International Journal of Informatics and Computation (IJICOM)
Vol. 8, No. 1, 2026
ISSN: 2685-8711, E-ISSN: 2714-5263 DOI: 10.35842/ijicom

Anomaly Detection in Sensor Data for Tomato Farming Using a Federated Learning - Autoencoder

Muhammad Rizki[1], Muh. Subhan[2], Akhiyar Waladi[3]

Abstract
Anomalies in tomato farming sensor data may interfere with soil condition monitoring and decision-making processes. This study developed a Federated Learning-Autoencoder model to detect anomalies in tomato farming soil sensor data. The public dataset consisted of three clients or sensor lines with humidity, temperature, and electrical conductivity parameters. The research stages included preprocessing, local model training, Federated Learning model training using Federated Averaging, model evaluation, and dashboard-based monitoring simulation. The evaluation used a confusion matrix, precision, recall, and F1-score. The local L01 P95 model obtained an average precision of 1.0000, an average recall of 0.8291, and an average F1-score of 0.9061. The Federated Learning F01 P95 model obtained an average precision of 1.0000, an average recall of 0.7865, and an average F1-score of 0.8803. The results showed that the Federated Learning model detected spike anomalies competitively without combining raw data among clients.

Keywords:
Anomaly Detection, Federated Learning, Autoencoder, Soil Sensor, Tomato Farming

1. Introduction

The rapid development of precision agriculture has transformed conventional farming into a data-driven ecosystem supported by Internet of Things (IoT) technologies. Modern agricultural systems increasingly deploy environmental and soil sensors to continuously monitor critical parameters such as soil moisture, temperature, humidity, and electrical conductivity. These sensor networks enable farmers to optimize irrigation, improve resource utilization, and increase crop productivity through timely decision-making. Tomato cultivation particularly benefits from IoT-based monitoring because tomato plants are highly sensitive to environmental fluctuations and irrigation management. Recent studies demonstrate that integrating smart sensors, cloud platforms, and machine learning technologies significantly improves agricultural efficiency and sustainability. However, the growing dependence on sensor-generated data also increases the importance of ensuring data quality and reliability throughout the monitoring process. [1], [2], [4], [11].

Tomato farming represents one of the most important applications of smart agriculture because tomato yield and fruit quality strongly depend on precise environmental control. Researchers have developed various IoT-based systems that continuously collect soil and climate data to support irrigation scheduling, disease prevention, and yield optimization. Sensor-guided irrigation systems have shown substantial improvements in water-use

Corresponding Author: Muhammad Rizki (rizkitwelve12@gmail.com)
1 Muhammad Rizki, Information Systems, Faculty of Science and Technology, Universitas Jambi, Indonesia, rizkitwelve12@gmail.com
2 Muh. Subhan, Information Systems, Faculty of Science and Technology, Universitas Jambi, Indonesia, muh.subhan@unja.ac.id
3 Akhiyar Waladi, Information Systems, Faculty of Science and Technology, Universitas Jambi, Indonesia, akhiyar.waladi@unja.ac.id

efficiency and crop performance, while machine learning-based decision support systems assist farmers in maintaining optimal growing conditions. Furthermore, recent tomato farming datasets include multivariate measurements collected under different irrigation regimes, creating valuable opportunities for data-driven analytics. Despite these advances, sensor networks deployed in agricultural environments remain vulnerable to hardware failures, communication interruptions, environmental interference, and calibration drift, all of which may generate abnormal readings that negatively affect management decisions. [2], [5], [11], [12], [19], [20].

The reliability of IoT sensor data has become a critical issue in smart farming environments. Agricultural sensor networks operate under harsh field conditions characterized by fluctuating temperatures, humidity variations, dust exposure, and unstable communication channels. These conditions often produce anomalous observations such as sudden spikes, missing values, out-of-range measurements, and inconsistent temporal patterns. When agricultural management systems utilize such corrupted data without validation, they may generate inaccurate recommendations regarding irrigation, fertilization, or disease prevention. In tomato cultivation, even small deviations in sensor readings can lead to inappropriate water management strategies that directly affect plant growth and productivity. Consequently, effective anomaly detection mechanisms are required to ensure the trustworthiness of sensor-generated information before it is used for decision support and automation. [1], [3], [4], [13].

Traditional anomaly detection approaches often rely on statistical thresholds or rule-based methods that require extensive domain knowledge and manual configuration. These approaches frequently struggle to capture complex nonlinear relationships among multiple environmental variables collected from modern agricultural IoT systems. As sensor networks become increasingly sophisticated and generate large volumes of multivariate time-series data, conventional methods face limitations in identifying subtle or previously unseen anomalies. Recent surveys on anomaly detection highlight the growing need for intelligent techniques capable of learning normal operational patterns directly from data without requiring extensive labeling efforts. This challenge becomes even more significant in agricultural environments where obtaining comprehensive anomaly labels is difficult, costly, and often impractical. Therefore, unsupervised learning approaches have emerged as promising alternatives for sensor anomaly detection applications. [13], [14].

Among unsupervised learning methods, Autoencoders have gained considerable attention because of their ability to learn compressed representations of normal data patterns and identify anomalies through reconstruction errors. Autoencoders train neural networks to reconstruct input data, allowing the model to capture latent structures and correlations among sensor variables. During operation, abnormal observations generally produce larger reconstruction errors because they deviate from the learned normal behavior. Several studies have demonstrated the effectiveness of Autoencoder-based methods in IoT anomaly detection scenarios, including smart farming environments. Recent advances in deep Autoencoders and latent space optimization further improve anomaly detection performance by enhancing feature extraction capabilities and reducing false detection rates. These characteristics make Autoencoders highly suitable for detecting anomalies in multivariate agricultural sensor datasets where normal operating conditions dominate the data distribution. [3], [6], [7], [15].

Although Autoencoder-based anomaly detection has shown promising results, most existing implementations assume centralized data collection architectures. In practical agricultural deployments, sensor data are often distributed across multiple farms, greenhouses, irrigation zones, or geographical locations. Centralizing all sensor data may introduce challenges related to data ownership, privacy concerns, communication costs, and storage requirements. The increasing scale of smart agriculture systems therefore necessitates distributed learning frameworks capable of utilizing decentralized data while

preserving local autonomy. This issue becomes particularly relevant for collaborative agricultural monitoring systems where multiple farms seek to benefit from collective intelligence without exposing sensitive operational data. Consequently, researchers have started exploring distributed machine learning paradigms that address these limitations. [8], [9], [17], [18].

Federated Learning (FL) offers a promising solution to distributed agricultural analytics by enabling collaborative model training without transferring raw data to a central server. Introduced as a communication-efficient framework for decentralized machine learning, FL allows participating clients to train local models and share only model parameters during aggregation. Recent studies demonstrate that FL can achieve competitive anomaly detection performance while maintaining data privacy and reducing communication overhead. Furthermore, federated approaches have shown effectiveness in IoT-edge environments characterized by distributed sensor infrastructures. In agricultural contexts, hierarchical and personalized federated learning architectures have been proposed to support crop monitoring across multiple farms and heterogeneous sensing environments. These developments suggest that FL provides a suitable foundation for building scalable anomaly detection systems in precision agriculture. [8], [9], [10], [16], [17], [18].

Despite the growing interest in both Autoencoder-based anomaly detection and Federated Learning, research combining these two approaches for tomato farming sensor data remains limited. Existing agricultural studies primarily focus on irrigation optimization, disease detection, crop monitoring, or yield prediction, while anomaly detection often receives less attention. Similarly, many anomaly detection studies evaluate Autoencoders or federated frameworks using generic IoT datasets rather than real agricultural sensor measurements collected under varying irrigation conditions. Moreover, few studies investigate how anomaly detection outputs can be translated into actionable monitoring information for end users. Therefore, a significant research gap exists in developing a Federated Learning–Autoencoder framework capable of detecting anomalies in distributed tomato farming sensor data while preserving data locality and supporting practical monitoring applications. Addressing this gap can improve the reliability of smart farming systems and enhance decision-making processes in precision tomato cultivation. [2], [3], [5], [8], [10], [18], [20].

2. Related Works

Several studies investigated the application of IoT technologies in precision agriculture. Mansoor et al. [1] reviewed the integration of smart sensors, IoT platforms, and data analytics in agricultural environments. They reported that sensor networks improved real-time monitoring and supported data-driven farming decisions. The study highlighted the importance of continuous sensing for irrigation and crop management. However, it also identified data quality, sensor reliability, and interoperability as major challenges. The review provided a broad overview but did not propose a specific anomaly detection mechanism for agricultural sensor data.

Researchers also applied IoT sensing and machine learning to tomato farming applications. Martelli et al. [2] developed a machine learning framework for smart irrigation management in processing tomato cultivation. Their approach utilized environmental and soil measurements to optimize irrigation decisions and improve water-use efficiency. Similarly, Dirlik et al. [11] evaluated sensor-guided irrigation strategies in greenhouse tomato production and demonstrated the benefits of soil moisture monitoring. These studies showed the value of sensor-based decision support systems. However, they assumed that sensor measurements were reliable and did not address the presence of anomalous or corrupted sensor readings.

Several studies focused on IoT-enabled monitoring systems for tomato cultivation. Kakakhel et al. [19] developed a smart farming system that collected temperature, humidity,

and soil moisture data to detect tomato leaf curl disease. Their system achieved high classification accuracy using machine learning algorithms. Swathika et al. [20] proposed a real-time IoT-machine learning pipeline for irrigation optimization, disease detection, and yield prediction. Both studies demonstrated the effectiveness of IoT and machine learning in tomato farming. Nevertheless, they concentrated on predictive tasks and disease identification rather than anomaly detection in sensor data streams.

Autoencoder-based anomaly detection has received considerable attention in smart farming environments. Adkisson et al. [3] proposed an Autoencoder-based anomaly detection framework for a smart farming ecosystem. The model learned normal sensor behavior and identified anomalies using reconstruction errors. The study showed that Autoencoders effectively detected abnormal sensor patterns without requiring labeled anomaly data. This capability represented a significant advantage for agricultural datasets where anomaly labels are often unavailable. However, the framework relied on centralized training and did not consider distributed sensor deployments.

Recent advances further improved Autoencoder-based anomaly detection methods. Chen and Guo [6] reviewed Autoencoder architectures and discussed their effectiveness in feature learning and unsupervised anomaly detection. Rhachi et al. [7] enhanced anomaly detection performance using deep Autoencoders combined with feature selection techniques in IoT networks. Walczyna et al. [15] explored latent space manipulation to improve anomaly detection accuracy. These studies demonstrated that Autoencoders captured complex nonlinear relationships within multivariate sensor data. Despite their strong performance, the studies focused primarily on general IoT scenarios and did not evaluate agricultural datasets under distributed learning settings.

Researchers also examined anomaly detection in multivariate IoT time-series data. Belay et al. [13] reviewed unsupervised anomaly detection approaches and compared their performance across different IoT applications. Wang et al. [14] presented a comprehensive survey of deep anomaly detection techniques for multivariate time-series analysis. Both studies emphasized that deep learning models outperformed conventional statistical methods when handling complex sensor patterns. They also highlighted the growing importance of anomaly detection in IoT systems. However, neither study specifically addressed tomato farming environments nor investigated federated learning frameworks.

Federated Learning emerged as a promising solution for decentralized machine learning. McMahan et al. [16] introduced the Federated Averaging (FedAvg) algorithm and demonstrated that distributed clients could collaboratively train models without sharing raw data. Later, Florencia and Mayer [9] applied Federated Learning to anomaly detection tasks involving distributed data sources. Xiang et al. [10] further developed a federated anomaly detection framework for IoT-edge environments. These studies showed that Federated Learning preserved data privacy while maintaining competitive model performance. However, they focused on generic IoT and network applications rather than agricultural sensor monitoring.

Recent agricultural studies explored Federated Learning for crop monitoring. Devaraj et al. [18] proposed the RuralAI architecture that combined distributed sensing, edge computing, and hierarchical Federated Learning for tomato crop health monitoring. Their framework enabled collaboration among geographically distributed farms while preserving local data ownership. The study demonstrated the feasibility of federated approaches in agricultural environments. Nevertheless, the work focused on crop health assessment and personalized learning rather than sensor anomaly detection. Furthermore, existing studies rarely integrated Federated Learning with Autoencoder-based anomaly detection for tomato farming datasets collected under different irrigation conditions. This limitation motivated the development of the proposed Federated Learning–Autoencoder framework for anomaly detection in distributed tomato farming sensor data.

3. Proposed Method

3.1 Research Design

This study adopted a quantitative approach and an experimental method. We used the quantitative approach to measure model performance using numerical metrics. We applied the experimental method to evaluate the capability of the Federated Learning–Autoencoder model in detecting anomalies in IoT sensor data from tomato farming. The study focused on model development, performance evaluation, and the visualization of detection results through a dashboard-based monitoring simulation. We conducted the research in several sequential stages. First, we reviewed relevant literature and collected the tomato farming IoT sensor dataset. Next, we performed data understanding and preprocessing. We then trained local Autoencoder models on each client and aggregated the model parameters using the Federated Learning approach. Afterward, we evaluated the global model using predefined performance metrics. Finally, we implemented the best-performing model in a monitoring dashboard simulation to display anomaly detection results. Each client participated in both local training and federated aggregation processes.

3.2 Data Preprocessing

We conducted data preprocessing to prepare the dataset for model training and evaluation. First, we separated the dataset into three clients based on the sensor lines, where Line 1, Line 2, and Line 3 represented Client 1, Client 2, and Client 3, respectively. This study selected this configuration because each line reflected a different irrigation condition and suited the Federated Learning scenario. Next, this experiment cleaned the data by checking for missing values, duplicate records, and format inconsistencies, and we removed records with incomplete sensor measurements. We then selected humidity, temperature, and electrical conductivity (EC) as the input features because these variables represented key soil conditions in tomato cultivation. Afterward, we divided the data from each client into training and testing sets using an 80:20 ratio. The training data were used to learn normal sensor patterns, while the testing data were used to evaluate anomaly detection performance. Finally, we normalized all features using RobustScaler to balance their scales and reduce the influence of extreme values before training the local and federated models.

3.3 Federated Learning-Autoencoder Model

This experiment utilized an Autoencoder model to learn normal patterns in humidity, temperature, and electrical conductivity (EC) data. The model reconstructed the input data, and we used the reconstruction error between the original and reconstructed values to detect anomalies. We designed a lightweight architecture to match the three input features and support efficient implementation in the Federated Learning environment. Table 1 presents the Autoencoder architecture.

Table 1. Autoencoder Architecture

Layer (Type)	Output Shape	Param #
input (InputLayer)	(None, 3)	0
enc_dense_1 (Dense)	(None, 8)	32
dropout_1 (Dropout)	(None, 8)	0
bottleneck (Dense)	(None, 2)	18
dec_dense_1 (Dense)	(None, 8)	24
dropout_2 (Dropout)	(None, 8)	0
output (Dense)	(None, 3)	27
Total params	/	101
Trainable params	/	101
Non-trainable params	/	0

Based on Table 1, the model processed three sensor features through the encoder, bottleneck, and decoder layers to reconstruct the input data. We used the same architecture for all clients to support Federated Learning aggregation. We conducted two training scenarios: local training on each client's data and Federated Learning training using FedAvg aggregation without sharing raw data. We then compared the anomaly detection performance of the local and federated models.

3.4 Spike Anomaly Detection and Evaluation Metrics

Spike anomaly detection was performed based on reconstruction error, which represents the difference between the original sensor value and the reconstructed value produced by the Autoencoder. A higher reconstruction error indicates a greater possibility that the data point deviates from the learned normal pattern. The reconstruction error for each feature is calculated using (1).

$$E_{ij} = \left(x_{ij} - \hat{x}_{ij}\right)^2 \quad (1)$$

In (1), E_{ij} is the error value of feature j in data point i, x_{ij} is the original value of feature j, and $\hat{x}_{ij}$ is the reconstructed value of feature j. A data point was categorized as a spike anomaly when at least one feature had a reconstruction error above the threshold. If all error values were below or equal to the threshold, the data point was categorized as normal.

Threshold values were determined using percentile thresholds P95, P97, and P99 to test model sensitivity. A lower threshold made the model more sensitive, whereas a higher threshold made the model more selective. Because the dataset did not contain actual anomaly labels, the test data were given 5% synthetic spike anomaly injection on each client as the evaluation ground truth. The injection was carried out by randomly selecting data points in the test data and adding a disturbance to one sensor feature as an increase or decrease of k x IQR based on each client training data, with k = 3 and non-negative final values. This technique is commonly used in IoT sensor data that do not have anomaly labels [13]. Spike anomalies represent sudden changes in sensor values at one time point [14]. Normal data were labeled 0, whereas anomaly data were labeled 1.

Model evaluation used a confusion matrix, precision, recall, and F1-score. The confusion matrix was used to observe classification results based on four conditions: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).

Table 2. Confusion Matrix

	Predicted Spike	**Predicted Normal**
Actual Spike	True Positive (TP)	False Negative (FN)
Actual Normal	False Positive (FP)	True Negative (TN)

In this study, TP indicated correctly detected spike anomalies, TN indicated correctly detected normal data, FP indicated normal data incorrectly classified as anomalies, and FN indicated spike anomalies that were not detected. The evaluation was conducted on both local models and Federated Learning models to compare their detection performance. The best model was selected based on the balance among F1-score, precision, and recall, as well as the stability of performance across different threshold values. Out-of-range

anomaly detection was not included in the model evaluation stage because it was applied as an additional rule-based mechanism in the monitoring simulation.

3.5 Dashboard-Based Monitoring Procedure

We conducted a monitoring simulation by gradually streaming historical sensor data to mimic periodic sensor readings. The selected Federated Learning–Autoencoder model analyzed the incoming data, and the dashboard displayed the sensor status, anomaly type, triggering parameter, detection basis, recommended action, and anomaly event logs. The simulation first streamed records containing client information, humidity, temperature, and electrical conductivity (EC). The system then applied a rule-based check by comparing sensor values with predefined reasonable limits. If a value exceeded the specified range, the system classified it as an out-of-range anomaly. Table 3 presents the limits used in the simulation.

Table 3. Out-of-Range Sensor Limits

Sensor Parameter	Unit	Out-of-Range Limit
Humidity	%	< 10 or > 90
Temperature	°C	< 10 or > 35
Electrical Conductivity	µS/cm	< 0 or > 2500

Based on Table 3, the system classified humidity as out-of-range when it fell below 10% or exceeded 90%, temperature when it dropped below 10°C or rose above 35°C, and electrical conductivity when it was below 0 µS/cm or above 2500 µS/cm. This rule-based filtering was applied directly to raw sensor values before normalization and Autoencoder processing. Data that passed this check were then normalized using a client-specific scaler to match the training distribution. After normalization, the Autoencoder generated reconstructed values and the system computed reconstruction errors. A spike anomaly was flagged when at least one feature exceeded the predefined error threshold. Finally, the dashboard presented the results as normal, spike anomaly, or out-of-range anomaly, along with triggering parameters, detection basis, recommended actions, and detailed event logs.

4. Experimental Setup

4.1 Data Collection

This study used the public stuard_soil_data dataset from the work titled Smart agriculture dataset in a tomato cultivation under different irrigation regimes [5]. The dataset contained soil sensor data collected from tomato cultivation using a Milesight EM500-SMTC device. It included sensor line identifiers, timestamps, device IDs, and soil measurements. The dataset had three sensor lines with different irrigation treatments. Line 1 followed IrriFrame recommendations, Line 2 received 60% of Line 1 irrigation, and Line 3 received 30% of Line 1 irrigation. We treated each line as a separate client in the Federated Learning setup. The dataset initially contained 32,666 records, and we processed it through preprocessing before training and evaluation.

4.2 Data Preprocessing

We applied data preprocessing to the three client datasets before model training and evaluation. This stage included client-based data separation, removal of missing values, feature selection, train-test splitting, and normalization using RobustScaler. We selected humidity, temperature, and electrical conductivity because they represent key soil conditions in tomato farming. We split each client dataset into training and testing sets using an 80:20 ratio. Table 4 summarizes the dataset after preprocessing.

Table 4. Summary of Data After Preprocessing

Client	Initial Data	Missing Values	Data After Cleaning	Training Data	Test Data
Client 1	10,966	2	10,964	8,771	2,193
Client 2	10,862	0	10,862	8,689	2,173
Client 3	10,838	2	10,836	8,668	2,168
Total	32,666	4	32,662	26,128	6,534

Table 4 shows that the number of data points in each client was relatively balanced. After data splitting, RobustScaler normalization was applied to EC, humidity, and temperature so that feature scales became comparable. The preprocessed data were then used as input for local model training and Federated Learning-Autoencoder training.

4.3 Training Configuration

The training configuration was used to observe the influence of epochs, batch size, learning rate, number of rounds, and local epochs on the model training process. In the local model scenario, the experiments were coded as L01, L02, and L03. Each experiment used a maximum of 70 epochs, with differences in batch size and learning rate. The local training configuration is shown in Table 5.

Table 5. Local Training Configuration

Experiment	Epoch	Batch Size	Learning Rate	Number of Clients
L01	70	16	0.001	3
L02	70	16	0.0005	3
L03	70	32	0.0001	3

Based on Table 5, experiment L01 used a batch size of 16 and a learning rate of 0.001, L02 used a batch size of 16 and a learning rate of 0.0005, and L03 used a batch size of 32 and a learning rate of 0.0001.

In the Federated Learning model, the experiments were coded as F01, F02, and F03. The three experiments used three clients and the FedAvg aggregation method. The main differences among the experiments were the number of rounds, local epochs, batch size, and learning rate. The Federated Learning training configuration is shown in Table 6.

Table 6. Federated Learning Training Configuration

Experiment	Number of Rounds	Local Epoch	Batch Size	Learning Rate	Number of Clients
F01	10	3	16	0.001	3
F02	15	5	16	0.001	3
F03	20	5	32	0.0005	3

Based on Table 6, experiment F01 used 10 rounds with 3 local epochs, F02 used 15 rounds with 5 local epochs, and F03 used 20 rounds with 5 local epochs. The different configurations were used to observe their effects on global model formation through

FedAvg. With these configurations, the Federated Learning model was trained gradually without combining raw data among clients.

5. Result and Analysis

5.1 Local Model Training Results

Local model training was conducted separately on each client. Final loss and final validation loss were used to observe the ability of the model to reconstruct normal data. The local model training results are shown in Table 7.

Table 7. Local Training Experiment Results

Experiment	Client	Max Epoch	Best Epoch	Batch Size	Final Loss	Final Val Loss
L01	Client 1	70	30	16	0.09399	0.01787
	Client 2	70	70	16	0.02886	0.00101
	Client 3	70	47	16	0.01187	0.00453
L02	Client 1	70	70	16	0.29983	0.29119
	Client 2	70	70	16	0.02892	0.00199
	Client 3	70	70	16	0.01206	0.00670
L03	Client 1	70	63	32	0.40835	0.39812
	Client 2	70	16	32	0.49780	0.14566
	Client 3	70	18	32	0.83899	0.22878

Based on Table 7, experiment L01 showed the most stable local training result because it produced low validation loss values on all clients. Experiment L02 was still relatively good on Client 2 and Client 3, but it was less optimal on Client 1. Experiment L03 produced higher loss and validation loss values than L01 and L02.

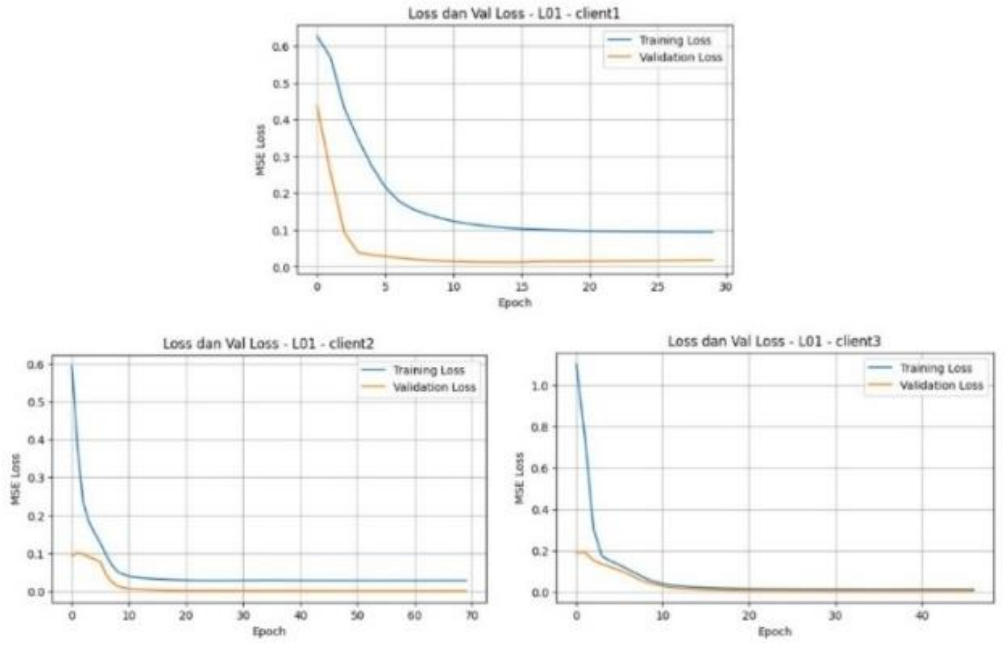

Fig. 1. Training and validation loss for local experiment L01

Fig. 1 shows that experiment L01 produced a relatively stable decrease in training loss and validation loss on all clients. Low validation loss values indicate that the model was able to reconstruct normal data well under the L01 configuration.

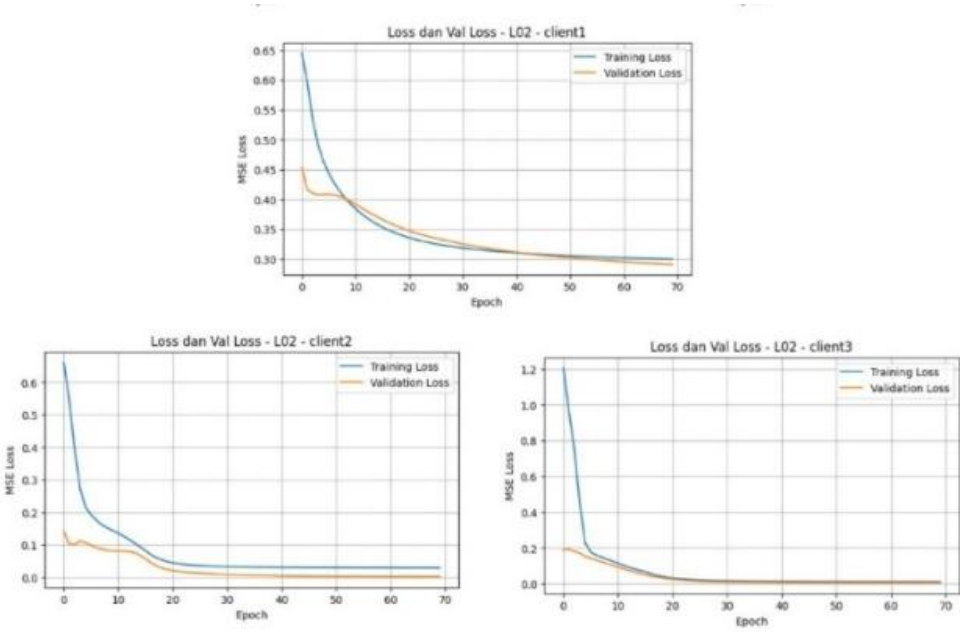

Fig. 2. Training and validation loss for local experiment L02

Fig. 2 shows that experiment L02 still produced decreasing training loss and validation loss, but the results were not evenly distributed across all clients. This condition can be seen from the loss value of one client that remained higher than those of the other clients.

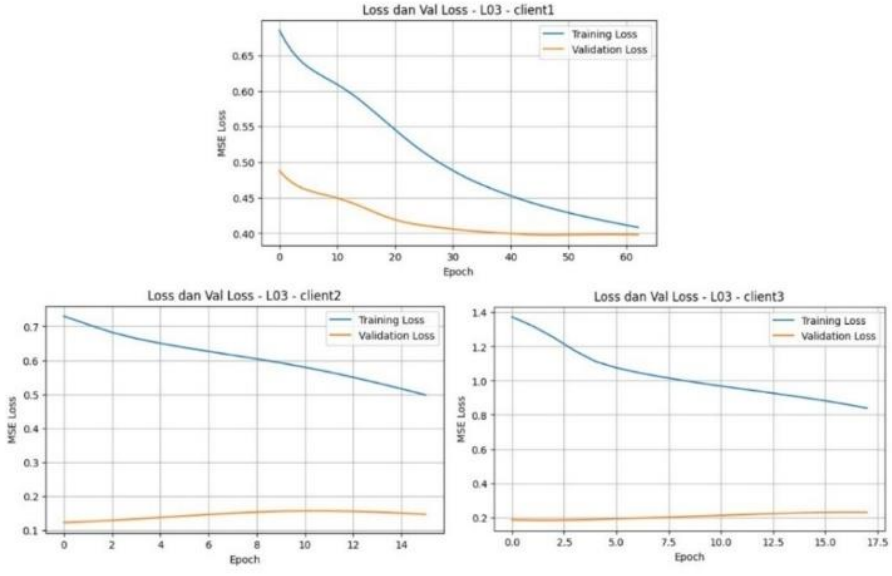

Fig. 3. Training and validation loss for local experiment L03

Fig. 3 shows that experiment L03 produced relatively higher training loss and validation loss values than L01 and L02. This condition indicates that the L03 configuration did not provide an optimal training process for the local model. Overall, L01 showed the most stable decline in training and validation loss compared with the other local experiments. However, loss results were not used as the final basis for selecting the best model because the main objective of this study was spike anomaly detection.

5.2 Federated Learning Model Training Results

Federated Learning model training was conducted through three scenarios, namely F01, F02, and F03. In each round, each client trained the model using local data. The model parameters were then sent to the server and aggregated using FedAvg. The Federated Learning model training results are shown in Table 8.

Table 8. Federated Learning Experiment Results

Experiment	Number of Rounds	Epoch	Batch Size	Final Train Loss	Final Val Loss
F01	10	3	16	0.04919	0.00763
F02	15	5	16	0.04222	0.00730
F03	20	5	32	0.26731	0.16172

Based on Table 8, experiment F02 produced the lowest final train loss and final validation loss, namely 0.04222 and 0.00730. Experiment F01 also showed good results with a final validation loss of 0.00763. Experiment F03 produced higher loss values, indicating that it was not optimal in the global model training stage.

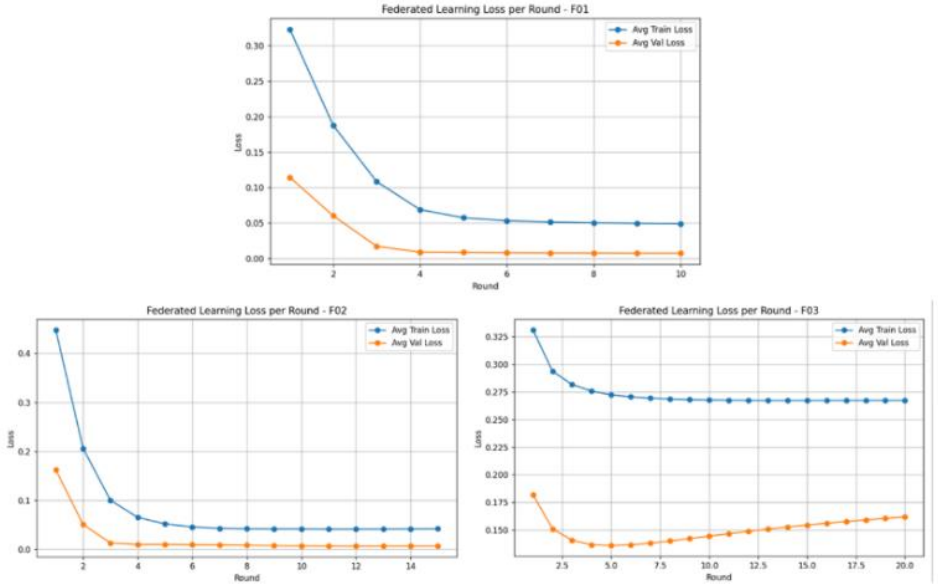

Fig. 4. Training and validation loss for Federated Learning experiments F01, F02, and F03

Fig. 4 shows that experiments F01 and F02 produced stable loss reductions during training. F02 became the best configuration at the training stage because it produced the lowest final loss value. In contrast, F03 showed a less optimal training pattern because its validation loss was still high and increased again in several final rounds. Nevertheless, the best model was determined through anomaly detection evaluation using precision, recall, and F1-score rather than training loss alone.

5.3 Spike Anomaly Detection Evaluation

This experiment evaluated the model using test data with a 5% synthetic spike anomaly injection for each client because the dataset did not include real anomaly labels. We created anomalies by randomly selecting test points and perturbing one feature using k × IQR (k = 3) from each client's training data, while keeping final values non-negative. We labeled normal data as 0 and injected anomalies as 1 to compute precision, recall, and F1-score. The test set contained 6,534 records, including 6,207 normal and 327 anomaly samples. This study detected spike anomalies using reconstruction error from humidity, temperature, and electrical conductivity (EC). This study classified a record as anomalous when at least one feature exceeded the error threshold. This paper tested three thresholds: P95, P97, and P99. Our proposed method reported results across three experiments (L01, L02, and L03). We also calculated average precision, recall, and F1-score to summarize performance across all clients. Table 9 presents the detailed results. results.

Table 9. Local Model Evaluation Results

Experiment	Percentile	Avg Precision	Avg Recall	Avg F1-Score
L01	P95	1.0000	0.8291	0.9061
	P97	1.0000	0.8260	0.9042
	P99	1.0000	0.7957	0.8863
L02	P95	0.7132	0.8108	0.6625
	P97	1.0000	0.8077	0.8928
	P99	1.0000	0.7138	0.8298
L03	P95	0.6096	0.9207	0.6892
	P97	1.0000	0.8383	0.9108
	P99	1.0000	0.6104	0.7431

Based on Table 9, the local model showed good performance in detecting spike anomalies. Experiment L01 with the P95 threshold obtained Avg Precision of 1.0000, Avg Recall of 0.8291, and Avg F1-Score of 0.9061. L01 P95 was selected as the best local model because it had stable performance, especially in maintaining precision without producing false positives.The Federated Learning model evaluation results are shown in Table 10. The evaluation was conducted on three experiments, namely F01, F02, and F03, using threshold values P95, P97, and P99.

Table 10. Federated Learning Model Evaluation Results

Experiment	Percentile	Avg Precision	Avg Recall	Avg F1-Score
F01	P95	1.0000	0.7865	0.8803
	P97	1.0000	0.7591	0.8628
	P99	1.0000	0.6648	0.7981
F02	P95	0.7763	0.8505	0.7660
	P97	0.7817	0.8383	0.7646
	P99	1.0000	0.8170	0.8990
F03	P95	0.2905	0.8993	0.4233
	P97	0.5431	0.7865	0.8803
	P99	1.0000	0.7591	0.8628

Based on Table 10, F01 showed the most stable performance with Avg Precision of 1.0000 across all thresholds, meaning it did not produce false positives. The best F01 performance was obtained at P95 with Avg Recall of 0.7865 and Avg F1-Score of 0.8803. Although F02 had the highest Avg F1-Score at P99, its precision values at P95 and P97 were lower. Therefore, F01 P95 was selected as the best Federated Learning model. The comparison between local model L01 P95 and Federated Learning model F01 P95 is shown in Table 11.

Table 11. Comparison of Local and Federated Learning Models

Model Type	Best Model	Percentile	Avg Precision	Avg Recall	Avg F1-Score
Locally trained model	L01	P95	1.0000	0.8291	0.9061
Federated Learning model	F01	P95	1.0000	0.7865	0.8803

Based on Table 11, the local model L01 P95 obtained an Avg F1-Score of 0.9061, while the Federated Learning model F01 P95 obtained 0.8803, with a difference of 0.0258. The local model performed slightly better, but the Federated Learning model remained competitive because it did not combine raw data among clients and both models obtained Avg Precision of 1.0000 without false positives.

The slightly lower recall and F1-score of the Federated Learning model likely resulted from heterogeneous data across clients. Each sensor line reflected different irrigation conditions, which produced different humidity, temperature, and electrical conductivity patterns. In this case, the global model needed to generalize across all clients, while local models learned only client-specific patterns. Despite this, the performance gap remained small and still showed competitive results while preserving data locality. These findings aligned with Adkisson et al. [3], who showed that Autoencoders detect anomalies in smart farming using reconstruction error. Our best local model achieved an F1-score of 0.9061. The Federated Learning model performed similarly and supported Florencia and Mayer [8], who confirmed the effectiveness of FL for distributed time-series anomaly detection without sharing raw data. Overall, these results confirmed the suitability of combining Federated Learning and Autoencoder for agricultural IoT anomaly detection. The detection error pattern is shown in Table 12.

Table 12. Confusion Matrix of L01 and F01 Models

Model (Percentile Threshold)	Client	TN	FP	FN	TP
L01 (P95)	Client 1	2,083	0	13	97
	Client 2	2,064	0	21	88
	Client 3	2,059	0	22	87
F01 (P95)	Client 1	2,083	0	22	88
	Client 2	2,064	0	21	88
	Client 3	2,060	0	27	82

Based on Table 12, neither model produced false positives, but both models still produced false negatives. The local model showed slightly better performance, whereas the Federated Learning model remained competitive because its results were close to the local model without combining raw data among clients.

6. Conclusion

This study successfully developed a Federated Learning-Autoencoder approach to detect anomalies in IoT sensor data for tomato farming consisting of three clients or sensor lines with different irrigation conditions. The parameters used included humidity, temperature, and electrical conductivity. Through the FedAvg method, the global model could be formed from the local training results of each client without combining raw data, making this approach suitable for distributed agricultural IoT sensor data.

The evaluation results showed that the best local model, L01 with the P95 threshold, obtained Avg Precision of 1.0000, Avg Recall of 0.8291, and Avg F1-Score of 0.9061. Meanwhile, the best Federated Learning model, F01 with the P95 threshold, obtained Avg Precision of 1.0000, Avg Recall of 0.7865, and Avg F1-Score of 0.8803. The Avg F1-Score difference of 0.0258 indicated that the Federated Learning model had competitive performance even without accessing all raw data centrally. In addition, the detection results were successfully presented in a monitoring dashboard simulation that displayed normal status, spike anomaly, out-of-range anomaly, recommended actions, and anomaly event logs. Future work may develop this approach further by testing it on direct field sensor data and integrating real-time monitoring.

Acknowledgment

The authors express their gratitude to Allah SWT for His blessings and guidance throughout the completion of this research. The authors also thank the Information Systems Study Program, Faculty of Science and Technology, Universitas Jambi, for the support and guidance provided during this study. The authors are also grateful to all parties who contributed to the completion of this research.

References

[1] S. Mansoor, S. Iqbal, S. M. Popescu, S. L. Kim, Y. S. Chung, and J. H. Baek, “Integration of smart sensors and IOT in precision agriculture: trends, challenges and future prospectives,” 2025, *Frontiers Media SA*. doi: 10.3389/fpls.2025.1587869.

[2] A. Martelli, D. Rapinesi, L. Verdi, I. I. M. Donati, A. Dalla Marta, and F. Altobelli, “Smart irrigation for management of processing tomato: a machine learning approach,” *Irrig. Sci.*, vol. 43, no. 6, pp. 1407–1424, Nov. 2025, doi: 10.1007/s00271-024-00993-9.

[3] M. Adkisson, J. C. Kimmel, M. Gupta, and M. Abdelsalam, “Autoencoder-based Anomaly Detection in Smart Farming Ecosystem,” Oct. 2022, [Online]. Available: http://arxiv.org/abs/2111.00099

[4] T. Domínguez-Bolaño, O. Campos, V. Barral, C. J. Escudero, and J. A. García-Naya, “An overview of IoT architectures, technologies, and existing open-source projects,” Nov. 01, 2022, *Elsevier B.V.* doi: 10.1016/j.iot.2022.100626.

[5] L. Belli *et al.*, “Smart agriculture dataset in a tomato cultivation under different irrigation regimes,” *Data Brief*, vol. 60, Jun. 2025, doi: 10.1016/j.dib.2025.111521.

[6] S. Chen and W. Guo, “Auto-Encoders in Deep Learning—A Review with New Perspectives,” Apr. 01, 2023, *MDPI*. doi: 10.3390/math11081777.

[7] H. Rhachi, Y. Balboul, and A. Bouayad, “Enhanced Anomaly Detection in IoT Networks Using Deep Autoencoders with Feature Selection Techniques,” *Sensors*, vol. 25, no. 10, May 2025, doi: 10.3390/s25103150.

[8] F. Liu *et al.*, “FedTADBench: Federated Time-Series Anomaly Detection Benchmark,” Dec. 2022, [Online]. Available: http://arxiv.org/abs/2212.09518

[9] C. Florencia and R. Mayer, *Anomaly Detection from Distributed Data Sources via Federated Learning*, vol. 450. in Lecture Notes in Networks and Systems, vol. 450. Cham: Springer International Publishing, 2022. doi: 10.1007/978-3-030-99587-4.

[10] H. Xiang *et al.*, "Federated Learning-Based Anomaly Detection with Isolation Forest in the IoT-Edge Continuum," *ACM Transactions on Multimedia Computing, Communications, and Applications*, Nov. 2024, doi: 10.1145/3702995.

[11] I. Dirlik, F. Uğurlar, and C. Kaya, "Sensor-Guided Smart Irrigation for Tomato Production: Comparing Low and Optimum Soil Moisture in Greenhouse Environments," *Food Energy Secur.*, vol. 14, no. 2, Mar. 2025, doi: 10.1002/fes3.70082.

[12] Y. Li, T. Wang, H. Zu, S. Liu, and L. Zu, "Explainable AI-driven interpretation of environmental drivers of tomato fruit expansion in smart greenhouses using IoT sensing," *Sci. Rep.*, vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-24800-3.

[13] M. A. Belay, S. S. Blakseth, A. Rasheed, and P. Salvo Rossi, "Unsupervised Anomaly Detection for IoT-Based Multivariate Time Series: Existing Solutions, Performance Analysis and Future Directions," Mar. 01, 2023, *MDPI*. doi: 10.3390/s23052844.

[14] F. Wang, Y. Jiang, R. Zhang, A. Wei, J. Xie, and X. Pang, "A Survey of Deep Anomaly Detection in Multivariate Time Series: Taxonomy, Applications, and Directions," Jan. 01, 2025, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/s25010190.

[15] T. Walczyna, D. Jankowski, and Z. Piotrowski, "Enhancing Anomaly Detection Through Latent Space Manipulation in Autoencoders: A Comparative Analysis," *Applied Sciences (Switzerland)*, vol. 15, no. 1, Jan. 2025, doi: 10.3390/app15010286.

[16] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," 2017.

[17] Y. Liu, L. Zhang, N. Ge, and G. Li, "A Systematic Literature Review on Federated Learning: From A Model Quality Perspective," Dec. 2022, [Online]. Available: http://arxiv.org/abs/2012.01973

[18] H. Devaraj et al., "RuralAI in Tomato Farming: Integrated Sensor System, Distributed Computing, and Hierarchical Federated Learning for Crop Health Monitoring," IEEE Sensors Letters, 2024. doi: 10.1109/LSENS.2024.3384935.

[19] M. Z. U. Kakakhel, M. Ablimit, E. Eli, and K. Ubul, "Smart-Farming based Prevention System for Tomato Leaf Curl Disease," in Proc. 2nd Int. Conf. Artif. Intell., Human-Computer Interact. Robot. (AIHCIR), 2023. doi: 10.1109/aihcir61661.2023.00065.

[20] R. Swathika, N. Radha, Y. R., N. S., and K. Stephano, "A Real Time IoT Machine Learning Pipeline for Tomato Crop Disease Detection Irrigation Optimization and Yield Prediction," in Proc. Int. Conf. Comput. Commun. Control Cyber-Phys. Syst. (I5CPS), 2026. doi: 10.1109/I5CPS67958.2026.11452615.

[21] A. M. Morales-Badajoz et al., "Astro Cultivators: Autonomous Growth System for Space Farming Based on Machine Vision and Multi-Sensor Fusion," in Proc. CPS-IoT Week Workshops, 2023. doi: 10.1145/3576914.3588338.