A Graph Correlated Anomaly Detection with Fuzzy Model for Distributed Wireless Sensor Networks

Yasir Abdullah R.[1,2,*], Mary Posonia A.[1], and Barakkath Nisha U.[2]
[1] Sathyabama Institute of Science and Technology, Chennai, India
[2] Sri Krishna College of Engineering and Technology, Coimbatore, India
Email: yasirsince1984@gmail.com (Y.A.R.), maryposonia.cse@sathyabama.ac.in (M.P.A.), ubnisha@gmail.com (B.N.U.)

***Abstract*—Wireless sensor networks have limited power for processing data, storage, and communication. Due to power shortages and anonymous attacks, sensor nodes may produce faulty or anomaly data which affects the accuracy of the entire system. Effective anomaly detection is essential to make an accurate prediction of the result. Moreover, clustering-based anomaly detection reduces energy consumption by avoiding individual sensory data reporting to the base station. The proposed methodology consists of two phases: Correlated graph clustering, and anomaly detection using a Fuzzy model. In the first phase, the spatial correlation of the sensor readings is used to generate a graph, partitioned into clusters. The intra-cluster and inter-cluster temporal correlations are analyzed to refine the optimized cluster structure. Finally, a fuzzy Mamdani model is used to classify the clusters as either normal or anomalous based on their membership values. The proposed approach leverages both spatial and temporal correlation between sensor measurements to form optimized clusters that are more effective for anomaly detection. The Experiments performed on a real-world dataset of WSNs indicate the efficacy of the proposed methodology, which shows significant improvement over traditional anomaly detection methods the electronic file of your paper will be formatted further for final publication.**

***Index Terms*—Anomaly detection, clustering, data aggregation, graph theory, wireless sensor networks**

I. Introduction

Wireless Sensor Networks (WSN) has multivariate sensing nodes to sense, transmit, and process network data. Distributed sensor nodes have self-powered battery devices. These Sensor nodes that constantly sense and analyze data use more energy. Moreover, Network connectivity and computation tasks use 80% of battery power [1, 2]. These sensor nodes collect data in the sensor field via wireless connections, then send the data to a reliable sink node for processing and analysis. There are three primary uses for a WSN in any application setting. In a WSN, it is the responsibility of each deployed sensor node to initially detect and record specific environmental factors. Secondly, it needs a place to permanently store the data it collects and a record of some basic processing steps. As a final step, all sensor nodes must link to both their neighbors and the sink nodes. However, internal and external factors typically affect the sensed and collected data in WSNs [3, 4].

Data generated by a sensor node may contain inconsistencies due to factors like noise, inaccuracy, and missing values, and there may be resource limitations, costs, and battery life. Therefore, the sensor node data in WSNs are unreliable and inconsistent. In a sensor node, an outlier or anomaly is a measurement that does not fit the pattern of previously collected data. Detecting anomalies in WSN sensor nodes placed in the field is crucial here because of the useful data it provides. Therefore, developing a reliable method for finding anomalies in WSNs is essential. Recently, numerous real-time applications have made use of different outlier detection approaches in WSN. For instance, if the features of the data are unknown, then specific rules must be used to identify outliers [5, 6].

Traditional anomaly detection methods are often limited in their ability to handle the large amounts of data generated by WSNs. We propose that Graph-based clustering is a common technique used in wireless sensor networks (WSNs) to organize nodes into groups for efficient data transmission and processing. In this technique, the WSN is modeled as a graph, where nodes are represented as vertices, and edges connect nodes that are within communication range of each other. Clustering is the initial stage of the suggested architecture, and it involves the use of an optimal clustering algorithm to divide datasets into groups based on their similarities and differences. The second step involves identifying and isolating abnormal clusters in the network through the use of an adaptive fuzzy model. Since then, the network's data accuracy and dependability have both improved. Also, by preventing the transmission of fraudulent data in a sensor network, the lifespan of the network is extended.

The proposed approach will allow for the completion of three primary goals:

1) A graph-based clustering is utilized to structure the optimal clusters for constructing initial structures.

Manuscript received April 15, 2023; revised May 4, 2023; accepted May 11, 2023.
*Corresponding author

2) The purpose of intra and inter-data aggregation is to filter out anomalous information before it enters the network.

3) Accurate anomaly detection with fuzzy logic-based decision-making.

The rest of the study is made up of different contexts, while Section II details some of the existing techniques in relation to the proposed methods. Section III explains the suggested methodology's network architecture and problem definition. In Section IV, we outline the proposed Graph-based clustering technique that makes use of fuzzy rule-based systems. Section V explains the performance of the experiments with the results. Section VI concludes the proposed research with an experimental analysis with state-art-techniques.

II. Related Works

Anomaly detection algorithms are divided into several categories, including statistics-based, clustering-based, proximity-based, supervised, unsupervised, reinforcement learning, and deep learning [7, 8]. Therefore, in this section, we summarize a few of the state-art- techniques related to anomaly detection in sensor networks. Clustering-based anomaly identification techniques begin by grouping similar data together, then labeling outliers as anomalies. The LEACH algorithm is a traditional clustering algorithm [9]. In this approach, nodes randomly select themselves as cluster heads and non-cluster heads are merged with the nearest clusters. The primary goal of cluster heads is to aggregate data and transmit accurate, consistent data to the base station.

Graph-based clustering also uses the k-means algorithm [10]. This method graphs the network and clusters nodes into *k* groups based on proximity. The cluster heads then aggregate and transmit data. This approach has certain benefits, such as being straightforward, achieving rapid convergence, and yielding strong results when working with vast amounts of data. However, there are also some drawbacks, such as a proclivity to converge towards a local optimal solution and being highly reliant on the initial selection of cluster centers. Self-Organization Map (SOM) [11] can map data onto a two-dimensional plane, allowing for improved clustering and visualization of results. However, there are some drawbacks to using this method, including its high complexity, which makes it highly dependent on the user's experience.

The CLIQUE algorithm [12] based clustering, examines large datasets and high-dimensional data efficiently. However, one potential disadvantage of this approach is that it may yield suboptimal clustering accuracy, which could ultimately affect the accuracy of the entire algorithm. A correlation technique for anomaly detection was suggested in [13]. The study solely looked at geographical correlation, the temporal correlation at each sensor reading, and attribute correlation with Fuzzy Model.

One method for finding outliers is the relative correlation clustering method [14]. This model is a hybrid of clustering and re-clustering, based on the comparative cascaded correlation method. The hierarchical clustering algorithm [15], is another widely used method that clusters nodes based on their similarities. Initially, each node is treated as a separate group, and the algorithm then iteratively merges groups that share similarities [16]. The output of this process is a hierarchical structure, represented as a tree where each level corresponds to a different degree of clustering.

The authors of [17] proposed a dynamically connected fuzzy system that takes into account temporal, attribute, and geographical correlations in order to identify multivariate outliers in sensor networks. During data consolidation in the cluster's leader, they deployed an outlier identification system. Spectral clustering [18] has emerged as a competitive clustering algorithm. However, it is not highly scalable for processing large modern datasets. Our proposed algorithm overcomes this limitation by combining the concepts of spatial-temporal clustering and spectral clustering techniques. We identify abnormal data as outliers. First, the approach identifies outlier candidates in the eigenspace with the least frequency of appearance of the second smallest eigenvalue. Then, distance-based outlier scores are assigned to each data point to rank the candidate outliers.

Clustering techniques based on minimum spanning trees for discovering clusters of variable shape, size, and density are studied in [19]. However, outlier identification strategies based on minimal spanning tree clustering might be computationally expensive. To alleviate some of these issues, the authors propose a new approach to detect outliers in datasets, inspired by the minimum-spanning tree. This approach effectively identifies local outliers that deviate from the predominant patterns within the dataset. Additionally, the approach is compatible with traditional distance-based outlier detection methods, thus addressing some of the issues associated with those methods. The authors in [20] proposed SOM-based anomaly detection which involves constructing a candidate model using a customized self-organizing network partition and then grouping normal and anomalous data points and then an adaptive fuzzy inference system is applied to separated clusters of CSOM with fuzzy logic rules to identify any anomalies.

In [21], the authors presented a new clustering algorithm called HPFuzzNDA, which automatically adjusts the number of clusters for all classes at each step to improve efficiency. By utilizing cluster modifications, HPFuzzNDA achieved higher classification accuracy and better results in terms of the F-score measure. The authors in [22] focused on using clustering technology in the ODCASC algorithm to predict anomalous node data while incorporating spatial correlation for decision-making. However, this algorithm solely emphasized spatial correlation and did not consider attribute relationships in multivariate data analysis, resulting in lower accuracy compared to our proposed approach.

Graph-based clustering in WSNs can offer several advantages, such as reducing energy consumption and increasing network scalability. However, the performance of the clustering algorithm depends on the network topology, the distribution of sensor nodes, and the

communication protocol used. Therefore, it is carefully designed and evaluated to ensure optimal performance in different network scenarios.

From the literature survey, traditional algorithms perform well with their methodology, but their detection rate and false alarm rate are not satisfactory. However, the proposed method addresses these limitations by improving detection accuracy, reducing complexity, and minimizing energy consumption. To achieve this, the proposed approach utilizes a correlated graph clustering and anomaly detection approach that incorporates a fuzzy model. By leveraging a graph-based clustering algorithm that accounts for spatial and temporal correlations among sensor measurements, the proposed method generates more efficient clusters for detecting anomalies.

III. Network Structure and Problem Statement

WSN clustering models organize nodes into groups called clusters, with each cluster having a leader known as a cluster head (CH) that mediates communications between the members of the cluster and the base station (BS). WSN clustering, as represented by the network model, consists of three distinct but interrelated phases: cluster creation; inter-cluster communication; and data transmission to the BS. The cluster creation phase entails the process of grouping nodes into clusters according to proximity, node density, or some other factor. Node energy, node degree, and node centrality are just a few of the metrics that are used to determine which nodes will serve as cluster heads. During the Inter-cluster communication phase, the cluster leaders interact and share information about their various groups.

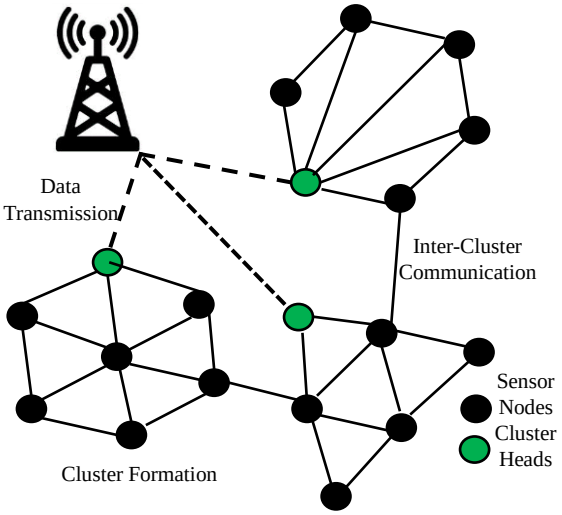


Fig. 1. Network structure.

During the data transmission phase, cluster leaders send the information gathered by their nodes to the network's server. The data is sent back to the base station where it is processed and action is taken. In distributed clustered structures, heterogeneous sensor nodes are considered. During data aggregation, the cluster head performs aggregation and anomaly detection processes hence it has more power than normal sensing nodes

Fig. 1 depicts the infrastructure used by a typical WSN to aggregate data at its Cluster Heads (CHs). Clusters drastically increase the lifespan of a WSN by prohibiting the blind transmission of all the data collected by each sensor node and so reducing energy consumption. Data in each cluster is forwarded and aggregated by the CH. The data aggregation process ensures data integrity by filtering out and deleting any anomalous instances that are received from a cluster of sensor nodes [23]. In the wake of anomaly detection, the collected data are transmitted to the home base for further examination and policymaking.

The proposed model for anomaly detection in wireless sensor networks (WSNs) uses a graph $G(S, D)$ to represent the network structure, where G(S, D)) consists of a set of sensor nodes S and a set of hop connection distance D. The sensor nodes S are defined as $S=\{s_1, s_2, \cdots, s_n\}$ where n is the total number of sensor nodes in the WSN. The hop distances D are defined as $D=\{(s_i, s_j)|s_i, s_j \in S$, and there is a hop connection between s_i and $s_j\}$ where (s_i, s_j) represents the link connecting sensor node s_i to sensor node s_j. The D also emphasize the connections between the sensor nodes in the WSN and can be used to capture the spatial correlation between the sensor measurements. By using this graph-based representation, the proposed model can leverage the spatial correlation between sensor measurements to partition the network into clusters and identify anomalous events. The edges in Graph G can also be used to calculate the minimum path to aggregate data to reach the cluster head, which can improve the efficiency of the anomaly detection process.

IV. Proposed Methodology

Anomaly detection is a critical challenge in various domains such as industrial automation, security monitoring, and medical diagnosis. The ability to identify anomalies rapidly and accurately in large datasets of sensor measurements can help in preventing system failures, detecting potential security breaches, and diagnosing medical conditions. However, conventional anomaly detection methods face difficulty in managing large volumes of data and intricate correlations between sensor measurements, which are typical in these applications. Hence, there is a need for more effective approaches that can handle these challenges and provide accurate detection of anomalies. The proposed approach involves two stages, correlated graph clustering and anomaly detection utilizing a fuzzy model shown in Fig. 2.

In the correlated graph clustering stage, a graph is created from the sensor measurements based on their spatial correlation [24]. The graph is then partitioned into clusters, forming the initial structure of the sensory values. In the temporal correlation analysis stage, the intra- and inter-cluster temporal correlations are analyzed and used to refine the cluster structure. Specifically, the temporal correlations are used to reform the clusters based on the temporal relationships between the sensor measurements.

In the cluster optimization stage, edge connectivity and vertex connectivity are applied in each cluster generated. An Optimized Spanning Tree (MST) technique is applied to determine the shortest path for sending aggregated data

to the cluster head. A set of spatially and temporally optimized clusters is the end result of this procedure being applied to each cluster individually.

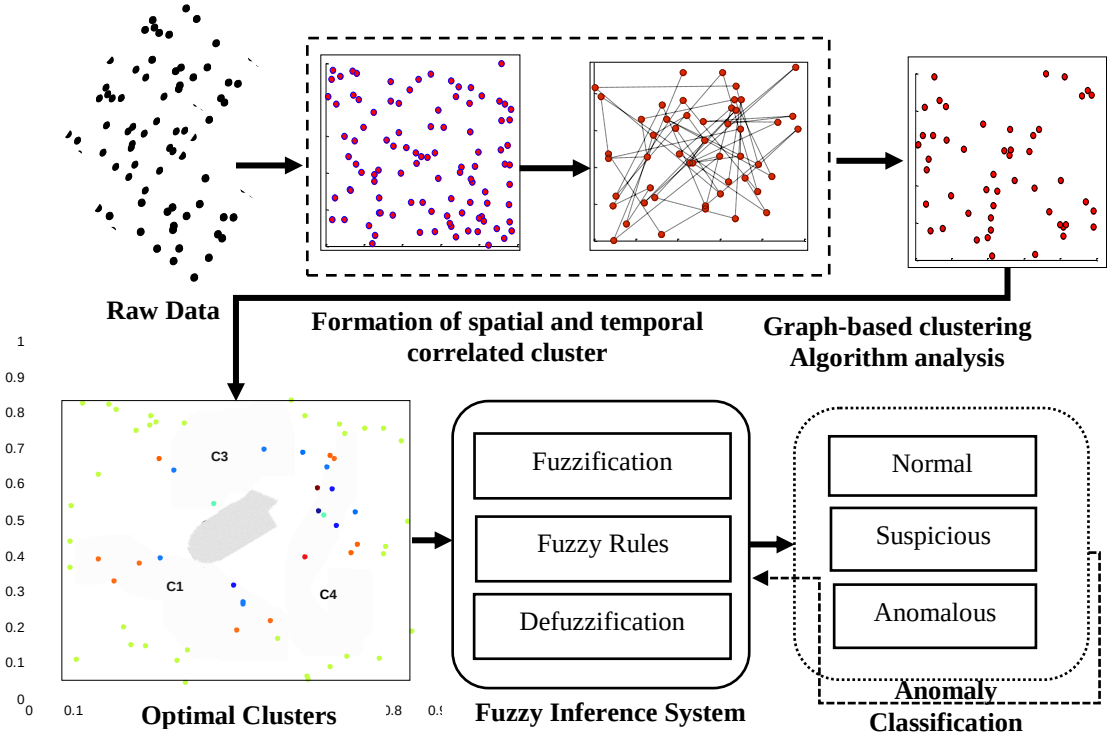


Fig. 2. Schematic diagram of the proposed methodology

In the anomaly detection stage, a fuzzy Mamdani model is used to classify the clusters as either normal or anomalous based on their membership values. The approach calculates membership values based on a set of linguistic rules that capture the spatiotemporal correlation among sensor measurements. The proposed approach utilizes the spatiotemporal correlation between sensor measurements to group enhanced clusters that are more efficient in detecting anomalies. By incorporating an optimized MST and a fuzzy model, the approach is able to further refine the cluster structure and accurately classify anomalous clusters.

A. Correlated Graph Clustering Method

Step 1: Create a graph $G(V, E)$ with n vertices sensor nodes and e edges hop connection [25]. Edges represent single-hop connections among nodes in intra-cluster and multi-hop connections among nodes in inter-clusters.

Step 2: Create the adjacency matrix **A** of G, it is a square grid in which the rows and columns show the vertices and the entries show the edges between the vertices. More specifically, the entry in the i^{th} row and j^{th} column of the matrix is 1 if there is a hop connection between vertices s_i and s_j and 0 otherwise.

Step 2.1: After the adjacency matrix **A**, compute the proximity matrix **P**, which gives the minimum and the maximum edges between each pair of vertices in G. To compute the proximity matrix **P**, distance or similarity is used for the nodes in the graph.

Pairwise adjacency matrix $\mathbf{P}_A$, where $\mathbf{P}_A(i, j) = 1$ if there is a hop connection between i and j, and $\mathbf{P}_A(i, j) = 0$ otherwise. Since G is an undirected graph, $\mathbf{P}_A$ is a symmetric matrix, and $\mathbf{P}_A(i, j) = \mathbf{P}_A(j, i)$.

Step 2.2: The proximity distance matrix $\mathbf{P}_D$ for graph G with n sensor nodes can be represented by an $n \times n$ matrix, where $\mathbf{P}_A(i, j)$ denotes the distance or proximity between two random nodes as shown in Eq. (1):

$$\mathbf{P}_D = \left\lfloor P_{(i,j)} \right\rfloor_{n \times n} \tag{1}$$

The values in the proximity matrix depend on the specific definition of distance or similarity used for the nodes in the cluster. If the similarity between nodes is defined based on some feature vectors associated with the nodes, the proximity matrix is calculated using similarity measures like Euclidean distance. Here we are using the Mahalnobis distance metric for similarity measure. It calculates distance with correlation by considering the correlation matrix instead of the covariance matrix.

Step 3: Spectral sparsification is performed in a sparser subgraph of the original graph G while approximately preserving certain spectral characteristics of the original graph. The eigenvalues and eigenvectors of the graph Laplacian, a matrix that represents the graph's connectivity and structure, are related to the spectral qualities.

Step 3.1: To find the adjacency matrix in spectral sparsification, let $\mathbf{S}_A$ be the $n \times n$ spectral adjacency matrix of G, where $\mathbf{S}_A(i, j)=1$ if there is a hop connection between two vertices and $\mathbf{S}_A(i, j)=0$ otherwise. The degree of vertex i is defined as $d_i = sum_j\ \mathbf{S}_A(i, j)$, and the degree matrix $\mathbf{D}_M$ is a diagonal matrix with $\mathbf{D}_M(i, j)=d_i$.

Step 3.2: The graph Laplacian is symmetric, positive semidefinite matrix **L** of *G* has the formula $\mathbf{L} = \mathbf{D}_M - \mathbf{S}_A$. The graph is highly connected and inferred from **L**'s eigenvalues and eigenvectors.

Step 3.3: To perform spectral sparsification, we aim to find a sparse subgraph of **G** that approximates the Laplacian matrix **L**. Low-rank approximation of **L** is obtained by the maximum values of eigenvalues and eigenvectors of **L**. Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ be the eigenvalues of **L**, and let O_1, O_2, ⋯, O_n be the corresponding orthonormal eigenvectors. The *k*-dimensional subspace spanned by the top *k* eigenvectors is denoted by V_k, and the projection of **L** onto V_k is denoted by $\mathbf{S}_k$.

The matrix $\mathbf{S}_k$ is a low-rank approximation of **L**, and it can be used to construct a sparse subgraph of G by only including the edges that correspond to the non-zero entries of $\mathbf{S}_k$. Specifically, we define a weight function $W(\text{Edge}_i)$ for each edge in *G*. $W(\text{Edge}_i) = \|v_i - v_j\|^2$ if there is an edge between vertices *i* and *j*, and $W(\text{Edge}_i) = 0$ otherwise. The weight function is based on the Mahalanobis distance between the corresponding eigenvectors of the vertices connected by the edge. As a next step, we find the minimum spanning tree of the weighted graph and only use its edges in the subgraph. Depending on factors like the desired approximation error and available processing resources, *k* can have a range of values.

Step 3.4: Use spectral sparsification to create a sparse graph $\bar{G} = (V, \bar{E})$ that approximates the original graph *G*, where $\bar{E}$ is a subset of *E*. Spectral sparsification is a technique for approximating a dense graph with a sparse graph while preserving certain properties. The resulting sparse graph can be used to speed up certain algorithms or reduce memory usage, while still maintaining the important structural properties of the original graph.

Step 4: Find the Shared Nearest Neighbors (SNN) of each vertex in $\bar{G}'$. Let $\bar{G} = (\bar{V}, \bar{E})$ be the sparsified graph obtained in Step 3, where $\bar{V}$ is the set of nearest sensor nodes and $\bar{E}$ is the set of minimal hop connection links. Then, we can define the proximity matrix $\mathbf{P}_D$ for $\bar{G}$ as an *n*'×*n*' matrix, where $P_D(i, j)$ represents the similarity or distance between vertices *i* and *j* in *G'*. This matrix is using the correlation coefficient and Euclidean distance. The SNN of a vertex v_i in $\bar{G}$ as the set of vertices that share at least one neighbor with v_i as shown in Eq. (2)

$$\text{SNN}(v_i) = \left\{ v_i \in \bar{V} \mid (v_i, v_k) \in \bar{E} \cap (v_j, v_k) \in \bar{E} \forall v_k \text{ in } \bar{V} \right\} \quad (2)$$

Using the proximity matrix $\mathbf{P}_D$, SNN is calculated of each vertex in $\bar{G}$ by finding the *k*-nearest neighbors of each vertex based on their similarity or distance. Specifically, we can define the *k*-nearest neighbor graph G_k as the graph obtained by connecting each vertex v_i in $\bar{G}$ to its *k* nearest neighbors in terms of $\mathbf{P}_D$, i.e.

(v_i, v_j) in E_k if $v_j \in$ *k*-nearest neighbors of $v_i \in \boldsymbol{P}_D$

Finally, we can group vertices that have at least *k* shared nearest neighbours into the same cluster, i.e.

$$C_i = \{\text{v}_\text{j} \text{ in } \bar{V} \mid |\text{SNN}(v_i) \cap \text{SNN}(v_j)| \geq k\}$$

The parameter *k* can be tuned to control the size and density of the resulting clusters and can be chosen based on the specific application and the desired trade-off between sensitivity and specificity.

Step 5: The centrality of a vertex and edges in a graph is a measure of its importance, which can be calculated in various ways. Betweenness centrality is calculated by considering the shortest paths between all pairs of nodes in the graph.

Step 5.1: The betweenness centrality of a vertex, SCV(γ), is computed with all-pairs shortest paths that pass through γ as

$$\text{BCV}(\gamma) = \frac{\sum(\alpha \neq \gamma \neq \beta)\sigma(\alpha, \beta \mid \gamma)}{\sigma(\alpha, \beta)} \quad (3)$$

where $\sigma(\alpha, \beta)$ is the count of minimal paths between nodes and $\sigma(\alpha, \beta \mid \gamma)$ is the possible count of shortest paths between nodes α and β that traverse to node γ. Vertices with high betweenness centrality are crucial for maintaining the connectivity of the graph, and their removal can result in the graph becoming disconnected or fragmented.

Step 5.2: The betweenness centrality of an edge link, BCE(δ), is computed based on all paths between pairs of sensor nodes that traverse to an edge link δ, divided by the minimal paths between those pairs of nodes in the graph as

$$\text{BCE}(\delta) = \sum(\alpha, \beta) \in V^2,\ \alpha \neq \beta \left(\frac{\sigma(\alpha, \beta|\delta)}{\sigma(\alpha, \beta)} \right) \quad (4)$$

where *V* is the set of sensor nodes in the graph, $\sigma(\alpha, \beta)$ is the total number of shortest paths from vertex α to vertex β, and $\sigma(\alpha, \beta|\delta)$ is the number of those shortest paths that traverse to an edge δ.

Step 6: Vertex connectivity is the minimal number of vertices that holds the graph as a single component. Edge connectivity is the least number of edges that holds the graph as a single component. To find highly connected components using vertex and edge connectivity, we can use the following steps:

1. Let HCC = {} be an empty set to store the highly connected components.
2. For each vertex *v* in the graph:
   Compute the vertex connectivity of *v*, denoted VC(*v*), as follows:
   i. Let *G'* = (*V'*, *E'*) be the subgraph of *G* by removing *v*.
   ii. Count the number of disconnected components in *G'*, denoted *c*(*G'*).
   If VC(*v*) ≥ *k*, where *k* is the desired threshold: Add *v* to a new set of highly connected vertices, denoted HCV.
3. For each edge *e* in the graph: Compute the edge connectivity of e, denoted EC(*e*), as follows:
   1) Let *G'* = (*V'*, *E'*) be the subgraph of *G* by removing *e*.
   2) Count the number of disconnected components in *G'*, denoted *c*(*G'*).

If EC(e) ≥ k, where k is the desired threshold: Add e to a new set of highly connected edges, denoted HCE.

4. Use HCV and HCE to form highly connected components:
   Let CC = {} be an empty set of components.
   For each vertex v in HCV:
   1) Let HCE(v) = {$e \in$ HCE: v is incident to e} be the set of highly connected edges incident to v.
   2) Let U = {u:(v, u) $\in E \vee (u, v) \in E$} be the HCE edge-incident vertices (v).
   3) Add the component {v} ∪ U to CC.
   For each edge e in HCE:
   1) Let U = {u: (u, e) $\in E \vee (e, u) \in E$} be the vertices of e.
   2) Let HCV(e) = {$v \in$ HCV: e is incident to v} be the set of highly connected vertices incident to e.
   3) Add the component {e} ∪ U ∪ U {$v \in$ HCV(e)} U(v) to CC.
5. Return the set of highly connected components HCC.

Step 7: Chameleon methodology performs agglomerative clustering based on cluster similarities. The algorithm proceeds as follows:

1. Initialize each data point as a cluster, each with a single point $C = \{C_i\}$, where $C_i = \{x_i\}$
2. Compute the similarity matrix S between every pair of clusters $S = \{s_{i,j}\}$, where $s_{i,j}$ = similarity (C_i, C_j)
3. While the number of available clusters is greater than the desired number of candidate clusters k:
   while $|C| > k$:
   1) Determine the clusters C_i and C_j with the highest similarity (i, j) = argmax $s_{i,j}$
   2) Join clusters C_i and C_j into a new cluster $C_m = C_i \cup C_j$
   3) Recompute the similarity between C_m and all other clusters
      for each k != i,j, $s_{k,m}$ = similarity(C_k, C_m)
   a. Update the similarity matrix by replacing the rows and columns of C_i and C_j with the row and column of C_m, respectively, and removing the (i,j) entry
   b. S = {$s_{k,l}$}, where $s_{k,l}$ is the similarity between clusters C_k and C_l, and $C_k = C_i \vee C_j \vee C_m$, and $C_l = C_i \vee C_j \vee C_m$, and $s_{i,j}$ entry is removed
4. Return cluster C.

The final optimal clusters are confirmed and given as input of the fuzzy inference system for anomaly detection. The overall procedure is summarized in the following algorithm.

Algorithm: Correlated Graph Clustering with Fuzzy Anomaly Detection

Input:

n: integer, the vertices (sensor nodes)
e: integer, the edges (hop connections) in the graph
h: integer, the maximum number of hops allowed for the proximity matrix
ε: float, the error tolerance for spectral sparsification
spatialCorrelationData: dictionary, where the keys are pairs of vertices (i, j) and the values are the spatial correlation values between those vertices
temporalCorrelationData: dictionary, where the keys are pairs of vertices (i, j) and the values are the temporal correlation values between those vertices

Output:

normalClusters: listings of normal cluster vertices.
anomalousClusters: listings of anomalous cluster vertices

Begin

1: **CreateGraph**(n, e**)** with n vertices and e edges G(V,E), V denotes the sensor nodes and E denotes the hop connection between the nodes
2: **Create ProximityMatrix(**G, h**)** of the graph G, PM(G) = {(i, j) : min-hop(i, j) ≤ h} where h is a threshold for the maximum number of hops allowed
3: **Perform SpectralSparsification(**G, ε**)** on the graph G, G_s = Sparsify(G, ε)
4: **FindSharedNeighbours(**G_s, spatialCorrelationData, temporalCorrelationData**)** For vertices v_1 and v_2, if they share more than one neighbour, and have both spatial and temporal correlation above a threshold, place them in the same cluster
5: Find the centrality of G,
ComputeVertexBetweenness(G_s**)** and **ComputeEdgeBetweenness(**G_s**)**
CB(G_s) = {(i, b_i) : $i \in$ V, b_i is the betweenness of vertex i}
CE(G_s) = {(e, b_e) : $e \in$ E, b_e is the betweenness of edge e}
6: **MSTAlgorithm(**G_s, sharedNeighbours**)** Generate MST for each cluster
7: **ChameleonAlgorithm(**MST, mahalanobisDistanceThreshold**),** Apply the Chameleon algorithm to reform clusters based on Mahalanobis distance
8: Find highly connected components with vertex and edges
vertex connectivity **FindVertexConnectedClusters(**G_s**)** and
edge connectivity **FindEdgeConnectedClusters(**G_s**)**
9: Generate optimal clusters using vertex connectivity and edge connectivity
10: Apply **Mamdani Fuzzy Inference System** to separate normal and anomalous clusters
a. For Each cluster in clusters

b. If IsNormalCluster(cluster, MFIS) **Then**
normalClusters.Add(cluster)
c. Else
anomalousClusters.Add(cluster)
d. End If
e. Next
End

B. Classification using Fuzzy Inference Systems

Fuzzy logic is a powerful tool for dealing with such systems by providing a way to represent and manipulate linguistic variables and their associated degrees of membership in a fuzzy set. A fuzzy rule base, fuzzifier, inference engine, and defuzzifier comprise the Mamdani FIS. The rule base encodes expert knowledge or data-driven patterns as if-then rules [26]. Membership functions turn crisp inputs into fuzzy sets. The inference engine generates fuzzy outputs from fuzzy inputs using fuzzy rules. Aggregation functions defuzzify fuzzy values.

In anomaly detection, the Mamdani fuzzy model can be used to identify anomalous behavior by modeling the normal behavior of the system and detecting deviations from this behavior. The model can be trained using real-time historical datasets to identify the typical behavior of the system and then used to detect anomalies in real-time data by comparing the current data to the model [27].

Fuzzification: This step transforms the crisp inputs into fuzzy sets. The degree of membership of an input x in a fuzzy set A is given by the membership function $\mu A(x)$.

Rule Evaluation: The fuzzy rules represent the knowledge of an expert or a set of experts. The rule evaluation stage calculates each rule's support based on the inputs' fuzzy set membership in the rule's antecedent. Rule R's support is:

$$W_R = \min(\mu A_1(x_1),\ \mu A_2(x_2),\ \cdots,\ \mu A_n(x_n))$$

where W_R is the weighted rule and $\mu A_i(x_i)$ is the membership function for an input x_i in A_i.

Aggregation: This step aggregates the outputs of all rules to form a single output fuzzy set. This is done using the maximum operator (OR) as follows:

$$\mu C(y) = \max(W_R^*\ \mu C(y|R))$$

where W_R^* is firing strength of the rule and $\mu C(y|R)$ is the degree of membership of output fuzzy set C for the rule R.

The crisp values will be converted into fuzzy linguistic variables by the fuzzy inference engine to construct IF-THEN rules in a predictable way that is in keeping with its underlying principle. Anomaly detection fuzzy if-then rules generated by the proposed approach are presented in Table I.

TABLE I: FUZZY RULES FOR ANOMALY DETECTION

Rules	IF clause				THEN clause
	DC	SC	CV	TT	
1	low	low	low	low	Anomalous
2	low	low	low	medium	Anomalous
3	low	low	low	high	Suspicious
4	low	low	medium	low	Anomalous
5	low	low	medium	medium	Suspicious
6	low	low	medium	high	Suspicious
7	low	low	high	low	Anomalous
8	low	low	high	medium	Suspicious
9	low	low	high	high	Suspicious
10	low	medium	low	low	Anomalous
11	low	medium	low	medium	Suspicious
			.....		
27	low	high	high	high	Normal
28	medium	low	low	low	Normal
29	medium	low	low	medium	Normal
30	medium	low	low	high	Normal
31	medium	low	medium	low	Suspicious
32	medium	low	medium	medium	Suspicious
33	medium	low	medium	high	Suspicious
34	medium	low	high	low	Suspicious
35	medium	low	high	medium	Suspicious
36	medium	low	high	high	Suspicious
37	medium	medium	low	low	Normal
			……		
53	medium	high	high	medium	Anomalous
54	medium	high	high	high	Anomalous
55	high	high	high	high	Normal
56	high	high	high	medium	Normal
57	high	high	high	low	Suspicious
58	high	high	medium	high	Normal
59	high	high	medium	medium	Suspicious
60	high	high	medium	low	Anomalous
61	high	high	low	high	Suspicious
62	high	high	low	medium	Anomalous
63	high	high	low	low	Anomalous
64	high	medium	high	medium	Normal
			……		
80	high	low	low	medium	Anomalous
81	high	low	low	low	Anomalous

TABLE II: SAMPLE INSTANCE OF FUZZY MEMBERSHIP CLASSIFICATION

Membership value instances						
Input matrix				Output vector		
DC	SC	CV	TT	Normal	Suspicious	Anomalous
High	High	Low	High	0.8	0.2	0.0
Low	Low	High	Medium	0.1	0.4	0.5
High	Low	High	Low	0.7	0.1	0.2
Medium	High	Low	High	0.6	0.3	0.1

For instance, the input matrix has four rows, each representing a different set of values for the input variables. The output vector has the same number of rows as the input matrix, with each row representing the corresponding output values based on the input values shown in Table II. The output vector shows the degree of membership of each cluster type (normal, suspicious, and

anomalous) based on the input values shown in Fig. 3 and Fig. 4.

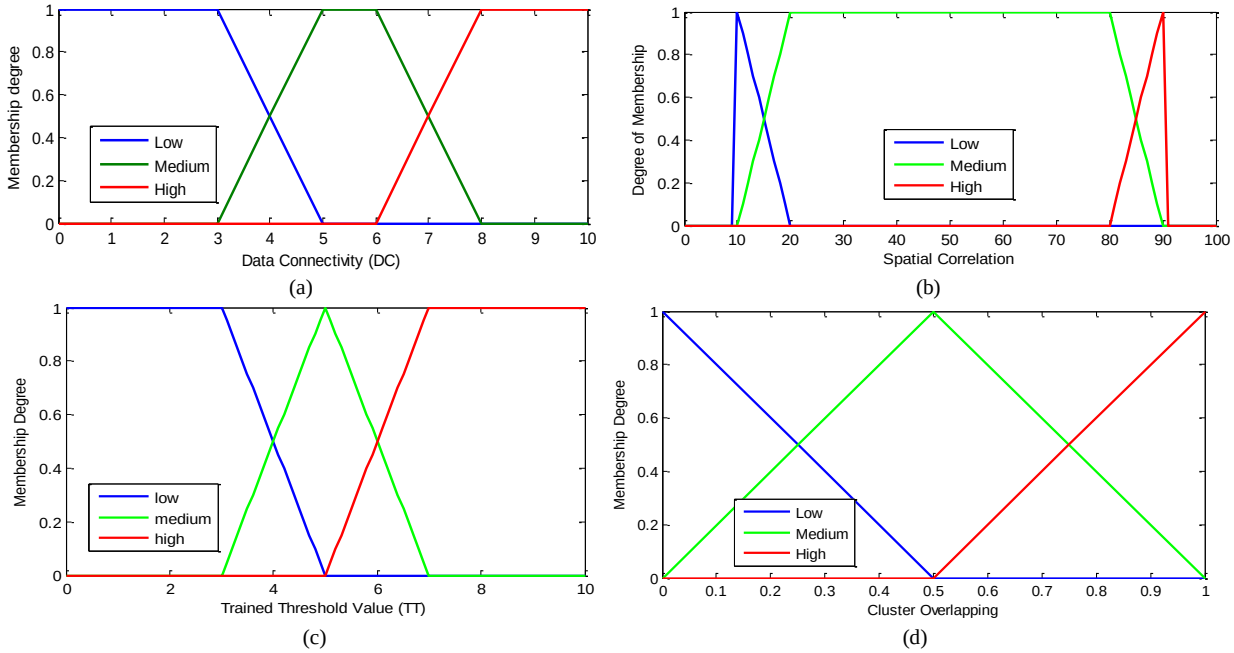


Fig. 3. Input membership functions: (a) Data connectivity, (b) spatial correlation, (c) trained threshold value, and (d) cluster overlapping.

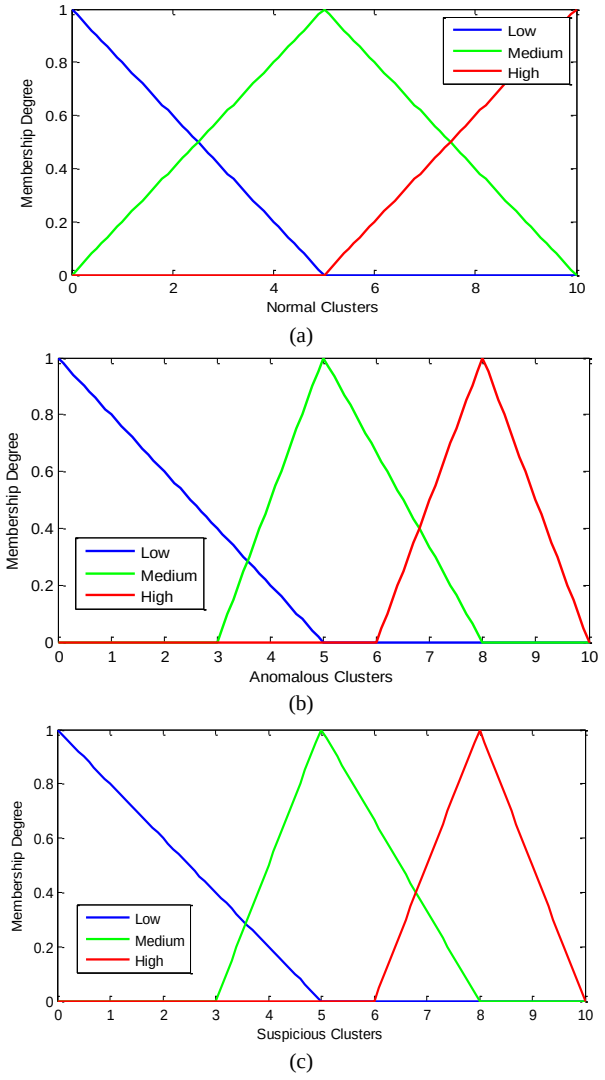


Fig. 4. Output membership function: (a) Normal cluster, (b) anomalous cluster, and (c) suspicious cluster.

V. Results and Discussion

Two real-world datasets are utilized to assess the effectiveness of the suggested methodology. The efficacy of the model is measured by its ability to detect outliers, as well as by its specificity, F1 Score, and false alarm rate. The distinctive inferential relevance of the proposed approach is further confirmed through a comparison with existing methods.

The Intel Berkeley Research Laboratory uses Mica2Dot sensors with weatherboards [28]. Mica2Dot sensors with weatherboards collect time-stamped topological data, humidity, temperature, light, and voltage every 31 seconds The ISSNIP dataset contains standard WSN motes' sensor data. Our experimentation uses two clusters of four sensor nodes installed indoors and outdoors, with one node as the cluster head. Temperature and humidity were measured at 5-second intervals for six hours. Random probability-based corruption produced conflicting data [29].

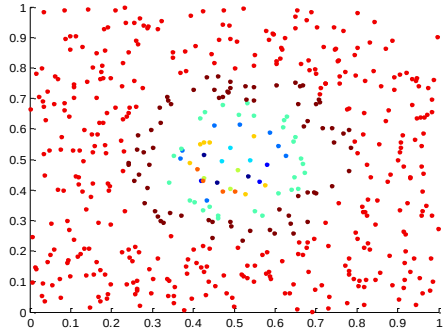
Our proposed system exhibited good performance in all performance metrics when evaluated on a dataset comprising 785,800 samples from IBRL and 5,210 samples from the ISSNIP dataset. This performance was maintained even as the dataset size increased.

Performance metrics such as accuracy, False Alarm Rate (FAR), precision, sensitivity, specificity, and F1 score are often used. In this situation, the performance of an anomaly detection model can be described by the terms True Positive, True Negative, False Positive, and False Negative. True Positive (TP) means that the model correctly finds an outlier in the data that is actually there. True Negative (TN) means that the model correctly says that a data point that is not abnormal is normal. False Positive (FP) is when the model mistakes a normal data point for an anomaly. False Negative (FN) is when the model misses an anomaly in the data that is actually there.

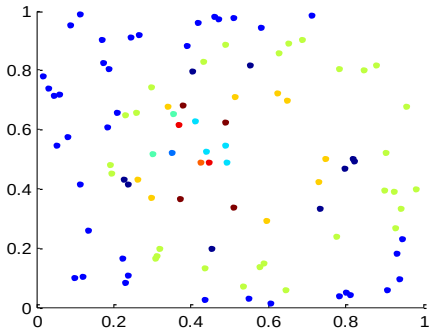
The following equations are used to figure out the performance evaluation metrics.

1: Accuracy = (TP + TN) / (TP + FP + TN + FN)
2: FAR= FP / (FP + TN)
3: Precision = TP / (TP + FP)
4: Sensitivity or recall = TP / (TP + FN)
5: Specificity = TN / (FP + TN)
6: F1 score = 2×Precision×Sensitivity/(Precision + Sensitivity)

Fig. 5 depict the spectral cluster nodes of the dataset before grouping them into clusters. During the training phase, two real dataset correlations are analyzed and thresholds are fixed for the testing phase. The connectivity and spatial-temporal correlations are illustrated in Fig. 6. For improving the efficacy of the proposed system, synthetic datasets are created and random anomalous data are inserted.

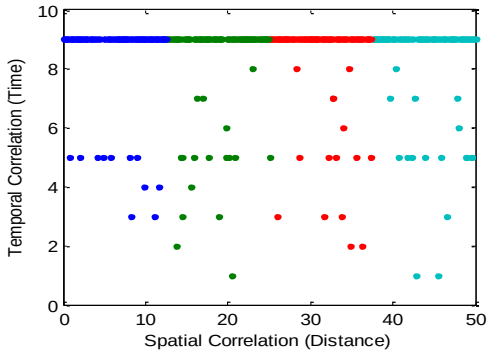


(a)

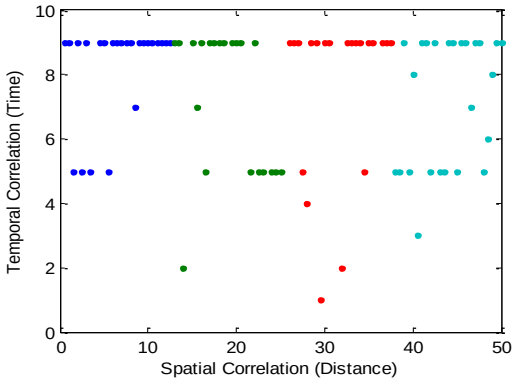


(b)

Fig. 5. (a) Spectral clusters for anomaly detection (IBRL). (b) Spectral clusters for anomaly detection (ISSNIP).



(a)



(b)

Fig. 6. (a) Spatial and temporal correlation of clusters (IBRL). (b) Spatial and Temporal correlation of clusters (ISSNIP).

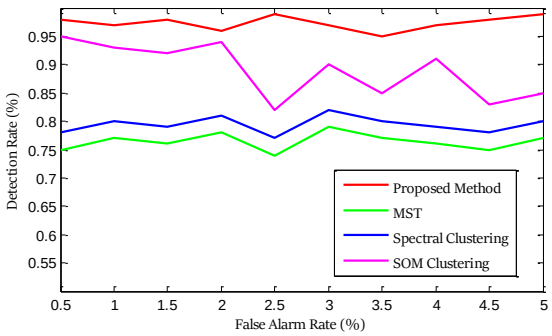


Fig. 7. Detection rate versus false alarm rate.

Fig. 7 shows a comparison of the performance of the proposed method with the existing based on the anomaly detection ratio. It shows that the proposed method works well than the method that is already being used.

TABLE III: OVERALL PERFORMANCE MEASURES OF PROPOSED METHODOLOGY

Metrics	IBRL: Anomalous Data Corruption Level (%)					ISSNIP: Anomalous Data Corruption Level (%)				
	5%	**4%**	**3%**	**2%**	**1%**	**5%**	**4%**	**3%**	**2%**	**1%**
Sensitivity	0.98	0.97	0.96	0.98	0.97	0.97	0.97	0.96	0.98	0.97
Specificity	0.99	0.99	0.99	0.99	0.99	0.98	0.99	0.98	0.99	0.99
Precision	0.96	0.98	0.96	0.97	0.97	0.97	0.98	0.96	0.97	0.98
FAR	0.04	0.03	0.04	0.02	0.06	0.06	0.03	0.04	0.04	0.07
Accuracy	0.99	0.99	0.98	0.99	0.99	0.96	0.99	0.97	0.99	0.99
F1 Score	0.97	0.97	0.96	0.97	0.97	0.97	0.98	0.95	0.96	0.96

TABLE IV: COMPARISON OF PROPOSED METHODOLOGY WITH EXISTING METHODOLOGY

Models	IBRL Dataset		ISSNIP Dataset		Computation complexity
	Detection accuracy	F1 Score	Detection accuracy	F1 Score	
Proposed method	99.43%	96%	98.65%	96%	$O(n^2\log n + \varepsilon^{-2} + kn^2\log n) + O(c)$
SOM clustering [19]	96.67%	87%	94.86%	81%	$O(kni)+ O(c)$
Spectral clustering [17]	94.34%	88%	92.87%	91%	$O(n^3 + kni)+ O(c)$
MST [18]	84.32%	82%	82.56%	79%	$O(n\log n) + O(c)$

The proposed method has the highest performance, followed by SOM clustering [19], spectral clustering [17], and MST [18]. Fig. 10 shows the proposed method efficacy and existing methodologies by varying anomalous data corruption levels. The Anomalous data corruption rate ranges from 10% to 50%, while the y-axis represents the detection rate and ranges from 0.8 to 1.0 (i.e., 80% to 100%). The legend shows which line represents each method, and a grid is added to the plot for better visualization.

Table III indicates the results of each performance metric for the proposed Methodology. This indicates the model's capability to detect abnormal instances with minimum error and maximum accuracy.

The performance comparison of the proposed methodology with IBRL and ISSNIP datasets by representing values of accuracy, F1 Score, and computational complexity is shown in Table IV. The overall computational complexity of the proposed methodology can be expressed as:

$$O(n^2 \log n + \varepsilon^{-2} + kn^2 \log n) + O(c)$$

where n is the number of sensor nodes, k is the number of cluster groups, e is the number of connection links, h is the maximum number of hops allowed, ε is the error tolerance for spectral sparsification and c is the final group of anomalous clusters. The computational complexity of the spectral clustering algorithm for complex data is $O(n^3 + kni) + O(c)$ with i iterations. The complexity comes from the computation of the similarity matrix. The computational complexity of the SOM clustering algorithm depends on the size of the input data and the neurons training in the network. For the dataset with n input data feature vectors and a network of m neurons, the computational complexity is $O(ikn) + O(c)$ with i iterations. The computational complexity of MST is $O(n^2)$ for a dense graph. However, the complexity can be reduced to $O(n \log n)$ in sparse graphs using efficient algorithms such as Kruskal's algorithm.

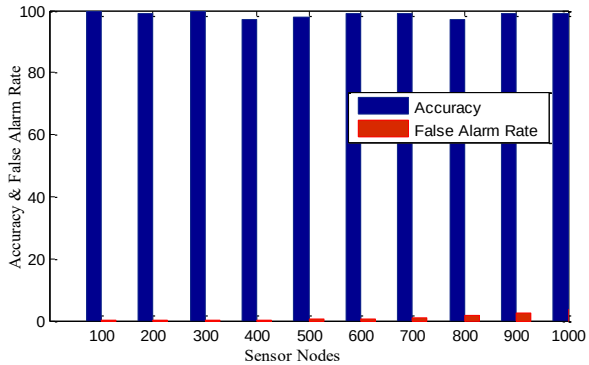


Fig. 8. Scalability of the proposed method.

Fig. 8 depicts the scalability of the proposed method by varying the data samples from 10% to 100% in the number of sensor nodes. It can be observed that the accuracy of the proposed method remains consistently high at around 97.8% even at the maximum size of sensor nodes. However, the false alarm rate of the methodology may slightly increase at the maximum size of sensor nodes. Overall, the proposed methodology exhibits good accuracy and a low false alarm rate in both vertical and horizontal scalability measures.

VI. Conclusion

The usage of wireless sensor networks provides unique issues due to the restricted power of sensor nodes, which might lead to inaccurate or anomalous data. Effective anomaly detection is crucial to ensure accurate prediction of results and reduce energy consumption. Clustering-based anomaly detection is a promising approach that can minimize individual sensory data reporting to the base station. An added advantage is to minimize individual sensory data reporting to the base station and reduce energy consumption. The proposed methodology in this study employs a correlated graph clustering phase and an anomaly detection phase using a Fuzzy model to classify clusters as normal or anomalous based on their membership values. By incorporating both spatial and temporal correlation between sensor measurements, this approach optimizes the cluster structure and significantly improves upon traditional anomaly detection methods in real-world WSN datasets. This research contributes to the development of an optimal anomaly detection method that improves data transmission reliability and accuracy. The effectiveness of the proposed approach may depend on the quality of the initial cluster formation and the accuracy of the Fuzzy model used in the anomaly detection phase; this limitation will be considered in the future research.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Yasir Abdullah and Mary Posonia made significant contributions to the research's design, implementation, analysis of the results, and manuscript writing. Additionally, Yasir Abdullah and Barakkath Nisha conducted the research and analyzed the dataset for experimental results. The final version of the manuscript has been approved by all authors.

References

[1] I. F. Akyildiz, Y. Sankarasubramaniam, and E. Cayircil, "Wireless sensor networks," *A Survey of Computer Networks*, vol. 38, no. 4, pp. 393-422, 2002.

[2] A. Ayadi, G. Oussama, O. Abdulfattah *et al*., "Outlier detection approaches for wireless sensor network," *A Survey of Computer Networks*, vol. 129, no. 1, pp. 31-333, Dec. 2017.

[3] A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer Communications*., vol. 30, no. 14-15, pp. 2826-2841, 2007.

[4] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computer Survey*, vol. 41, no. 3, pp. 1-58, 2009.

[5] Y. Zhang, N. Meratnia, and P. Havinga, "Outlier detection techniques for wireless sensor networks: A survey," *IEEE Communication Survey Tutor*, vol. 12, no. 2, pp. 159-170, 2010.

[6] M. Xie, S. Han, B. Tian, and S. Parvin, "Anomaly detection in wireless sensor networks: A survey*," Journal of Network and Computer Applications*., vol. 34, no. 4, pp. 1302-1325, 2001.

[7] S. Rajasegarar C. Leckie, and M. Palaniswami, "Anomaly detection in wireless sensor networks," *IEEE Wireless Communications*., vol. 15, no. 4, pp. 34-40, 2008.
[8] C. C. Aggarwal and P. S. Yu, "Outlier Detection for High Dimensional Data," *Sig: ACM, -mod Record*, vol. 30, no. 2, pp. 37-46. 2001.
[9] S. Varshney and R. Kuma, "Variants of LEACH routing protocol in WSN: A comparative analysis," in *Proc. of 8th Int. Conf. on Cloud Computing, Data Science & Eng*., 2008, pp. 199-204.
[10] M. Wazid and A. K. Das, "An efficient hybrid anomaly detection scheme using K-means clustering for wireless sensor networks," *Wireless Personal Communication*., vol. 90, no. 4, pp. 1971-2000, 2016.
[11] Y. Pan, G. Li, Y. Xu, *et al*., "Abnormal data detection method for environmental sensor networks based on DBSCAN," *Computing Application Software*, vol. 29, no. 11, pp. 69-72, 2012.
[12] D. Duan, Y. Li, R. Li, *et al*., "Incremental K-clique clustering in dynamic social networks," *Artificial Intelligence. Review*, vol. 38, no. 2, pp.129-147, 2012.
[13] B. N. Usman, U. Natarajan, V. Ramalingam, *et al*., "Fuzzy-based flat anomaly diagnosis and relief measures in distributed wireless sensor network," *Int. Journal of Fuzzy Systems*, vol. 19, no. 2, pp. 1528–1545, 2017.
[14] B. N. Usman, U. Natarajan, V. Ramalingam, *et al*., "Robust estimation of incorrect data using relative correlation clustering technique in wireless sensor networks", in *Proc. of IEEE Int. Conf. on Communication and Network Technologies*, 2014, pp. 314-318.
[15] L. Xu, Y. Yehb, Y. Lee, *et al*., "A hierarchical framework using approximated local outlier factor for efficient anomaly detection," *Procedia Computer Science*, vol. 19, pp. 1174-1181, 2013.
[16] R. Y. Abdullah, A. M. Posonia, and U. Barakkath, "An adaptive mountain clustering-based anomaly detection for distributed wireless sensor networks", in *Proc. of Int. Conf. on Communication, Control and Information Sciences*, 2021, pp. 1-6.
[17] B. N. Usman, U. Natarajan, V. Ramalingam, *et al*., "Improving data accuracy using proactive correlated fuzzy system in wireless sensor networks," *KSII Trans. on Internet Information Systems*, vol. 9, no. 9, pp. 3515-3537, 2015.
[18] X. Wang, X. Wang, and M. Wilkes, "A *k*-Nearest Neighbour spectral clustering-based outlier detection technique," in *New Developments in Unsupervised Outlier Detection: Algorithms and Applications*, Singapore: Springer, 2021.
[19] X. Wang, X. Wang, and M. Wilkes, "A minimum spanning tree clustering-inspired outlier detection technique," in *New Developments in Unsupervised Outlier Detection*, Singapore: Springer, 2020.
[20] R. Y. Abdulla, A. M. Posonia, and U. Barakkath, "An enhanced anomaly forecasting in distributed wireless sensor network using fuzzy model," *Int. Journal of Fuzzy Systems*, vol. 24, no. 7, pp. 3327-3347, 2022.
[21] S. Akhmedova, V. Stanovov, and Y. Kamiya, "Hybrid clustering approach based on fuzzy logic and evolutionary computation for anomaly detection," *Algorithms*, vol. 15, no. 10, #342, 2022.
[22] M. Li and A. Sharma, "Abnormal data detection in sensor networks based on DNN algorithm and cluster analysis," *Journal of Sensors*, vol. 2022, #1718436, 2022.
[23] N. Deo, *Graph Theory with Applications to Engineering and Computer Science*. Mineola, New York: Dover Publications, 1936.
[24] C. C. Noble and D. J. Cook, "Graph-based anomaly detection," in *Proc. Ninth ACM SIGKDD, KDD'03*, 2003, pp. 631-636.
[25] R. Balakrishnan and K. Ranganathan, *A Textbook of Graph Theory*, 2nd ed. New York: Springer, 2020.
[26] B. Shanmugam and N. B. Idris, "Improved intrusion detection system using fuzzy logic for detecting anomaly and misuse type of attacks," in *Proc. Int. Conf. of Soft Computing and Pattern Recognition*, 2009, pp. 212-217.
[27] J. S. R. Jang, C. T. Sun, and E. Mizutani, *Neuro-fuzzy and Soft* Computing, Academic Press, 1997.
[28] IBRL Dataset. [Online]. Available: http://db.csail.mit.edu/labdata/labdata.html, Retrieved on June 13, 2022.
[29] ISSNIP Dataset: [Online]. Available: https://home.uncg.edu/cmp/downloads/lwsndr.html, Retrieved on September 5, 2022.

R. Yasir Abdullah received his bachelor of engineering in electronics & communication engineering from Anna University, Chennai. He got his master of engineering in computer science with distinction from Anna University, Coimbatore, Tamilnadu, India in 2009. He is pursuing Ph.D. degree in the domain of data quality in sensor networks at Sathyabama Institute of Science & Technology, Chennai. Currently, he is working as an assistant professor at Sri Krishna College of Engineering & Technology, Coimbatore, India. His research interests lie in wireless networks, information security, computer networks, sensor networks, etc. He has a total of sixteen years of teaching experience which includes 11 years of research experience. He has published various papers in SCI and Scopus indexed journals which include 22 papers in international journals, and 5 papers in national journals, and presented 23 papers at international conferences and 10 papers at national conferences. His research interests include sensor networks, data mining, data analytics, neural networks, soft computing, network security, and networks. He has published several book chapters in Scopus indexed Book series.

Mary Posonia. A completed her B.E. degree in 2002 from Manonmaniam Sundaranar University, M.E. degree in 2004 from Sathyabama University, and Ph.D. deegree in 2018 from Sathyabama Institute of Science and Technology. She has got 20 years of teaching experience and presently she is working at Sathyabama Institute of Science and Technology as an associate professor in the Department of Computer Science and Engineering. Her area of interest includes machine learning, data mining, and networking, and got more than 40 publications in Scopus and web of science journals.

Barakkath Nisha Usman is a senior member of IEEE, she had contributed technically to various events conducted by IEEE. She completed her Ph.D. degree in the area of wireless sensor networks from Anna University, Chennai in March 2017. She received her bachelor of engineering in computer science engineering in 2008 with distinction from Anna University, Chennai. She got her master of engineering in Computer Science Engineering from Anna University, Chennai. She received Gold Medal for securing First Rank in her post-graduation. She was working as associate professor in well-established engineering colleges for the past fourteen years. Currently, she is working as an associate professor in the Department of Information Technology at Sri Krishna College of Engineering & Technology (Autonomous), Coimbatore. Her area of Interest includes Ad hoc networks, wireless sensor networks, data mining, and natural language processing, etc. she has published various papers in reputed SCI, SCOPUS indexed journals and she has presented papers in various national and international conferences. She is an active member of various professional societies. She reviewed more than 25 papers as a reviewer for international journals and conferences.