Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SCALABILITY AND EFFICIENCY OF CLUSTERING ALGORITHMS FOR LARGE-SCALE IoT DATA: A COMPARATIVE ANALYSIS

PRABHAT DAS[1,2], KARTHIK KOVURI[3], SAJAL SAHA[4]

[1]PhD Scholar, Department of Information Technology, The Assam Kaziranga University, India

[2]Faculty, Department of Computer Science and Engineering, Adamas University, India

[3]Professor, Department of Computer Science and Engineering, The Assam Kaziranga University, India

[4]Professor, Department of Computer Science and Engineering, Adamas University, India

E-mail: [1,2]prabhatdash0376@gmail.com, [3]karthik@kzu.ac.in, [3]sajal1.saha@adamasuniversity.ac.in

ABSTRACT

This research investigates the scalability and efficiency of clustering algorithms applied to large-scale Internet of Things (IoT) data. A comprehensive evaluation is conducted on fourteen clustering algorithms—Affinity Propagation, Agglomerative, BIRCH, Bisecting K-Means, DBSCAN, Fuzzy C-Means, Gaussian Mixtures, HDBSCAN, K-Means, Mean-Shift, OPTICS, Overlapping K-Means, Spectral Clustering, and Ward-Hierarchical—across datasets ranging from 40,000 to 100,000 sensor readings. The study systematically analyzes execution time and clustering performance to determine their suitability for large-scale IoT applications. Results indicate that K-Means, Ward-Hierarchical, and BIRCH exhibit strong scalability and computational efficiency, whereas Affinity Propagation and Spectral Clustering face significant challenges with increasing dataset size. These findings provide valuable guidance for selecting optimal clustering techniques in IoT-based data analytics, considering factors such as computational constraints, dataset characteristics, and clustering granularity.

Keywords: *Clustering Algorithms, IoT Data Clustering, Comparative Analysis, Sensor Data Analysis, Bibliometric Analysis, Machine Learning in IoT, Multi-Dimensional Data Clustering*

1. INTRODUCTION

Clustering of IoT data represents a form of unsupervised classification widely studied in the fields of data mining and machine learning due to its relevance for summarization, learning, segmentation, and market partitioning [1]. This technique involves grouping items or data into clusters where elements within each group share similar traits, while data between groups exhibit distinct differences [2]. Clustering plays a crucial role by identifying inherent groupings within unlabeled data, which is vital for uncovering hidden patterns and improving system efficiency. Given the critical and sensitive nature of IoT data, precise handling is imperative. IoT data is characterized by the "5Vs": Volume, Velocity, Veracity, Variety, and Value, making it complex to process and analyze. Defined as large volumes of data generated in real time, IoT data requires processing to extract valuable insights that support informed decision-making.

The Figure 1 represented below visually demonstrates how IoT application features integrate within connected systems, highlighting the importance of clustering in organizing and optimizing such environments. The illustration underscores the role of clustering in enhancing decision-making and improving the performance of IoT systems within interconnected networks.

In today's generation a huge chunk of data is being generated by people, things and via their technology interactions. IoT data being a multi variated data also come with multiple variable and constraints which needs to be processed further. Although the information obtained from various sources is beneficial to individuals and businesses, data management and analysis is a time-consuming process. As a result, IoT data still has several flaws in terms of data management [4]. Different scholars have offered a number of solutions to these problems. However, clustering of IoT data is the most successful technique to date. Finding homogenous groupings of data items is the basic objective of the clustering activity. Many academics in the field of IoT have given many clustering algorithms, but the primary challenge is that the

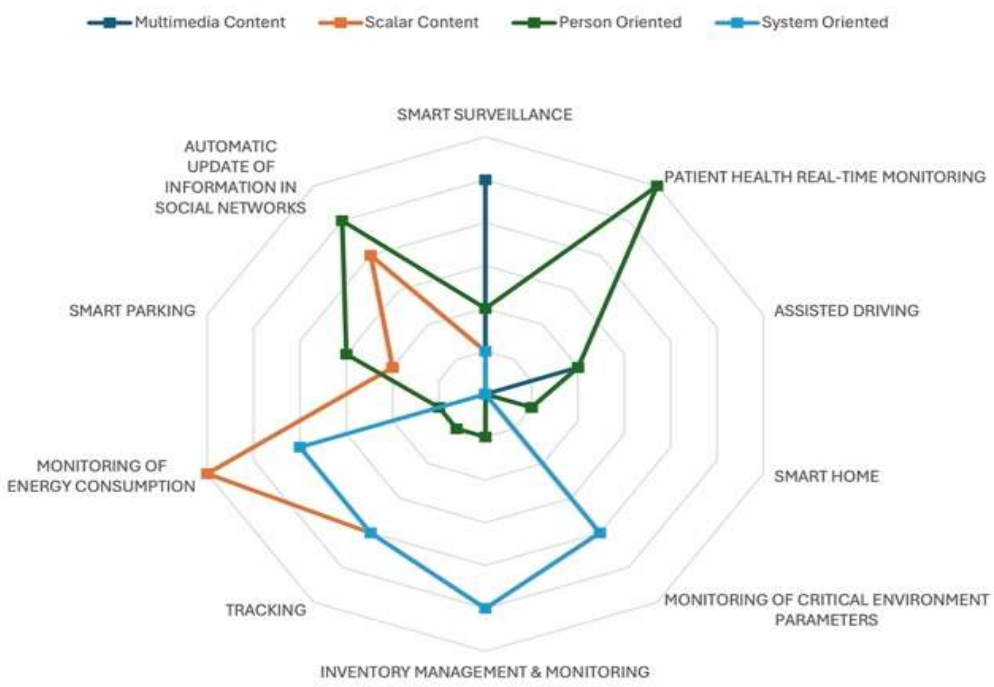

Figure 1: Integration and Impact of IoT Application Features in Connected Ecosystems [3].

kind and capacity of the data are unclear. For this reason, it is crucial to plan and develop a powerful algorithm to manage this IoT data.

The following bases can be used to categorize clustering algorithms: Overlapping-based, Hierarchical-based, Density-based, Fuzzy based, and Distribution-based. This study's aim is to present a comprehensive analysis and performance-based comparison of several large data clustering techniques. This will assist researchers in choosing the finest clustering algorithm for a certain circumstance and assist them in designing an effective clustering algorithm by taking into account the benefits and drawbacks of each method.

The significance of clustering in the field of the Internet of Things (IoT) cannot be overstated. Clustering is particularly valuable in IoT for several reasons: it enables the efficient processing and analysis of large datasets, facilitates anomaly detection and predictive maintenance, enhances data security by identifying unusual patterns, and improves decision-making by providing actionable insights [5]. Moreover, clustering can significantly reduce the complexity and dimensionality of IoT data, making it more manageable and interpretable. As such, the deployment of clustering algorithms stands as a cornerstone in harnessing the full potential of IoT, driving innovations, and optimizing operations across various industries [6].

The citation network analysis was conducted to explore highly cited publications on Clustering Algorithms in Internet of Things (IoT) research between 2011 to 2024. The bibliometric data has been extracted from the Scopus database, and a visualization has been generated using VOSviewer 1.6.19 software. Out of 2,213 documents, 892 documents met the criterion of having at least 5 citations. Notably, the document authored by Feng Chen et al. [7] emerged as the most influential, with 408 citations, positioned at the bottom right of the visualization. Following closely is the publication authored by Zhihua Cui et al. [8], located at the center, garnering 379 citations.

Another prominent node in the network, positioned at the top right, represents the document authored by Trupti Mayee Behera et al. [9], with 356 citations. These key nodes indicate significant contributions and high impact within the realm of clustering algorithms in IoT research. Additionally, the visualization highlights interconnected research clusters, showcasing evolving trends, influential collaborations, and emerging areas of interest within the field.

Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

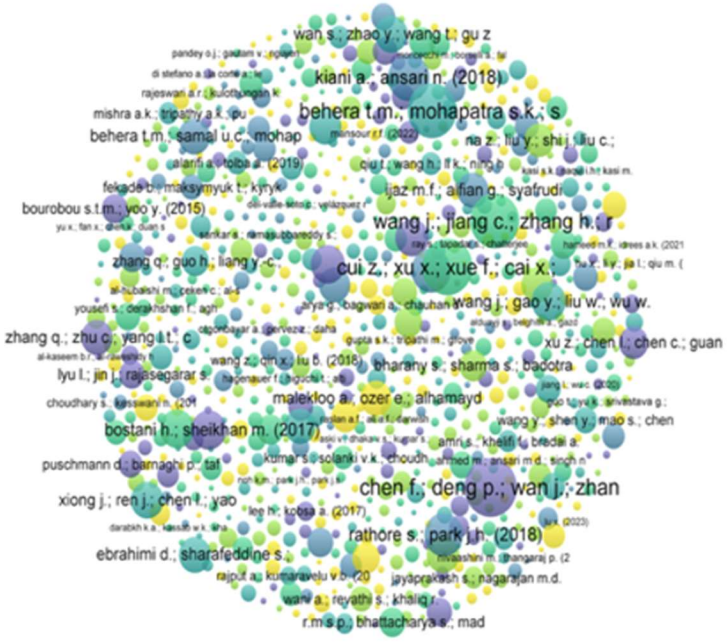

Figure 2: Visualization Map of highly cited publications in application of clustering algorithms in IoT.

2. CLUSTERING OVERVIEW

Before delving into recent works, it is essential to understand the clustering algorithms that form the foundation of this study. Clustering techniques vary in their approach to grouping data, each with distinct computational properties and applicability. Some algorithms prioritize speed and scalability for large datasets, while others focus on uncovering complex structures at a higher computational cost. The choice of method depends on factors like data distribution, noise tolerance, and clustering granularity. A clear understanding of these algorithms is crucial for evaluating their efficiency, scalability, and suitability for large-scale IoT data analysis.

The following section provides a comprehensive discussion on the taxonomy of the 14 clustering algorithms, categorizing them based on their methodologies. By classifying these techniques into partition-based, hierarchical, density-based, and model-based approaches, we establish a structured framework for comparing their strengths,

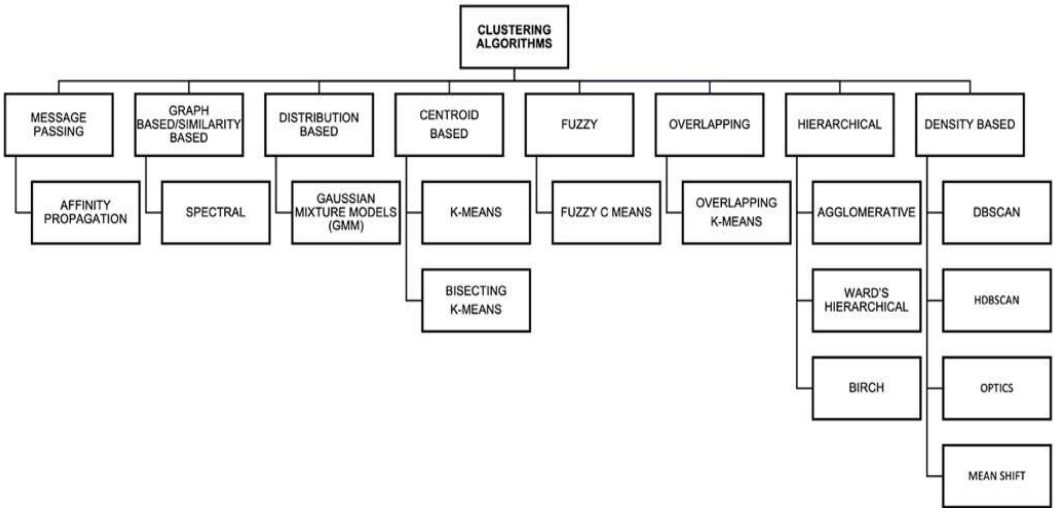

Figure 3: Taxonomy of different algorithms used for clustering [10].

limitations, and applications. This categorization lays the foundation for analyzing their performance, ensuring a systematic approach to selecting the most suitable algorithm for IoT-driven clustering tasks.

2.1 K-Means

Allocates each point to the proximity cluster center and then computing those centers as the mean of the given points, the K-Means algorithm divides data into K clusters. This process is repeated until cluster assignments stabilize. Step by step explanation of K-Means clustering algorithm [11]:

Step 1: Initialize: Randomly select K centroids.

Step 2: Assign: For each point x_i, find the nearest centroid c_j and assign x_i to cluster j.

$$j = \underset{j}{\operatorname{argmin}} \ \| x_i - c_j \|^2$$

Step 3: Update: Recalculate centroids as the mean of all points assigned to that centroid's cluster.

$$c_j = \frac{1}{|S_j|} \sum_{x_i \in S_j} x_i$$

Step 4: Repeat steps 2 and 3 until convergence.

2.2 Affinity Propagation

Affinity Propagation exchanges communications between data points until a group of exemplars (representative examples) appears, utilizing which affinity propagation builds clusters. Each data point selects the exemplar it most closely resembles based on these shared characteristics. Step by step explanation of affinity propagation clustering algorithm [12]:

Step 1: Similarity Computation: Compute similarity $s(i,k)$ between data points.

Step 2: Message Update: Iteratively update "responsibility" $r(i,k)$ and "availability" $a(i,k)$.
Responsibility:

$$r(i,k) \leftarrow s(i,k) - \max_{k' \neq k} \{a(i,k') + s(i,k')\}$$

Availability:

$$a(i,k) \leftarrow \min \left\{ 0, r(k,k) + \sum_{i' \notin \{i,k\}} \max\{0, r(i',k)\} \right\}$$

Step 3: Criterion for Exemplars: Determine exemplars based on the sum of responsibility and availability.

Step 4: Assign: Assign points to their respective exemplars.

2.3 Mean-Shift

The non-parametric Mean Shift Clustering algorithm that finds dense regions of data points by iteratively shifting points towards the mode (the highest density of points) using a sliding window. It automatically discovers the number of clusters based on the data's spatial distribution. Step by step explanation of mean-shift clustering algorithm [13]:

Step 1: Choose Kernel: Select a kernel function $K(x)$ and bandwidth h.

Step 2: For each data point x:

$$m(x) = \frac{\sum_{x_i \in N(x)} K\left(\frac{x - x_i}{h}\right) x_i}{\sum_{x_i \in N(x)} K\left(\frac{x - x_i}{h}\right)}$$

Shift x towards $m(x)$.

Step 3: Cluster Identification: Group points that converge to the same region.

2.4 Spectral Clustering

By performing dimensionality reduction using the eigenvalues of a similarity matrix, spectral clustering is a clustering technique that finds clusters in lower-dimensional space. It's particularly effective for discovering clusters that are not linearly separable. Step by step explanation of spectral clustering algorithm [14]:

Step 1: Construct Similarity Graph: Create affinity matrix A.

Step 2: Graph Laplacian: Compute $L = D - A$, where D is the degree matrix.

Step 3: Eigenvalue Decomposition: Find eigenvectors of L.

Step 3: K-Means on Eigenvectors: Apply K-Means clustering.

Step 4: Assign Clusters: Based on the K-Means result.

2.5 Ward's Hierarchical Clustering

It, groups data points into clusters based on the principle of minimizing the sum of squared differences within all clusters. Step by step explanation of ward's hierarchical clustering algorithm [15]: Step 1: Initialize: Each point as its own cluster.

Step 1: Initialize: Each point as its own cluster.

Step 2: Find Pair: Identify pair of clusters to merge, minimizing increase in total variance.

$$\Delta(\text{Var}) = \text{Var}(C_{\text{merged}}) - \text{Var}(C_i) - \text{Var}(C_j)$$

Step 3: Merge: Combine the pair.

Step 4: Repeat: Until desired number of clusters.

2.6 Agglomerative Clustering

In this algorithm each observation starts within a cluster of its own when employing a bottom-up hierarchical clustering technique called agglomerative clustering. Pairings of clusters are merged as one moves up the hierarchy. Up until every point is combined becomes a solitary cluster or a stopping requirement is satisfied, the procedure iteratively continues. Step by step explanation of agglomerative clustering algorithm [16]:

Step 1: Initialize: Consider each data point as a single cluster.

Step 2: Compute Distance: Use a linkage criterion to calculate the distance between clusters.

Step 3: Merge: Join the two closest clusters.

Step 4: Update Distance: Recalculate the distance matrix for the newly formed cluster.

Step 5: Repeat: Until the desired number of clusters is achieved.

2.7 DBSCAN

It is an algorithm used to recognize clusters of closely located points and to differentiate these from outliers, which are points in areas of low density. This method relies on two main parameters: MinPts, which is the minimum number of points needed to define a cluster, and epsilon (ε), which is the search radius around a point to determine if it's part of a dense region. Step by step explanation of DBSCAN clustering algorithm [17]:

Step 1: Identify Core Points: A point is a core point if it has more than $MinPts$ within a radius ϵ.

Step 2: Expand Clusters: Form a cluster by recursively adding all directly reachable points from core points.

Step 3: Assign Non-Core Points: Label non-core points as border points or noise.

2.8 HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise)

It is a method that expands on DBSCAN by making it a hierarchical clustering technique that doesn't require a preset distance value. (\epsilon). It identifies clusters of varying densities and is robust to noise and outliers. Step by step explanation of HDBSCAN clustering algorithm [18]:

Step 1: Transform Space: Convert the space according to the density/sparsity of the data.

Step 2: Build Minimum Spanning Tree: Create a minimum spanning tree from the weighted graph of the data.

Step 3: Convert Tree to Hierarchy: Derive a hierarchy of connected components.

Step 4: Condense Tree: Prune the hierarchy to find significant clusters.

Step 5: Extract Clusters: Determine clusters from the condensed tree based on stability.

2.9 HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise)

Is an algorithm similar to DBSCAN but instead creates an ordering of points to allow for variable density clustering and extraction of hierarchy and reachability plots, which provide a visualization of cluster structures. Step by step explanation of OPTICS clustering algorithm [19]:

Step 1: Order Points: Arrange points to identify the spatial structure.

Step 2: Core Distance: For a point p, compute the smallest distance within which $MinPts$ are contained.

Step 3: Reachability Distance: Calculate the reachability distance for each point.

Step 4: Build Reachability-Plot: Generate a plot based on the reachability distances.

Step 5: Extract Clusters: Identify clusters based on valleys in the reachability plot.

2.10 Gaussian Mixture Models (GMM)

A probabilistic approach known as Gaussian Mixture Models (GMM) clustering makes the assumption that each data point is the result of a blend of multiple Gaussian distributions with unknown parameters. It finds the greatest probability estimations for the parameters using the Expectation-Maximization (EM) technique, enabling soft-clustering of the data. Detailed description of the GMM clustering algorithm [20]:

Step 1: Initialize Parameters: Choose initial values for the means, variances, and mixing coefficients.

Step 2: Expectation Step (E-Step): Assign each data point a probability of belonging to each cluster.

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j)}$$

Step 3: Maximization Step (M-Step): Update the parameters to maximize the likelihood of the data.

$$\mu_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n$$

Step 4: Repeat: Until the log-likelihood converges.

2.11 BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)

Incrementally constructs a tree structure (CF Tree) that captures the essence of the data. It then applies a global clustering algorithm to the leaf nodes of this tree, efficiently handling noise and discovering clusters with varying shapes and sizes. Step by step explanation of BIRCH clustering algorithm [21]:

Step 1: Build CF Tree: Sequentially insert data points into a CF Tree, adhering to the threshold and branching factor limits.

Step 2: Condense Tree: Optionally, condense the tree to improve the quality of clustering.

Step 3: Global Clustering: Apply a global clustering algorithm on the leaf entries.

Step 4: Refine Clustering: Optionally, refine the clustering by reassigning points or adjusting the tree.

2.12 Bisecting K-Means

Bisecting K-Means is a variant of the K-Means clustering algorithm that iteratively splits clusters with the conventional K-Means technique, into two. With K=2, selecting the best split until the required number of clusters is obtained at each stage. It combines elements of hierarchical clustering with the efficiency of K-Means. Step by step explanation of bisecting k-means clustering algorithm [22]:

Step 1: Select a Cluster: Choose a cluster to bisect.

Step 2: Bisect: Apply K-Means with $K = 2$ on the selected cluster.

Step 3: Choose Next: Select the next cluster to bisect, based on some criterion.

Step 4: Repeat: Continue until the desired number of clusters is formed.

2.13 Fuzzy C Means:

It is a clustering technique that applies fuzzy sets to cluster assignment, which sets it apart from K-Means and permits data points to belong to several clusters with different levels of membership. By using this method, the objective function that calculates the distance between a data point and a cluster center, weighted by the point's membership degree, is minimized. Step by step explanation of fuzzy C means clustering algorithm [23]:

Step 1: Initialize Membership Matrix U: Randomly assign membership levels for each data point to each cluster.

Step 2: Calculate Centroids: Update the cluster centers based on the membership degrees.

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^m x_i}{\sum_{i=1}^{N} u_{ij}^m}$$

Step 3: Update Membership U: Recalculate the membership of each data point to each cluster.

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\| x_i - c_j \|}{\| x_i - c_k \|} \right)^{\frac{2}{m-1}}}$$

Step 4: Repeat: Until the membership matrix stabilizes.

2.14 Overlapping k-Means:

An improvement on the standard K-Means approach, overlapping K-Means clustering enables the identification of clusters in which a single data point can be a member of many groups, rather than being assigned to just one. This method is particularly useful in scenarios where data naturally belongs to overlapping groups or categories. Step by step explanation of overlapping k-means clustering algorithm [24]:

Step 1: Initialize: Allow data points to have membership in multiple clusters.

Step 2: Assign Membership: Update the membership of each data point based on its distance to each cluster center.

Step 3: Update Centers: Recalculate the cluster centers considering the degrees of membership.

Step 4: Repeat: Iteratively adjust memberships and centers until convergence.

Table 1: Different Parameters used by the clustering algorithms [11]-[24].

Parameter	Description	Applicable Algorithms
Number of Clusters (K or Equivalent)	Many clustering algorithms require specifying the cluster numbers that needs to be identified.	K-Means, Spectral Clustering, Gaussian Mixture Models, BIRCH, Bisecting K-Means, Fuzzy C Means, Overlapping K-Means
Distance Metric	The choice of distance metric can significantly affect cluster formation, especially in algorithms relying on the notion of distance for membership or centroid computation.	K-Means, Agglomerative, DBSCAN, HDBSCAN, OPTICS, Fuzzy C Means
Density Parameters	Parameters defining density criteria to identify core points, reachable points, and noise, including the minimal number of points within a specified radius (ϵ).	DBSCAN, HDBSCAN, OPTICS
Similarity or Affinity Measure	Influences how clusters are formed according to the resemblance (or affinity) between data points.	Affinity Propagation, Spectral Clustering
Kernel Function and Bandwidth	The choice of kernel function and its bandwidth defines the weighting of points within a region to find directions of maximum density.	Mean-Shift
Linkage Criterion	Determines how the distance between clusters is calculated, influencing which clusters are merged as the algorithm progresses.	Agglomerative Clustering, Ward's Hierarchical Clustering
Convergence Criterion or Iteration Limit	Most algorithms repeat their operations until a maximum number of iterations is achieved or a convergence requirement is satisfied.	Nearly all algorithms
Cluster Stability or Quality Measures	Parameters or internal measures that help determine the "best" number of clusters or assess the quality/stability of identified clusters.	HDBSCAN, BIRCH, Gaussian Mixture Models

3. LITERATURE REVIEW

The efficiency of clustering algorithms largely depends on their scalability and execution time, particularly when managing large, high-velocity datasets like those generated by IoT devices. Partition-based methods, such as K-Means and Bisecting K-Means, are widely favored for their simplicity and linear time complexity, making them suitable for high-dimensional data [25][26]. However, these methods typically assume spherical cluster shapes and require predefining the number of clusters, limitations that hinder their adaptability to dynamic IoT environments.

BIRCH, originally designed for very large databases, offers memory-efficient incremental clustering [27]. Yet, despite its promise, it has seen limited empirical validation on massive, heterogeneous IoT datasets. Hierarchical clustering techniques, including Agglomerative Clustering and Ward's method, often suffer from high computational complexity ($O(n^2)$ or worse), making them impractical for real-time applications involving sensor networks [28].

Density-based approaches such as DBSCAN, OPTICS, and HDBSCAN are effective for identifying arbitrarily shaped clusters and noise [17][19]. Nevertheless, as dataset size increases, these algorithms become computationally expensive. Moreover, while OPTICS mitigates DBSCAN's fixed epsilon limitation, its applications have largely been confined to geospatial and biological datasets, without sufficient exploration in large-scale IoT contexts [29].

Other advanced methods like Affinity Propagation and Spectral Clustering can produce high-quality results but impose excessive computational demands, limiting their scalability for IoT systems [30]. Despite their strengths, these algorithms have not been rigorously tested under the continuous data streams and real-time demands characteristic of IoT sensor networks.

Many prior studies have investigated clustering techniques across different domains; however, the majority focus on traditional, structured datasets rather than on vast, dynamically evolving IoT data. For instance, Nhat et al. [31] evaluated BIRCH for clustering financial transactions, demonstrating efficient handling of streaming data but without addressing IoT-specific challenges such as temporal irregularity and sensor heterogeneity. Marella et al. [32] applied OPTICS for anomaly detection in network traffic but did not analyze performance at IoT scale.

Similarly, Wang et al. [33] proposed a two-phase GIS-based HDBSCAN clustering approach for traffic analysis, while Yadav et al. [34] surveyed clustering algorithms without concentrating on execution time or scalability for IoT. Studies like those by Sheng et al. [35] and Chakraborty et al. [36] highlight the effectiveness of K-Means and Affinity Propagation in healthcare and image processing respectively, yet neglect to evaluate their scalability under continuous, high-volume IoT conditions.

Across these prior works, a critical shortcoming persists: the absence of a comprehensive, empirical comparison of clustering algorithms designed specifically for large-scale, real-time IoT datasets. To the best of our knowledge, no existing study systematically benchmarks a wide range of clustering algorithms while analyzing both execution time and scalability in the context of dynamic IoT environments [37][38].

This research bridges the gap by evaluating 14 clustering algorithms across datasets comprising 40,000 to 100,000 IoT sensor readings, offering insights into their performance under real-world, large-scale conditions. Unlike earlier studies limited to financial, healthcare, or geospatial domains, this work directly addresses IoT-specific demands such as real-time adaptability, scalability, and execution efficiency.

This work provides an empirical comparison to guide researchers and practitioners in selecting clustering techniques based on dataset size, computational resources, and effectiveness for large-scale IoT applications. Although clustering has been applied in finance, healthcare, and geospatial studies, evaluations tailored to massive IoT sensor datasets remain scarce. Table 2 summarizes key prior studies and their limitations, while Figure 4 highlights research gaps. Prior research often lacked scalability focus for IoT data or was confined to structured datasets. This study addresses these gaps, while future work can explore energy-efficient clustering and adaptive methods for evolving IoT environments.

Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2: Comparison of Related Works Highlighting Gaps in IoT Sensor Data Clustering Research

Study	Dataset Type	Algorithm(s) Used	Focus Area	Identified Gap
Nhat et al. [31]	Financial transactions	BIRCH	Streaming data clustering	Did not apply to IoT sensor networks
Marella et al. [32]	Network traffic	OPTICS	Anomaly detection in noisy data	Not tested on IoT temporal data
Wang et al. [33]	Traffic accident GIS data	HDBSCAN	Accident black-spot identification	Focused on geospatial data, not IoT
Yadav et al. [34]	General datasets	Multiple clustering algorithms	Broad algorithm review	No scalability analysis for IoT data
Sheng et al. [35]	Healthcare data	K-Means	Patient data clustering	Did not consider real-time IoT environments
Chakraborty et al. [36]	Image datasets	Affinity Propagation	Image segmentation	Irrelevant to IoT sensor data needs
Various others [37][38]	Structured datasets	Various methods	Small dataset analysis	No benchmarking on large-scale IoT datasets

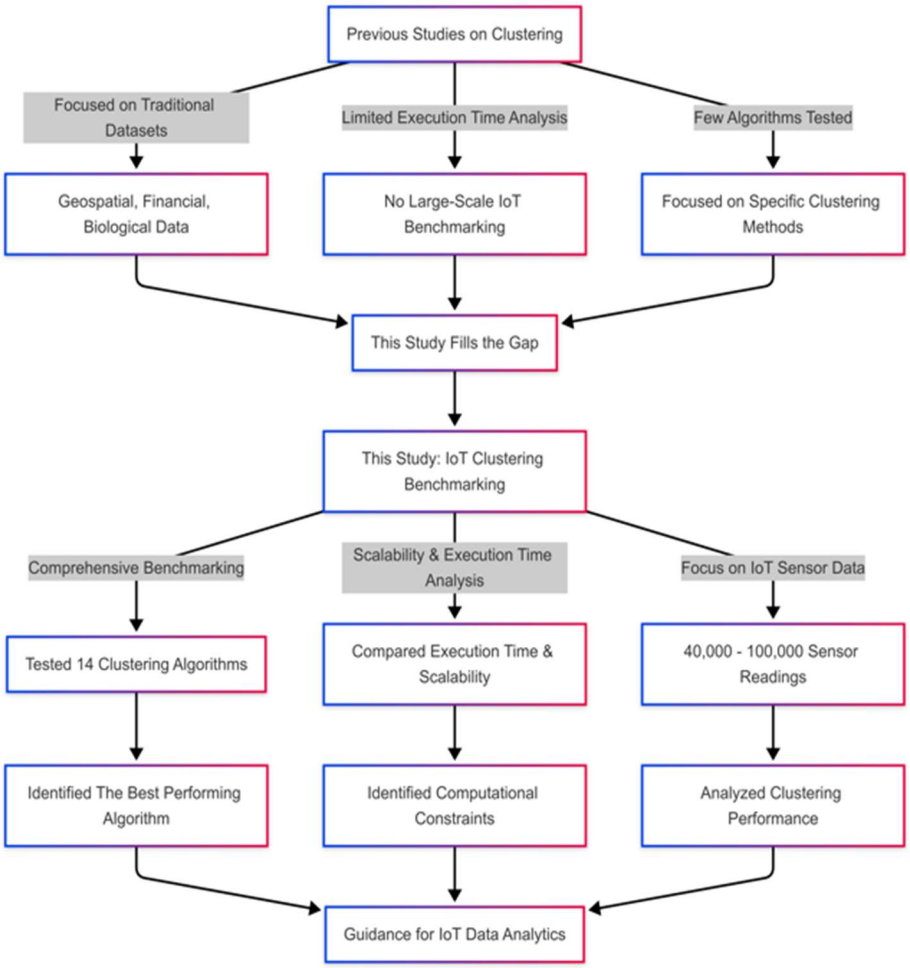

Figure 4: Comparative Analysis of Previous Studies and This Study's Contributions to IoT Clustering Benchmarking

4. METHODOLOGY

To analyse the above-mentioned clustering algorithms on an existing IoT dataset, the following methodology is employed. This approach is designed to effectively manage large volumes of real-time data and extract meaningful insights by grouping similar data points. The process is systematically broken down into a series of well-defined steps, each targeting a specific aspect of the data analysis pipeline. The detailed explanation of each stage ensures a comprehensive understanding of how the clustering algorithms are applied to derive actionable insights from the IoT data.

3.1 Creation of Oracle Cloud Computing Instances for infrastructure setup

We created an Oracle Cloud Computing, VM instance having a configuration of 256GB RAM and 16 core processor on a Windows Server Operating System, which was managed through the Oracle Cloud Consol. The VM instance served as the backbone for deploying and testing the clustering algorithms. This infrastructure setup in the Oracle Cloud Infrastructure platform provided us scalability, flexibility, and reliability for running workloads in a cloud environment [39].

3.2 Implementation of 14 clustering algorithms in Python

The implementation of various clustering algorithms in Python was done using scikit-learn (sklearn) [40] for efficient computation and analysis.

3.3 Description of the IoT sensor data used for analysis

Three identical, especially designed sensor arrays gathered the environmental data. Every sensor array was linked to a Raspberry Pi and positioned in several areas with diverse environmental circumstances.

Sensor readings and data details:

- The sensors collected seven readings: humidity, liquid petroleum gas (LPG), carbon monoxide (CO), temperature, smoke, motion and light.
- Measurements were consistently recorded at regular time intervals over a one-week period, resulting in a total of 405,184 data tuples.
- Each tuple includes:
  - A unique device identifier.
  - A timestamp.
  - The seven sensor readings.
- The data was published using the MQTT protocol, an industry standard for messaging in internet of things (IoT) applications, used for faster data transmission.

3.4 Data Pre-processing

Minimal pre-processing was required as the sensor data was consistently collected with no missing values. To ensure fair distance computations across all algorithms, sensor readings were normalized using Min-Max scaling, bringing all features into a standard range between 0 and 1.

3.5 Experimental Protocol

Each clustering algorithm was executed independently on the complete IoT dataset under identical hardware and software conditions. To account for variations in execution time due to resource allocation or system fluctuations, each experiment was repeated three times, and the average value was considered. No dimensionality reduction techniques or manual feature engineering were applied to maintain the integrity of the original dataset. To test scalability, experiments were conducted on sub-samples of the dataset, progressively increasing the data size from **40,000 to 100,000** records in increments of 20,000 (i.e., 40k, 60k, 80k, and 100k).

3.6 Evaluation Metrics

The following evaluation metric was utilized:

- **Execution Time:** Measured using Python's time module to capture the total time taken to complete clustering.
- **Scalability:** Observed based on the change in execution time as the dataset size increased from 40k to 100k.

3.7 Comparison and Analysis Procedure

All algorithms were tested on the same dataset splits and in the same execution environment to maintain fairness in comparison. The resulting performance metrics were tabulated and visualized using bar plots and line graphs to highlight the scalability and efficiency of different clustering techniques. Comparative trends were critically analysed to identify algorithms best suited for large-scale IoT data.

Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

5. RESULTS AND DISCUSSION

This study investigated the performance of various clustering algorithms on a large IoT dataset containing sensor measurements from multiple devices. The dataset was segmented into different sizes (40,000, 50,000, 60,000, 70,000, 80,000, 90,000, and 100,000 data points) and fourteen clustering algorithms were applied: affinity propagation, agglomerative, birch, bisecting k-means, dbscan, fuzzy c-means, gaussian mixtures, hdbscan, k-means, mean-shift, optics, overlapping k-means, spectral, and ward-hierarchical. The time taken by each algorithm to cluster the data was

Table 3: Represents the time taken in seconds by the various clustering algorithms

Data Size	40000	50000	60000	70000	80000	90000	100000
K-Means	2.53	2.55	2.6	2.72	2.85	2.91	3.01
Affinity Propagation	2856.87	*	*	*	*	*	*
Mean-Shift	241.72	464.7	865.77	1238.81	1509.29	2061.17	2287.86
Spectral	308.51	617.48	1076.55	1710.55	2244.41	*	*
Ward Hierarchical	53.31	87.33	130.77	183.51	256.54	332.82	419.25
Agglomerative	53.34	85.16	125.96	179.61	253.62	323.16	417.98
DBSCAN	13.58	16.02	20.69	29.07	37.85	46.67	53.81
HDBSCAN	5.35	9.25	6.49	7.21	8.17	9.66	10.36
Optics	83.22	131.32	173.39	249.03	296.82	383.59	455.14
Gaussian Mixtures	2.74	3.24	2.55	4.33	3.78	3.67	3.96
BIRCH	2.83	3.08	3.4	3.68	3.94	4.26	4.47
Bisecting K-Means	2.63	3	2.89	3.19	3.5	3.44	3.5
Fuzzy C Means	2.59	3.01	3.37	3.63	3.62	3.89	3.98
Overlapping K-Means	2.78	4.36	3.1	2.95	6.44	3.27	3.11

*Failed to execute

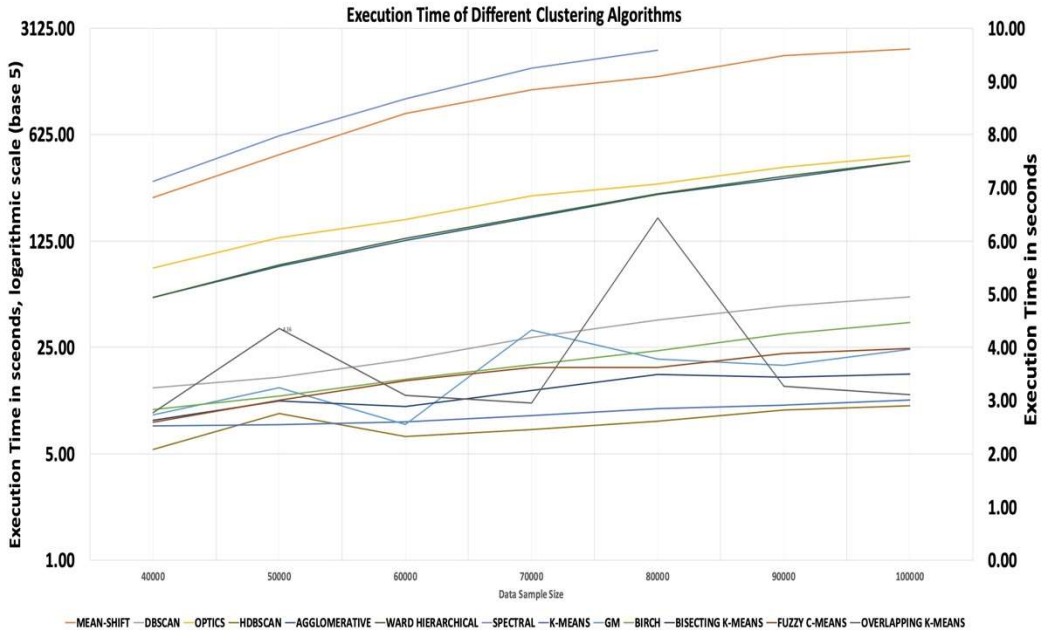

Figure 5: Performance of various clustering algorithms with reference to the execution time

recorded. The following observations were made after applying the above-mentioned clustering algorithms: The graphical representation below illustrates the execution time, measured in seconds on a logarithmic scale (base 5), of several clustering algorithms. Mean-Shift, DBSCAN, OPTICS, HDBSCAN, Agglomerative, Ward Hierarchical, and Spectral are depicted on the primary y-axis. Notably, the Agglomerative and Ward Hierarchical algorithms exhibit similar performance, resulting in overlapping data points on the graph.

Conversely, K-Means, Gaussian Mixtures (GM), BIRCH, Bisecting K-Means, Fuzzy C Means, and Overlapping K-Means are visualized on the secondary y-axis, representing their respective execution times in seconds.

The results show that Ward-Hierarchical, Agglomerative, K-Means, DBSCAN, OPTICS, Bisecting K-Means, HDBSCAN, BIRCH, Gaussian Mixtures, Fuzzy C-Means, and Overlapping K-Means were able to cluster the data in a reasonable time for all dataset sizes. Affinity Propagation and Spectral Clustering were significantly slower and failed to complete the clustering process for dataset sizes larger than 80,000 and 50,000 data points, respectively, which is represented by *(asterisk shaded region) in the above-mentioned table. These findings indicate that while a majority of the algorithms scaled well with increasing data volume, a few faced considerable limitations when processing very large datasets.

The findings of this study suggest that several clustering algorithms are suitable for analysing large IoT datasets containing sensor measurements. Their ability to manage large volumes of data efficiently makes them strong candidates for real-world deployment where real-time or near-real-time decision-making is necessary. K-Means, Ward-Hierarchical, Agglomerative, DBSCAN, OPTICS, BIRCH, Bisecting K-Means, and Fuzzy C-Means all displayed good scalability, handling dataset sizes up to 100,000 data points efficiently. Among all these algorithms, the clusters formed by BIRCH were observed to be more distinct and better separated, enhancing the interpretability of the clustering outcomes. These algorithms may therefore be preferable for real-world IoT applications where timely processing of large data streams is crucial to ensure responsive and adaptive system behaviour.

In contrast, Affinity Propagation and Spectral Clustering exhibited poor scalability, failing to handle larger datasets. While these algorithms may be suitable for smaller datasets or when computational resources are not a constraint, they may not be ideal choices for large-scale IoT data analysis where both speed and resource efficiency are critical requirements. It is crucial to consider that the choice of clustering algorithm for a specific application will rely on a number of variables, such as the dataset's size and composition, the required level of clustering granularity, the computational budget, and the real-time responsiveness needed by the application.

The findings of this investigation provide useful insights into the performance of clustering algorithms on large IoT datasets, helping researchers and practitioners select suitable algorithms for their needs. The algorithms are illustrated through three-dimensional graphs, with each algorithm shown in a separate subplot labelled from 'a' to 'g', offering a clear visual comparison of their performance across different dataset sizes.

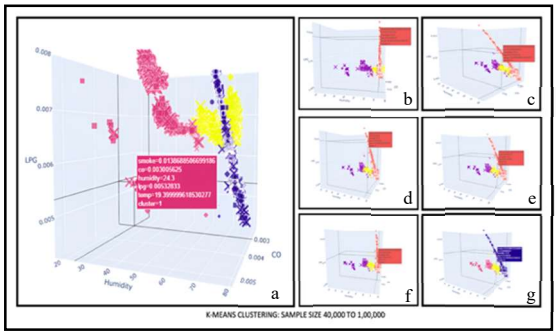

Figure 6. K-Means Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

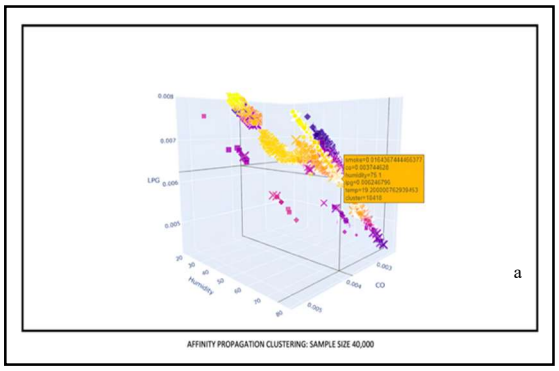

Figure 7. Affinity Propagation Clustering, Sample Size: (a: 40,000)

Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

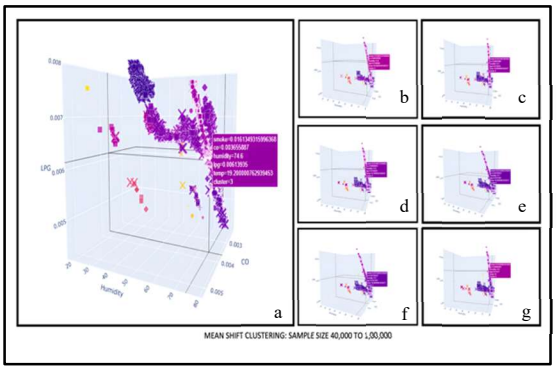

Figure 8. Mean-Shift Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

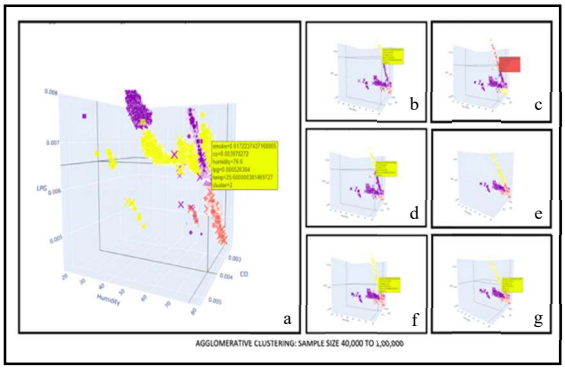

Figure 11. Agglomerative Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

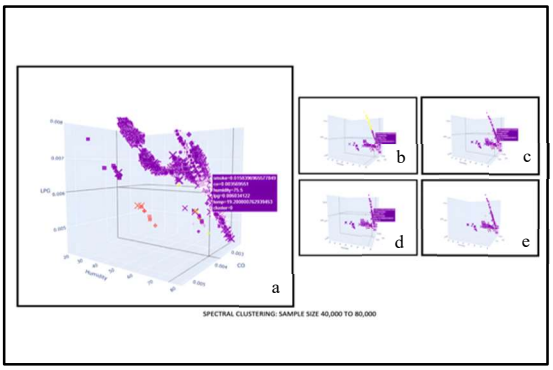

Figure 9. Spectral Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000)

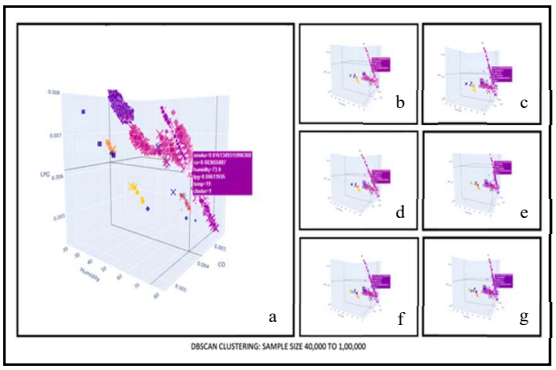

Figure 12. DBSCAN Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

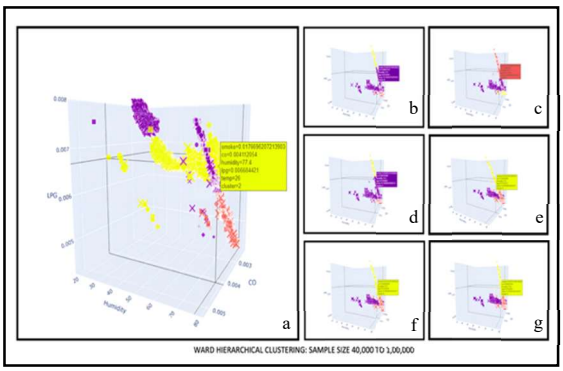

Figure 10. Ward Hierarchical Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

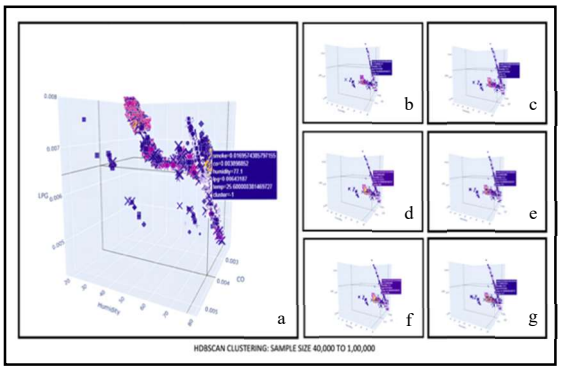

Figure 13. HDBSCAN Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

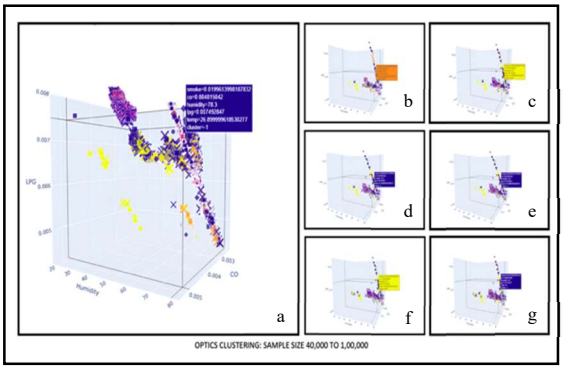

Figure 14. OPTICS Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

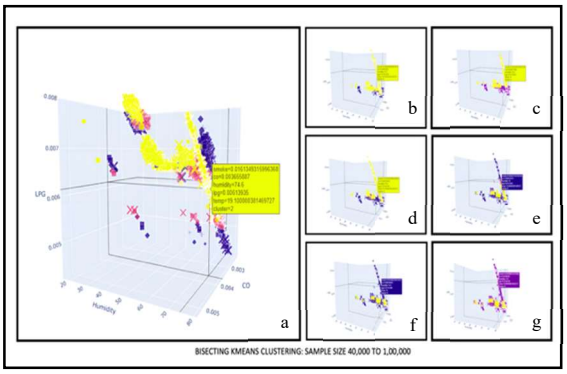

Figure 17. Bisecting K-Means Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000

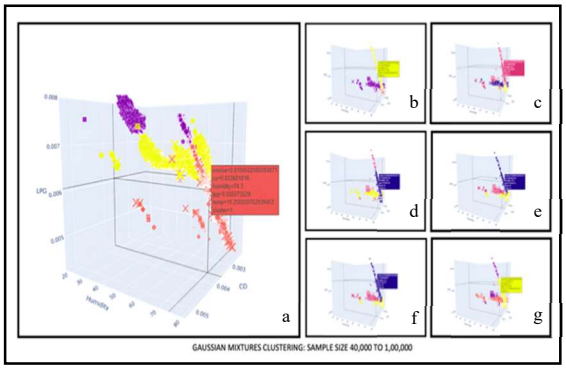

Figure 15. Gaussian Mixtures Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

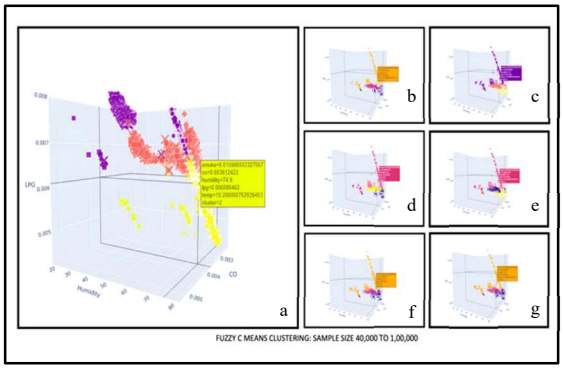

Figure 18. Fuzzy C Means Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

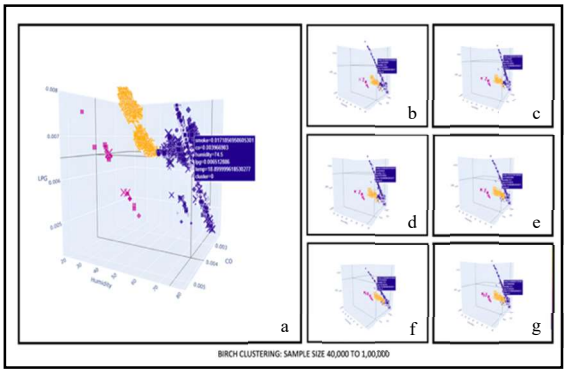

Figure 16. Birch Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

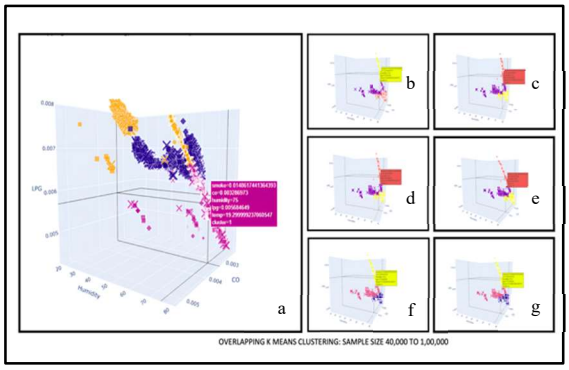

Figure 19. Overlapping K Means Clustering, Sample Size: (a: 40,000; b:50,000; c: 60,000; d: 70,000; e: 80,000; f: 90,000; g: 1,00,000)

Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 4: Number of clusters formed by clustering algorithms in each dataset sample.

Data Size	40000	50000	60000	70000	80000	90000	100000
K-Means	3	3	3	3	3	3	3
Mean-Shift	13	16	17	19	19	19	18
Affinity Propagation	21985	*	*	*	*	*	*
Spectral	3	3	3	3	3	*	*
Ward Hierarchical	3	3	3	3	3	3	3
Agglomerative	3	3	3	3	3	3	3
DBSCAN	7	7	8	7	8	8	9
HDBSCAN	2244	2754	3376	4095	4744	4095	5295
Optics	2811	3498	4248	5011	5011	6346	6964
Gaussian Mixtures	3	3	3	3	3	3	3
BIRCH	3	3	3	3	3	3	3
Bisecting K-Means	3	3	3	3	3	3	3
Fuzzy C Means	3	3	3	3	3	3	3
Overlapping K-Means	3	3	3	3	3	3	3

*Failed to execute

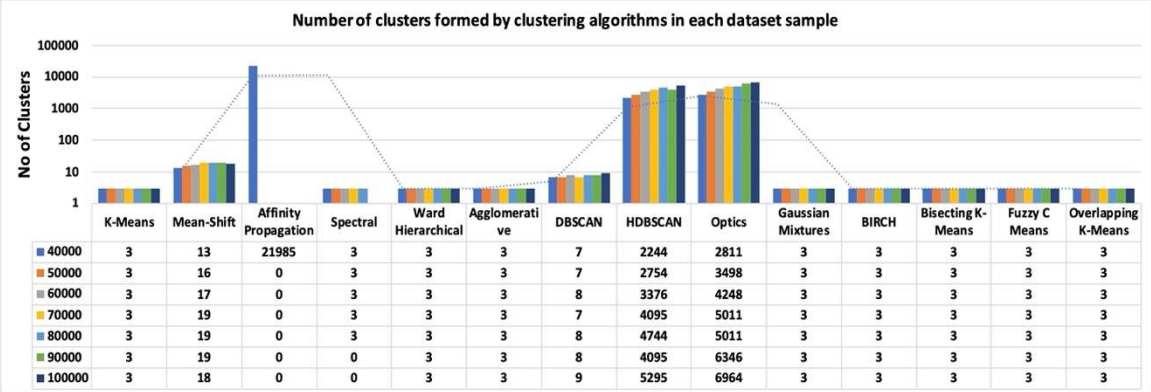

Figure 20. Graphical Representation Of Number Of Clusters Formed By Clustering Algorithms In Each Dataset Sample.

5.1 Additional Considerations

The performance of clustering algorithms can also be influenced by the specific attributes of the data, such as the number of features, the presence of noise or outliers, and the distribution of the data points.

This study focused on the time taken by the algorithms and other factors such as the quality of the clustering results.

The number of clusters generated by each clustering algorithm for each sample size was also observed to analyze how different techniques adapt to varying dataset sizes.

Here is a table representing the time complexity and space complexity of the following clustering algorithms: Affinity Propagation, Agglomerative, BIRCH, Bisecting K-Means, DBSCAN, Fuzzy C-Means, Gaussian Mixtures, HDBSCAN, K-Means, Mean-Shift, OPTICS, Overlapping K-Means, Spectral, and Ward-Hierarchical:

Table 5: Worst Case Time and Space Complexity of Clustering Algorithms

Algorithm	Time Complexity	Space Complexity
K-Means	O(nkT)	O(nk)
Affinity Propagation	O(n^2T)	O(n^2)
Mean-Shift	O(n^2d)	O(nd)
Spectral	O(n^3)	O(n^2)
Ward-Hierarchical	O(n^3)	O(n^2)
Agglomerative	O(n^2 log n)	O(n^2)
DBSCAN	O(n log n)	O(n)
HDBSCAN	O(n log n)	O(n)
OPTICS	O(n log n)	O(n)
Gaussian Mixtures	O(nKT)	O(nk)
BIRCH	O(n)	O(n)
Bisecting K-Means	O(nk log k)	O(nk)
Fuzzy C-Means	$O(nT^2)$	O(nT)
Overlapping K-Means	O(nkT)	O(nk)

Note: The variables k, n, d, and T represents clusters, number of data points, and iterations, respectively in the data.

5.2 Difference from Prior Research

This study differs from prior research by focusing on the scalability and execution time of clustering algorithms when applied to large-scale IoT sensor data, rather than evaluating clustering quality through internal validation metrics.

Previous studies typically analysed clustering algorithms using small-sized datasets or non-IoT datasets, often without considering the challenges posed by real-time data volumes in IoT environments. In contrast, this research systematically evaluated the performance of fourteen clustering algorithms on datasets ranging from 40,000 to 100,000 data points, emphasizing practical aspects such as clustering speed and computational feasibility.

While earlier work often concentrated on theoretical assessments, this study conducted experiments using real-world IoT data collected from sensor arrays deployed in diverse environmental conditions and transmitted using the MQTT protocol. This approach provides a more application-oriented understanding of how clustering algorithms perform under realistic IoT operational scenarios.

The findings reveal that algorithms like K-Means, BIRCH, DBSCAN, OPTICS, Ward-Hierarchical, Agglomerative, Bisecting K-Means, and Fuzzy C-Means offer better scalability and faster execution, making them more suitable for real-time IoT data analysis. In contrast, algorithms such as Affinity Propagation and Spectral Clustering exhibited limited scalability when handling larger datasets.

Thus, this work extends the current research landscape by offering practical, real-world insights into algorithm selection for large-volume IoT data clustering tasks.

7 Conclusion and Future Scope

This study conducted a comprehensive comparative analysis of fourteen clustering algorithms on a large IoT sensor dataset, addressing the critical challenge of identifying scalable and efficient clustering techniques for real-time IoT data analysis. The findings reveal that algorithms such as OPTICS, DBSCAN, K-Means, Ward-Hierarchical, Agglomerative, HDBSCAN, Gaussian Mixtures, BIRCH, Overlapping K-Means, Bisecting K-Means, and Fuzzy C-Means exhibited strong scalability, successfully clustering datasets containing up to 100,000 data points. Among them, BIRCH consistently produced more distinct and meaningful clusters, suggesting its particular suitability for large-scale IoT environments where timely and reliable pattern recognition is essential.

Conversely, Affinity Propagation and Spectral Clustering were found to lack scalability, failing to cluster larger datasets effectively. This highlights a crucial insight: while some algorithms may perform well on smaller or non-IoT datasets, their performance does not necessarily translate to high-volume, high-velocity IoT contexts. Therefore, this study reinforces the necessity of carefully aligning the choice of clustering algorithm with the specific size, structure, and demands of IoT data streams.

By systematically evaluating execution time and practical scalability rather than focusing solely on theoretical metrics, this research provides actionable guidance to practitioners and researchers who must manage increasingly large IoT deployments. It bridges an important gap by empirically

demonstrating how different clustering techniques behave under real-world IoT data loads, thus answering the primary research questions posed at the outset.

While this study emphasizes the scalability and computational efficiency of clustering algorithms, future research should address several limitations. First, the qualitative evaluation of clustering results including the interpretability and real-world usefulness of the formed clusters remains unexplored. Future studies should incorporate domain-specific validation measures to assess clustering quality beyond time performance.

Additionally, further investigation is needed into how data characteristics such as noise, the presence of outliers, feature dimensionality, and temporal variations impact algorithm performance. Exploring adaptive clustering frameworks that can dynamically adjust parameters based on incoming data patterns, and testing algorithms on streaming or continuously updating IoT data, would provide deeper insights. Expanding the dataset diversity by including more complex, multi-modal IoT data can also make the findings more generalizable.

REFERENCES:

[1] Reddy, Chandan K. Data clustering: algorithms and applications. Chapman and Hall/CRC, 2018.

[2] Jain, Anil K., and Richard C. Dubes. Algorithms for clustering data. Prentice-Hall, Inc., 1988.

[3] Floris, Alessandro, and Luigi Atzori. "Managing the quality of experience in the multimedia Internet of Things: A layered-based approach." Sensors 16.12 (2016): 2057.

[4] Debauche, Olivier, et al. "Data management and internet of things: A methodological review in smart farming." Internet of Things 14 (2021): 100378.

[5] Patel, Keyur K., Sunil M. Patel, and P. Scholar. "Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges." International journal of engineering science and computing 6.5 (2016).

[6] Lyu, Lingjuan, et al. "Fog-empowered anomaly detection in IoT using hyperellipsoidal clustering." IEEE Internet of Things Journal 4.5 (2017): 1174-1184.

[7] Chen, Feng, et al. "Data mining for the internet of things: literature review and challenges." International Journal of Distributed Sensor Networks 11.8 (2015): 431047.

[8] Cui, Zhihua, et al. "Personalized recommendation system based on collaborative filtering for IoT scenarios." IEEE Transactions on Services Computing 13.4 (2020): 685-695.

[9] Behera, Trupti Mayee, et al. "Residual energy-based cluster-head selection in WSNs for IoT application." IEEE Internet of Things Journal 6.3 (2019): 5132-5139.

[10] Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar. Introduction to Data Mining. Pearson Education India, 2016.

[11] Hartigan, John A., and Manchek A. Wong. "Algorithm AS 136: A k-means clustering algorithm." Journal of the royal statistical society. series c (applied statistics) 28.1 (1979): 100-108.

[12] Dueck, Delbert. Affinity propagation: clustering data by passing messages. Diss. 2009.

[13] Cheng, Yizong. "Mean shift, mode seeking, and clustering." IEEE transactions on pattern analysis and machine intelligence 17.8 (1995): 790-799.

[14] Ng, Andrew, Michael Jordan, and Yair Weiss. "On spectral clustering: Analysis and an algorithm." Advances in neural information processing systems 14 (2001).

[15] Miyamoto, Sadaaki, et al. "Ward method of hierarchical clustering for non-Euclidean similarity measures." 2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR). IEEE, 2015.

[16] Murtagh, Fionn, and Pierre Legendre. "Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion?." Journal of classification 31 (2014): 274-295.

[17] Ester, Martin, et al. "A density-based algorithm for discovering clusters in large spatial databases with noise." kdd. Vol. 96. No. 34. 1996.

[18] Campello, Ricardo JGB, Davoud Moulavi, and Jörg Sander. "Density-based clustering based on hierarchical density estimates." Pacific-Asia conference on knowledge discovery and data mining. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[19] Ankerst, Mihael, et al. "OPTICS: Ordering points to identify the clustering structure." ACM Sigmod record 28.2 (1999): 49-60.

[20] Andriyanov, Nikita, Alexander Tashlinsky, and Vitaly Dementiev. "Detailed clustering based on Gaussian mixture models." Intelligent Systems and Applications: Proceedings of the 2020

Intelligent Systems Conference (IntelliSys) Volume 2. Springer International Publishing, 2021.

[21] Zhang, Tian, Raghu Ramakrishnan, and Miron Livny. "BIRCH: an efficient data clustering method for very large databases." ACM sigmod record 25.2 (1996): 103-114.

[22] Patil, Ruchika, and Amreen Khan. "Bisecting K-means for clustering web log data." International Journal of Computer Applications 116.19 (2015).

[23] Bezdek, James C., Robert Ehrlich, and William Full. "FCM: The fuzzy c-means clustering algorithm." Computers & geosciences 10.2-3 (1984): 191-203.

[24] Baadel, Said, Fadi Thabtah, and Joan Lu. "Overlapping clustering: A review." 2016 SAI Computing Conference (SAI). IEEE, 2016.

[25] Oyewole, Gbeminiyi John, and George Alex Thopil. "Data clustering: application and trends." Artificial intelligence review 56.7 (2023): 6439-6475.

[26] Ikotun, Abiodun M., et al. "K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data." Information Sciences 622 (2023): 178-210.

[27] Basu, Sulagna, and Aritra Bandyopadhyay. "Birch Clustering Approach for Detection of Random Valued Impulse Noise." Journal of The Institution of Engineers (India): Series B (2024): 1-9.

[28] Ran, Xingcheng, et al. "Comprehensive survey on hierarchical clustering algorithms and the recent developments." Artificial Intelligence Review 56.8 (2023): 8219-8264.

[29] Caudillo-Cos, Camilo Alberto, et al. "Defining urban boundaries through DBSCAN and Shannon's entropy: The case of the Mexican National Urban System." Cities 149 (2024): 104969.

[30] Ge, Haimiao, et al. "Affinity propagation based on structural similarity index and local outlier factor for hyperspectral image clustering." Remote Sensing 14.5 (2022): 1195.

[31] Nhat, Nguyen Minh. "Applied Density-Based Clustering Techniques for Classifying High-Risk Customers: A Case Study of Commercial Banks in Vietnam." Journal of Applied Data Sciences 5.4 (2024): 1639-1653.

[32] Marella, Deepika, et al. "Unveiling Network Anomalies: A Comparative Study of Real-time Log-Based Detection Approach." 2023 IEEE 20th India Council International Conference (INDICON). IEEE, 2023.

[33] Ezugwu, Absalom E., et al. "A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects." Engineering Applications of Artificial Intelligence 110 (2022): 104743.

[34] Yadav, Dinesh Kumar, et al. "A comparative study of various clustering algorithms using in machine learning." Advances in Electronics, Computer, Physical and Chemical Sciences. CRC Press, 2025. 458-465.

[35] Sheng, Yiyang, et al. "Augmenting K-Means Clustering With Qualitative Data to Discover the Engagement Patterns of Older Adults With Multimorbidity When Using Digital Health Technologies: Proof-of-Concept Trial." Journal of Medical Internet Research 26 (2024): e46287.

[36] Chakraborty, Shouvik, Kalyani Mali, and Sushmita Mitra. "Affinity Propagation in Semi-Supervised Segmentation: A Biomedical Application." IEEE Transactions on Systems, Man, and Cybernetics: Systems (2024).

[37] Hassan, Bryar A., et al. "From A-to-Z review of clustering validation indices." Neurocomputing 601 (2024): 128198.

[38] Oyewole, Gbeminiyi John, and George Alex Thopil. "Data clustering: application and trends." Artificial intelligence review 56.7 (2023): 6439-6475.

[39] "Welcome to Oracle Cloud Infrastructure." Oracle, 2024, https://docs.oracle.com/en-us/iaas/Content/GSG/Concepts/baremetalintro.htm. Accessed 15 October. 2024.

[40] "scikit-learn: Machine Learning in Python — scikit-learn 1.4.1 Documentation." scikit-learn, n.d., https://scikit-learn.org/stable/. Accessed 7 October. 2024.