol{\theta^j_{[t+1]}} = \boldsymbol{\theta^j_{[t]}} - \frac{\alpha \cdot \hat{m}_t}{\sqrt{\hat{v}_t{}^j} + \epsilon^j} \tag{6}$$

Here, $\hat{m}_t$ is the biased-corrected estimate of the first moment m_t, and $\hat{v}_t$ is the biased-corrected estimate of the second moment v_t.

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \tag{7}$$

$$v = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \tag{8}$$

$$\hat{m}_t = m_t / (1 - \beta_1) \tag{9}$$

$$\hat{v}_t = v_t / (1 - \beta_2) \tag{10}$$

We employed minibatch FedSGD with an SGD optimizer set at a learning rate of 0.05. Each iteration involved randomly selecting 32 samples from the 16,032 samples available on each node. The model was then trained on this minibatch for a single epoch to compute gradients. This approach allowed for efficient utilization of local datasets while maintaining convergence stability through minibatch processing.

In models utilizing the SGD optimizer, at iteration t, given momentum p, velocity v, and learning rate η update for node j is given by:

$$\boldsymbol{\theta^j_{[t+1]}} = \boldsymbol{\theta^j_{[t]}} + p.v - \eta.g_t^j \tag{11}$$

We define $\nabla J^j_{[t]}(\theta)$ as the weight delta for the local model for round t:

$$\nabla J^j_{[t]}(\boldsymbol{\theta}) = \boldsymbol{\theta^j_{[t+1]}} - \boldsymbol{\theta^j_{[t]}} \tag{12}$$

2) Global aggregation: We investigated diverse strategies for aggregation and convergence in the context of federated learning.

FedAvg follows the standard aggregation.

$$\boldsymbol{\theta_{[t+1]}} = \Psi(\{f^j(\boldsymbol{\theta})\}_{1 \le j \le K}) = \frac{1}{K} \sum_{j=1}^{K} \boldsymbol{\theta^j_{[t+1]}} \tag{13}$$

For FedSGD, we aggregate gradients computed on each node. The aggregation equation simplifies to:

$$\Psi\left(\{g^j_{[t+1]}\}_{1 \le j \le K}\right) = \frac{1}{K} \sum_{j=1}^{K} g^j_{[t+1]} \tag{14}$$

To incorporate the aggregated gradients for updating the global model weights, denoted by $\theta_{[t+1]}$, we use the following equation:

$$\boldsymbol{\theta_{[t+1]}} = \boldsymbol{\theta_{[t]}} + p \cdot v - \eta \cdot \Psi\left(\{g^j_{[t+1]}\}_{1 \le j \le K}\right) \tag{15}$$

Our equation for aggregating in Weighted FedAvg learning is outlined below:

$$\boldsymbol{\theta_{[t+1]}} = \Psi(\{f^j(\boldsymbol{\theta})\}_{1 \le j \le K}) = \frac{\sum_{j=1}^{K} tp^j_{[t]} \cdot \boldsymbol{\theta^j_{[t+1]}}}{\sum_{j=1}^{K} tp^j_{[t]}} \tag{16}$$

In selective Fed-Delta, we compute the weight delta $J^j_{[t]}(\theta)$ for each local node j at iteration t using Eq. (12). These deltas are then sorted in descending order of the number of true positives $tp^j_{[t]}$ detected at each node, ensuring that updates from nodes with higher true positives contribute more to the aggregation process.

After sorting, we determine each node's contribution to the global model update, $\Delta\theta^j_{[t]}$, as follows: we first try adding $\nabla J^j_{[t]}(\theta)/K$ to the global model parameters. This improves the model performance; we accept the update to the global model weights and proceed to the next weight deltas $J^{j+1}_{[t]}(\theta)$. This approach selectively integrates updates that enhance the model into the global parameters.

$$\boldsymbol{\Delta\theta^j_{[t]}} = \begin{cases} \frac{1}{K}.J^j_{[t]}(\boldsymbol{\theta}) & \text{if the global performance improves} \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

Subsequently, these individual node contributions are aggregated. Upon aggregation, the combined contributions determine the new model weights.

$$\Psi(\{\Delta\theta^j_{[t]}\}_{1 \le j \le K}) = \sum_{j=1}^{K} \Delta\theta^j_{[t]} \tag{18}$$

$$\boldsymbol{\theta_{[t+1]}} = \boldsymbol{\theta_{[t]}} + \Psi(\{\Delta\boldsymbol{\theta^j_{[t]}}\}_{1 \le j \le K}) \tag{19}$$

Algorithm 1 outlines the key steps of Weighted FedAvg with K clients for T global rounds, while Algorithm 2 presents the equivalent for Selective Fed-Delta.

In our experiments, $T = 10$ communication rounds were needed for both weighted FedAvg and FedAvg to converge. Selective Fed-Delta converged in $T = 5$ rounds. FedSGD required up to 50 rounds, but the time taken per round was less than the other methods.

In step 4 of the Fed-Delta algorithm, the temporary weights $\theta^j_{[t]}$ for each node j represent the weights that would result from training on that specific node. However, these updates are not retained locally; only the weight differentials are transmitted to the central server. The nodes persist in utilizing the global model weights until step 14. In this latter step, model updates are made based on the global performance of fault classification.

Algorithm 1 Performance-Driven FL: Boosting contributions from high-performing nodes

Initialization: 1: The Central server initializes and broadcasts an initial model $\theta_{[1]}$ to all the nodes.
2: $t = 1$ to T
Local Model updates for iteration t:
3: each node $j \in \{1, 2, \ldots, K\}$ /* $\theta^j_{[t]}$ is the weight parameter of local node j
4: $\theta^j_{[t+1]}$ is obtained using Eq. (6)
5: $\theta^j_{[t+1]}$ is sent to the central server
6: $tp^j_{[t]}$ is sent to the central server

Model update using aggregation at iteration t:
7: $\theta_{[t+1]} = \frac{\sum_{j=1}^{K} tp^j_{[t]} \cdot \theta^j_{[t+1]}}{\sum_{j=1}^{K} tp^j_{[t]}}$ using Eq. (16)
8: broadcast $\theta_{[t+1]}$ to all the fog nodes

Algorithm 2 Intelligent FL for adaptive network performance

Initialization:
1: The Central server initializes and broadcasts an initial model $\theta_{[1]}$ to all the nodes.
2: $t = 1$ to T **Local Model updates for iteration t:**
3: each node $j \in \{1, 2, \ldots, K\}$ /* $\theta^j_{[t]}$ is the weight parameter of local node j */
/* ${\theta'_{[t]}}^j$ is the temporary weight parameter used to compute the delta update for node j */
4: ${\theta'_{[t]}}^j$ is obtained using Eq. (6)
5: $\nabla J^j_{[t]}(\theta)$ is obtained using Eq. (12) $= {\theta'_{[t]}}^j - \theta^j_{[t]}$
6: $\nabla J^j_{[t]}(\theta)$ is sent to the central server
7: $tp^j_{[t]}$ is sent to the central server
Selective Integration at iteration t:
8: an empty list $\Delta\Theta = []$
each node $j \in \{1, 2, \ldots, K\}$ 9: Append $(\nabla J^j_{[t]}(\theta), tp^j_{[t]})$ to $\Delta\Theta$
10: Sort $\Delta\Theta$ in descending order by $tp^j_{[t]}$
11: Broadcast the sorted list $\Delta\Theta$ to all nodes
each $(\nabla J^i_{[t]}(\theta), tp^i_{[t]})$ in $\Delta\Theta$ 12: Compute tentative update $\frac{1}{K}\nabla J^i_{[t]}(\theta)$
13: Evaluate the performance across all nodes $j \in \{1, 2, \ldots, N\}$ using $\theta^j_{[t]} + \frac{1}{K}\nabla J^i_{[t]}(\theta)$
14: total global performance improves $\Delta\theta^i_{[t]} = \frac{1}{K}\nabla J^i_{[t]}(\theta)$
Accept the update: $\theta^j_{[t]} \leftarrow \theta^j_{[t]} + \Delta\theta^i_{[t]}$ Reject the update: $\Delta\theta^j_{[t]} \leftarrow 0$
15: $\Delta\theta_{[t]} = \sum_{j=1}^{K} \Delta\theta^j_{[t]}$ using Eq. (18)
16: $\theta_{[t+1]} = \theta_{[t]} + \Delta\theta_{[t]}$ Eq. (19)
17: broadcast $\theta_{[t+1]}$ to all the fog nodes.

During Fed-Delta training, a stricter fault detection threshold of 0.7 in step 14 of Algorithm 2 ensured higher precision in updates by considering only highly confident positive predictions, thus reducing false positives. However, we used the conservative threshold of 0.5 in all the approaches for final detection to balance sensitivity and specificity.

After the fault detection using the FL models, we perform fault classification using the ensemble classifier. In this work, we considered four specialized models for ensemble fault detection as illustrated in Table III. Their architectures have been described below:

(i) The LSTM-based AutoEncoder model uses two LSTM layers (64 and 32 units) for feature extraction from input sequences, followed by a classifier with two dense layers (16 units with ReLU and 1 unit with Sigmoid) for binary classification. It is compiled with the Adam optimizer and binary cross-entropy loss. The LSTM layers capture temporal patterns, while the dense layers ensure precise classification of bias faults.

(ii) An LSTM-based Backpropagation Neural Network (BPNN) integrates wavelet denoising and a sigmoid output layer for detecting and analyzing hard faults. This model combines LSTM for sequential data, wavelet denoising for signal preprocessing, and a sigmoid layer for binary classification, making it adept at identifying hard faults.

(iii) The three-stacked Recurrent Neural Network (RNN) model, enhanced with batch normalization, dropout layers, and a sigmoid classification layer, effectively detects drift faults by capturing temporal dependencies and mitigating overfitting.

(iv) The extended CNN-LSTM model includes a Conv1D layer for local feature extraction, MaxPooling1D for dimensionality reduction, an LSTM layer for sequential dependencies, a hidden dense layer with ReLU, dropout to prevent overfitting, and an output dense layer for predictions. This architecture effectively identifies random faults by combining local anomaly detection with long-term dependency capture.

Table III: specialized models for Ensemble fault classification

Model Description	**Applicable Fault Type**
An LSTM-based AutoEncoder and a binary Classification Model [41]–[43]	Bias Fault
An LSTM-based BPNN network with wavelet denoising and sigmoid classification output layer	Malfunction (Hard) Fault
A three-stacked RNN model with Batch Normalization, DropOut and a Sigmoid classification layer	Drift Fault
An extended CNN network using LSTM layers for temporal analysis	Random Fault

VI. Performance Evaluation

This section presents the experimental results for the proposed novel FL methods for fault detection and ensemble model for fault classification. The proposed FL methods namely FedAvg, Fed-SGD, Weighted Fed-Avg and Selective Fed-Delta having different strengths and each model is uniquely identified for specific task. First, we discussed the dataset used and the experimental conditions used to evaluate the performance. Then, we focused on the performance analysis of the proposed FL-based fault detection methods. To demonstrate the superiority of our proposed methods, we compare them with the standard approach. Additionally, we have analyzed and compared the effect on computation cost, training time, and overhead while performing the experiment. Finally, we evaluate the performance of the ensemble fault classification models and compare the performance with conventional models.

A. Dataset

To facilitate our study, we have utilized a temperature and light sensor dataset generously provided by the Creative Commons (CC) consortium [44]. This dataset encompasses readings collected from sensors manufactured by three distinct companies: Intel, Santander, and Sensorscope. It is pertinent to note that our analysis primarily revolves around temperature

sensor data, given its paramount importance in environmental monitoring and predictive modeling applications. The datasets employed for fault identification predominantly random faults (1), malfunction faults (2), encompass bias faults (4) and drift faults (8). In Table IV, we provide a detailed exposition of the dataset structure of mixed faults observed within an Intel sensor node, elucidating how the data is organized and the requisite labeling schema. The sensor nodes communicate in a decentralized manner, creating a decentralized training setup for performance evaluation.

Table IV: **Detailed Preview of Intel Sensor Dataset**

Timestamp	Mote_id	has_fault_type	Temperature	Light
2004-02-29 T00:00:00	1	0	19.26	45.08
2004-03-01 T15:50:30	1	4	37.597	97.52
2004-03-01 T21:36:30	1	8	0.045	39.5600
2004-03-02 T08:30:00	1	1	84.365	323.84
2004-03-02 T22:22:30	1	2	21.107	39.56

B. Experimental Setup for FL Framework

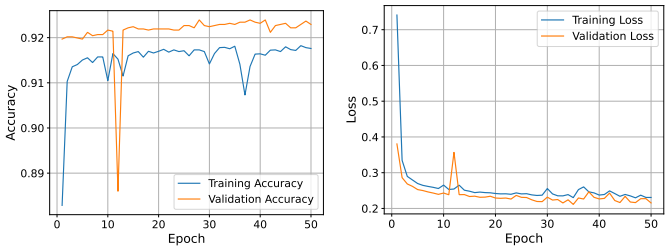
In the decentralized training setup, we employed an LSTM-CNN model to train on K nodes out of a total of N = 10 nodes in the network. Utilizing time series data recorded using Intel lab sensors [45], [46], we sampled data from the nodes and created datasets for each node. Each node possessed 20,160 temperature sensor recordings at one-minute intervals. We employed overlapping 120-minute windows, shifting by one minute each time, to create 20,041 time sequences per node. These sequences constituted the datasets for training, with each node's dataset containing 3,629 faults. We performed a 20% train-test split, resulting in 16,032 data points for training and 4,009 points for validation. Python and Keras served as our primary tools for model training and implementation.

We monitored GPU utilization using a python script leveraging the NVIDIA Management Library (NVML) which continuously tracked and recorded GPU utilization metrics at 0.5-second intervals. By running this script in a parallel thread, we obtained a comprehensive indicator of the computing power consumed during training. The communication overhead time is calculated based on the conventional channel capacity of 180 KBps and a standard buffer for queuing.

C. Performance Analysis of Proposed Fault Detection Approaches

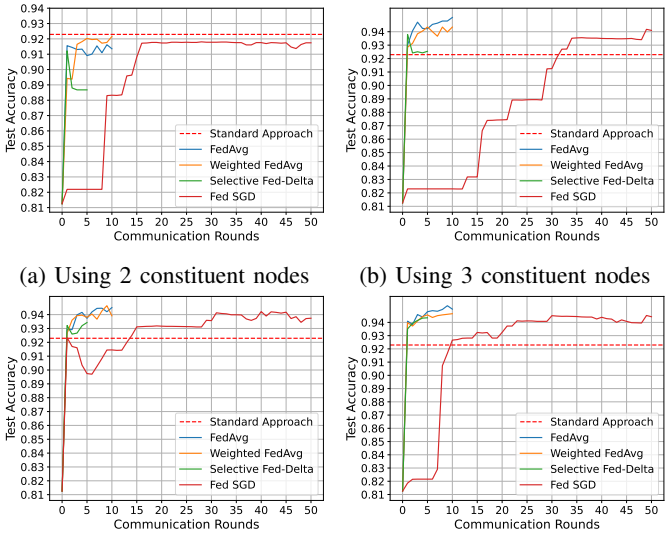
Through a comprehensive evaluation, we compared each of the meticulously implemented proposed FL methods, FedAvd, Fed SGD, Weighted FedAvg and Selective Fed-Delta (Algorithm 1 and Algorithm 2) in our study to the standard approach. First, we evaluate the performance of the model trained by the standard LSTM-CNN approach to compute the accuracy and loss during fault detection.

When trained using the standard approach of training the model, the fault detection model achieves a 92.29% local accuracy on its training node in fault detection tasks as shown in Fig. 4. The model accurately predicted faults in 395 out of 695 instances and correctly identified the absence of faults in 3,305 out of 3,314 instances of its node. Moreover, it also identified 3,969 faults out of 7,058 faults and correctly identified the absence of fault 32,893 times out of 33,032 globally across all 10 nodes for a total global accuracy of 91.95%, as depicted in Table VI. Its performance highlights its suitability for integration into federated learning protocols [47], [48], emphasizing reliability and efficacy in fault diagnosis within AgIoT environments [49], [50]. As, we have seen that this model is able to detect only 3,969 faults out of actual 7,058 faults which is only 56%. Hence, even the standard approach claimed overall good detection accuracy because the number of faults was very low, but it failed to provide correct results when the number of faults actually increases.



(a) LSTM-CNN Model Accuracy (b) LSTM-CNN Model Loss

Fig. 4: Training of the standard LSTM-CNN Model

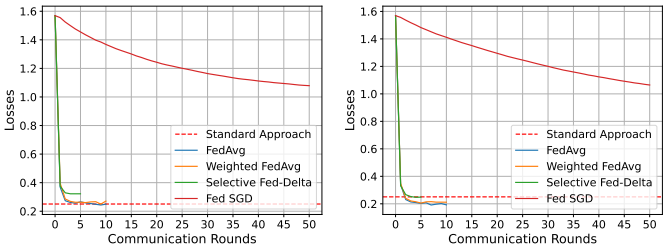


(a) Using 2 constituent nodes (b) Using 3 constituent nodes

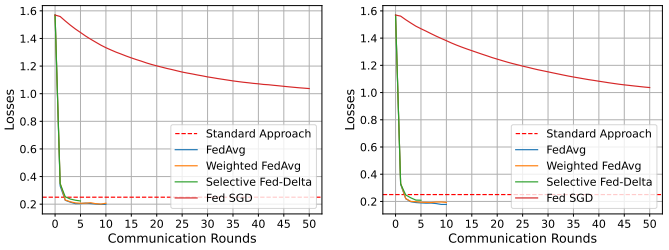
(c) Using 4 constituent nodes (d) Using 5 constituent nodes

Fig. 5: Comparison of Constituent Accuracies across K Nodes.

Now, we compare the standard approach to the performance of the FL approaches. Moreover, we have considered several key metrics for evaluating the performance of the FL methods, including constituent accuracy, global accuracy, constituent node losses, global losses, total training time, communication overhead, GPU utilization, and most importantly, the number of true positives (correct fault predictions in the network). Constituent accuracies and losses quantify the model's efficacy on the subset of K nodes engaged in training. In contrast,

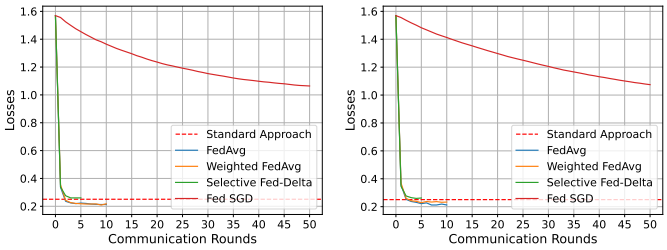


(a) Using 2 constituent nodes (b) Using 3 constituent nodes

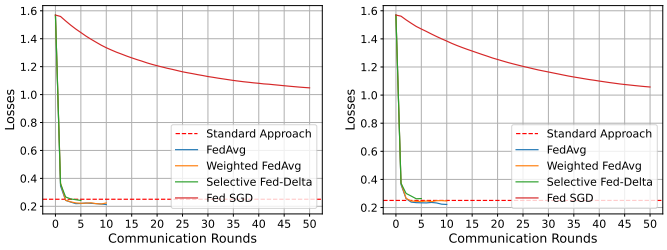


(c) Using 4 constituent nodes (d) Using 5 constituent nodes

Fig. 6: Comparison of Constituent Losses across K Nodes.

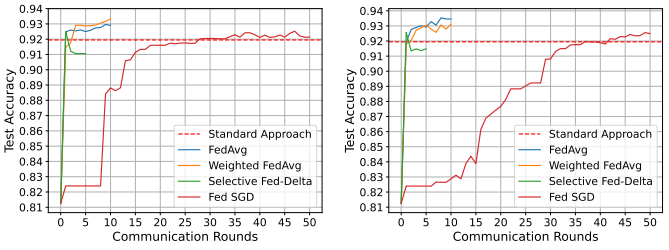


(a) Using 2 constituent nodes (b) Using 3 constituent nodes

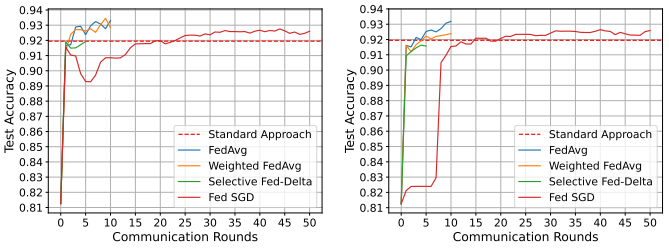


(c) Using 4 constituent nodes (d) Using 5 constituent nodes

Fig. 8: Comparison of Global Losses across all N nodes.



(a) Using 2 constituent nodes (b) Using 3 constituent nodes



(c) Using 4 constituent nodes (d) Using 5 constituent nodes

Fig. 7: Comparison of Total Global Accuracies across all N nodes.

global accuracies and losses evaluate its performance across the entire network of N nodes. These metrics are pivotal in assessing the model's overall performance in federated learning. Analyzing these metrics allows us to gauge training efficacy with varying numbers of participating nodes relative to using the entire network. Furthermore, we aim to compare the model's performance on familiar nodes with its performance on all nodes, encompassing those not previously encountered by the model.

We limit the training to a maximum of five nodes due to negligible performance differences observed between using four and five nodes, indicating convergence of model performance at $K = 5$. Furthermore, scaling beyond this threshold resulted in disproportionately higher resource consumption without commensurate training accuracy and correct fault detection gains across all existing and proposed methods. The accuracies and losses across different methods and varying numbers of K constituent nodes are depicted in Table V as well as Fig. 5, 6, 7 and 8. These figures are divided into four sub-figures and each subfigure depicts the effect on performance with respect to the number of communication rounds while increasing the number of training nodes. The performance of the standard model is the same in all subfigures because it trains the model at once with entire available data, so an increased number of training nodes does not affect its performance. Fig. 5 and 6 represents the local performance on consituent nodes only where model trained and tested on same nodes. Additionally, The training effect on all the nodes are shown in Fig. 7 and 8. It represents the global performance irrespective to the training nodes.

The accuracies and losses across different methods and varying numbers of K constituent nodes are depicted in Fig. 5 and 6, against the validation accuracy and loss of the standard approach evaluated on its training node. Fig. 5 represents the accuracies of the standard model and FL-based models. The accuracy of standard model remains at 92.67% for all the cases. However, the effect of increasing the number of training/constituent nodes can be seen in all subfigures. The training accuracy of FedAvg, FedSGD, Weighted FedAVG, Selective Fed-Delta models are 91.35%, 91.74%, 92.16% and 88.67% respectively on two constituent nodes as represented in Fig. 5a. Their performmance gradually improves when the number of constituent nodes increases as shown in Fig. 5b and 5c. Finally, Fig. 5d depicts the training effect where proposed methods demonstrated competitive constituent accuracies, achieving 94.65% and 94.35% for Weighted FedAvg and Selective Fed-Delta respectively, compared to 95% for FedAvg and 94.42% for FedSGD with 5 actively contributing nodes, their distinctive strength lies in their enhanced fault detection capabilities. However, FedAvg is outperforming other methods in terms of accuracy but required 15 communication round in the training but selective fed-delta is producing similar results

Table V: Performance Metrics of Different Federated Learning Algorithms in terms of efficiency metrics, including training time, communication overhead, computation cost, accuracies and losses

Algorithm	**Number of Constituent Nodes**	**Training Time**	**Communication Overhead**	**Computation Cost**	**Constituent Nodes' Accuracy**	**Constituent Nodes' Loss**	**Global Network Accuracy**	**Global Network Loss**
Standard Approach	–	123.9 s	NA	3918	92.29%	0.2154	91.95%	0.2504
FedSGD	2 nodes	99.53 s	42.35 s	20421.0	91.74%	1.0789	92.13%	1.0641
	3 nodes	153.51 s	45.88 s	22368.5	94.09%	1.0645	92.49%	1.0746
	4 nodes	210.24 s	49.41 s	28051.5	93.73%	1.0371	92.60%	1.0481
	5 nodes	268.91 s	52.94 s	31638.0	94.42%	1.0367	92.58%	1.0575
FedAvg	2 nodes	86.39 s	8.47 s	6022.5	91.35%	0.2493	92.90%	0.2130
	3 nodes	136.18 s	9.18 s	8189.5	95.06%	0.1924	93.45%	0.2122
	4 nodes	183.80 s	9.88 s	10494.5	94.50%	0.1992	93.31%	0.2133
	5 nodes	216.48 s	10.59 s	11334.5	95.00%	0.1779	93.18%	0.2219
Weighted FedAvg	2 nodes	89.37 s	8.47 s	6010.5	92.16%	0.2708	93.32%	0.2178
	3 nodes	156.94 s	9.18 s	9181.0	94.34%	0.2116	93.10%	0.2329
	4 nodes	177.94 s	9.88 s	10212.5	93.91%	0.2006	92.82%	0.2239
	5 nodes	237.27 s	10.59 s	11585.0	94.65%	0.1929	92.38%	0.2465
Selective Fed-Delta	2 nodes	129.23 s	7.06 s	6262.5	88.67%	0.3220	91.05%	0.2602
	3 nodes	196.09 s	10.59 s	7621.0	92.54%	0.2487	91.48%	0.2622
	4 nodes	263.55 s	14.17 s	10565.0	93.42%	0.2229	91.92%	0.2419
	5 nodes	336.16 s	17.65 s	12883.0	94.35%	0.2088	91.57%	0.2639

in 5 communication rounds only.

In addition to the constituent accuracy, the performance has been evaluated based on the constituent losses as shown in Fig. 6. It illustrates the effect of training on different nodes on FL-based models. We can see high losses when training the model on two constituent nodes (Fig. 6a). Further, achieved significant improvement when number of constituent nodes increases and reaches to 5 (Fig. 6d). In this case, FedAvg, Weighted FedAvg and Selective Fed-Delta have minimize respective losses 21.1%, 11.2% and 2.7% when compared with the standard approach. Here also, FedAvg is outperforming and achieved minimum losses when it fully trained on 5 constituent nodes.

Furthermore, we contrast these metrics among actively contributing nodes with the global accuracy and loss, which reflect the overall performance metrics across all network nodes, including both contributing and non-contributing ones. This comparison visualised in Fig. 7 and 8 provides insights into how network configuration impacts model performance on a broader scale. When we look in the results, the overall performance degrades a bit in terms of accuracy and losses when tested all the models on global network. This observation considers a practical case that models couldn't be trained on all available sites on the globe. But, when we test the performance on some other site where model is not trained, the performance may vary a bit.

We have demonstrated comparisons between the constituent accuracies and global accuracies in Fig. 9, as well as between constituent losses and global losses in Fig. 10. These comparisons were made for each approach analyzed using varying numbers of constituent nodes. This analysis was highlights the differences and overall global performance across all methods.

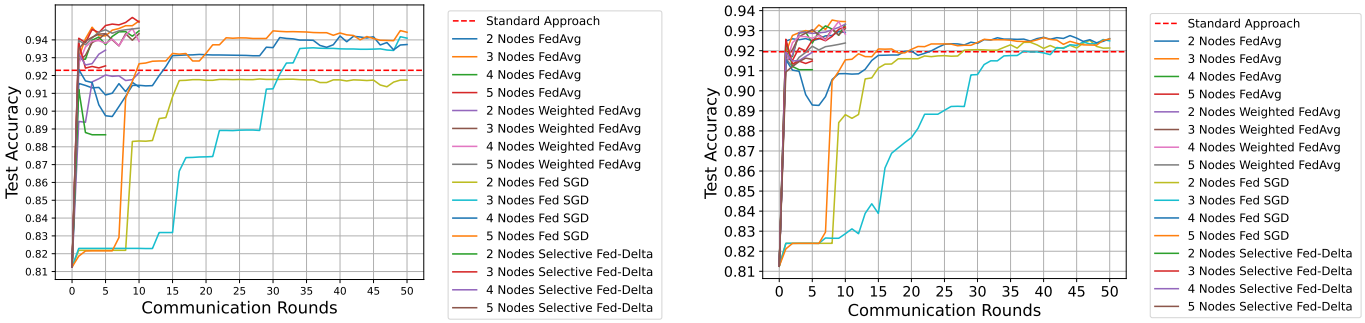
D. Effect of proposed FL methods on Fault Detection Capabilities, Cost, Training Time, and Communication Overhead

Fig. 12 provides a comprehensive analysis of the total training times, communication overhead times and relative computation costs consumed. The total training time of each framework represents the cumulative duration required for training across all nodes in that framework. Computation costs are the total expense associated with using processors over the entire monitoring period. Table V provides a comprehensive overview of the efficiency metrics, including training time, communication overhead, computation cost, accuracies and losses. Table VI details efficacy metrics as global prediction values of the correct fault detections out of the 7058 fault occurrences in the dataset.

We assessed the performance of our proposed Federated Learning methods using a sparse fault occurrence dataset, comprising only 7,058 fault occurrences out of 40,090 total data points. Weighted FedAvg and Selective Fed-Delta accurately identified 5,814 and 5,936 faults respectively, outperforming FedAvg, which detected 5,472 faults, and FedSGD, which accurately predicted 4,819 faults. These findings represent a significant advancement over the standard approach, which could only predict 3,969 faults out of the total 7,058 faults. Fig. 11 provides a graphical representation of these findings.

Table VI: Comparison of Correct Fault Predictions of Proposed Federated Learning Methods

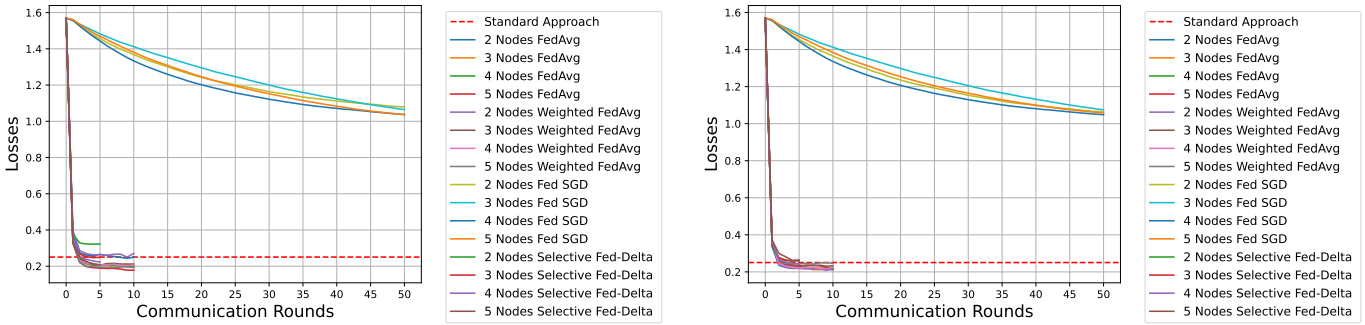
Algorithm	Number of Constituent Nodes	**Number of Correct Fault Predictions**
Standard Approach	-	**3,969**
FedSGD	2 nodes	**4,156**
	3 nodes	**4,404**
	4 nodes	**4,558**
	5 nodes	**4,819**
FedAvg	2 nodes	**4,741**
	3 nodes	**5,189**
	4 nodes	**5,409**
	5 nodes	**5,472**
Weighted FedAvg	2 nodes	**5,385**
	3 nodes	**5,302**
	4 nodes	**5,597**
	5 nodes	**5,814**
Selective Fed-Delta	2 nodes	**5,680**
	3 nodes	**5,621**
	4 nodes	**5,954**
	5 nodes	**5,936**



(a) Constituent Nodes' Accuracies analysis

(b) Global Accuracies analysis

Fig. 9: Comprehensive analysis of constituent and global node accuracies across different Federated Learning approaches.



(a) Constituent nodes' Losses analysis

(b) Global Losses analysis

Fig. 10: Comprehensive analysis of constituent and global node losses across different Federated Learning approaches.

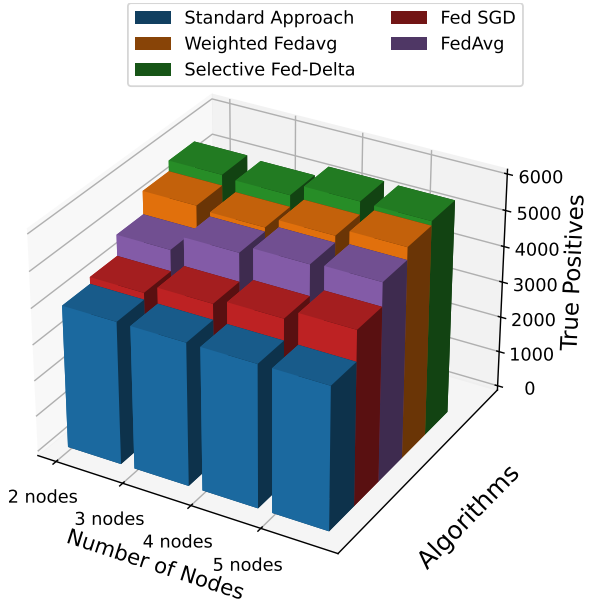


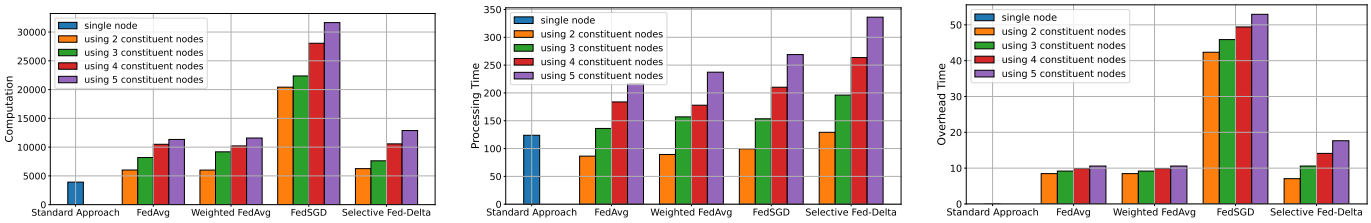
Fig. 11: Evaluation of the total number of correct fault predictions by each approach over the entire network of $N = 10$ nodes.

While our methods exhibit improved fault detection capabilities, achieving higher precision in identifying faults compared to FedAvg and FedSGD, there is a slight trade-off in global accuracy. Standard FedAvg achieves a global accuracy of 93.18%, and FedSGD has an accuracy of 92.58% with 5 actively contributing nodes, whereas our methods achieve 92.38% for Weighted FedAvg and 91.57% for Selective Fed-Delta. Despite this reduction, the enhanced fault detection ability underscores the relevance of our methods in scenarios where precise fault detection is paramount.

Furthermore, Tables V and VI, and Fig. 11 demonstrate that as the number of constituent nodes increases, both constituent accuracies and correct fault predictions improve. This underscores the scalability of our methods and their capacity to leverage data from additional nodes to enhance overall performance.

E. Performance Analysis of Ensemble Fault Classification Models

In previous subsections, we have analyzed the performance of fault detection methods in various aspects. After detecting the fault, it is imperative to find and analyze the exact class of the fault. Therefore, this subsection presents the performance evaluation of the proposed novel ensemble fault classification model. The performance of ensemble fault classification has been illustrated in Fig. 13 and Table VII where a mixed model can determine the type of fault with an average fault classification accuracy of 96%. The three-stacked RNN model identifies the drift fault and achieves fault classification accuracy close to 99%. Moreover, this model is able to complete the task in a maximum of 5 epochs only. Additionally, Extended CNN network + LSTM model achieves a fault classifications accuracy of 98.63%. This model could detect the random fault



(a) Computation cost across different approaches

(b) Total Training time of different approaches

(c) Communication overhead in different approaches

Fig. 12: Analysis of computation cost, total training time, and communication overhead across different Federated Learning approaches.

in a maximum of 15 epochs. Further, experimental results show that the biased fault and malfunctioning fault are also identified in 50 epochs, with more than 94% accuracy.

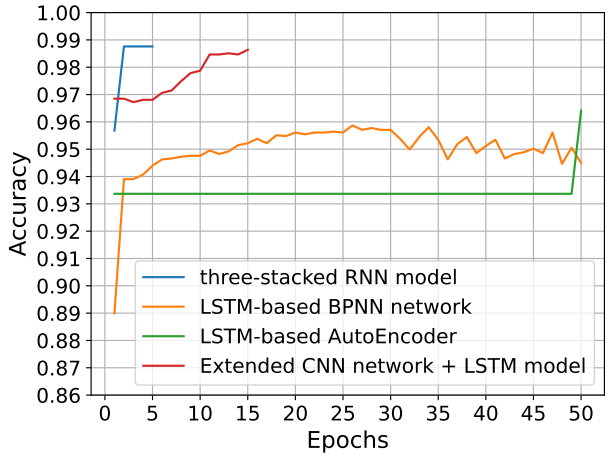


Fig. 13: Accuracies of the Ensemble models

Table VII: Performance of specialized models for Ensemble Fault Classification Model

Model Description	**Applicable Fault Type**	**Classification Accuracy**
LSTM-based AutoEncoder + binary Classification [41]–[43]	Bias Fault	96.40%
LSTM-based BPNN with wavelet denoising and sigmoid function	Malfunction (Hard) Fault	94.49%
3-stacked RNN with Batch Normalization, DropOut & Sigmoid function	Drift Fault	98.76%
Extended CNN network using LSTM	Random Fault	98.63%

On the other hand, Table VIII represents the performance of conventional fault diagnosis models where the maximum achievable accuracy is about 94%, but its lower limit is near 80% only. Therefore, a huge difference can be observed between convention model and proposed ensemble fault classification model. Hence, proposed fault classification models significantly outperform the conventional fault diagnosis models.

VII. Conclusion and Future Work

This research work presents innovative FL methodologies for fault diagnosis in dynamic AgIoT landscapes and offers a robust solution for precisely detecting faults and optimizing resource utilization. The proposed work addressed critical

Table VIII: Performance of Conventional Fault Classification Methodologies [3].

Model Description	**Applicable Fault Type**	**Classification Accuracy**
Linear AutoEncoder [6]	Drift Fault	91.06%
3-Layered Logistic Regression [17]	Bias Fault	79.97%
SVR model using a linear kernel and regularization [5]	Malfunction (Hard) Fault	82.07%
Bernoulli Restricted Boltzmann Machine+Logistic Regression [15]	Random Fault	82.12%
3 Stacked SimpleRNN Layers with a dense layer and sigmoid function [3]	Drift Fault	94.05%
Autoencoder binary classifier [3]	Bias Fault	91.37%
BPNN with wavelet denoising [3]	Malfunction (Hard) Fault	82.06%
DBF + CNN [3]	Random Fault	81.77%

shortcomings in fault detection when fault occurrences are sparse, as these methods can achieve high accuracies by showing many false negatives or ignoring numerous fault occurrences. This is a common challenge in real-world AgIoT scenarios; however, the timely detection of these faults is crucial. The proposed FedAvg framework achieves higher accuracy and significantly improves fault detection precision over the conventional approach. Enhancement in fault detection is critical for applications where identifying faults is more important than achieving the highest overall accuracy. Our methods address this challenge while showcasing competitive accuracies. The proposed Selective Fed-Delta method stands out for its exceptional fault detection precision, outperforming other methods even with fewer active nodes and precisely detecting almost twice the number of faults as the standard approach. Weighted FedAvg offers a balanced approach among the proposed FL methods, improving precision while maintaining high overall accuracy.

Moreover, the ensemble classification model for fault classification enhances accuracy and robustness by combining multiple classification models. Hence, the ensemble classifier offers improved performance and reliability by aggregating multiple predictions. This makes the ensemble classifier ideal for fault classification over complex environments such as AgIoT networks. The results underscore the scalability and adaptability of our proposed methods, as demonstrated by increasing constituent accuracies and correct fault predictions with an expanding number of constituent nodes. Thus, it provides a framework that can be adapted to various fault scenarios beyond AgIoT networks.

Looking ahead, our research paves the way for further advancements in FL-based fault diagnosis methodologies. Future works can explore adaptive FL strategies that automatically adjust to differing environmental conditions and node distributions. The FL frameworks can be expanded to better manage and utilize heterogeneous data. Conducting longitudinal studies will be essential to understand the efficacy of FL models in the IoT fields. The future scope also involves more advanced integration techniques for ensemble classifiers to improve the efficiency and collaborative performance of the local network.

REFERENCES

[1] Mohamed Torky and Aboul Ella Hassanein. Integrating blockchain and the internet of things in precision agriculture: Analysis, opportunities, and challenges. *Computers and Electronics in Agriculture*, 178:105476, 2020.

[2] Tamoghna Ojha, Sudip Misra, and Narendra Singh Raghuwanshi. Internet of things for agricultural applications: The state of the art. *IEEE Internet of Things Journal*, 8(14):10973–10997, 2021.

[3] X. Zou, W. Liu, Z. Huo, S. Wang, Z. Chen, C. Xin, Y. Bai, Z. Liang, Y. Gong, Y. Qian, and L. Shu. Current status and prospects of research on sensor fault diagnosis of agricultural internet of things. *Sensors*, 23(5), 2023.

[4] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3347–3366, 2023.

[5] F. Deng, S. Guo, R. Zhou, and J. Chen. Sensor multifault diagnosis with improved support vector machines. *IEEE Transactions on Automation Science and Engineering*, 14:1053–1063, 2017.

[6] J. Loy-Benitez, Q. Li, K. J. Nam, and C. K. Yoo. Sustainable subway indoor air quality monitoring and fault-tolerant ventilation control using a sparse autoencoder-driven sensor self-validation. *Sustainable Cities and Society*, 52:101847, 2020.

[7] Bill CP Lau, Eden WM Ma, and Tommy WS Chow. Probabilistic fault detector for wireless sensor network. *Expert Systems with Applications*, 41(8), 2014.

[8] Amina Antonacci, Fabiana Arduini, Danila Moscone, Giuseppe Palleschi, and Viviana Scognamiglio. Nanostructured (bio) sensors for smart agriculture. *TrAC Trends in Analytical Chemistry*, 98:95–103, 2018.

[9] Martin Holmberg, Fabrizio AM Davide, Corrado Di Natale, Arnaldo D'Amico, Fredrik Winquist, and Ingemar Lundström. Drift counteraction in odour recognition applications: lifelong calibration method. *Sensors and Actuators B: Chemical*, 42(3):185–194, 1997.

[10] Daoliang Li, Ying Wang, Jinxing Wang, Cong Wang, and Yanqing Duan. Recent advances in sensor fault diagnosis: A review. *Sensors and Actuators A: Physical*, 309:111990, 2020.

[11] Han Qiu Zhang and Yong Yan. A wavelet-based approach to abrupt fault detection and diagnosis of sensors. *IEEE Transactions on Instrumentation and Measurement*, 50(5):1389–1396, 2001.

[12] Zhiwen Chen, Chunhua Yang, Tao Peng, Hanbing Dan, Changgeng Li, and Weihua Gui. A cumulative canonical correlation analysis-based sensor precision degradation detection method. *IEEE Transactions on Industrial Electronics*, 66(8):6321–6330, 2018.

[13] Sana Ullah Jan, Young-Doo Lee, Jungpil Shin, and Insoo Koo. Sensor fault classification based on support vector machine and statistical time-domain features. *IEEE Access*, 5:8682–8690, 2017.

[14] Jun Hu, Zidong Wang, and Huijun Gao. Joint state and fault estimation for time-varying nonlinear systems with randomly occurring faults and sensor saturations. *Automatica*, 97:150–160, 2018.

[15] Y. Wang, Z. Pan, X. Yuan, C. Yang, and W. Gui. A novel deep learning based fault diagnosis approach for chemical process with extended deep belief network. *ISA Transactions*, 96:457–467, 2020.

[16] J. Zhou and P. Zhang. Hard fault isolation of gps navigation system based on gray prediction for agricultural robot. *Trans. Chin. Soc. Agric.*, 41, 2010.

[17] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[18] J. Kullaa. Detection, identification, and quantification of sensor fault in a sensor network. *Mech. Syst. Signal Process.*, 40, 2013.

[19] Shangjun Ma, Wenkai Liu, Wei Cai, Zhaowei Shang, and Geng Liu. Lightweight deep residual cnn for fault diagnosis of rotating machinery based on depthwise separable convolutions. *IEEE Access*, 7, 2019.

[20] Xing Yang, Lei Shu, Kailiang Li, Zhiqiang Huo, and Yu Zhang. Sa1d-cnn: A separable and attention based lightweight sensor fault diagnosis method for solar insecticidal lamp internet of things. *IEEE Open Journal of the Industrial Electronics Society*, 3, 2022.

[21] Chun Lo, Jerome P Lynch, and Mingyan Liu. Reference-free detection of spike faults in wireless sensor networks. In *2011 4th International Symposium on Resilient Control Systems*. IEEE, 2011.

[22] Salah Zidi, Tarek Moulahi, and Bechir Alaya. Fault detection in wireless sensor networks through svm classifier. *IEEE Sensors Journal*, 18(1), 2017.

[23] Sana Ullah Jan, Young Doo Lee, and In Soo Koo. A distributed sensor-fault detection and diagnosis framework using machine learning. *Information Sciences*, 547, 2021.

[24] Indrajit Banerjee, Prasenjit Chanak, Hafizur Rahaman, and Tuhina Samanta. Effective fault detection and routing scheme for wireless sensor networks. *Computers & Electrical Engineering*, 40(2), 2014.

[25] Hedde HWJ Bosman, Giovanni Iacca, Arturo Tejada, Heinrich J Wörtche, and Antonio Liotta. Ensembles of incremental learners to detect anomalies in ad hoc sensor networks. *ad hoc networks*, 35, 2015.

[26] Mien Van, Shuzhi Sam Ge, and Dariusz Ceglarek. Fault estimation and accommodation for virtual sensor bias fault in image-based visual servoing using particle filter. *IEEE Transactions on Industrial Informatics*, 14(4), 2017.

[27] Rui Xiong, Quanqing Yu, Weixiang Shen, Cheng Lin, and Fengchun Sun. A sensor fault diagnosis method for a lithium-ion battery pack in electric vehicles. *IEEE Transactions on Power Electronics*, 34(10), 2019.

[28] Yong Yu, Yongzheng Zhao, Bo Wang, Xiaolei Huang, and Dianguo Xu. Current sensor fault diagnosis and tolerant control for vsi-based induction motor drives. *IEEE Transactions on Power Electronics*, 33(5), 2017.

[29] Wu Deng, Zhongxian Li, Xinyan Li, Huayue Chen, and Huimin Zhao. Compound fault diagnosis using optimized mckd and sparse representation for rolling bearings. *IEEE Transactions on Instrumentation and Measurement*, 71, 2022.

[30] Hongjiang Cui, Ying Guan, and Huayue Chen. Rolling element fault diagnosis based on vmd and sensitivity mckd. *IEEE Access*, 9, 2021.

[31] Forrest Iandola and Kurt Keutzer. Small neural nets are beautiful: enabling embedded systems with small deep-neural-network architectures. In *Proceedings of the Twelfth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis Companion*, pages 1–10, 2017.

[32] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[33] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[34] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

[35] Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, and Daniel J Inman. 1d convolutional neural networks and applications: A survey. *Mechanical systems and signal processing*, 151, 2021.

[36] Yaseen Ahmed Mohammed Alsumaidaee, Chong Tak Yaw, Siaw Paw Koh, Sieh Kiong Tiong, Chai Phing Chen, Talal Yusaf, Ahmed N Abdalla, Kharudin Ali, and Avinash Ashwin Raj. Detection of corona faults in switchgear by using 1d-cnn, lstm, and 1d-cnn-lstm methods. *Sensors (Basel, Switzerland)*, 23, 2023.

[37] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31, 2019.

[38] Alireza M Javid, Sandipan Das, Mikael Skoglund, and Saikat Chatterjee. A relu dense layer to improve the performance of neural networks. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021.

[39] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data, 2023.

[40] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[41] H Du Nguyen, Kim Phuc Tran, Sébastien Thomassey, and Moez Hamad. Forecasting and anomaly detection approaches using lstm and lstm autoencoder techniques with the applications in supply chain management. *International Journal of Information Management*, 57:102282, 2021.

[42] Pooyan Mobtahej, Xulong Zhang, Maryam Hamidi, Jing Zhang, et al. An lstm-autoencoder architecture for anomaly detection applied on compressors audio data. *Computational and Mathematical Methods*, 2022, 2022.

[43] Mahmoud Said Elsayed, Nhien-An Le-Khac, Soumyabrata Dev, and Anca Delia Jurcut. Network anomaly detection using lstm based autoencoder. In *Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, pages 37–45, 2020.

[44] Creative Commons Corporation. Creative Commons Dataset. https://www.dropbox.com/s/l98aln9g87caatk/datasets-release-v3.zip?dl=0file˙subpath=[Online; accessed 30-March-2024].

[45] Bas de Bruijn, Tuan Anh Nguyen, Doina Bucur, and Kenji Tei. Benchmark datasets for fault detection and classification in sensor data. In *5th International Conference on Sensor Networks*, 2016.

[46] Intel Berkeley Research lab. Intel Lab Data. https://db.csail.mit.edu/labdata/labdata.html, 2004. [Online; accessed 30-March-2024].

[47] J. Liu, J. Huang, Y. Zhou, and et al. From distributed machine learning to federated learning: a survey. *Knowl Inf Syst*, 64(4), 2022.

[48] Z. Cui, J. Wen, Y. Lan, and et al. Communication-efficient federated recommendation model based on many-objective evolutionary algorithm. *Expert Syst Appl*, 201, 2022.

[49] H. Zhou, G. Yang, H. Dai, and G. Liu. PFLF: Privacy-preserving federated learning framework for edge computing. *IEEE Trans Inf Forensics Secur*, 17, 2022.

[50] A. Li, L. Zhang, J. Wang, F. Han, and X. Li. Privacy-preserving efficient federated-learning model debugging. *IEEE Trans Parallel Distrib Syst*, 33(10), 2022.

Parthasarthi Aggarwal received his B.Tech. degree in Electronics Engineering with a minor in Language and Communication from the Indian Institute of Technology (BHU), Varanasi, Uttar Pradesh. He is currently working as a Member of Technical Staff at Oracle. During his undergraduate tenure at IIT BHU, he actively participated in numerous research projects, focusing on the Internet of Things (IoT), Electronics, and Ubiquitous Computing across various laboratories. His research on biomedical wearable electronics was presented at the IEEE ECBIOS conference in 2024. His primary research involved IoT, Embedded System Design, Ubiquitous Computing, and Machine Learning, with a focus on real-life applications of AI in embedded system design and machine learning. Additionally, he has worked on projects involving soft computing, wireless technologies, and computer vision.

Jubin Banerjee received his B.Tech. degree in Electronics Engineering from the Indian Institute of Technology (BHU), Varanasi, Uttar Pradesh. He is currently employed as a Software Engineer at Microsoft, where he contributes to the development of innovative technologies. During his undergraduate studies at IIT BHU, he actively engaged in multiple research projects, gaining substantial experience in the fields of Internet of Things (IoT), Electronics, and Ubiquitous Computing across various laboratories. His research involved integrating IoT with wireless technologies, advancing embedded system design, and applying machine learning and artificial intelligence. Jubin's primary research interests encompass wireless technologies, IoT, ubiquitous computing, embedded system design, and machine learning, with a main focus on Natural Language Processing and Computer Vision.

Naveen Kumar Gupta is currently working as Assistant Professor in Department of Information Technology, Dr B R Ambedkar National Institute of Technology Jalandhar, Jalandhar, India. He also worked as Assistant Professor in Jaypee Institute of Information Technology, Noida, India. He completed his doctorate from Motilal Nehru National Institute of Technology Allahabad, Prayagraj, India in 2020. He received his B. Tech. degree in Computer Science and Engineering from Inderprastha Engineering College Ghaziabad, India in 2011 and M. Tech in Information Technology from Madan Mohan Malaviya University of Technology Gorakhpur, India in 2014. He has five years of teaching experience. He has published various research papers in reputed international journals and conferences. His main research interest lies in Wireless Sensor Networks, IoT, Ad hoc Routing protocols and Blockchain Technology. He is a member of IEEE and serves as a regular reviewer for various reputed journals including IEEE Transactions on Network and Service Management, international journal of communication systems, computer communications, journal of supercomputing, ad hoc & sensor wireless networks.

Om Jee Pandey received the B.Tech. degree in electronics and communication engineering from Uttar Pradesh Technical University, Lucknow, India, in 2008, the M.Tech. degree in digital communications from the ABV-Indian Institute of Information Technology and Management, Gwalior, India, in 2013, and the Ph.D. degree from the Department of Electrical Engineering, Indian Institute of Technology Kanpur, Kanpur, India, in 2019. He worked as a Postdoctoral Fellow with the Communications Theories Research Group, Department of Electrical and Computer Engineering, University of Saskatchewan, Saskatoon, SK, Canada. From 2008 to 2011, he worked with Escorts Ltd., and FANUC India Private Ltd. He was a Senior Lecturer with the Jaipur Engineering College and Research Center, Jaipur, from 2013 to 2014. He is currently working as an Assistant Professor with the Department of Electronics Engineering, Indian Institute of Technology (BHU), Varsnasi, Uttar Pradesh. His research interests include wireless sensor networks, low-power wide-area networks, unmanned aerial vehicle networks, mobile and pervasive computing, cyber physical systems and Internet of Things, cloud and fog computing, UAV-assisted optical communications, and social networks. He is a senior member of IEEE and serves as a regular reviewer for various reputed journals including computer communications, computer networks, pervasive & mobile computing and Ad Hoc Networks.