Journal of Trends in Computer Science and Smart Technology (ISSN: 2582-4104)
www.irojournals.com/tcsst/

MVRODC: Multi-Variate Regression based Outlier Detection and Classification on IoT Sensor Data — A Temporal Node Level Approach

Veera Brahmam M.[1], Gopikrishnan S.[2]

School of Computer Science and Engineering, VIT-AP University, Amaravathi, Andhra Pradesh, India.

E-mail: [1]veerabrahmam.20phd7156@vitap.ac.in, [2]gopikrishnan.s@vitap.ac.in

Orcid ID: [1]0000-0002-7693-1165, [2]0000-0001-9082-9012

Abstract

The reliability and correctness of data detected by sensors are essential for the efficient use of Internet of Things (IoT) and Wireless Sensor Network (WSN) technologies. However, sensor readings are often affected by errors due to hardware failure or actual environmental events. This causes outliers that can affect decision-making and system efficiency. To address these problems, the present study proposes a novel outlier detection and classification technique called Multivariate Regression-based Outlier Detection and Classification (MVRODC). MVRODC uses similarity measures derived from Multiple Linear Regression (MLR) along with an adaptive buffer to model temporal relationships. This ensures that outliers are detected and classified into two different categories in real-time into two categories: errors and actual events. Inter-sensor feature correlations across multiple sensor streams (temperature, humidity, air quality, and light) are exploited along with temporal prediction consistency to enable robust real-time outlier detection and classification. The MVRODC technique ensures that relevant outliers caused by actual events are retained, allowing for the detection of environmental changes while ignoring erroneous data. This filtering technique saves energy because sending erroneous data consumes as much energy as sending legitimate data. Experimentally, MVRODC performs better than existing outlier detection techniques, achieving superior results in terms of detection rate, false alarm rate, accuracy, error detection rate, and event detection rate.

Keywords: Internet of Things, Outliers, Errors, Events, Detection Rate, Multiple Linear Regression.

1. Introduction

The widespread use of IoT and WSNs has transformed the way we gather data in different domains, including environmental monitoring, smart cities, and industrial automation. Reliable sensor data are fundamental for accurate decision-making; however, ensuring the quality of sensor data remains a challenge. Sensor readings are frequently affected by various errors from sensor malfunctions. In IoT, an outlier is typically defined as a data point that

DOI: https://doi.org/10.36548/jtcsst.2026.3.001
Received: 02.04.2026, received in revised form: 05.05.2026, accepted: 24.05.2026, published: 05.06.2026

deviates considerably from the expected data distribution [1]. Outliers may arise from multiple sources, including measurement errors and true environmental events [2]. Separation of these outliers is important in applications that require real-time data processing.

There exist multiple classification schemes that can be applied to outlier detection methods. One of the most popular criteria is related to the purpose of detecting abnormal sensor values or interesting events [2]. The other classification scheme depends on the ability of the method to work in either an online mode or an offline mode. Moreover, outlier detection approaches can be further classified according to their learning technique (supervised or unsupervised learning) and the implementation strategy (distributed or centralized). Among all these approaches, online outlier detection has gained a lot of popularity due to the possibility of real-time monitoring and fast reaction.

Real-time outlier detection is crucial in many application scenarios, such as monitoring heart rate and blood pressure in the healthcare sector, monitoring machine temperature, pressure, and vibration in industrial maintenance applications, and controlling traffic, energy consumption and environmental conditions in smart cities [3]. Early detection of outliers facilitates preventive measures, prevents machine failure, and improves resource utilization. Although substantial progress has been achieved in real-time outlier detection, many existing approaches do not simultaneously leverage inter-sensor feature correlations across multiple sensor streams and temporal prediction consistency, thereby limiting their effectiveness in robust real-time outlier detection and classification. It is very important to consider energy efficiency when handling and transmitting data. One of the main challenges faced is due to the resource constraints of IoT sensor nodes, primarily their limited energy capacity. Offline approaches to anomaly detection, which consume large amounts of computational energy, cannot be applied in such scenarios. Moreover, most traditional approaches lack any mechanism for automating the model update process and may involve considerable delay or work with univariate data models.

To counter these difficulties, this study introduces a new model of node-level online outlier detection and classification, namely MVRODC. This method utilizes the correlation between sensor data streams, which can be obtained using MLR with an adaptive buffer approach. More precisely, the temporal correlation between the estimated value of the current detected data vector and the last vector in the adaptive buffer is calculated. If the similarity exceeds a certain level, the current reading is considered valid; otherwise, it is considered an outlier. The similarities between these outliers are examined further. If more than half of the outliers share a similar pattern, they are classified as events; otherwise they are considered errors due to their temporary or permanent nature. The proposed methodology facilitates efficient real-time detection and classification of sensor errors and environmental events without causing energy waste, as there is no need to communicate invalid readings. Given that the amount of energy needed to transfer invalid information is the same as that required to transfer valid information, the exclusion of invalid information at the node level facilitates energy savings [4].

1.1 Motivation

This investigation is motivated by the following significant challenges associated with IoT systems:

- Environment factors that interfere with sensors or malfunctioning sensors are numerous causes of outliers in IoT sensor data. It is crucial that these incorrectly detected data be detected and fixed in order to maintain data quality and reliability.
- Fast and accurate responses to data outliers are critical in a variety of applications, such as Industrial IoT, environmental monitoring, and smart cities. Figuring out whether an outlier is due to malfunctioning sensors (errors), or something important happening in the environment (an event) significantly improves decision-making effectiveness.
- This work significantly improves previous methodologies by addressing the important requirement for automated real-time outlier management in IoT systems.

1.2 Contributions Made

- The suggested technique facilitates a quick response to outlier data by detecting outliers immediately as they flow in from IoT sensors instead of waiting to process a large batch of information.
- The proposed approach goes beyond the fundamental concept of outlier detection by classifying outliers as events or errors. This classification is important for understanding the source of outliers and taking the necessary steps.
- The total absence of human intervention improves the efficiency and reliability of IoT systems by eliminating the need for manual supervision and the associated risks of human error.

The remainder of the paper is structured into four sections: Section 2 reviews previous work on outlier detection and classification in sensor networks. The proposed methodology is discussed in Section 3, while Section 4 discusses and evaluates the experimental results. Section 5 gives the conclusion of the paper.

2. Related Work

This subsection presents some prominent research efforts related to outlier detection and classification in WSN and IoT environments, including temporal modeling, online detection, learning techniques, and implementation difficulties. Prior investigations concentrated almost entirely on interpolation and clustering methods for detecting outlier sensor observations. In an attempt to solve the issue of distinguishing faulty sensors from environmental occurrences, Al Samara et al. [5] recommended a new technique called EEODCA, which employs DBSCAN clustering along with IDW interpolation and takes advantage of the connection between spatial-temporal factors. Although this is an efficient method in controlled environments, it relies on synchronized sampling and therefore cannot be implemented in dynamic WSNs. Al Samara et al. [6] recently developed a novel technique called OPTICS-K, which builds on this work by integrating OPTICS clustering, inter-cluster distance, and KNN with Kriging interpolation. Despite its high accuracy, the method is computationally expensive and relies heavily on homogeneity and synchronization.

Several research works focused on the detection of outliers using online analysis and stream processing. Hu et al. [7] proposed an adaptive updating KNN Gaussian mixture

approach where Gaussian components were adaptively maintained and the use of Mahalanobis distance was performed using dynamic thresholds for online outlier detection. The method is ideal for local and incremental learning; nevertheless, the algorithm did not comprehend multivariate dependencies and temporal-spatial patterns. Dani et al. [8] devised an online outlier detection approach through the use of a recursive principal component analysis with the aid of first order perturbation to update eigenspace structures without retaining any history of the data. Nevertheless, sensitivity to abrupt changes in the data and the lack of outlier classification made it inefficient. Similarly, Mazarei et al. [9] suggested the use of the online boxplot method using histogram approximation and CDF computation. However, it only focuses on one-dimensional outlier detection.

Due to their ability to represent intricate temporal and non-linear relationships, deep learning methods have been utilized extensively. This includes the introduction of a methodology by Shu et al. [10]. The authors suggested a framework for continuous outlier detection based on a hybrid C-LSTM model for complex network data streams that shows excellent performance when applied to actual data. However, its generalizability and explainability need further research. The work of Abhaya et al. [11] enhanced the autoencoder detection method by integrating Density Peak Clustering and Self-Organizing Maps to find potential outliers, resulting in lower reconstruction bias and higher precision. Despite all the benefits, the algorithm is highly dependent on clustering and thresholding methods. Antonius et al. [12] developed the novel Bat-optimized CNN-BiLSTM framework that helps to represent spatiotemporal features of data collected through IoT devices and leads to outstanding results, but shows low adaptability to rapidly changing environments.

Outlier detection and classification using hybrid and ensemble learning have been the subject of various research papers. Samara et al. [13] proposed an online model called SA-O2DCA that integrates the use of three different models, including K-means clustering, Isolation Forest, and Newton interpolation, along with a new approach called a seasonal model switching mechanism. Though the method appears promising due to its enhanced adaptability properties, its limitation with respect to its dependence on predefined seasonal patterns constrains its adaptability. Malki et al. [14] proposed two models, Prophet and Light GBM, for anomaly detection in the energy consumption pattern of smart homes which outperformed classical VAR models but were sensitive to both dataset sizes and the sampling rate.

Some of the papers concentrate on security centric anomaly detection. For example, Lai et al. [15] present a framework for DoS detection in a network, which uses a noise-aware Multi-class Passive Aggressive learning algorithm and improves robustness when handling noisy data, but external environmental parameters are not considered. Similarly, Rodriguez et al. [16] propose a dual-model-based framework for anomaly detection and classification in Industrial IoT networks by taking into account contextual information; at the same time, the problem of classifying unknown anomalies remains unsolved in their research.

The use of statistical and density-based methods is still popular due to their minimum computational overhead. In particular, krelza et al. [17] propose an approach called Statistical Hierarchical Clustering that allows detecting anomalies in a stream and dividing them into clusters with minimum computational overhead, but does not provide any distinction between errors and events. In turn, Gupta et al. [18] introduce the Outlierness Factor based on the neighborhood method that successfully distinguishes between those two types of anomalies but lacks flexibility. Another work by Singh et al. [19] design ADINOF as a solution for in-network anomaly detection in data streams with minimum memory consumption, yet their approach

cannot classify anomalies. Application-driven methods show even more examples of the versatility of outlier detection techniques. The work of Pecksen et al. [20] is focused on the development of machine learning approaches to predict early fires in electrical panels with the help of IoT data, while MollaSalilew et al. [21] applied multiple classifiers to detect faults in gas turbine operations.

Detecting KNN as the best classifier, they managed to achieve better results compared to other algorithms tested. Wei et al. [22] presented an interesting hybrid model that combines LSTM and autoencoders for indoor air quality monitoring; although it performs efficiently for univariate data, the approach shows weaknesses when working with multivariate time series. Finally, Rollo et al. [23] proposed Sliding Window Anomaly Detection for correction of environmental data, despite the univariate nature of their model.

2.1 Research Gap and Novelty Analysis

Existing online outlier detection approaches focus mainly on adaptive updating, clustering, density estimation, or prediction error analysis. Although several methods support online processing and incremental model updates, most approaches do not simultaneously provide multivariate temporal analysis, adaptive model updating, and outlier classification within a lightweight framework suitable for resource-constrained IoT environments. Furthermore, none of the reviewed methods explicitly exploit the similarity between consecutive regression predictions as an outlier indicator. These limitations motivate the development of the proposed MVRODC framework, which combines adaptive buffering, online multiple linear regression, prediction similarity analysis, and anomaly classification into a unified online detection architecture. A detailed comparison of MVRODC with existing online outlier detection methods are summarized in Table 1.

Table 1. Comparison of MVRODC with Existing Online Outlier Detection Methods

Method	Online Processing	Adaptive Update	Multivariae Analysis	Outlier Classification	Prediction Similarity	Edge-Friendly
Hu et al. [7]	Yes	Yes	No	No	No	Moderate
Dani et al. [8]	Yes	Yes	Yes	No	No	Moderate
Singh et al. [19]	Yes	Yes	Yes	No	No	High
Samara et al. [13]	Yes	Yes	Yes	Yes	No	Moderate
Proposed MVRODC	Yes	Yes	Yes	Yes	Yes	High

3. Proposed Work

The Figure1 illustrates the overall architecture of the proposed MVRODC framework. Temperature, humidity and light measurements collected at the IoT sensor node are processed locally using an adaptive buffer and a multivariate regression model. A similarity score is computed between successive predictions and is evaluated using a predefined threshold. Normal observations are transmitted to the fusion center or base station, whereas detected outliers are further classified as errors or events. Error readings are discarded at the node level, while event observations are retained and flagged for higher-level analysis. Unlike existing online outlier detection methods that primarily rely on residual analysis, clustering, or density estimation, the proposed MVRODC framework integrates adaptive buffering with prediction

similarity analysis and outlier classification. This combination enables continuous online outlier monitoring while maintaining low computational and memory requirements suitable for edge-based IoT deployments.

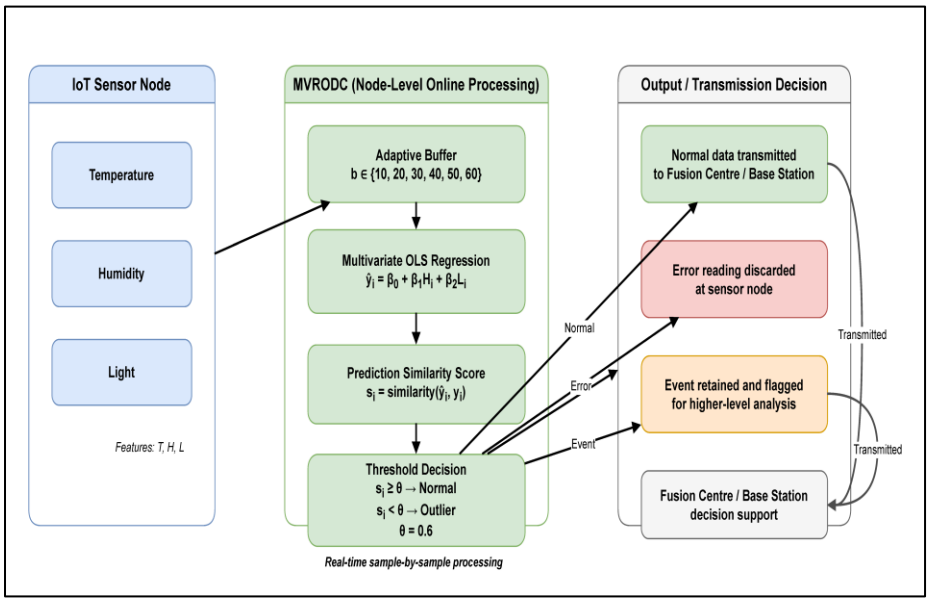


Figure 1. Proposed MVRODC System Architecture

Initially, a collection of b samples is used, where b represents the size of an adaptive buffer that will be applied in the subsequent outlier detection process. During this stage, any noisy or incomplete data can be removed. A major challenge involves finding errors directly at the node level, especially in real time, where the goal is to detect outliers in the current sensor reading without relying on historical records or external storage. To overcome this, the proposed approach introduces an adaptive buffer for sensor data. Rather than retaining the entire history of sensor data, the adaptive buffer maintains only a sliding window of the most recent samples. The size of buffer b is variable and adapts to values 10, 20, 30, 40, 50, and 60. This mechanism enables the system to forecast upcoming readings and compare them with previous predictions to detect potential outliers. The complete processing flow for outlier detection is illustrated in Figure 2. In this work, the threshold θ is set to 0.6. This value is chosen based on the correlation strength of the readings [24]:

- If the similarity between consecutive predictive readings is between 0.20 and 0.29, the readings are considered weakly correlated.
- If the similarity lies between 0.30 and 0.39, the readings are moderately correlated.
- Readings with similarity between 0.40 and 0.59 are classified as strongly correlated.
- A similarity of 0.6 or greater indicates that the readings are very strongly correlated.

Figure 2 shows the workings of detecting outliers using our proposed MVRODC algorithm. Let $D = \{(temp_i, hum_i, light_i)\}_{i=1}^{n}$ denote the multivariate sensor stream sampled over time, where D is the complete data set. In time index i, we use an adaptive buffer (sliding window) of recent samples of length b to train a regression model that predicts the current temperature from humidity and light. We evaluated multiple buffer sizes $b \in \{10,20,30,40,50,60\}$with step size $\sigma = 1$ in a real-time setting.

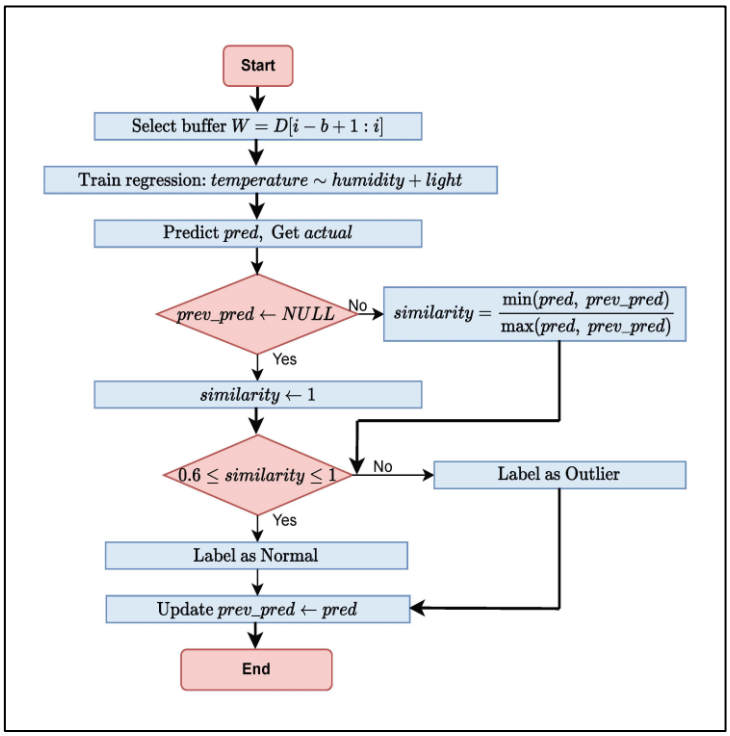


Figure 2. Flowchart of Adaptive Buffer Multivariate Regression Outlier Detection

3.1 Adaptive Buffering (Sliding Window)

Figure 3 illustrates the operation of the adaptive buffer mechanism. During the initial filling phase, observations are accumulated until the buffer reaches its predefined capacity. Once the buffer is full, a sliding-window strategy is employed in which the oldest observation is discarded and the newest observation is incorporated. This enables real-time processing while maintaining a fixed memory footprint. For each buffer size b and for each time index i such that $i \geq b$, we form a buffer window $W_i = \{i - b + 1, \ldots, i\}$.

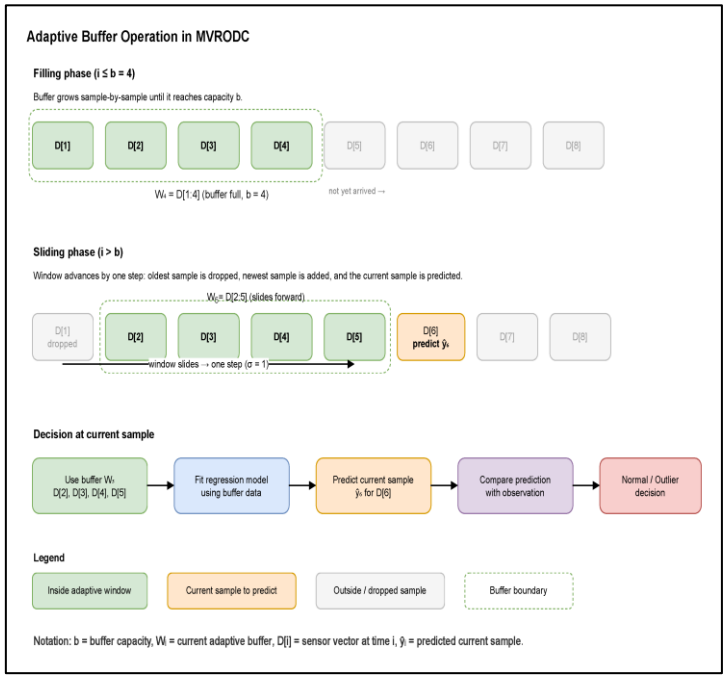


Figure 3. MVRODC Adaptive Buffer Working Procedure

Only the most recent b observations are retained for model construction, ensuring online processing capability with limited memory usage. The b points in Wi are used to train the regression model and only the most recent point i (the current sample) is classified. For example, when b = 4 and i = 8, $W_8 = D[5:8]$ which represents the most recent observations used for regression modeling. This strategy allows the regression model to operate continuously in streaming environments. while ensuring that the model is always trained on the most recent observations. For example, when b=4, the buffer gradually fills during the initial stage as $W_1 = D[1:1], W_2 = D[1:2], W_3 = D[1:3]$, and $W_4 = D[1:4]$. Once the buffer reaches capacity, the sliding window mechanism begins, resulting in $W_5 = D[2:5]$.

3.2 Multivariate Linear Regression Model

Figure 4 presents the regression-based prediction workflow used by MVRODC. For each incoming observation, an adaptive buffer window is selected and used to construct the regression design matrix.

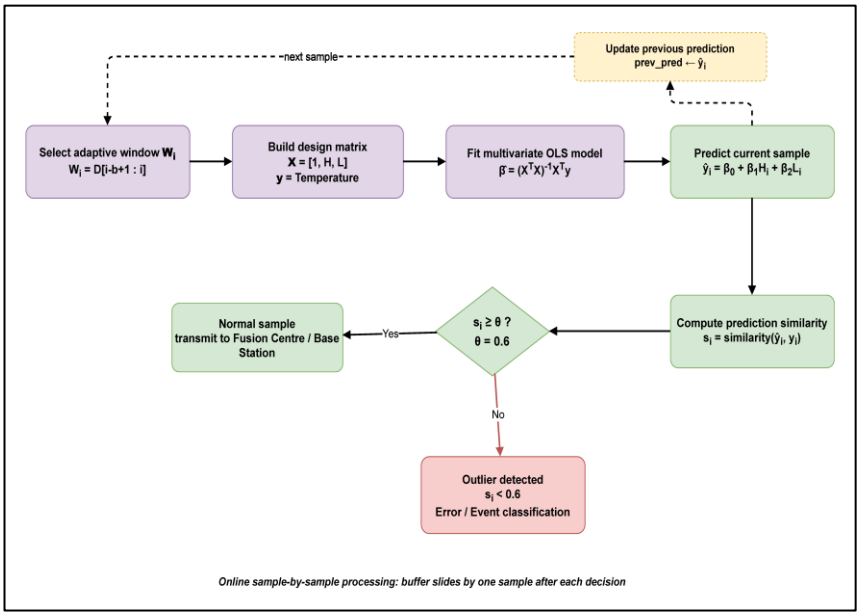


Figure 4. MVRODC Regression Workflow

The ordinary least squares model is fitted using historical observations and the current reading is predicted. Subsequently, a similarity score is computed and compared with the decision threshold to determine whether the observation is normal or anomalous. Within each window Wi, we fit an ordinary least squares (OLS) regression model of the form shown in Eq.1

$$temp_t = \beta_0 + \beta_1 hum_t + \beta_2 light_t + \varepsilon_t, \quad \forall t \in W_i \tag{1}$$

Temperature is highly reactive to its surroundings, particularly humidity and light. Humidity plays a big role in the air's thermal conditions, and light intensity is essentially a stand-in for solar radiation, both of which dramatically swing the temperature. By modelling the temperature in this way, using humidity and light as its drivers, the system can capture the underlying relationships in the environment and obtain the estimated coefficients $\widehat{\beta_0}, \widehat{\beta_1}, \widehat{\beta_2}$.

The regression coefficients $(\beta_0, \beta_1, \beta_2)$ are estimated using the least-squares formulation as shown in Eq.2.

$$\hat{\beta} = (X^T X)^{-1} X^T Y \tag{2}$$

Where X is the design matrix including the intercept, Y denotes the temperature observations, and $\hat{\beta}$ represents the estimated regression coefficients.

3.3 Real-time Prediction

Using the model trained on W_i, the predicted temperature for the current point i is given by Eq.3

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 hum_i + \hat{\beta}_2 light_i \tag{3}$$

The actual temperature measured at time i is $y_i = \text{temp}_i$. For temporal consistency, we also retain the previous prediction $\widehat{y_{i-1}}$ from the immediately preceding window W_{i-1}.

3.4 Similarity Measure between Consecutive Predictions

Instead of directly comparing observed and predicted values, MVRODC evaluates the temporal consistency between consecutive predictions. Conventional regression-based outlier detection relies on the residual deviation between observed and predicted values, which is highly sensitive to environmental drift and measurement noise. The proposed MVRODC differentiates itself by evaluating the temporal consistency of consecutive regression predictions. Since regression predictions reflect the learned system behavior within each adaptive window, any rapid changes in prediction stability show structural changes in sensor dynamics. This technique allows for robust outlier detection while reducing false alarms caused by gradual environmental changes. The proposed MVRODC defines a similarity measure s_i as shown in Eq.4 to quantify the consistency of model predictions over time,

$$s_i = \begin{cases} 1\,, & if\ i = b \\ \frac{\min(\hat{y}_i, \hat{y}_{i-1})}{\max(\hat{y}_i, \hat{y}_{i-1})}\,, & if\ i > b \end{cases} \tag{4}$$

si is designed to range from 0 to 1, The higher values represent a stronger association between consecutive predictions, while the smaller values indicate sudden changes. The min–max similarity ratio offers a consistent and bounded stability measure that is not affected by scale, remains robust against gradual environmental changes while detecting sudden deviations.

3.5 Decision Rule (Fixed Similarity Threshold)

We apply a fixed threshold $\theta = 0.6$ to the similarity score. Then the classification rule is as shown in Eq.5

$$status_i = \begin{cases} normal\,, & if\ \theta \le s_i \le 1 \\ outlier, & otherwise \end{cases} \tag{5}$$

Thus, the vector observed at the current time moment *i* is identified as an outlier when its prediction deviates significantly from the previous prediction, i.e., when $s_i \notin [\theta, 1]$. Otherwise, it is classified as normal. Algorithm 1 is developed based on the sequence of steps described in Sections 3.1 to 3.5. Both Eq. 5 and Alg.1 represent the same decision rule.

Eq. 5 provides the mathematical formulation of the decision process, whereas Algorithm 1 presents its procedural implementation for online outlier detection.

Algorithm 1: Adaptive Buffer Multivariate Regression Outlier Detection

1: INPUT: Dataset *D*, buffer size b, similarity threshold θ, step size σ
2: OUTPUT: Normal samples, Outliers
3: $pred_{i-1} \leftarrow$ NULL
4: for $i = b$ to n step σ do
5: Select buffer window $W \leftarrow \{D_{i-b+1}, \ldots\ldots, D_i\}$
6: Train regression model using temperature ~ humidity + light
7: $pred_i \leftarrow$ predicted temperature
8: actual $\leftarrow D_i$.temperature
9: if $pred_{i-1}$= NULL then
10: $s_i \leftarrow 1$
11: else
12: $s_i \leftarrow \frac{\min(pred_i, pred_{i-1})}{\max(pred_i, pred_{i-1})}$
13: end if
14: if $\theta \leq s_i \leq 1$ then
15: Label D_i as Normal
16: else
17: Label $D_i$as Outlier
18: end if
19: $pred_{i-1} \leftarrow pred_i$
20: end for

3.6 Error and Event Classification Mechanism

The Error and Event Classification algorithm is a post-processing step that refines the results of Alg. 1. Its objective is to determine whether a detected outlier is caused by sensor faults or random noise (error), or whether it corresponds to a meaningful environmental or contextual change (event). Alg. 2 clearly explains the classification of outliers. Figure 5 shows the flow chart for the classification of outliers into errors and events.

Algorithm 2 steps:

3.6.1 Inputs and Initialization

The algorithm takes as input the set of outliers O_b obtained from each adaptive buffer b. Two parameters are required:

- Similarity threshold (θ): The minimum similarity value required to consider two outliers as related.
- Minimum support fraction ($\alpha = 0.5$): If an outlier is similar to at least more than half of the previous outliers, it is classified as an event.

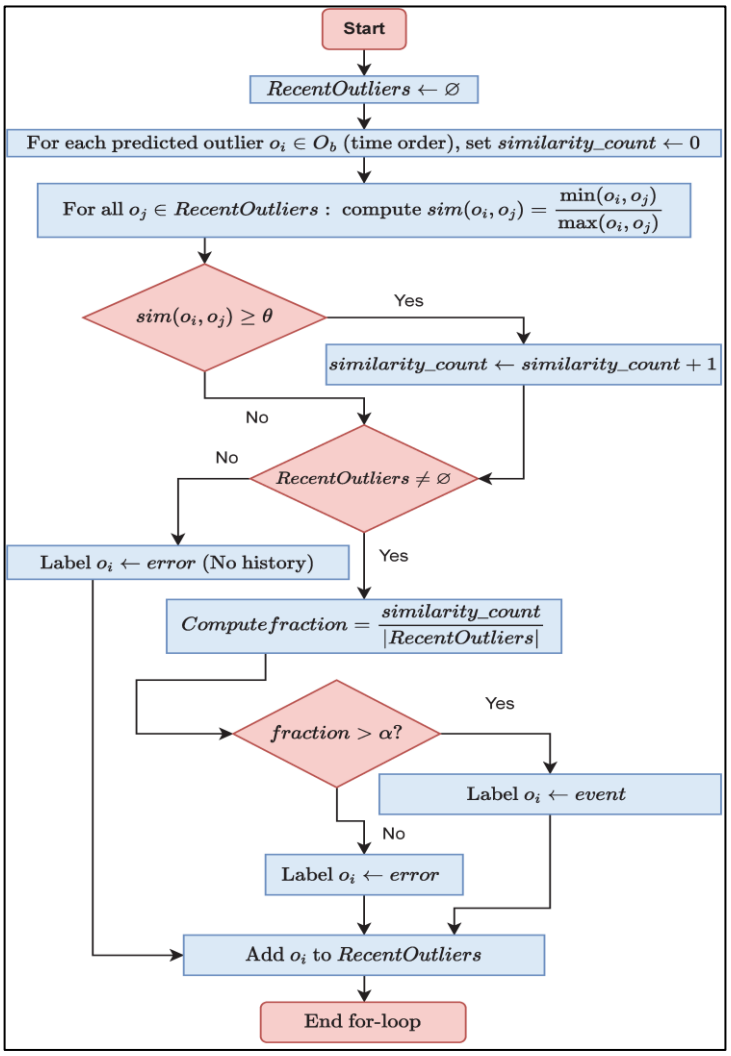


Figure 5. Flowchart of Error and Event Classification of Outliers

3.6.2 Iterating through Outliers

For each outlier $o_i \in O_b$, the algorithm computes its similarity to the set of previously detected outliers stored in *RecentOutliers*. A data structure *RecentOutliers* is initialized to store previously observed outliers in the buffer.

3.6.3 Similarity Computation

The similarity between two outliers o_i and o_j is computed as: $sim(o_i, o_j) = \frac{\min(o_i, o_j)}{\max(o_i, o_j)}$. This measure ranges between 0 and 1, providing a normalized comparison irrespective of the absolute values.

3.6.4 Counting Similar Outliers

For each $o_j \in$ RecentOutliers, if $sim(o_i, o_j) \geq \theta$, the counter $similarit\_count$ increases.

Algorithm 2: Error and Event Classification of Outliers

1: INPUT: Outliers O_b, similarity threshold θ, support fraction α
2: OUTPUT: Errors, Events
3: for each buffer b in B do
4: Initialize *RecentOutliers* ← ∅
5: for each outlier o_i in O_b (time order) do
6: *similarity_count* ← 0
7: if *RecentOutliers* ≠ ∅ then
8: for each o_j in *RecentOutliers* do
9: $sim \leftarrow \frac{\min(o_i,o_j)}{\max(o_i,o_j)}$
10: if $sim \geq \theta$ then
11: *similarity_count* ← *similarity_count* + 1
12: end if
13: end for
14: $fraction \leftarrow \frac{similarity\_count}{|RecentOutliers|}$
15: if *fraction* > α then
16: Label o_i as Event
17: else
18: Label o_i as Error
19: end if
20: else
21: Label o_i as Error ▷ No history
22: end if
23: Add o_i to *RecentOutliers*
24: end for
25: end for

3.6.5 Classification Rule

If RecentOutliers is not empty, the fraction of similar outliers is computed as: $fraction = \frac{similarity\_count}{|RecentOutliers|}$

- If $fraction > \alpha$, the current outlier oi is classified as an event.
- Otherwise, oi is classified as an error.

If no history exists (i.e. $RecentOutliers = \emptyset$), the outlier is directly labelled as an error.

3.6.6 Updating History

After classification, the current outlier o_i is added to *RecentOutliers* to be used for future evaluations.

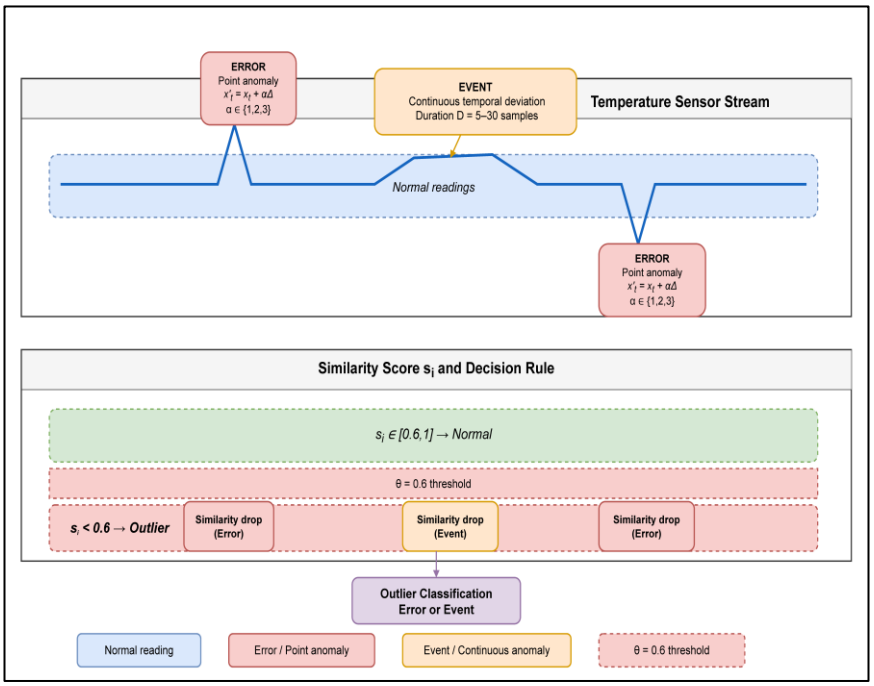


Figure 6. Temporal Anomaly Visualization and Similarity-Based Decision Rule

Figure 6 provides a visual representation of the anomaly detection process. Point anomalies appear as isolated spikes that deviate significantly from normal observations, whereas event anomalies manifest as sustained deviations over multiple consecutive samples. These deviations cause a reduction in the prediction similarity score. Observations that produce similarity values below the threshold ($\theta = 0.6$) are labeled as outliers and subsequently classified as errors or events.

3.7 Significance

- Errors represent random isolated deviations in sensor data, often caused by communication noise, temporary faults, or sensor malfunctions.
- Events represent consistent deviations that occur repeatedly over time, representing meaningful changes in the environment (e.g., a sudden rise in temperature or prolonged high humidity).

The rules for error and event classification are presented in Table 2.

Table 2. Rules for Error vs Event Classification

Type	Rule	Decision
Error	Sudden spikes or abrupt drops in sensor readings without temporal continuity	Outlier = Error [24]
Error	Sensor readings exceeding the valid physical operating range	Outlier = Error [24]
Event	Smooth and continuous variation in sensor readings over time	Outlier = Event [5]
Event	Significant deviation between predicted and previous predicted values beyond the similarity threshold limits	Outlier = Event [5]

4. Results and Discussion

A real-time dataset was gathered from the sensor node, which captured measurements from three onboard sensors: temperature, humidity, and air quality. The readings were recorded at 60-second intervals for environmental monitoring purposes. Data collection was carried out on our university campus between March 29, 2022, and September 15, 2022, as part of a smart campus initiative. In total, approximately 21,000 samples were collected. The hardware configuration of the sensor node is presented in Table 3. A synthetic dataset was generated from the original real-time data by introducing 300 artificial outliers (200 errors and 100 events) as the initial dataset used in this study did not contain any outliers. The complete dataset (including outliers) was then imported into MATLAB for further analysis. In all MATLAB simulations, "adaptive buffer" sizes of 10, 20, 30, 40, 50 and 60 were considered. In our research, the term adaptive buffer refers to the ability of a buffer to dynamically adjust its size or contents during the data collection process, depending on the requirements of the system or application. Instead of using a fixed buffer size, we adopted an experimental strategy by testing different buffer lengths (b= 10, 20, 30, 40, 50, and 60). The choice of b within this range is due to hardware constraints—larger buffer sizes decrease the efficiency of the IoT sensor node and increase computational overhead. Hence, we restricted b to values between 10 and 60.

This range ensures the balance between detection accuracy and efficient processing. The proposed MVRODC framework is also designed to handle 3500 samples (including outliers) of real time data from a total of 21000 samples, with various given buffer sizes. 300 outliers are randomly injected into every 3500 sample real time dataset. This classification procedure improves the reliability of IoT monitoring systems by differentiating between noise-induced errors and contextual events, which in turn leads to more accurate decision-making. Table 2 shows the rules for the classification of errors and events. In addition to the sensor node in Table 3, the authors also used nodes N1, N2, N4, and N33 from the IBRL dataset [25]. The authors took 10000 samples from each node and trained the model with every 2500 samples (including outliers). 200 artificial outliers are randomly injected into every 2500 samples, out of the total 10000 samples from the given IBRL nodes.

Table 3. Sensor Node Specification

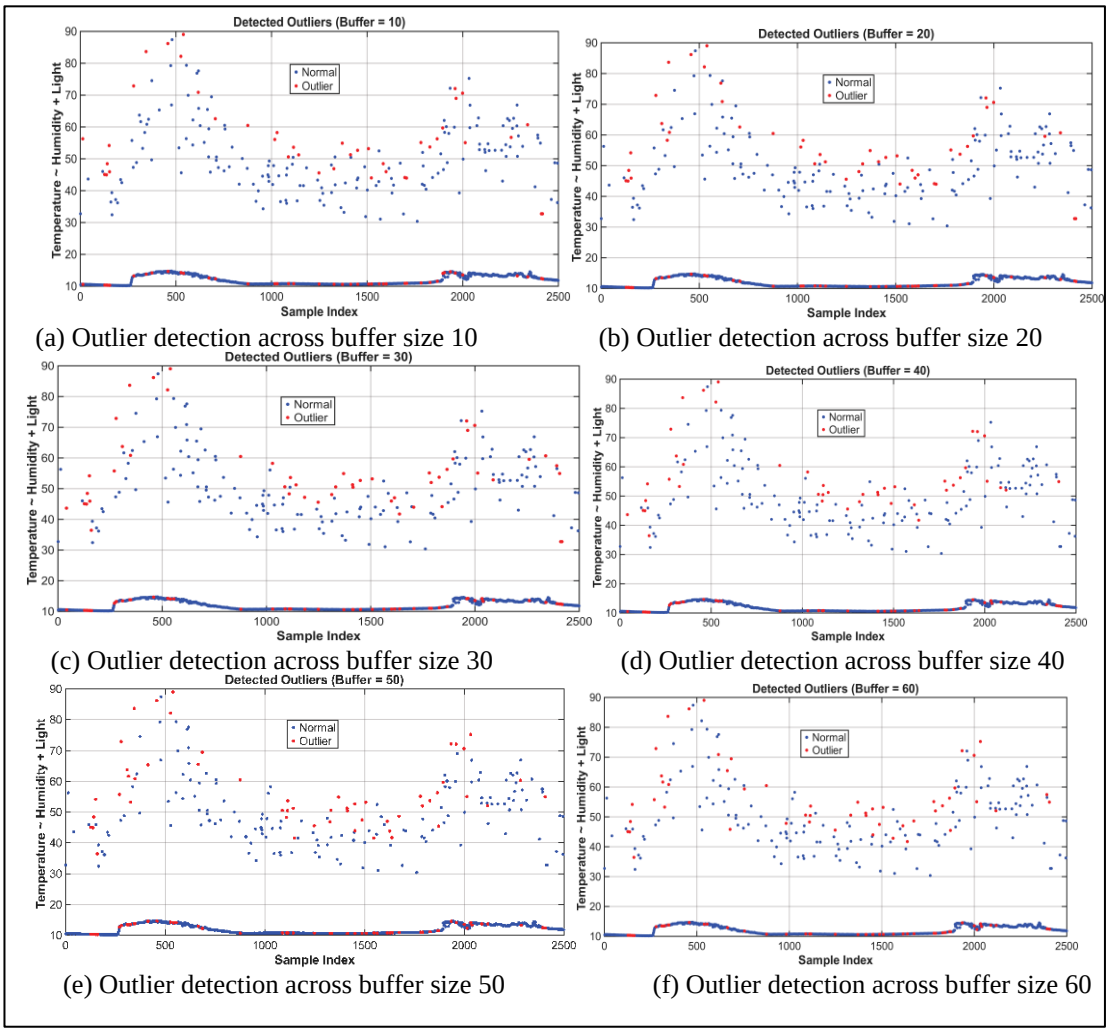
Name	Specification
ETS IoT Module	Xtensa dual-core 32-bit
Onboard LoRa module	RF96 (300kbps) with 865–867 MHz
Onboard WiFi module	802.11n (150mbps) with 2412–2484 MHz
Onboard Bluetooth module	BLE V4.2 BR-EDR
Onboard Temperature sensor	-40°C to 90°C with Accuracy ±0.3°C
Onboard Humidity sensor	0%RH–100%RH with Accuracy ±2%RH
Onboard Air Quality sensor	MQ135 to measure NH3, NOx, CO2

According to Figures 7 and 8, the model seems to perform better when *b* is more than 20 samples. The model acquired richer historical information due to the increase in buffer size (e.g., after b=20) learning to make more stable predictions. This, in turn, leads to far better detection of unusual deviations. However, too much increase in the buffer size demands more computing power. Hence, the buffer size is bounded in our study for this reason. The intention behind including a smaller buffer size (b = 10) is to study the behavior of the proposed method under limited historical information. The main goal of outlier detection in IoT/WSNs is to design effective methods that reduce communication costs while considering the limited resources of the sensor nodes, including memory and energy constraints.

4.1 Synthetic Outlier Generation

The anomaly magnitude was generated using a Gaussian distribution $\Delta \sim \mathcal{N}(0, \sigma_a^2)$ where Δ denotes the magnitude of the anomaly perturbation and σ_a controls the intensity of the anomaly. The higher values of σ_a produce more deviations from the normal sensor behavior. The parameter σ_a was selected relative to the empirical standard deviation of the corresponding sensor attribute to ensure realistic anomaly magnitudes while preserving the statistical characteristics of the original data set. For errors: $x_t' = x_t + \alpha$, For events: $x_t' = x_t + \alpha,\ t = t_0, \ldots, t_0 + D$, where x_t denotes the original sensor observation at the time instant t, x_t' denotes the modified observation after anomaly injection, α represents the anomaly intensity factor and Δ is a random perturbation. generated from a Gaussian distribution, D denotes the duration of the event, t_0 denotes the starting time of the event anomaly. Point anomalies are generated by perturbing a single observation and are treated as sensor errors. In contrast, event anomalies are injected over multiple consecutive observations, thereby preserving temporal continuity and emulating realistic environmental changes.

The duration of an event anomaly is randomly selected from a uniform distribution $D \sim U(5,30)$ where D denotes the number of consecutive observations affected by the anomaly. In this process, $D \in U(5,30)$ means that the event occupies at least part of the smallest and part of the largest buffer.



(a) Outlier detection across buffer size 10

(b) Outlier detection across buffer size 20

(c) Outlier detection across buffer size 30

(d) Outlier detection across buffer size 40

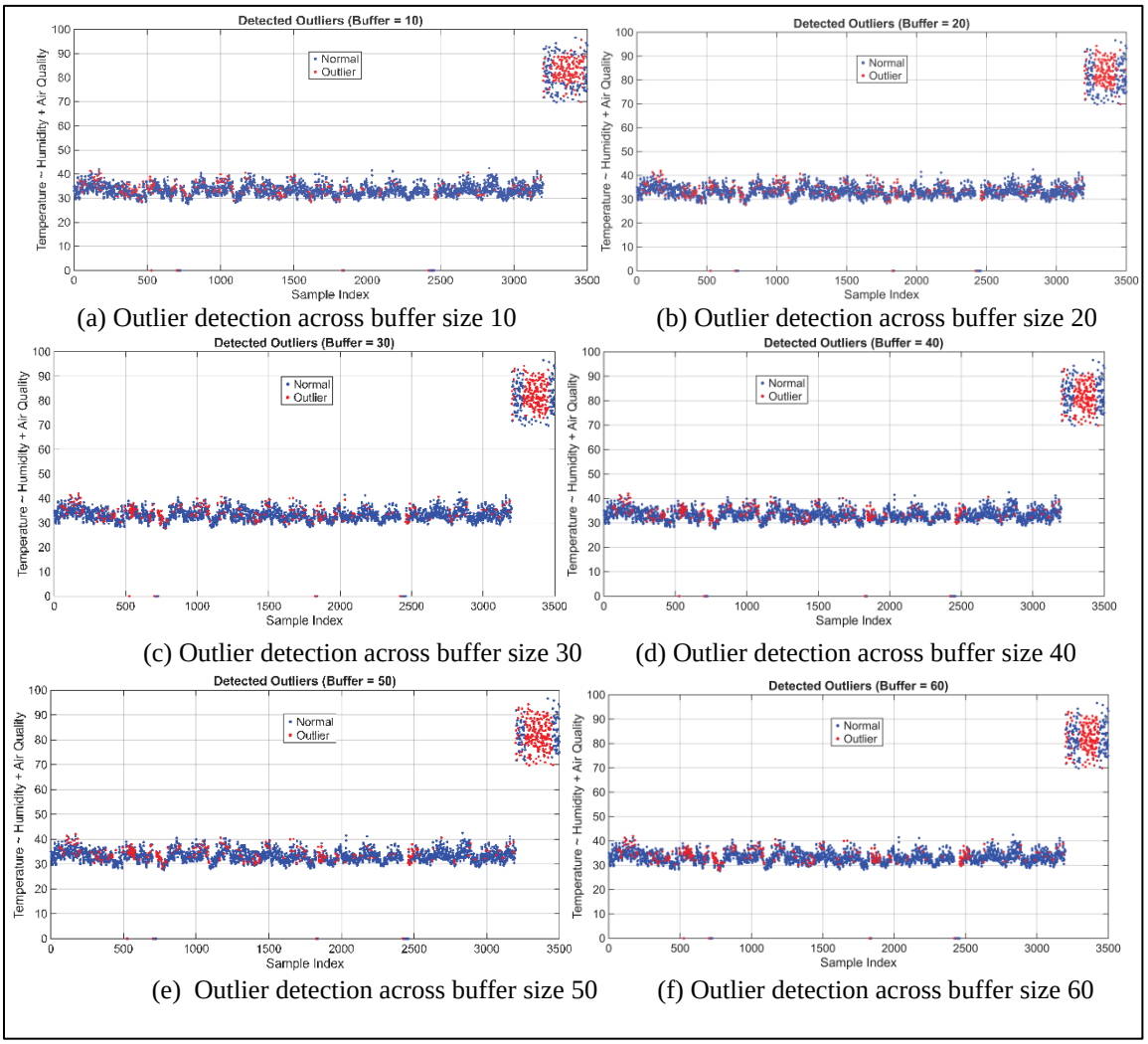
(e) Outlier detection across buffer size 50

(f) Outlier detection across buffer size 60

Figure 7. Outlier Detection across Different Buffer Sizes on IBRL Data Set

4.2 Evaluation

To measure the performance of outlier detection and classification approaches, several evaluation metrics are used. One is the Detection Rate (DR), also known as Recall or True Positive Rate, which represents the proportion of abnormal readings correctly identified as outliers. Another is the False Alarm Rate (FAR), which indicates the percentage of normal readings incorrectly flagged as outliers, as defined in [26]. In addition, the Accuracy metric (ACC) is employed, representing the overall proportion of correct detections, as defined in [24]. We observe that our proposed model yields the highest DR and lowest FAR with B=30 from both the smart campus and IBRL datasets. Performance metrics such as DR, FAR, and ACC are evaluated using the confusion matrix illustrated in Figure 9. to assess the outlier detection performance of MVRODC.



(a) Outlier detection across buffer size 10 (b) Outlier detection across buffer size 20

(c) Outlier detection across buffer size 30 (d) Outlier detection across buffer size 40

(e) Outlier detection across buffer size 50 (f) Outlier detection across buffer size 60

Figure 8. Outlier Detection Across Different Buffer Sizes on Smart Campus Data Set

- True Positive (TP): An actual outlier correctly identified by the algorithm.
- False Positive (FP): A normal data point incorrectly classified as an outlier.
- True Negative (TN): A normal data point correctly identified as a non-outlier.
- False Negative (FN): An actual outlier that the algorithm fails to detect.

The methods used for comparison with the proposed MVRODC for outlier detection and classification are Seasonal Adapted Online Outlier Detection and classification (SA-O2DCA) [13], Online Boxplot Derived Outlier Detection (OBDOD) [9], and KNN based Approximate Outlier Detection (KNN-AOD) [27]. The proposed MVRODC outperformed all other methods in terms of DR, FAR, and ACC, as shown in Figures10 and 11. The training model is constructed using 2,500 samples each from 10,000 IBRL data vectors with three attributes (temperature, humidity, and light) against various buffer sizes(B). A synthetic dataset comprising 200 outliers (100 errors and 100 events) is also employed. In this experiment, the temporal correlation between the consecutive predictive readings is assessed, and the performance metrics DR and FAR are calculated based on the confusion matrix shown in Fig.9. Eqs. 6, 7, and 8 show the formulas for DR, FAR, and ACC. Additionally, the training model is constructed using 3,500 samples each from 21,000 smart campus data vectors with three attributes (temperature, humidity, and air quality) against various buffer sizes (B). A synthetic dataset comprising 300 outliers (200 errors and 100 events) is also employed. The online MVRODC evaluation yields an average DR of 96.5%, FAR of 2%, and ACC of 97.85% across all sets of 2,500 samples taken from a total of 10,000 samples of the IBRL dataset, and an average DR of 97%, FAR of 1.5%, and ACC of 98.4% across all sets of 3,500 samples from a total of 21,000 samples of the smart campus dataset.

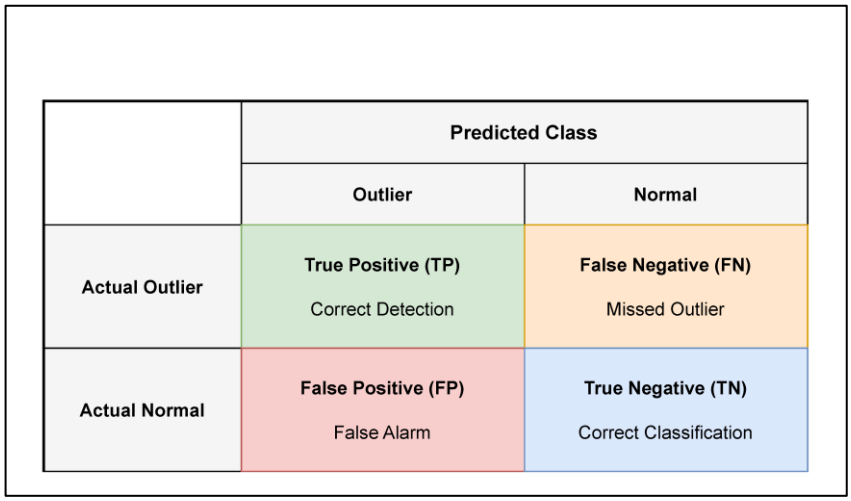


Figure 9. Confusion Matrix Used for Outlier Detection of MVRODC

$$DR = \frac{TP}{TP+FN} \tag{6}$$

$$FAR = \frac{FP}{FP+TN} \tag{7}$$

$$ACC = \frac{TP+TN}{TP+TN+FP+FN} \tag{8}$$

The current implementation of OBDOD relies on manual selection of parameters, such as the number of histogram bins, which may affect accuracy; future work aims to automate this parameter tuning to improve usability and scalability. The authors also note that the current algorithm is suitable for univariate data, but its applicability in handling multivariate time series data and detecting a broader range of events is still an area for development. This limitation indicates the need for future research to improve automation, extend functionality, and validate across broader contexts.

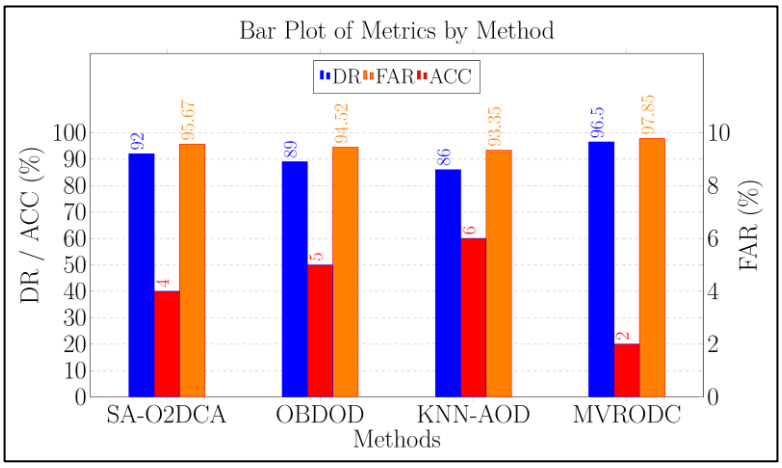


Figure 10. Performance Evaluation Using the Average of All IBRL Samples Sets, Taken from a Total Of 10,000 Samples

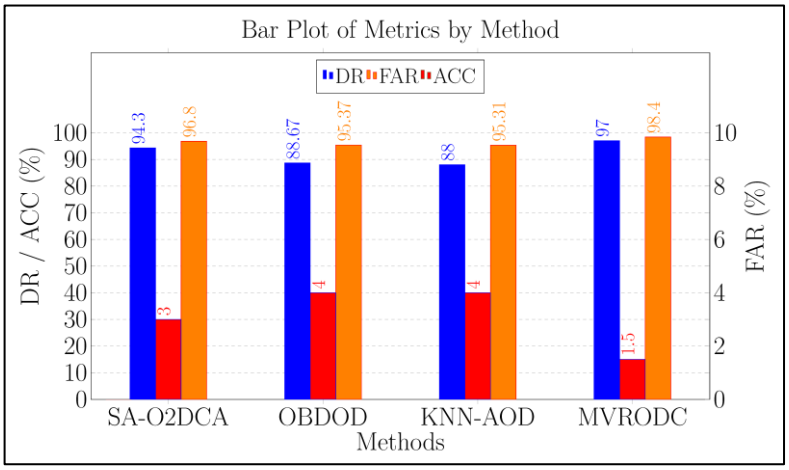


Figure 11. Performance Evaluation Using the Average of all 3,500 Smart Campus Samples Sets, Taken from a Total Of 21,000 Samples

Since KNN-AOD employs an approximate KNN search and dynamic grid adjustments, there is an inherent trade-off between accuracy and efficiency. Although the framework is designed to maintain high efficiency and acceptable accuracy, it may occasionally produce false positives or negatives in outlier detection due to approximation errors. SA-O2DCA assumes regular seasonal changes and may require adaptations for irregular or non-seasonal patterns. Parameter selection can be challenging and environment-dependent. The dynamic switch and correlation computation of models may introduce increased computational complexity. Tables 4 and 5 give a performance comparison of all methods, including the proposed MVRODC on four sets of 2500 samples taken from a total of 10,000 samples from IBRL nodes and six sets of 3500 samples taken from a total of 21,000 samples from the smart campus data set, respectively.

Table 4. Performance Comparison of DR, FAR and ACC across All Sets of 2500 Samples

Dataset	SA-O²DCA			OBDOD			KNN-AOD			MVRODC		
	DR (%)	FAR (%)	ACC (%)	DR (%)	FAR (%)	ACC (%)	DR (%)	FAR (%)	ACC (%)	DR (%)	FAR (%)	ACC (%)
SET-1	94	6	94.0	88	6	93.5	86	7	92.4	96	2	97.8
SET-2	90	3	96.4	87	5	94.4	88	8	91.7	97	3	96.9
SET-3	90	4	95.52	91	5	94.7	84	6	93.2	95	1	98.7
SET-4	94	3	96.76	90	4	95.5	86	3	96.1	98	2	98.0

Average of all the metrics in Tables. 4 and 5, illustrated in Figures10 and11. During the online classification phase, four distinct outcomes are defined:

- True Positive (TP): The algorithm correctly detects an event that actually occurred.
- False Positive (FP): The algorithm incorrectly signals an event when none occurred.
- True Negative (TN): The algorithm correctly identifies the absence of an event.
- False Negative (FN): The algorithm fails to detect an event that actually occurred.

Both ErDR and EvDR are computed from Table 6, as shown in Eqs.9 and 10. The authors classified the outliers detected from Alg.1 into errors and events using Alg.2. The metrics used to classify errors and events are shown in Eqs.9 and 10. The proposed MVRODC model outperforms other models in terms of ErDR and EvDR.The proposed MVRODC classifies outliers as errors and events at the node level. If more than half of the outliers in the buffer are similar, they are treated as events, and otherwise they are treated as errors. The authors checked the similarity of every outlier with other outliers in the buffer. This method has the advantage over other methods that the similarity between any two outliers is checked. For events, the authors choose extremely high values and, for errors, small values have been chosen. For IBRL datasets, 100 errors and 100

Table 5. Performance Comparison of DR, FAR and ACC across All Sets of 3500 Samples

Dataset	SA-O²DCA			OBDOD			KNN-AOD			MVRODC		
	DR (%)	FAR (%)	ACC (%)	DR (%)	FAR (%)	ACC (%)	DR (%)	FAR (%)	ACC (%)	DR (%)	FAR (%)	ACC (%)
SET-1	95	4	95.91	90	4	95.49	86	5	94.23	98	1	98.91
SET-2	93	2	97.57	88	3	96.23	88	3	96.23	96	3	96.91
SET-3	94	3	96.74	91	3	96.49	86	4	95.14	97	1	98.83
SET-4	93	2	97.57	90	4	95.49	90	5	94.57	98	1	98.91
SET-5	96	3	96.91	88	4	95.31	91	3	96.49	96	2	97.83
SET-6	95	4	95.91	85	6	93.23	87	4	95.23	97	1	98.83

events (total 200) are randomly injected into every set of 2500 samples. For the Smart campus data set, 200 errors and 100 events (total 300) are randomly injected into every set of 3500 samples.

Table 6. Confusion Matrix Used for Outlier Classification

Category	Classified as Event	Classified as Error
Observed Event	True Positive (TP)	False Negative (FN)
Observed Error	False Positive (FP)	True Negative (TN)

$$ErDR = \frac{TN}{TN+FP} \tag{9}$$

$$EvDR = \frac{TP}{TP+FN} \tag{10}$$

The proposed MVRODC model correctly identified 94 out of 100 events and 92 out of 100 errors, on average, for every set of 2500 samples of IBRL nodes. Additionally, it correctly identified 95 out of 100 events and 188 out of 200 errors, on average, for every set of 3500 samples from the smart campus dataset. Figures 12 and 13 show that the metrics for event and error detection rates are evaluated using the average of 2,500 IBRL data samples (taken from

a total of 10,000 samples) and the average of 3,500 smart campus data samples (taken from a total of 21,000 samples), respectively.

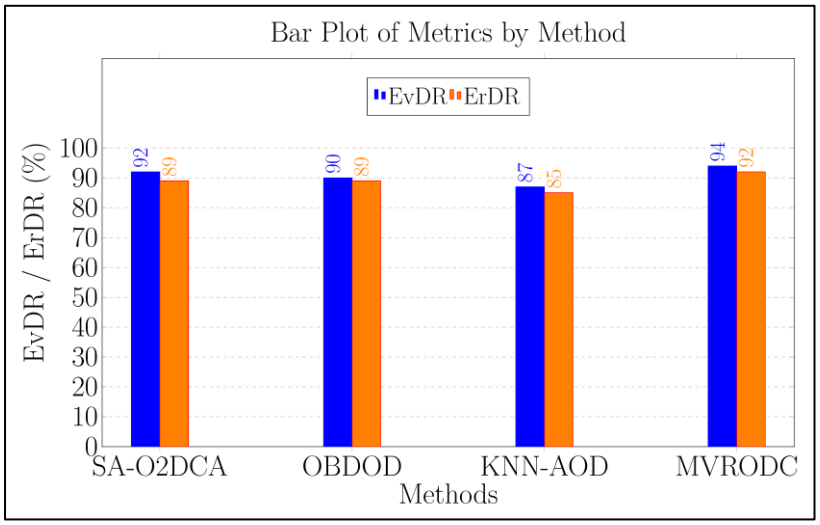


Figure 12. The Metrics for Event and Error Detection Rate are Evaluated Using the Average of 2,500 IBRL Data Samples, Taken from a Total of 10,000 Samples

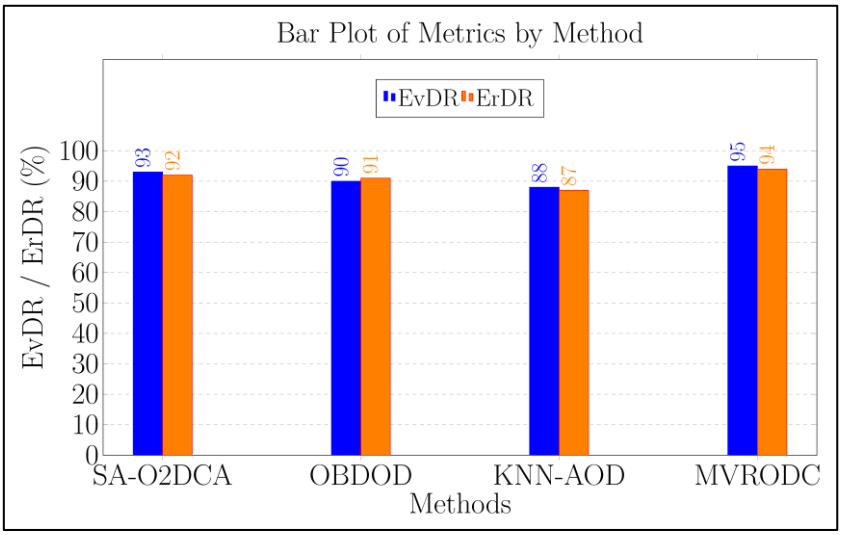


Figure 13. The Metrics for Event and Error Detection Rate are Evaluated Using the Average of 3,500 Smart Campus Data Samples, Taken from a total of 21,000 Samples

5. Conclusion

The study introduces a novel paradigm called MVRODC, which effectively detects and classifies outliers into errors and events at the node level in IoT sensor networks dynamically. The MVRODC algorithm outperforms existing outlier methods in terms of DR, FAR, ACC, ErDR, and EvDR. This approach not only improves the accuracy of real-time data collection but also conserves energy by avoiding the transmission of erroneous data, thus extending the lifetime of the sensor network. A limitation of this method is its suitability for linearly correlated data, not nonlinear data. Future work should explore methods suitable for nonlinear and high-dimensional data. The authors used a fixed threshold in the MVRODC method to ensure computational simplicity in IoT resource-constrained environments, where devices often have limited battery lifespans and processing power. However, this fixed threshold is not

always suitable for dynamically detecting outliers during data collection. Therefore, future work should incorporate an adaptive threshold that changes dynamically according to data collection. Additionally, it should explore updates to model parameters, explicit spatial correlation modeling, and data compression techniques to further enhance performance and scalability. This research indicates that the sensor node is capable of sending data directly to the Fusion Center/Base Station without relying on intermediate hops. However, to address multi-hop networks, it is essential to develop a new data collection framework tailored for multivariate sensors in IoT applications.

References

[1] Zafeirelli, Sofia, and Dimitris Kavroudakis. "Comparison of Outlier Detection Approaches in a Smart Cities Sensor Data Context." International Journal on Smart Sensing and Intelligent Systems 1 (2024).

[2] Garca, J. C., Luis A. Rivera, and Jonny Perez. "A Literature Review on Outlier Detection in Wireless Sensor Networks." Journal of Advances in Information Technology 15, no. 3 (2024).

[3] Xue, Puning, Luyang Shi, Zhigang Zhou, Jing Liu, and Xin Chen. "An Online Fault Detection and Diagnosis Method of Sensors in District Heating Substations Based on Long Short-Term Memory Network and Adaptive Threshold Selection Algorithm." Energy and Buildings 308 (2024): 114009.

[4] Alduais, Nayef Abdulwahab Mohammed, Jiwa Abdullah, and Ansar Jamil. "RDCM: An Efficient Real-Time Data Collection Model for IoT/WSN Edge with Multivariate Sensors." IEEE Access 7 (2019): 89063-89082.

[5] Al Samara, Mustafa, Ismail Bennis, Abdelhafid Abouaissa, and Pascal Lorenz. "Enhanced Efficient Outlier Detection and Classification Approach for WSNs." Simulation Modelling Practice and Theory 120 (2022): 102618.

[6] Al Samara, Mustafa, Ismail Bennis, Abdelhafid Abouaissa, and Pascal Lorenz. "Complete Outlier Detection and Classification Framework for WSNs Based on OPTICS." Journal of Network and Computer Applications 211 (2023): 103563.

[7] Hu, Zhichao, Xiangzhan Yu, Likun Liu, Yu Zhang, and Haining Yu. "ASOD: An Adaptive Stream Outlier Detection Method Using Online Strategy." Journal of Cloud Computing 13, no. 1 (2024): 120.

[8] Dani, Yasi, Agus Yodi Gunawan, Masayu Leylia Khodra, and Sapto Wahyu Indratno. "Detecting Outliers Using Modified Recursive Pca Algorithm for Dynamic Streaming Data." In MENDEL, vol. 29, no. 2, 2023, 237-244.

[9] Mazarei, Arefeh, Ricardo Sousa, João Mendes-Moreira, Slavo Molchanov, and Hugo Miguel Ferreira. "Online Boxplot Derived Outlier Detection." International journal of data science and analytics 19, no. 1 (2025): 83-97.

[10] Shu, Zhinian, and Xiaorong Li. "The Detection Method of Continuous Outliers in Complex Network Data Streams Based on C-LSTM." International Journal of System Assurance Engineering and Management 15, no. 9 (2024): 4582-4593.

[11] Abhaya, Abhaya, and Bidyut Kr Patra. "An Efficient Method for Autoencoder Based Outlier Detection." Expert Systems with Applications 213 (2023): 118904.

[12] Antonius, Franciskus, J. C. Sekhar, Vuda Sreenivasa Rao, Rahul Pradhan, S. Narendran, Ricardo Fernando Cosio Borda, and Susan Silvera-Arcos. "Unleashing the Power of Bat Optimized CNN-BiLSTM Model for Advanced Network Anomaly Detection: Enhancing Security and Performance in IoT Environments." Alexandria Engineering Journal 84 (2023): 333-342.

[13] Samara, Mustafa Al, Ismail Bennis, Abdelhafid Abouaissa, and Pascal Lorenz. "Sa-o2dca: Seasonal Adapted Online Outlier Detection and Classification Approach for WSN." Journal of Network and Systems Management 32, no. 2 (2024): 31.

[14] Malki, Amer, El-Sayed Atlam, and Ibrahim Gad. "Machine Learning Approach of Detecting Anomalies and Forecasting Time-Series of IoT Devices." Alexandria Engineering Journal 61, no. 11 (2022): 8973-8986.

[15] Lai, Trinh Thuc, Tuan Phong Tran, Jaehyuk Cho, and Myungsik Yoo. "DoS Attack Detection Using Online Learning Techniques in Wireless Sensor Networks." Alexandria Engineering Journal 85 (2023): 307-319.

[16] Rodríguez, Martha, Diana P. Tobón, and Danny Múnera. "A Framework for Anomaly Classification in Industrial Internet of Things Systems." Internet of Things 29 (2025): 101446.

[17] Krleža, Dalibor, Boris Vrdoljak, and Mario Brčić. "Statistical Hierarchical Clustering Algorithm for Outlier Detection in Evolving Data Streams." Machine Learning 110, no. 1 (2021): 139-184.

[18] Gupta, Umang, Vandana Bhattacharjee, and Partha Sarathi Bishnu. "Outlier Detection in Wireless Sensor Networks Based on Neighbourhood." Wireless Personal Communications 116, no. 1 (2021): 443-454.

[19] Singh, Manmohan, and Rajendra Pamula. "ADINOF: Adaptive Density Summarizing Incremental Natural Outlier Detection in Data Stream." Neural Computing and Applications 33, no. 15 (2021): 9607-9623.

[20] Pekşen, Muhammed Fatih, Ulaş Yurtsever, and Yılmaz Uyaroğlu. "Enhancing Electrical Panel Anomaly Detection for Predictive Maintenance with Machine Learning and IoT." Alexandria Engineering Journal 96 (2024): 112-123.

[21] Salilew, Waleligne Molla, Zainal Ambri Abdul Karim, and Tamiru Alemu Lemma. "Investigation of Fault Detection and Isolation Accuracy of Different Machine Learning Techniques with Different Data Processing Methods for Gas Turbine." Alexandria Engineering Journal 61, no. 12 (2022): 12635-12651.

[22] Wei, Yuanyuan, Julian Jang-Jaccard, Wen Xu, Fariza Sabrina, Seyit Camtepe, and Mikael Boulic. "LSTM-Autoencoder-Based Anomaly Detection for Indoor Air Quality Time-Series Data." IEEE Sensors Journal 23, no. 4 (2023): 3787-3800.

[23] Rollo, Federica, Chiara Bachechi, and Laura Po. "Anomaly Detection and Repairing for Improving Air Quality Monitoring." Sensors 23, no. 2 (2023): 640.

[24] Brahmam, M. Veera, and S. Gopikrishnan. "NODSTAC: Novel Outlier Detection Technique Based on Spatial, Temporal and Attribute Correlations on IoT Bigdata." The Computer Journal 67, no. 3 (2024): 947-960.

[25] Intel Berkeley Research Lab. “Intel Lab Data.” (2004).

[26] Chander, Bhanu, and G. Kumaravelan. "Outlier Detection Strategies for WSNs: A Survey." Journal of King Saud University-Computer and Information Sciences 34, no. 8 (2022): 5684-5707.

[27] Zhu, Rui, Xiaoling Ji, Danyang Yu, Zhiyuan Tan, Liang Zhao, Jiajia Li, and Xiufeng Xia. "KNN-Based Approximate Outlier Detection Algorithm over IoT Streaming Data." IEEE Access 8 (2020): 42749-42759.