International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 24s, 2025
https://theaspd.com/index.php

Automatic Outlier Detection And Data Consistency Maintenance Technique For Iot Healthcare Applications

[1*]**E.Nandhinipriya,** [2*]**Dr. R. Manikandan,** [3*]**Dr. K. Kamali**
[1]Assitant Professor, Dept. of Computer science and Engineering, Annamalai University. Chidambaram, Tamilnadu.
[2]Associate Professor, Dept of Computer science and Engineering, Annamalai University. Chidambaram, Tamilnadu.rmkmanikandan1111@gmail.com
[3]Assistant Professor, Dept of Computer and Information science, Annamalai University,Chidambaram, Tamilnadu. Kamaliaucse2006@gmail.com

Abstract
In a typical health-care system, various sensor nodes are employed for monitoring different vital parameters of a patient. In Internet of Things (IoT) based healthcare systems, data correctness is essential since clinical decisions often depend on real-time physiological data collected from sensors and wearable devices. In this paper, an automatic outlier detection and data consistency maintenance (AOD-DCM) technique for IoT-WSN is proposed. It consists of data outlier detection and data inconsistency checking phases. For data outlier detection, the Principal Component Analysis (PCA) algorithm is applied. For data inconsistency checking from existing data readings, the Deep Auto Encoder (DAE) model is designed. To fine tune the parameters of DAE, the bio inspired Brown Bear Optimization (BBO) algorithm is applied. Experimental results show that DAE-BBO technique achieves higher detection accuracy and correctness of data, when compared to the existing outlier detection techniques.
Keywords: *Internet of Things (IoT) , Healthcare data, Outlier detection, Consistency, Deep Auto Encoder (DAE), Brown Bear Optimization (BBO)*

1. INTRODUCTION

During Industry 4.0 revolution, real-world objects are progressively integrated and automated via the Internet of Things (IoT). This paradigm allows millions of devices equipped with sensors and actuators to connect through wired or wireless communication, enabling seamless data transmission. IoT devices produce massive volumes of data with diverse modalities and quality. These devices are anticipated to generate around 79.4 zettabytes (ZBs) of real-time data, highlighting the immense scale and potential of the IoT ecosystem [1]. In healthcare, IoT is driving transformative variations by allowing remote diagnostics, continuous patient monitoring, and personalized treatment. Smart medical devices, like glucose meters, wearable ECG monitors, and connected inhalers, gather real-time health metrics and send them to cloud-based systems for analysis. This enables clinicians to make timely, data-driven decisions and enhance patient outcomes. IoT also supports patient flow optimization, hospital asset tracking, and predictive maintenance of medical equipment.

In spite of these benefits, the security and reliability of the collected data are vital for successful deployment. Imprecise or tampered data can compromise treatment and diagnosis. So, ensuring data integrity, quality, and privacy is crucial in building trustworthy and efficient IoT-enabled healthcare systems [2].

In IoT-based healthcare systems, data correctness is essential since clinical decisions often depend on real-time physiological data collected from sensors and wearable devices. However, ensuring correctness in these dynamic and distributed environment is difficult. One major problem is sensor noise and drift, which can cause inaccuracies over time because of hardware degradation or environmental factors. Moreover, data transmission errors due to unreliable network connectivity may cause packet loss, duplication, or corruption. Another significant concern is device heterogeneity, in which different vendors use variable standards, sampling rates, and data formats, causing inconsistencies in collected data. Temporal misalignment between devices further worsens the problem, particularly in scenarios requiring synchronized measurements (like heart rate and oxygen levels).

Additionally, malfunctioning sensors or unauthorized tampering may cause anomalous values, compromising the entire system's integrity. Incorrect or inconsistent data may cause delayed diagnosis, false alarms, or even life-threatening decisions. Consequently, detecting outliers automatically and maintaining data consistency are vital for ensuring reliability [3].

IoT devices in healthcare are frequently installed in harsh or dynamic environments, in which establishing reliable communication and maintaining data integrity is complex. Originally developed for military usage, IoT systems have extended to critical domains like healthcare, disaster monitoring, and smart cities. However, the efficiency of these systems heavily based on the quality and reliability of the sensor data they collect. IoT sensor nodes are susceptible to internal problems like hardware faults, energy limitations, and noise, along with external threats such as denial-of-service or replay attacks [4].

These factors cause the generation of noisy, missing, or incorrect data—commonly called as outliers. Outliers can severely compromise real-time decision-making in healthcare applications, in which timely and accurate data is vital for monitoring vital signs, detecting anomalies, and triggering emergency responses. Conventional outlier detection methods depending on thresholds or statistical rules fail to scale, adapt, and generalize across diverse and high-dimensional IoT data. These approaches often impacted due to high false alarm rates and limited detection accuracy. Hence, there is a pressing requirement for advanced outlier detection techniques [5].

In IoT-based healthcare applications, ensuring data consistency is vital for the accuracy, reliability, and safety of medical decisions. Working with a consistent dataset creates the foundation for developing reliable clinical applications and procedures. Data consistency refers to correct storage formats, valid schema structures, the preservation of relationships amongst health records, accurate representation of physiological phenomena, and alignment with intended clinical use [6]. Healthcare IoT environments are fundamentally heterogeneous, including a wide range of sensors, devices, and standards.

This diversity often causes transmission delays, inconsistencies in data formats, and semantic mismatches. Hence, consistency should be ensured at multiple levels—data characteristics, data quality, and information quality. Inconsistent or corrupted data—because of sensor faults, transmission errors, or false data injection (FDI) attacks—can cause delayed treatment, misdiagnosis, or system malfunction. Unlike model-based systems in which clean data can be simulated, data-driven healthcare applications must struggle with real-world noise and anomalies. Therefore, identifying and correcting inconsistencies before further analysis is vital. Integrating automated outlier detection and consistency maintenance mechanisms confirms that only valid, trustworthy data drives healthcare decisions, ultimately improving patient safety and care effectiveness [7].

2. RELATED WORKS

A novel method is proposed in [7] to detect errors introduced during software refactoring and improve the security and correctness of IoT software. This approach uses both control flow and data flow analyses to observe changes in the program structure before and after refactoring. In addition, synchronization dependency analysis is utilized to identify modifications in synchronization dependencies that may influence software behaviour. Three specialized detection algorithms are developed for verifying the correctness of refactoring actions. The method is evaluated using four real-world benchmark IoT applications. Simulation results confirm the method's efficiency in maintaining the integrity and security of refactored IoT software systems.

An advanced maintenance analysis model [8] is proposed to significantly improve fault detection and risk evaluation in industrial systems. The model connects machine failure reports with sensor-generated data and excels in managing inaccurate and fuzzy data inputs, which are common in real-world industrial environments. Conventional failure evaluation models recognized FM10, FM8, and FM7 as the highest priority faults. Though, the model re-ranks failures based on weighted risk factors using a fuzzy Failure Mode and Effects Analysis (FMEA) approach combined with the MULTIMOORA decision-making method. Consequently, FM4, FM10, and FM6 are prioritized for initial corrective action. The study proves that applying fuzzy set theory and MULTIMOORA enhances the accuracy of failure ranking and helps recognize both root and contributing causes.

In [9], two strategies are proposed for managing data replication and consistency in fog computing environments. These strategies dynamically find the optimal number and placement of data replicas for every IoT datum to lessen latency and synchronization costs, ensuring the required consistency levels are satisfied. A simulation platform is developed using iFogSim, which enables users to implement and test customized replication strategies. These techniques lessen service latency by up to 30% in small-scale fog infrastructures and by 13% in large-scale deployments than iFogStor, a baseline strategy without replication mechanisms. It highlights the importance of intelligent data management in enhancing the performance of distributed IoT systems.

An algorithm [10] is proposed for fault diagnosis that utilizes IoT sensor data to diagnose mechanical faults such as rotor imbalance, bearing defects, shaft misalignment, and belt looseness. A preprocessing step depends on descriptive statistics to reduce the dimensionality of the sensor data, thus improving the computational efficiency of the algorithm. Both current and vibration data are utilized for ensuring reliable fault classification. Amongst several machine learning models tested, XGBoost version 1.7.6 produces the highest classification performance. It improves both the accuracy and speed of fault detection, providing significant contributions to IIoT safety and operational reliability.

A novel unsupervised outlier detection framework is introduced in [11] using Autoencoder-based models with contrastive loss. This technique integrates the strength of autoencoders for latent feature extraction and contrastive learning for improved discriminative capability. The model lessens a total loss function that comprises both reconstruction loss and contrastive loss, reassuring compact clustering of normal data and separation of anomalies. A statistically driven thresholding method flags outliers depending on reconstruction error. The model is assessed on the Statlog dataset through precision, recall, and F1-score, and proves high effectiveness in identifying outliers.

An unsupervised outlier detection approach is proposed in [12] using a deep Variational Autoencoder (VAE). The model exploits the VAE's ability for learning low-dimensional latent representations of input data and accurately reconstruct them. The process initiates with standardizing the input data, which is followed by encoding and decoding via the VAE. The reconstruction error between the input and output is estimated for scoring anomalies. Particularly, the model is exclusively trained on normal data without any labelled anomalies, following a fully unsupervised approach. The method attains approximately 90% precision and an F1-score of 79%, when tested on the Statlog Landsat Satellite dataset, performing on par with leading outlier detection methods.

2.1 Research Gaps

In spite of significant development in outlier detection, several problems remain unaddressed in IoT-based healthcare applications. Communication cost is a major concern—sending raw sensor data to a central server consumes significantly more energy when compared to local computation. This is unmanageable for healthcare IoT networks that depend on energy-constrained devices. Dynamic network topology due to node failures, mobility, and heterogeneous sensing capabilities affects network stability and interrupts existing detection models. Resource constraints, like limited memory, power, and processing capacity of sensor nodes, restrict the use of traditional, computation-heavy outlier detection algorithms.

Distributed streaming data is difficult as several existing techniques assume static or offline datasets. IoT healthcare environments yield real-time, high-velocity data streams that necessitate adaptive, online methods capable of handling incomplete and transient information. High-dimensional sensor data, common in healthcare such as ECG, temperature, and blood pressure, increases computational burden and can worsen model performance because of the curse of dimensionality.

Additionally, data inconsistency and label scarcity in medical IoT datasets affect the development of supervised or semi-supervised models. Privacy-preserving outlier detection needs further improvement, especially for sensitive patient data. The absence of robust benchmark datasets for IoT healthcare restricts fair evaluation and comparison of detection methods.

3. PROPOSED METHODOLOGY

3.1 Overview

In this paper, an automatic outlier detection and data consistency maintenance (AOD-DCM) technique for IoT-WSN is proposed.

It consists of two phases:

- Data Outlier detection
- Data inconsistency checking

For data outlier detection, the PCA algorithm is applied. For data inconsistency checking from existing data readings, the Deep Auto Encoder (DAE) model is designed. To fine tune the parameters of DAE, the bio inspired Brown Bear Optimization (BBO) algorithm is applied

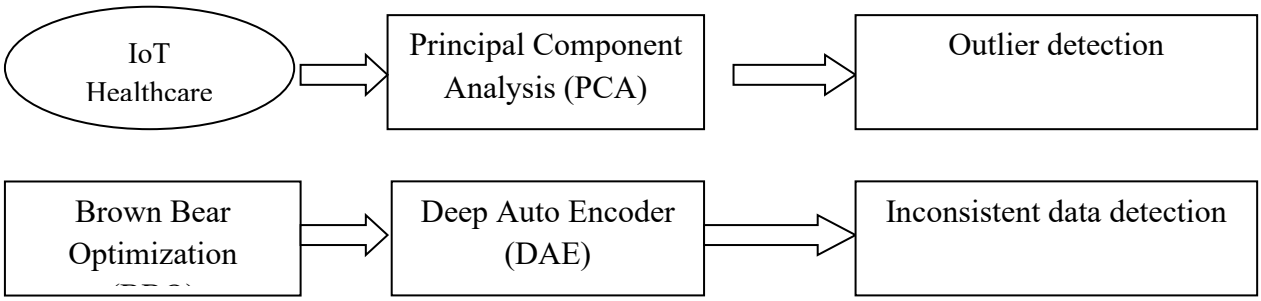

Figure 1 Block diagram of AOD-DCM technique

3.2 PCA for Dimensionality reduction

For outlier detection, PCA technique is applied which is an unsupervised non-linear technique.

PCA is utilized to remove the features with many interrelated variables while maintaining as much of the original variance as possible. This is attained by generating a new set of uncorrelated variables called principal components (PCs), which are organized so that the first few PCs capture the majority of the variance from all the original variables.

The objective of PCA is to create a d x k transformation matrix P, which maps the dataset vector into a new subspace with less dimensions compared to the original one. For instance, when the data contains two axes (M_1 and M_2), the aim is to decrease it to PC_1 and PC_2 that capture the most variance from the dataset. The new dimensions should be less than the original ones.

$x = [x1, x2, x3, \dots \dots \dots \dots , x_d], x \epsilon RD$

The first step is to determine a linear function $\alpha' 1\, X$ of the samples of the dataset with maximum variance, in such a way that α is a vector with constants $\alpha_{11}, \alpha_{12}, \alpha_{13}, \alpha_{14}$ and $'$ indicates the transpose.

A linear function $\alpha'_2 x$ is considered that is uncorrelated with $\alpha'_1 x$ but has the maximum variance. A linear function $\alpha'_k x$ with the greatest variance is uncorrelated with every preceding functions $\alpha_1 x, \alpha'_{2x}, \dots .. \alpha'_{k-1x}$. The k^{th} PC is denoted by the k^{th} derived variable, $\alpha' \kappa x$.

Although it is possible to recognize many principal components, most of the variation in x is usually captured by a set of m principal components, where m is much larger than p.

It is possible to find as many possible principal components, but most of the variation in x is expected to be explained by m number of principal components.

The dataset was standardized through the Standard Scaler method from preprocessing class of Scikit-Learn, and then converted so that all features have equivalent weights.

3.3 DAE architecture

This DAE architecture combines autoencoder-based representation learning with the discriminative strength of contrastive loss. The method is developed for effectively differentiating between normal and anomalous data points by learning informative latent representations.

Given an input dataset $X = \{x_1, x_2, \dots, x_n\}$, in which each data point $x_a \in R^d$, the autoencoder contains two core components:

- an encoder f_θ that projects the input into a latent space, and
- a decoder g_ϕ that reconstructs the original input from its latent representation.

The encoder converts each x_i into a latent vector $z_a = f_\theta (x_a) \in R^m$. Then, the decoder reconstructs the input as $\hat{x}_a = g_\phi (z_a)$.

The reconstruction loss measures the difference between the original data and its reconstruction, which is given by,

$$L_r = \frac{1}{m}\sum_{a=1}^{m} \|x_a - \hat{x}_a\|^2 \quad (1)$$

For ensuring that similar data points are mapped to neighbouring locations in the latent space—while dissimilar points contain distinct representations—a contrastive loss is introduced. For a given pair (x_a, x_b), it is given by,

$$L_c = \frac{1}{m}\sum_{a=1}^{m}\sum_{b=1}^{m} y_{ab}\|z_a - z_b\|^2 + (1 - y_{ab}) \max(0, \gamma - \|z_a - z_b\|)^2 \quad (2)$$

Where $y_{ab} = 1$ when x_a and x_b form a positive pair (similar), and 0 otherwise. α is the parameter that defines the margin between dissimilar pairs.

The final total loss function combines both reconstruction and contrastive losses:

$L_T = L_r + \lambda * (L_c)$ (3)

where λ indicates a hyperparameter used for balancing the contribution of the two loss components.

3.3.1 Training Procedure

The algorithm starts by initializing the encoder and decoder parameters, which is denoted as θ and ϕ, respectively. Training proceeds over multiple epochs, during which the input data is split into mini-batches. In each mini-batch, pairs of data points—both negative (dissimilar) and positive (similar)—are sampled for computing the contrastive loss.

For each data pair (x_a, x_b), their related latent representations z_a and z_b are obtained through the encoder f_θ. The reconstructed outputs $\hat{x}_a$ are generated by feeding the latent vectors through the decoder $g\phi$. The reconstruction loss L_r is computed as the mean squared error between the reconstructed and original inputs.

The contrastive loss L_c is calculated to promote clustering of similar data points in the latent space and separation of dissimilar ones. The total loss L_T is then obtained by integrating the reconstruction and contrastive losses, with a weighting factor λ.

The encoder and decoder parameters (θ and ϕ) are then updated through gradient descent with a learning rate η for minimizing the total loss. This optimization is iterated for each mini-batch across all training epochs. When training is completed, the optimized encoder f_θ and decoder g_ϕ are returned.

During training, the objective is to minimize L_T, in which the contrastive loss boosts compact representations for normal samples and separation for potential outliers. During inference, a test data point x_a is flagged as an outlier when its reconstruction error $\|x_a - \hat{x}_a\|$ exceeds a predefined threshold. Anomalies in IoT data are effectively detected by integrating autoencoder-based reconstruction with contrastive representation learning. The inclusion of contrastive loss improves the model's capability to learn discriminative features, thus enhancing overall outlier detection performance.

The IoT ecosystem creates huge volumes of data from a wide array of interconnected devices. This data is vital for applications like anomaly detection, predictive maintenance, and security monitoring. Though, the velocity, sheer volume, and diversity of IoT data cause challenges for precise outlier detection. Autoencoder-based model is enhanced with Contrastive Loss for detecting outliers in IoT data.

To identify anomalies, the learned latent representations are used with the help of the trained encoder f_θ and decoder g_ϕ obtained during the training phase. During inference, every incoming IoT data point $x_a \in R^d$ is fed through the encoder for producing a latent vector $z_a = f_\theta(x_a)$, which is then reconstructed as $\hat{x}_a = g_\phi(z_a)$.

The reconstruction error for each data point is given by,

$$r_a = \|x_a - \hat{x}_a\| \quad (4)$$

For determining if a data point is an outlier, its reconstruction error r_a is compared against a predefined threshold τ. A point is considered as an outlier when its error is more than this threshold:

$$O(x_a) = \begin{cases} 1 & if\ r_a > \tau \\ 0 & otherwise \end{cases} \quad (5)$$

τ is selected using statistical analysis of reconstruction errors on a validation dataset or adjusted depending on the requirements of the specific IoT application.

Algorithm 1 Training Procedure for DAE

Input: Training data $X = \{x_1, x_2, \ldots, x_n\}$, learning rate η, margin α, balance parameter λ, batch size B
Ensure: f_θ and g_ϕ
1: Initialize parameters θ and ϕ
2: For number of training epochs do
3: for each mini-batch $\{x_1, x_2, \ldots, x_B\}$ do
4: Sample negative and positive pairs for contrastive loss
5: for each pair (xi , xj) in the mini-batch do
6: Calculate latent representations z_a and z_b
7: Calculate reconstruction: $\hat{x}_a$
8: Calculate reconstruction loss L_r
9: Calculate contrastive loss L_c
10: end for
11: Calculate total loss L_T
12: Update θ and ϕ using gradient descent
13: end for

14: end for
15: return f_θ and $g_\phi = 0$

3.3.2 Outlier Detection

Algorithm 2 discussed the detailed algorithm for outlier detection using the trained autoencoder and contrastive loss.

Algorithm 2 Outlier Detection in IoT

Input: IoT data $X_t = \{x_1, x_2, \ldots, x_m\}$, trained encoder f_θ, trained decoder g_φ, threshold τ Ensure: Outlier labels for test data
1: for each x_i do
2: Calculate latent representation: z_a
3: Reconstruct data point: $\hat{x}_a$
4: Calculate reconstruction error: r_a
5: Determine outlier label
6: If ri > τ
7: outlier(x_a) = 1
8: Else
9: outlier(x_a) = 0
10: end for
11: return Outlier labels for $X_t = 0$

3.4 Brown-bear Optimization (BBO) Algorithm

To fine tune the parameters of DAE, the bio inspired BBO algorithm is applied.

The proposed optimization algorithm has two primary phases, which is inspired by the pedal scent marking and sniffing behaviours found in brown bears. The pedal scent marking phase is classified into three subcategories, each happening with equal probability. In this algorithm, every group of brown bears within a territory is considered as an individual solution set in the population. The pedal scent marks left by every group signify the decision variables in the corresponding solution set. The territory itself represents the search space of the problem. Like other population-based optimization approaches, the algorithm starts with the initialization of the population. Multiple bear groups are generated randomly within the territory, each having a predefined number of pedal scent marks. These marks are unique to every group and are distributed within the territory's bounds, which relate to the defined limits of the decision variables in the optimization problem. These groups' random initialization within the territory is mathematically given by,

$$P_{i,j} = P_{i,j}^{min} + \lambda(P_{i,j}^{max} - P_{i,j}^{min}) \quad (6)$$

where $P_{i,j}$ is the ith group's jth pedal mark of brown-bears. $P_{i,j}^{max}$ and $P_{i,j}^{min}$ are the maximum and minimum range of pedal marks, respectively. λ indicates any random number evenly distributed within the range [0,1]. When N_{pop} is the total number of groups in a territory and D is the total number of pedal marks in each group, the solution set P is indicated as

$$P = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,D} \\ P_{2,1} & P_{2,2} & \cdots & P_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ P_{N_{pop},1} & P_{N_{pop},2} & \cdots & P_{N_{pop},D} \end{bmatrix} \quad (7)$$

3.4.1 Pedal Scent Marking Behavior

The pedal scent marking behaviour showed by brown bears is remarkably unique. It includes a distinct walking pattern characterized by a specific gait, deliberate foot placement, and the twisting of feet into ground depressions. These three behavioural traits are mathematically modelled for representing the scent marking process, with each trait happening with equal probability. Let N_I be the total number of algorithm iterations. Each characteristic is modelled to occur over one-third of the total iterations.

Pedal scent marking is displayed mainly by male brown bears. For simplification, each group is presumed to comprise one male member. This male bear shows a unique walking gait, leading to distinctly recognizable pedal scent marks. It is assumed that this behaviour, depending on the characteristic gait,

dominates the first one-third of the total iterations N_I. The mathematical model for this behaviour is defined by,

$$P^n_{i,j,k} = P^o_{i,j,k} - (\theta_k \gamma_{i,j,k} P^o_{i,j,k}) \quad (8)$$

Where $P^n_{i,j,k}$ is the updated position of the jth pedal scent mark for the ith group of brown bears during the kth iteration, whereas $P^o_{i,j,k}$ is its previous position in the same iteration. The variable $\gamma_{i,j,k}$ is a random value uniformly distributed in the range of 0 to 1, related to the jth pedal mark of the ith group at iteration k. The term θ_k is the occurrence factor for the kth iteration that increases linearly when the iterations progress. It is defined as the proportion of the current iteration number C_I to the total number of iterations N_I, expressed as,

$$\theta_k = \frac{C_I}{N_I} \quad (9)$$

During the second phase—spanning from one-third to two-thirds of the total number of iterations—the pedal scent marks are updated depending on the careful stepping behaviour, reflecting the tendency of brown bears to repetitively step on their own previous scent marks to improve their visibility and recognizability to other members of the group. The aim of this action is to strengthen the scent marks, making them more prominent.

The mathematical formulation for this characteristic is given by:

$$P^n_{i,j,k} = P^o_{i,j,k} - F_k(P^b_{i,j,k} - L_k P^w_{i,j,k}) \quad (10)$$

where $P^b_{i,j,k}$ and $P^w_{i,j,k}$ are the best and worst jth pedal scent marks, respectively, among all bear groups in the kth iteration. The term F_k is the step factor for the current iteration influenced by the occurrence factor θ_k and is defined as:

$$F_k = \beta_{1,k} \theta_k \quad (11)$$

Where $\beta_{1,k}$ is a randomly generated number within the range [0,1] for the kth iteration. The term L_k is the step length at iteration k, which plays a vital role in determining how the pedal scent marks are adjusted using information from both the worst and best marks in the population.

The step length L_k can be 1 or 2. Depending on this value, the male brown bear of a given group alters its movement carefully—either backward or forward—to leave new pedal scent marks accordingly. The step length is given by,

$$L_k = \text{round}\ (1 + \beta_{2,k}) \quad (12)$$

where $\beta_{2,k}$ indicates a random number, which is uniformly distributed between 0 and 1 for the k^{th} iteration.

The updating of pedal scent marks is governed by the twisting feet behaviour, from the final third of the total iterations up to the last iteration. In this phase, the male brown bear from every group twists its feet into previously created ground depressions. This behaviour helps strengthen and solidify the pedal scent marks, which are later utilized by other group members to construct scent maps.

The previous marks selected for reinforcement are depending on their proximity to the best and worst marks in the population. Bears prefer to strengthen those nearer to the best marks and further from the worst. The angular velocity with which a bear accomplishes this twisting motion is given by:

$$\omega_{i,k} = 2\ \pi \cdot \delta_{i,k}\ \theta_k \quad (13)$$

where $\omega_{i,k}$ is the angular velocity of the ith bear during the kth iteration, and $\delta_{i,k}$ is a random number uniformly distributed in the range [0,1].

A brown bear twists its feet onto preceding pedal marks that are nearer to the best-performing marks and farther from the worst-performing ones. This behaviour is given by,

$$P^n_{i,j,k} = P^o_{i,j,k} + \omega_{i,k}\left(P^b_{j,k} - \left|P^o_{i,j,k}\right|\right) - \omega_{i,k}\left(P^w_{j,k} - \left|P^o_{i,j,k}\right|\right) \quad (14)$$

After this phase, the better-performing groups of bears are chosen to participate in the next stage of the algorithm.

3.4.2 Sniffing Behaviour

Sniffing behaviour is a common trait showed by all members within a brown bear group. It enables communication amongst group members by enabling them to detect and interpret information from pedal scent marks. This behaviour plays a vital role in guiding their movement across the territory.

To navigate, bears initiate by randomly sniffing pedal scent marks situated throughout the territory. They are inclined to move toward the marks related to their own group, overlooking those from other groups. This behaviour is mathematically modelled by choosing two random candidate solutions and updating the bear's movement depending on the following rule:

$$P_{i,j,k}^{n} = \begin{cases} P_{m,j,k}^{o} + \delta_{j,k}\left(P_{m,j,k}^{o} - P_{n,j,k}^{o}\right) & if\ f\left(P_{m,k}^{o}\right) < f\left(P_{n,k}^{o}\right) \\ P_{m,j,k}^{o} + \delta_{j,k}\left(P_{n,j,k}^{o} - P_{m,j,k}^{o}\right) & if\ f\left(P_{n,k}^{o}\right) < f\left(P_{m,k}^{o}\right) \end{cases} \quad (15)$$

where $P_{m,j,k}^{n}$ indicates the updated value of the jth pedal mark for the mth group during the kth iteration. $P_{m,j,k}^{o}$ and $P_{m,j,k}^{o}$ indicate the corresponding pedal marks for the mth and nth groups (with m ≠ n), whereas $f\left(P_{m,k}^{o}\right)$ and $f\left(P_{n,k}^{o}\right)$ indicate their corresponding fitness values. $\delta_{j,k}$ indicates a uniformly distributed random number in the range [0,1] for the jth pedal mark in the kth iteration.

This sniffing phase is implemented across all groups in the population. After the update, the best-performing groups from both the new and previous populations are reserved and passed on to the next iteration. The update and selection process in the previous behaviours continues until a predefined termination criterion is satisfied.

Algorithm 3 Pseudo code of proposed BOA

Inputs:

- N_I = Total number of iterations
- N_{pop} = Population size
- D =Number of decision variables
- Variable boundaries

Ensure: Fitness function f(P) is defined

1. Initialize a random population of bear groups
2. Form the population matrix P
3. Set the best fitness value $f_{best} = \infty$
4. Initialize iteration counter k = 1
5. Repeat until $k \leq N_I$:
6. For each individual i = 1 to N_{pop}:
7. Check and implement boundary constraints on pedal marks
8. Evaluate fitness: f(Pi)
9. If $f(P_i) < f_{best}$:
10. Update fbest = f(Pi)
11. Set Pkbest = P(i,:)
12. End if
13. End for
14. *// Pedal Scent Marking Behaviour //*
15. Find best and worst performing groups in the current population.
16. Calculate occurrence factor θ_k
17. For each group I = 1 to N_{pop}:
18. If $0 < \theta k \leq 1/3$:
19. Implement Characteristic Gait update
20. Else if $1/3 < \theta_k \leq 2/3$:
21. Implement Careful Stepping update
22. Else if $2/3 < \theta_k \leq 1$:
23. Apply Twisting Feet update
24. End if
25. End for
26. Choose the better-performing group of bears from current and preceding populations.
27. *// Sniffing Behaviour //*
28. For each group m = 1 to Npop:
29. Randomly choose another group n ≠ m

30. Update $P_{m,k}$
31. End for
32. Choose the best group of bears from updated population.
33. Increment iteration counter: k = k + 1
34. End while
35. Output: Final best fitness value f_{best} and corresponding solution P_{best}

4. EXPERIMENTAL RESULTS

The proposed DAE-BBO technique for outlier detection and consistency maintenance has been implemented in Python 3.0 with Google Colab environment. The IoT healthcare security dataset [13] has been used in the experiments. It contains healthcare normal and malicious traffic dataset.

4.2 Classification Results

The performance of the DAE-BBO classifier has been compared with the DRL [1] and VAE [12] classifiers. The classification performance is evaluated in terms of Accuracy and F1-score metrics.

$$\text{Accuracy} = \frac{TN + TP}{FP + FN + TP + TN} \quad (16)$$

$$\text{F1-score} = 2x\frac{precisionXrecall}{precision+ recall} \quad (17)$$

Here,

$$\text{Where, precision} = \frac{TP}{TP + FP}, \quad \text{recall} = \frac{TP}{TP + FN}$$

Table 1 and Figure 2 show the classification results of accuracy and F1-score for these 3 approaches

Techniques	Accuracy	F1-score
DAE-BBO	98.6	97.3
DRL	96.5	95.5
VAE	97.4	96.2

Table 1 Classification Results

Figure 2 Classification Results

As seen from Figure 2, the proposed DAE-BBO technique attains highest accuracy of 98.6% and highest F1-score of 97.3%, when compared to the other two techniques.

4.3 Quality of Service (QoS) metrics by varying the IoT devices

Apart from the classification results, the results of QoS metrics packet delivery ratio (PDR), average residual energy, computational cost and correctness of data are measured by varying the IoT devices from 20 to 100.

Table 2 and Figure 3 show the results of PDR for varying the devices from 20 to 100.

Number of devices	DAE-BBO	VAE	DRL
20	0.9741	0.9549	0.9357
40	0.9635	0.9458	0.9278
60	0.9601	0.9413	0.9215
80	0.9572	0.9317	0.9152
100	0.9534	0.9265	0.9085

Table 2 Results of PDR

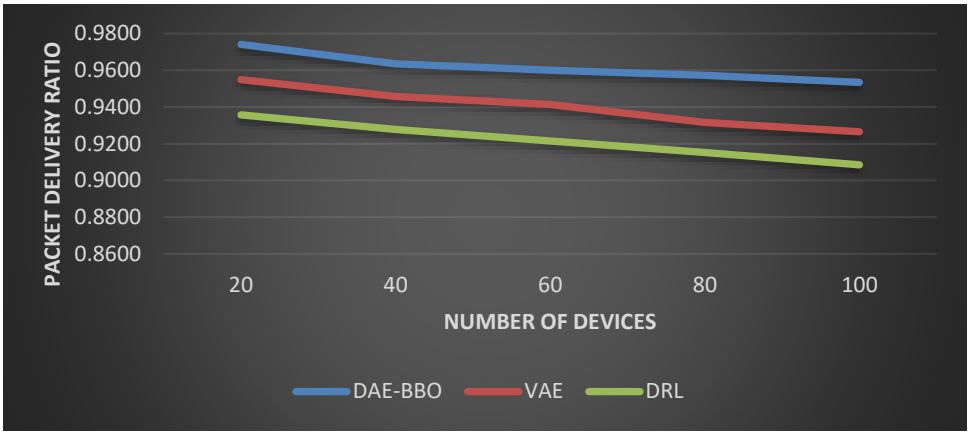

Figure 3 Results of PDR

As it can be seen from the figure, the DA-BBO attains 2.2% higher PDR than VAE and 4.1% higher PDR than DRL.

Table 3 and Figure 4 show the results of average residual energy for varying the devices from 20 to 100.

Number of devices	DAE-BBO (Joules)	VAE (Joules)	DRL (Joules)
20	8.14	7.77	7.28
40	7.76	7.25	6.78
60	7.38	6.58	6.16
80	6.89	6.19	5.85
100	6.52	5.81	5.42

Table 3 Results of Residual energy

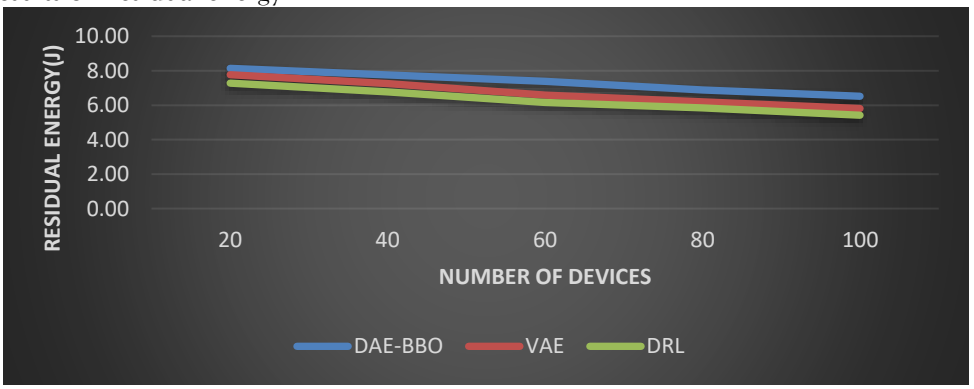

Figure 4 Results of residual energy

As it can be seen from the figure, the DAE-BBO attains 8.6% higher residual energy than VAE and 14.3% higher residual energy than DRL.

Table 4 and Figure 5 show the results of computational cost (in sec) for varying the devices from 20 to 100.

Number of devices	DAE-BBO (sec)	VAE (sec)	DRL (sec)
20	22.5	24.5	26.1
40	24.3	25.7	26.7
60	24.8	26.3	28.2
80	25.5	26.8	28.7
100	26.2	27.6	29.5

Table 4 Results of Computational cost

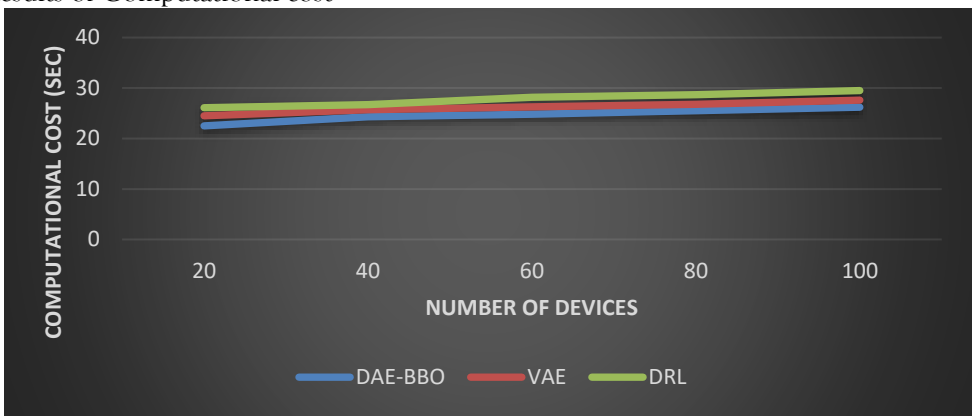

Figure 5 Results of computational cost

As it can be seen from the figure, the DAE-BBO attains 6% lesser cost than VAE and 11% lesser cost than DRL.

Table 5 and Figure 6 show the results of correctness of received data for varying the devices from 20 to 100.

Number of devices	DAE-BBO (%)	VAE (%)	DRL (%)
20	98.4	97.2	95.6
40	97.5	96.4	95.2
60	97.3	95.5	94.7
80	97.1	94.6	94.1
100	96.8	94.1	93.5

Table 5 Results of correctness of data

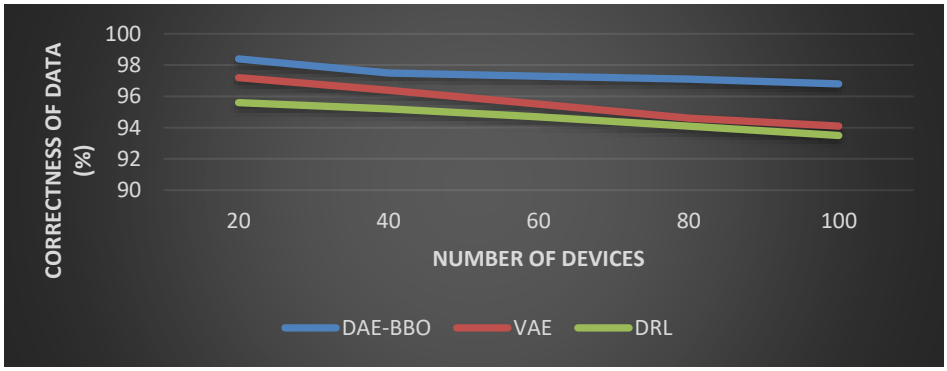

Figure 6 Results of correctness of data

As it can be seen from the figure, the DAE-BBO attains 2% higher correctness than VAE and 3% higher correctness than DRL.

5. CONCLUSION

In this paper, an automatic outlier detection and data consistency maintenance (AOD-DCM) technique using DAE-BBO technique for IoT-WSN has bee proposed. For data outlier detection, the PCA algorithm

is applied. For data inconsistency checking from existing data readings, the DAE model is designed. To fine tune the parameters of DAE, the BBO algorithm is applied. The IoT healthcare security dataset has been used in the experiments. The performance of the DAE-BBO technique has been compared with the DRL and VAE techniques. Experimental results show that DAE-BBO technique achieves higher detection accuracy and correctness of data with reduced computational cost, when compared to the existing outlier detection techniques.

REFERENCES

[1] A. F. Y. Mohammed, S. M. Sultan, J. Lee, and S. Lim, "Deep reinforcement-learning-based IoT sensor data cleaning framework for enhanced data analytics," *Sensors*, vol. 23, no. 4, p. 1791, 2023. [Online]. Available: https://doi.org/10.3390/s23041791
[2] I. Ullah, D. Adhikari, X. Su, F. Palmieri, C. Wu, and C. Choi, "Integration of data science with the intelligent IoT (IIoT): Current challenges and future perspectives," *Digital Communications and Networks*, vol. 11, pp. 280–298, 2025.
[3] S. Elkateb, A. Métwalli, A. Shendy, and A. E. B. Abu-Elanien, "Machine learning and IoT–based predictive maintenance approach for industrial applications," *Alexandria Engineering Journal*, 2024.
[4] M. A. Samara, I. Bennis, A. Abouaissa, and P. Lorenz, "A survey of outlier detection techniques in IoT: Review and classification," *Journal of Sensor and Actuator Networks*, vol. 11, no. 1, p. 4, 2022. [Online]. Available: https://doi.org/10.3390/jsan11010004
[5] Y. H. Reddy, M. H. Srinivas, A. Ali, and A. Z. Sha, "A review on outliers in IoT," *South Asian Research Journal of Engineering and Technology*, vol. 4, no. 6, pp. 134–141, 2022.
[6] G. Chicco, "Data consistency for data-driven smart energy assessment," *Frontiers in Big Data*, vol. 4, p. 683682, 2021. [Online]. Available: https://doi.org/10.3389/fdata.2021.683682
[7] Y. Zhang, S. Sun, D. Zhang, J. Qiu, and Z. Tian, "A consistency-guaranteed approach for Internet of Things software refactoring," *International Journal of Distributed Sensor Networks*, vol. 16, no. 1, 2020. [Online]. Available: https://doi.org/10.1177/1550147720901680
[8] A. Aboshosha, A. Haggag, N. George, and H. A. Hamad, "IoT-based data-driven predictive maintenance relying on fuzzy system and artificial neural networks," *Scientific Reports*, vol. 13, p. 12186, 2023. [Online]. Available: https://doi.org/10.1038/s41598-023-38887-z
[9] M. I. Naas, L. Lemarchand, P. Raipin, and J. Boukhobza, "IoT data replication and consistency management in fog computing," *Journal of Grid Computing*, vol. 19, no. 3, pp. 33, 2021.
[10] S.-H. Sung, S. Hong, H.-R. Choi, D.-M. Park, and S. Kim, "Enhancing fault diagnosis in IoT sensor data through advanced preprocessing techniques," *Electronics*, vol. 13, no. 16, p. 3289, 2024. [Online]. Available: https://doi.org/10.3390/electronics13163289
[11] E. Nikougoftar, "Outlier detection in IoT using trained autoencoder and contrastive loss," *ResearchSquare*, 2024.
[12] W. Gouda, S. Tahir, S. Alanazi, M. Almufareh, and G. Alwakid, "Unsupervised outlier detection in IoT using deep VAE," *Sensors*, vol. 22, no. 17, p. 6617, 2022. [Online]. Available: https://doi.org/10.3390/s22176617
[13] https://www.kaggle.com/datasets/faisalmalik/iot-healthcare-security-dataset