STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **14**, August 2025, pp 970–1017.
Published online in International Academic Press (www.IAPress.org)

Hybrid Outlier Detection Framework Based on Optimized KMeans and HDBSCAN Using Bat Algorithm and LSTM Autoencoder

Mai Abdrabo*, Hossam Refaat, Mohammed Abdallah Makhlouf, Osama Farouk

Information System Department, Faculty of computers and information, Suez Canal University, Ismailia, Egypt

Abstract Outlier detection is a critical task in data mining, especially in domains such as healthcare, cybersecurity, and fraud detection, where abnormal instances can signify crucial insights. Traditional approaches, including DBSCAN, Isolation Forest, and statistical techniques like Z-Score and IQR, often suffer from issues such as sensitivity to parameters, limited adaptability, and reduced effectiveness in high-dimensional or complex data. To overcome these limitations, this paper proposes a hybrid outlier detection framework that combines KMeans clustering with HDBSCAN, enhanced through Bat Algorithm-based optimization for dynamic selection of clustering parameters (eps and minsamples).
The proposed method is evaluated alongside IS-DBSCAN, Autoencoders, and advanced graph-based approaches like Cluster Catch Digraphs (CCDs) with Outbound and Inbound Outlyingness Scores (OOS and IOS) use in this study. It explores and compares two advanced outlier detection approaches applied to two real-world datasets: the Online Retail and the Diabetes 130-US hospitals datasets. The first approach utilizes a scalable Spark-based DBSCAN algorithm, while the second integrates KMeans clustering with HDBSCAN, optimized via the Bat Algorithm (KMeans + HDBSCAN (BAT)). A Spark-based implementation of DBSCAN.These methods were evaluated on two real-world datasets—Diabetes and Online Retail—using Silhouette Score (SII) and classification Accuracy (Acc) as performance metrics with performanceperformance (F1 = 0.972, AUPRC = 0.947). Experimental results demonstrate that the proposed hybrid approach significantly outperforms the Spark-based DBSCAN in both clustering quality and classification performance, achieving a Silhouette score of 0.67 and accuracy of 66.8% on the Diabetes dataset performance (F1 = 0.66.2, AUC = 0.72.26%), and 0.59 and 97.35% respectively on the Online Retail dataset.For MINIST dataset The model achieved high performance (F1 = 0.92, AUC = 0.96), outperforming Isolation Forest, with notable improvements in clustering quality as BAT iterations increased. These results highlight the effectiveness of integrating KMeans for initialization, HDBSCAN for density-based clustering, and the Bat Optimization algorithm for fine-tuning key parameters.

Keywords Outlier Detection , hybrid KMeans + HDBSCAN (BAT optimized) ,IS-DBSCAN ,Autoencoder ,Cluster Catch Digraphs (CCDs) , Outlyingness Score,Inbound/Outbound Score (IOS/OOS) , Local Coulomb Outlier Factor (LCOF) ,Spark

DOI: 10.19139/soic-2310-5070-2581

1. Introduction

Outlier detection plays a pivotal role across critical domains such as healthcare, finance, cybersecurity, and industrial systems, where the early identification of abnormal patterns is essential for informed decision-making. As data continues to grow in volume, complexity, and dimensionality, traditional outlier detection techniques often struggle with issues related to scalability, robustness, and adaptability. Classical statistical approaches such as Z-Score and Interquartile Range (IQR) [1] are simple and interpretable but are generally ineffective in modeling complex, non-linear distributions. Similarly, tree-based models like Isolation Forest (IF) [2] and density-based methods such as Local Outlier Factor (LOF)[3] demonstrate limited performance in high-dimensional or noisy datasets.

*Correspondence to: Mai Abdrabo (Email: mai_abdrabo86@yahoo.com). Information System Department, Faculty of computers and information, Suez Canal University, Ismailia, Egypt.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

Recently, clustering-based techniques have attracted increasing attention for outlier detection due to their ability to uncover underlying data structures without requiring labeled samples. Algorithms such as DBSCAN and its variants [4] offer the advantage of discovering arbitrarily shaped clusters while identifying outliers based on density deviations. However, the performance of DBSCAN is often highly sensitive to its parameter settings (e.g., `eps` and `min_samples`), leading to instability across diverse datasets and scenarios.

To address these limitations, we propose a **hybrid outlier detection framework** that integrates KMeans with HDBSCAN, where the critical clustering parameters are dynamically optimized using the Bat Algorithm [5]. The rationale behind this integration is to leverage the global partitioning capability of KMeans to estimate initial cluster seeds, which are subsequently refined through HDBSCAN's hierarchical density-based clustering [6]. This hybridization enables the model to construct adaptive clustering boundaries, improving its capacity to capture subtle anomalies in heterogeneous data.

In addition to spatial clustering, we incorporate a **Long Short-Term Memory (LSTM) Autoencoder** to enhance the modeling of temporal behavioral patterns, especially relevant in domains such as healthcare or customer analytics where monthly transaction or behavior data are sequential in nature. The LSTM Autoencoder is designed as follows:

- The **encoder** LSTM processes a sequence of 12 months of behavioral/spending data and compresses it into a fixed-length latent vector.
- The **decoder** LSTM reconstructs the original sequence from this latent representation.
- The reconstruction process is optimized using **Mean Squared Error (MSE)** loss.

The resulting latent vector encapsulates key temporal dependencies and behavioral dynamics, and it is concatenated with the static feature space before clustering. This enriched feature representation significantly improves both clustering quality and the precision of anomaly detection.

Figure 1 provides an overview of the proposed hybrid outlier detection framework, highlighting the data flow from raw input to final detection results. The integration of the LSTM Autoencoder is not merely auxiliary but represents a core innovation in how time-series behavior is embedded into the detection pipeline. This design improves the model's ability to differentiate between normal variations and genuine anomalies—especially in applications requiring behavioral interpretability. To validate the proposed framework, we conduct a comprehensive comparative study against several state-of-the-art outlier detection techniques, including:

- IS-DBSCAN [7],
- Autoencoder-based anomaly detection models [8],
- Graph-based strategies such as Cluster Catch Digraphs (CCDs) [9],
- Local Coulomb Outlier Factor (LCOF) [10].

Furthermore, we adopt novel evaluation metrics including **Inbound and Outbound Outlyingness Scores (IOS and OOS)** [11] to capture both local and global anomaly behaviors in a more nuanced manner.

Experimental evaluations conducted on a real-world healthcare dataset comprising 10,000 samples demonstrate that our hybrid KMeans + HDBSCAN (Bat-optimized) framework consistently outperforms baseline methods across multiple performance indicators (Precision, Recall, F1-score, and Accuracy). The combined use of optimization, deep temporal encoding, and adaptive clustering not only improves detection accuracy but also enhances scalability and robustness, making the proposed solution well-suited for real-world, large-scale anomaly detection tasks as shown in figure 1.

Organization. The remainder of this paper is organized as follows: Section 2 presents the background and literature review. Section 3 details the proposed methodology. Proposed framework is provided in Section 4, while Section 5 offers Evaluation Metrics.Section 6 (Discussion)offers an ablation study on the impact of the LSTM Autoencoder and detect outliers using kmeans +hdbscan optimized bat bat .Proposed framework demonstrates strong adaptability and performance, several limitations are presented in section 7. Finally, Section 8 concludes the paper.

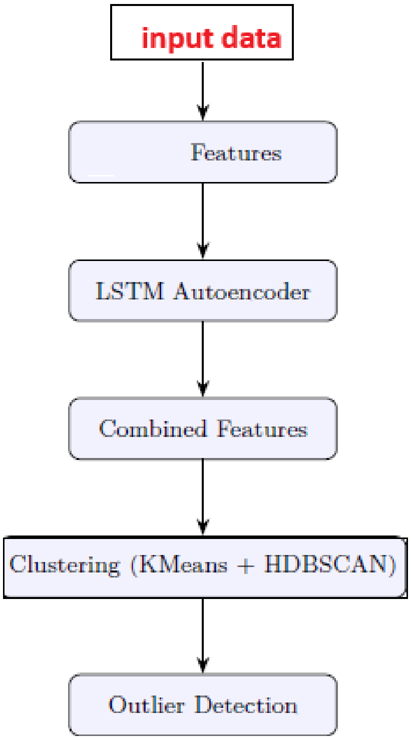

Figure 1. The proposed hybrid outlier detection framework

2. Background and Literature Review

Outlier detection has witnessed significant progress in recent years, particularly through clustering-based and graph-based techniques that aim to improve robustness and adaptability in diverse data environments. This section provides a structured overview of existing methods, followed by a comparative summary.

2.1. Review of Existing Approaches

2.1.1. Statistical Methods Classical statistical methods such as Z-Score, Interquartile Range (IQR), and Mahalanobis distance are computationally efficient and easy to interpret. However, they rely heavily on distributional assumptions, making them less effective in high-dimensional or non-Gaussian settings [1]. In contrast, more recent approaches such as Isolation Forest (IF) [2] and Local Outlier Factor (LOF) [3] adopt tree-based and density-based paradigms, respectively. Although widely used, their performance degrades in the presence of noise or high-dimensional feature spaces.

2.1.2. IS-DBSCAN IS-DBSCAN (Influenced Space-DBSCAN) enhances the classic DBSCAN by introducing a density influence (DI) measure [12]:

$$\mathrm{DI}(p) = \sum_{q \in N(p)} \frac{1}{1 + d(p, q)} \tag{1}$$

Here, $N(p)$ represents the neighborhood of point p, and $d(p, q)$ denotes the distance between points p and q. Points are then categorized into core, border, or outliers based on their DI value. This formulation allows for improved sensitivity to local densities.

2.1.3. HDBSCAN (Hierarchical DBSCAN) HDBSCAN [13] generalizes DBSCAN by using a hierarchy built from mutual reachability distances:

$$d_{\text{mreach}}(a, b) = \max(\text{core}_k(a), \text{core}_k(b), d(a, b)) \tag{2}$$

where $\text{core}_k(p)$ is the distance to the k-th nearest neighbor of point p. A minimum spanning tree (MST) is constructed using mutual reachability distances, and a condensed tree is extracted to represent the hierarchical clusters. Outliers are those points that do not belong to any stable cluster.

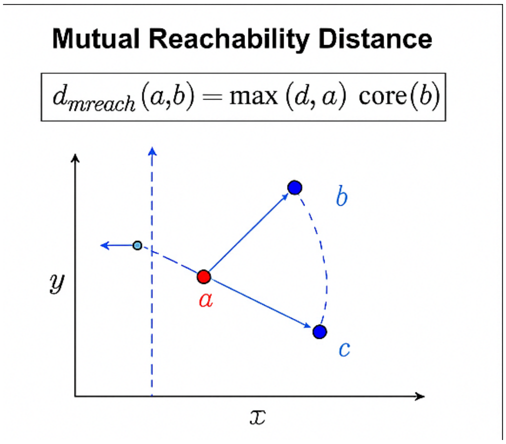
Mutual Reachability Distance
$d_{mreach}(a,b) = \max\ (d, a)\ \text{core}(b)$
y
b
a
c
x

Figure 2. Visualization of mutual reachability distance $d_{\text{mreach}}(a, b)$ combining Euclidean and core distances.

2.1.4. Autoencoder-Based Detection Autoencoders [14] are neural networks trained to reconstruct input data. They capture latent structures in high-dimensional datasets. Outliers are detected by computing the reconstruction error:

$$L(x, \hat{x}) = \|x - \hat{x}\|^2 \tag{3}$$

Instances with errors above a threshold are classified as outliers. This approach is especially effective in modeling non-linear data but may suffer from overfitting or poor generalization in limited data scenarios.

2.1.5. Cluster Catch Digraphs (CCDs) CCDs [15] leverage graph theory to model inter-point relationships. Each point is a vertex, and directed edges are drawn to closer or more central neighbors:

$$E = \{(i, j) \mid d(i, j) \leq d(i, k), \forall k \in \text{neighbors of } i\} \tag{4}$$

Outliers are nodes with few or no incoming edges, indicating sparse connectivity and weak integration within dense regions.

2.1.6. Local Coulomb Outlier Factor (LCOF) Inspired by Coulomb's law, LCOF [16] defines an outlier score as the inverse squared distance to neighbors:

$$LCOF(i) = \sum_{j \in N(i)} \frac{1}{d(i, j)^2} \tag{5}$$

Points with low accumulated repulsion (i.e., isolated from their neighbors) are marked as outliers. This method is parameter-free and robust to varying data distributions.

2.1.7. Inbound and Outbound Outlyingness Scores (IOS & OOS) These scores [**?**] are derived from graph structures and quantify how strongly a point deviates from the data. OOS measures how far a point is from others, while IOS assesses how distant a point appears from the viewpoint of its neighbors. High values in both scores indicate a strong likelihood of being an outlier.

2.1.8. Copula-Based Outlier Detection (COPOD) Copula-Based Outlier Detection (COPOD) is a recent unsupervised method introduced by Li et al. (2020) [17]. It is entirely parameter-free and based on empirical copula modeling to estimate tail probabilities in each feature dimension. By aggregating these extreme value scores, COPOD can effectively detect outliers in various data distributions.

The method is designed to be:

- **Fully automatic**: no need for parameter tuning.
- **Scalable**: suitable for high-dimensional data.
- **Interpretable**: through feature-wise tail probability contributions.

COPOD outperforms several classical and deep learning models in benchmark studies, particularly in unsupervised contexts. It has been integrated into the `pyod` library and is widely used in practice due to its balance of robustness, interpretability, and efficiency.

2.1.9. LSTM Autoencoder for Sequential Anomaly Detection LSTM Autoencoders have emerged as a powerful tool for detecting anomalies in sequential and time-series data. Unlike traditional autoencoders, LSTM-based architectures are specifically designed to capture temporal dependencies and long-range correlations through gated memory cells.

The typical architecture involves:

- An **encoder LSTM** that processes input sequences and compresses them into fixed-length latent vectors.
- A **decoder LSTM** that reconstructs the original sequence from this compressed representation.
- A reconstruction loss function (usually MSE) used to identify anomalies based on reconstruction error.

Data points or sequences that yield high reconstruction errors are considered anomalous, particularly when the model fails to learn regular temporal patterns. LSTM Autoencoders have been effectively applied in domains such as financial fraud detection, patient monitoring, and industrial system fault prediction [18, 19].

2.2. Evaluation Metrics

Evaluation of outlier detection models requires both internal and external metrics. Internal indices like the Silhouette Coefficient [20] and Davies–Bouldin Index (DBI) [21] assess cluster quality. Silhouette scores closer to 1 imply compact and well-separated clusters, while lower DBI values indicate better-defined clusters.

External metrics such as Precision, Recall, F1-score, and Accuracy are crucial when ground truth is available [22]. These are defined as follows:

- **Precision**: The proportion of true outliers among detected ones.
- **Recall**: The proportion of actual outliers that were correctly identified.
- **F1-Score**: Harmonic mean of Precision and Recall.
- **Accuracy**: Overall proportion of correct predictions.

Imbalance-Aware Evaluation Metrics In highly imbalanced datasets—common in anomaly detection scenarios—the use of standard accuracy-based metrics may lead to misleading conclusions. As a result, imbalance-aware metrics such as the Area Under the Receiver Operating Characteristic Curve (AUC-ROC) and the Area Under the Precision-Recall Curve (AUPRC) are widely recommended ,These metrics are particularly suitable for evaluating unsupervised and semi-supervised outlier detection models, where the positive (anomalous) class is often underrepresented [23, 24].

Class Imbalance in Anomaly Detection An inherent challenge in real-world anomaly detection tasks is the significant **class imbalance**, where anomalous cases represent a minor fraction of the overall data. A notable example is hospital readmission prediction, where the positive class (readmitted patients) is much rarer than the negative class. In such cases, **accuracy becomes a misleading metric**, as a model may achieve high accuracy by simply predicting the dominant (normal) class. Therefore, **imbalance-aware metrics must be used**, especially when the anomalous class is rare and critical to detect, such as hospital readmissions.

To address this, we adopt **imbalance-aware evaluation metrics** such as the *Area Under the Precision-Recall Curve (AUPRC)*, which provides a more informative measure of performance in skewed datasets. Additionally, we employ *Recall@K*, which quantifies the proportion of true anomalies detected among the top-K highest-scoring samples. These metrics offer a more realistic assessment of a model's ability to identify rare but critical anomalies [23, 24, 25].

2.3. *Recent Literature and Trends*

Recent studies have introduced novel perspectives on anomaly detection. The Random Clustering-Based Outlier Detector [26] applies probabilistic theory through randomized clustering iterations. Another work extends anomaly detection to functional data using an AA + kNN model to cluster and identify curve anomalies [27].

Cluster Catch Digraphs have been further refined through new scores such as IOS and OOS [28], improving interpretability in high-dimensional spaces. LCOF has also been redefined as a fully parameter-free alternative for diverse datasets [29].

ADBench [30], a recent large-scale benchmark, evaluates 30 anomaly detection methods across different supervision levels. Surprisingly, several simple unsupervised techniques outperformed deep supervised models, underscoring the need for appropriate model selection and highlighting the underexplored potential of clustering-based approaches.

A comparative study [31, 32] focused on tuning unsupervised one-class classifiers, including GMM and GLOSH, showed that performance varies significantly with data characteristics. However, these studies were oriented toward supervised or semi-supervised settings, unlike the present study which targets unsupervised, clustering-based anomaly detection.

Table 1. Comparative Analysis of Outlier Detection Methods Across Key Criteria

Method	**Scalable**	**Param-Free**	**High-Dim.**	**Interpretable**	**Noise Sens.**
Z-Score / IQR / Mahalanobis	✓	✓		✓	High
Isolation Forest (IF)	✓		✓	✓	Moderate
Local Outlier Factor (LOF)				✓	High
IS-DBSCAN				✓	High
HDBSCAN	✓		✓	✓	Low
Autoencoder-Based	✓		✓		Moderate
Cluster Catch Digraphs (CCDs)		✓	✓	✓	Low
Local Coulomb Outlier Factor (LCOF)	✓	✓	✓	✓	Low
IOS / OOS (CCDs)		✓	✓	✓	Low
COPOD	✓	✓	✓	✓	Low
Autoencoder-Based (LSTM)	✓		✓		Moderate

Table 2. . Detecting and dealing with rejected feature in diabetes data set.

Feature	Number of missing	Need to rejected or not
Race	1813	Not rejected
Gender	2	Not rejected
Weight	78844	Need to rejected almost 99% missing
payer$_code$	32231	Need to rejected almost 40% missing
medical$_specialty$	39935	Need to rejected almost 40% missing
diag$_1$	18	Not rejected
diag$_2$	288	Not rejected transformed into high-level clinical categories
diag$_3$	1125	Not rejected transformed into high-level clinical categories

2.4. *Comparative Analysis of Reviewed Methods*

To provide a consolidated view of the discussed methods, Table 1 presents a qualitative comparison across key aspects: scalability, parameter.

2.5. *Summary*

The reviewed methods span a wide spectrum—from simple statistical detectors to advanced neural and graph-based models. While deep learning and functional approaches bring representational power, graph-based detectors such as CCDs and LCOF offer strong interpretability and adaptability without extensive tuning. The present study leverages the strengths of density- and structure-based clustering to construct a robust hybrid outlier detection pipeline optimized via metaheuristics.

3. Proposed Methodology

3.1. *Datasets description and preprocessing*

3.1.1. Dataset1: The data set titled " Diabetes". It represents ten years (1999-2008) of clinical care in 130 US hospitals and integrated delivery networks. It includes more than 50 features representing patient and hospital outcomes. Information was extracted from the database for encounters that satisfied the following criteria.

(1) It is an inpatient encounter (a hospital admission).
(2) It is a diabetic encounter, one during which any kind of diabetes was entered into the system as a diagnosis.
(3) The length of stay was at least 1 day and at most 14 days.
(4) Laboratory tests were performed during the encounter.
(5) Medications were administered during the encounter.

The data contains such attributes as patient number, race, gender, age, admission type, time in hospital, medical specialty of admitting physician, number of lab tests performed, HbA1c test result, diagnosis, number of medications, diabetic medications, number of outpatient, inpatient, and emergency visits in the year before the hospitalization[33].

Data Preprocessing and Feature Engineering The preprocessing pipeline for the diabetes dataset was designed to ensure data integrity, enhance interpretability, and enable compatibility with both traditional machine learning models and deep learning architectures. The dataset was first imported while treating special placeholders (e.g., `"?"`) as missing values. Features with more than 50% missing entries were dropped, and the remaining missing values were forward-filled to preserve temporal consistency within patient records.

To construct a binary classification target, the `readmitted` attribute was mapped to 0 (not readmitted) and 1 (readmitted). Identifiers such as `encounter_id` and `patient_nbr` were excluded from modeling to avoid information leakage.

Diagnosis Code Grouping: The diagnostic features `diag_1`, `diag_2`, and `diag_3`, originally represented as raw ICD-9 codes, were transformed into high-level clinical categories using medically informed grouping rules. This reduced dimensionality and improved semantic interpretability. The grouping was based on ICD-9 ranges:

- 390–459 → Circulatory System
- 460–519 → Respiratory System
- 250 → Diabetes
- 140–239 → Neoplasms
- 800–999 → Injury/Poisoning
- 240–279 → Endocrine Disorders
- Other codes → Mapped as "Other"

Feature Engineering: Several domain-specific features were engineered to capture key clinical and behavioral patterns:

- **Polypharmacy:** A binary indicator set to 1 if the patient was prescribed more than five distinct medications during the encounter. This feature is widely used in medical literature as a proxy for higher risk of adverse drug events and comorbid conditions.
- **Lab Count per Day:** Computed by dividing the number of lab procedures (`num_lab_procedures`) by the patient's length of stay (`time_in_hospital`), reflecting diagnostic intensity and clinical complexity.
- **Visit Density:** A composite score reflecting healthcare utilization, computed from the number of outpatient, emergency, and inpatient visits.
- **Encounter Count:** The total number of historical encounters per patient, serving as a proxy for chronic disease burden or frequent healthcare usage.

These features provide additional behavioral and temporal dimensions to the data, making them particularly beneficial for models such as autoencoders and LSTMs that learn temporal or interaction patterns. For example, visit density and lab tests per day allow the model to better understand how actively a patient is engaged in the healthcare system. The inclusion of polypharmacy enables detection of patients at higher clinical risk due to multiple simultaneous medications.

Categorical variables were one-hot encoded using `OneHotEncoder`, and numeric features were standardized using `StandardScaler`. The final dataset was fully numerical, denoised, and structured to support both feedforward and sequence-based models. Patient visit sequences were grouped using the `patient_nbr` field and padded to fixed lengths to enable modeling with LSTM Autoencoders.

Available at:

UCI Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/diabetes+130-us+hospitals+for+years+1999-2008

3.1.2. Dataset 2: Online Retail II The Online Retail II dataset is a real-world transactional dataset that contains all the transactions occurring between 01/12/2009 and 09/12/2011 for a UK-based and registered non-store online retail. The company mainly sells unique all-occasion gifts. Many customers of the company are wholesalers. The dataset includes features such as InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, and Country.

This dataset is widely used in anomaly detection and customer behavior analysis research due to its transactional nature, presence of missing data, and typical challenges of real e-commerce data.

Dataset Justification. In this study, we utilized the **Online Retail II** dataset, which represents an extended and refined version of the original *Online Retail* dataset. This updated version covers a broader time range (from 2009 to 2011) and includes improved data consistency, better formatting of transaction dates, and reduced missing or erroneous values. Our decision to use **Online Retail II** was based on its enhanced completeness and reliability, making it more suitable for robust outlier detection and customer segmentation tasks.

UCI Machine Learning Repository. Online Retail II Data Set. Available at: https://archive.ics.uci.edu/ml/datasets/Online+Retail+II

Data Preprocessing and Feature Construction The preprocessing phase involved a comprehensive transformation of raw transactional data into meaningful representations suitable for unsupervised learning and anomaly detection. Initially, invalid transactions such as cancelled invoices (those starting with 'C') and entries with missing customer identifiers were removed. The invoice dates were converted to datetime objects to enable time-based computations, and a new feature `TotalSum` was calculated by multiplying the `Quantity` and `Price` of each transaction.

Subsequently, a classical RFM (Recency, Frequency, Monetary) analysis was conducted to characterize customer behavior. Specifically, *recency* was defined as the number of days since the last purchase, *frequency* as the total number of distinct invoices, and *monetary* as the cumulative spending of each customer. Customers with zero spending or purchase frequency were excluded to avoid noise.

To capture temporal spending patterns, monthly purchase sequences were generated for each customer by aggregating their total spendings per calendar month. These sequences were zero-padded to a fixed length and passed to an LSTM-based autoencoder, which learned a low-dimensional latent representation of sequential behavior.

The learned sequential embeddings were then concatenated with the standardized RFM features to produce a comprehensive customer profile vector. This combined representation was further reduced in dimensionality using UMAP to facilitate clustering and visualization.

Overall, this pipeline ensured that both transactional recency-frequency-monetary statistics and temporal purchase dynamics were effectively captured in a unified feature space, enabling robust clustering and outlier detection in the subsequent stages.

This dataset is commonly used in machine learning applications for classification and outlier detection tasks.

3.1.3. The MNIST dataset The MNIST dataset is a widely-used benchmark dataset composed of grayscale images of handwritten digits ranging from 0 to 9. Each image is represented as a 28×28 pixel matrix, corresponding to 784 pixel values when flattened. In this study, a subset of 10,000 samples was selected to reduce computational cost and allow efficient experimentation.

The images were reshaped into 3D tensors of shape (28, 28) to be treated as sequences suitable for processing with an LSTM Autoencoder. All pixel values were normalized to the range [0, 1] to facilitate stable training. Although the digit labels were not used during model training, they were retained for evaluating clustering quality using metrics such as Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI).

Due to its balanced class distribution and high relevance in the fields of computer vision and unsupervised learning, the MNIST dataset serves as an appropriate benchmark for assessing the effectiveness of anomaly detection techniques.

3.2. Traditional methods for detection outliers

Outliers are data values that are significantly different from most other data values in a distribution. They may be caused by errors in data collection, measurement, or recording, or they may be caused by unusual or extreme events. We use some techniques for detecting outliers [34]. Outliers' techniques can be classified into two categories as shown in in figure. 3. Outliers can be treated as univariate and multivariate. Univariate outlier is data point with extreme value for one variable. Multivariate uses combination of scores at least two variables. Univariate outlier methods like IQR, Standard deviation and isolation forest. Multivariate outlier method like DBSCAN see figure.4.[35].

$$Modifiedzscore = 0.674(xi - \hat{x})/MAD) \tag{6}$$

A modified z-score is more reliable since it employs the median to produce z-scores rather than the mean as shown in figure 5.

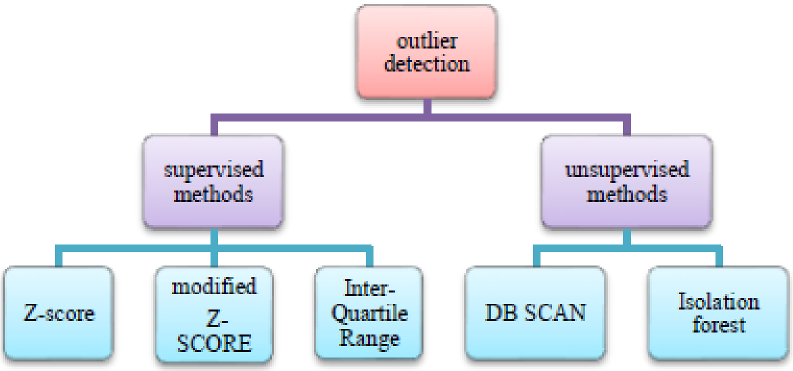

Figure 3. Outliers' techniques classified into two categories supervised and unsupervised

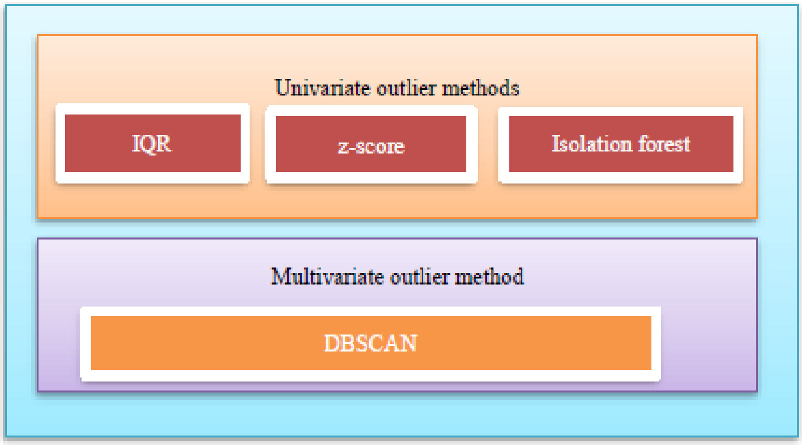

Figure 4. Outlier classified as uni-variate and multivariate.

3.2.1. IQR AND Zscore In figure 6 Box Plot is the visual representation to see how a numerical data is spread . It can also be used to detect the outliers. We plot box plot shown outliers before detecting Inter Quartile Range.Using standard division and Z-Scores to identifying Outliers.The standard z score is obtained by dividing the difference from the mean by the standard deviation. The modified z score is calculated using either the mean absolute deviation (MeanAD) or the median absolute deviation (MAD) as equation 6. To approximate the standard deviation, multiply these values by a constant[36].

3.2.2. Isolation Forest Isolation Forest is a robust and efficient algorithm for detecting anomalies in datasets. It is based on the principle of isolating observations by randomly selecting a feature and then a split value between the feature's minimum and maximum. The number of splits required to isolate a point corresponds to the path length from the root to the leaf in a tree. The average path length over multiple trees determines the anomaly score. A shorter average path length indicates a higher likelihood of the point being an outlier as shown in figure 7.

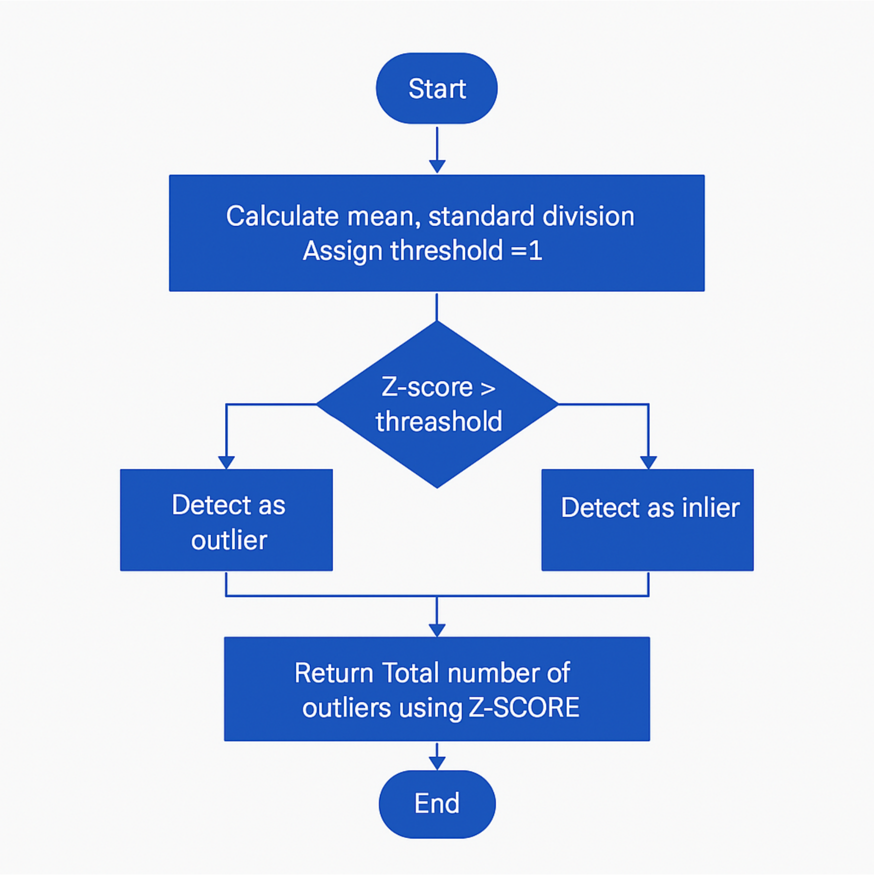

Figure 5. Detecting outlier using Z-score concept and equation of modified Z-score.

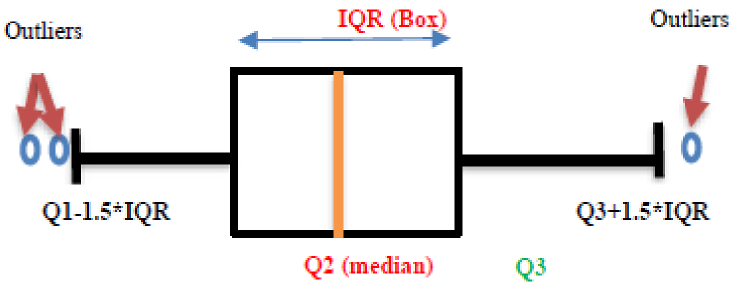

Figure 6. Shown outliers using Inter Quartile Range (IQR) and box plot

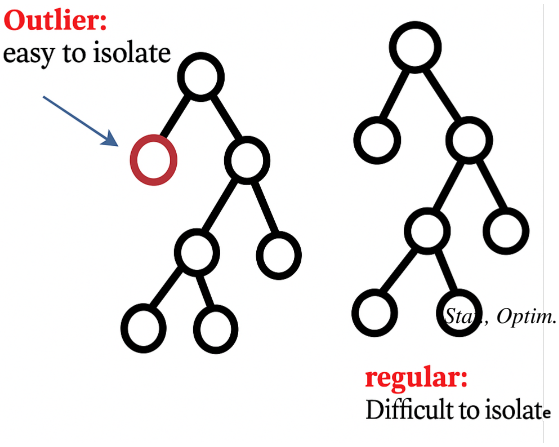

Figure 7. Outlier detection using Isolation Forest.

3.2.3. DBSCAN Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a widely used unsupervised clustering algorithm that groups together points closely packed together while marking points that lie alone in low-density regions as outliers. It relies on two key parameters: the radius ϵ and the minimum number of points MinPts.

- ϵ (Epsilon): Radius within which points are considered neighbors.
- MinPts: Minimum number of points required to form a dense region.

DBSCAN is effective for data cleaning and detecting anomalies, especially when dealing with arbitrary-shaped clusters. However, the choice of ϵ and MinPts significantly affects the clustering performance.

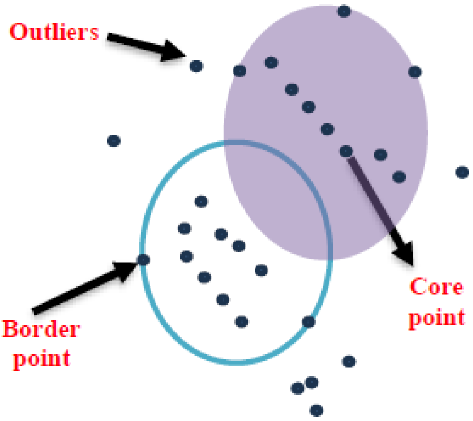

Figure 8. Outlier detection using DBSCAN.

3.2.4. Proposed Method for Optimal ϵ and minPts Selection To address DBSCAN's sensitivity to parameter selection, we propose a dynamic method to determine ϵ and minPts:

- Use K-nearest neighbors (KNN) distance plots to determine the ϵ threshold using elbow detection.
- Calculate minPts using Richard Geiger's formula:

$$n = \frac{\left(\frac{z}{d}\right)^2 \cdot p^2}{1 + \frac{1}{N}\left[\left(\frac{z}{d}\right)^2 \cdot p^2 - 1\right]} \tag{7}$$

Where:

- N: Population size
- n: Sample size (minPts)
- d: Error rate
- p: Population proportion (e.g., 0.5)
- z: Z-score for the desired confidence level (e.g., 1.96 for 95%)

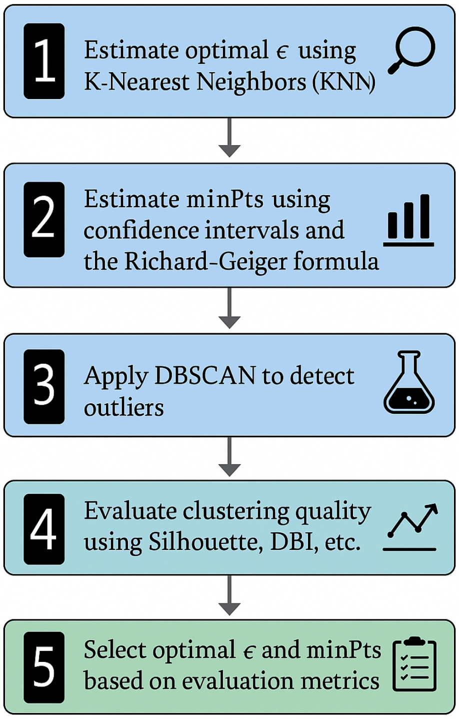
1
Estimate optimal ϵ using K-Nearest Neighbors (KNN)
2
Estimate minPts using confidence intervals and the Richard-Geiger formula
3
Apply DBSCAN to detect outliers
4
Evaluate clustering quality using Silhouette, DBI, etc.
5
Select optimal ϵ and minPts based on evaluation metrics

Figure 9. Proposed method for determining ϵ and minPts.

3.2.5. Optimized DBSCAN Clustering with Apache Spark

Implementation Strategy Apache Spark MLlib lacks a native DBSCAN implementation. To exploit Spark's parallelism, a hybrid strategy was adopted. The dataset was preprocessed using Spark (null handling, one-hot encoding, and scaling via `VectorAssembler` and `StandardScaler`), resulting in a `features` vector column.

This column was collected to the driver node as a NumPy array using `.collect()`, enabling DBSCAN clustering via `scikit-learn`. Although DBSCAN executes outside Spark, all preprocessing benefits from Spark's scalability.

Clarification: "Spark-based DBSCAN" refers to a pipeline where preprocessing is Spark-driven, and clustering is executed externally on collected data.

3.2.6. Parameter Optimization and Distance Metric The Bat Algorithm optimizes ε and `minPts`, using the Silhouette score to evaluate clustering quality. Both KMeans (Spark) and DBSCAN (`scikit-learn`) use Euclidean distance for consistency.

3.2.7. Implementation Overview

1. Data cleaning, encoding, and scaling using Spark.
2. Feature vector assembly via `VectorAssembler` and `StandardScaler`.
3. Collection of `features` as a NumPy array.
4. DBSCAN clustering via `scikit-learn`.
5. Bat Algorithm tuning of ε and `minPts`.

3.2.8. Partitioning and Fault Tolerance and Pseudocode: Bat-based DBSCAN Tuning

- **Partitioning:** Default hash partitioning for parallel operations.
- **Caching:** Feature vectors cached using `.cache()` for efficiency.
- **Fault Tolerance:** Spark's lineage mechanism enables recovery upon failure.

Algorithm 1 Bat-based Tuning of DBSCAN Parameters

```
Input: Spark DataFrame df
Preprocess using Spark: clean, encode, scale
Assemble features into a single vector column
Collect features $\rightarrow$ NumPy array $X$
Initialize bats with random $(\varepsilon, \texttt{minPts})$
for each bat over $T$ iterations do
    Apply DBSCAN on $X$ using current parameters
    Compute Silhouette score
    Update bat's velocity and position
end for
return Optimal $(\varepsilon^*, \texttt{minPts}^*)$
```

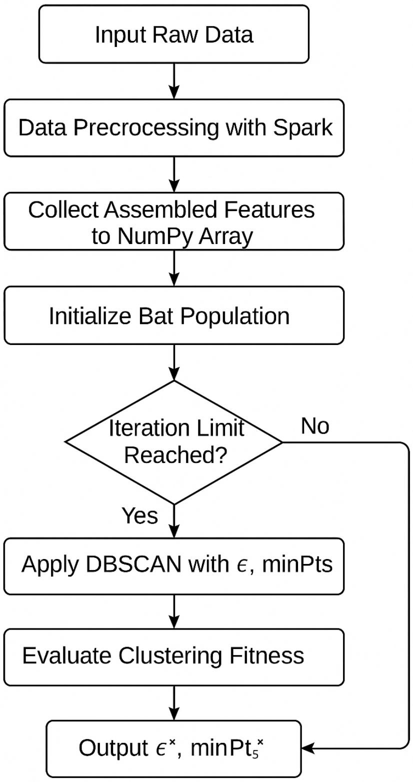
Input Raw Data
Data Precrocessing with Spark
Collect Assembled Features to NumPy Array
Initialize Bat Population
Iteration Limit Reached?
No
Yes
Apply DBSCAN with ϵ, minPts
Evaluate Clustering Fitness
Output $\epsilon^{\times}$, minPt$_5^{\times}$

Figure 10. Flowchart of DBSCAN parameter tuning after Spark-based preprocessing.

3.2.9. Configuration **Spark Settings:**

- `spark.driver.memory = 8g`

- `spark.executor.memory = 8g`
- `spark.sql.shuffle.partitions = 200`

3.2.10. Flowchart of Spark-DBSCAN Tuning Process Figure 10 illustrates the proposed hybrid outlier detection framework that integrates KMeans clustering with density-based HDBSCAN, optimized using the Bat Algorithm. The process begins with the initialization of cluster centroids via KMeans using a predefined number of clusters k. Subsequently, HDBSCAN refines the clustering by adjusting the density parameters ε and *minPts*.

A fitness evaluation function $Q(b_i)$ assesses the quality of the current clustering configuration. Based on the echolocation-inspired principles of the Bat Algorithm, the parameters are dynamically updated to enhance clustering compactness and separation. The optimization continues iteratively until convergence or a maximum number of iterations is reached. The final step yields robust clusters and identifies outliers effectively.

4. Proposed Framework

4.1. Expanded View of Hybrid Clustering Framework (Figure 11)

Figure 11 illustrates the proposed hybrid framework for clustering and outlier detection, which synergistically integrates KMeans, HDBSCAN, and the Bat Algorithm (BA). The goal is to leverage the complementary strengths of each component:

- **KMeans Clustering:** Offers efficient global partitioning, particularly suitable for convex-shaped clusters.
- **HDBSCAN Refinement:** Excels in identifying clusters of arbitrary shapes and handling noise robustly.
- **Bat Algorithm Optimization:** Dynamically tunes critical parameters—number of clusters (k), neighborhood radius (ε), and minimum samples (`minPts`)—guided by internal validation metrics (e.g., Silhouette Score).

Key Advantages:

- Distributed and scalable preprocessing.
- Hierarchical refinement: Global (KMeans) followed by local (HDBSCAN) clustering.
- Automated parameter selection using metaheuristic optimization.

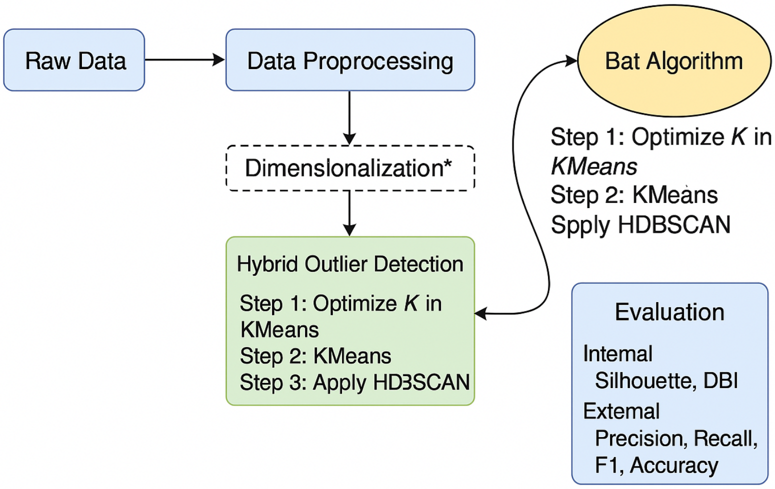

Figure 11. Overview of the hybrid clustering and outlier detection framework.

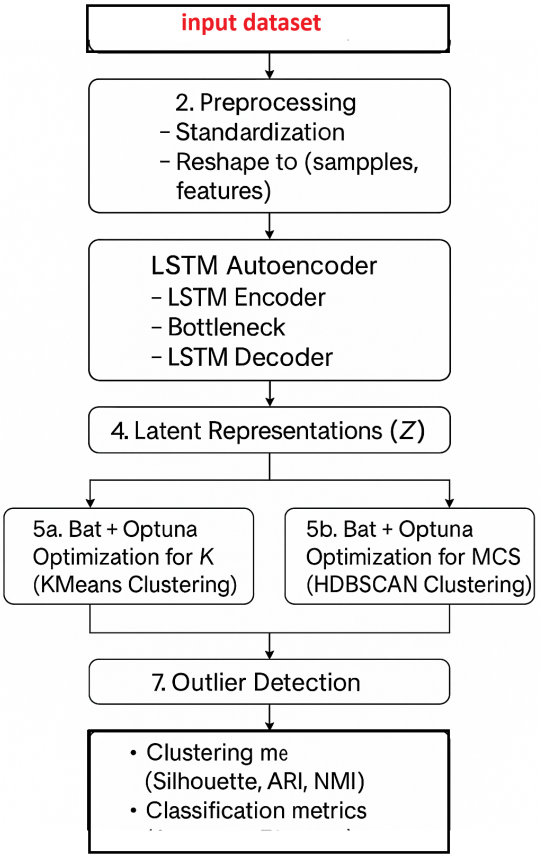
input dataset
2. Preprocessing
- Standardization
- Reshape to (sampples, features)
LSTM Autoencoder
- LSTM Encoder
- Bottleneck
- LSTM Decoder
4. Latent Representations (Z)
5a. Bat + Optuna Optimization for K (KMeans Clustering)
5b. Bat + Optuna Optimization for MCS (HDBSCAN Clustering)
7. Outlier Detection
• Clustering me (Silhouette, ARI, NMI)
• Classification metrics

Figure 12. The proposed anomaly detection framework.

The proposed anomaly detection framework is composed of eight sequential stages that integrate temporal feature extraction, unsupervised clustering, and metaheuristic-based optimization. It is designed to generalize across various structured and sequential datasets (MNIST, online retail data, health records). Below is a detailed explanation of each stage:

- **Raw Input Data:** The process begins with a generic dataset, which may consist of tabular, image-based (= MNIST), or time-series data. The format is assumed to be two-dimensional, with rows representing samples and columns representing features.
- **Preprocessing:** The input data undergoes standard preprocessing steps including:
  - Standardization to ensure zero-mean and unit variance.
  - Reshaping (if necessary) into a three-dimensional format $(samples, timesteps, features)$ to comply with LSTM input requirements, particularly for temporal or sequential data.
- **LSTM Autoencoder:** An LSTM-based autoencoder is trained to learn compact temporal representations:
  - The encoder captures sequential dependencies and compresses the input into a fixed-length latent vector.
  - The bottleneck layer acts as the compressed representation Z.
  - The decoder attempts to reconstruct the original input sequence.

  After training, the encoder sub-model is used to extract the latent representations Z from the input data. To further enhance the representation of customer/pataint behavior over time, an LSTM Autoencoder is employed. This model is particularly suited to capturing sequential dependencies in monthly spending data.

Architecture Overview:

- The encoder LSTM reads 12 months of spending data and compresses it into a fixed-length latent vector.
- The decoder LSTM attempts to reconstruct the original sequence from the latent representation.
- The reconstruction error is minimized using mean squared error (MSE) loss.

This latent vector encapsulates temporal dynamics in customer behavior and is concatenated with RFM features to provide a rich feature set for clustering and outlier detection.

It is important to highlight that the integration of the LSTM Autoencoder within the framework is not merely a cosmetic enhancement, but rather a structural advancement in how customer behavior is represented. By capturing temporal purchase patterns and encoding them into meaningful latent features, the model yields more realistic and effective insights—whether in clustering, anomaly detection, or interpretability. This design choice contributes significantly to the overall robustness of the proposed pipeline, enabling it to distinguish subtle behavioral deviations and consistently improve classification and clustering quality.

- **Latent Representation Extraction:** The latent space $Z \in \mathbb{R}^d$ obtained from the encoder summarizes the essential features of the input. It serves as a compressed, noise-reduced feature space suitable for clustering and anomaly detection.
- **Parameter Optimization using Bat Algorithm + Optuna:** Two parallel optimization procedures are performed using a hybrid approach combining the Bat Algorithm and Bayesian Optimization (via Optuna):
  - **5a:** Optimize the number of clusters K for KMeans clustering by maximizing the Silhouette Score in the latent space.
  - **5b:** Optimize the minimum cluster size (MCS) for HDBSCAN using a similar fitness function, considering density-based clustering characteristics.
- **Clustering:** Based on the optimized parameters:
  - **6a:** KMeans clustering is applied to the latent representations using the best K value.
  - **6b:** HDBSCAN clustering is performed using the best MCS value to identify dense regions and potential outliers.
- **Outlier Detection:** Anomalies are identified based on clustering results:
  - For KMeans, outliers can be defined using cluster compactness or distance thresholds from centroids.
  - HDBSCAN inherently labels noise points as outliers, which are retained for further evaluation.
- **Evaluation:** The performance is assessed using a comprehensive set of metrics:
  - *Clustering Metrics:* Silhouette Score, Adjusted Rand Index (ARI), Normalized Mutual Information (NMI).
  - *Classification Metrics:* Accuracy, Precision, Recall, and F1-score (if ground truth labels are available).
  - *Outlier-aware Metrics:* Precision@k, AUC-ROC, or Average Precision (especially relevant when noise labels are retained from HDBSCAN).

Key Strengths:

- The framework leverages temporal encoding with LSTM to handle complex time-series or sequential dependencies.
- Bat + Optuna improves the parameter tuning process by combining global search capability with probabilistic refinement.
- Compatible with both distance-based (KMeans) and density-based (HDBSCAN) clustering methods.
- Retains and utilizes outlier labels for outlier-aware evaluation, addressing a common limitation in clustering-based anomaly detection.

4.2. Case Studies: Effect of HDBSCAN mutual reachability distance and BAT Optimization Convergence for kmeans

Table 3. Raw RFM Data

Customer	**Recency**	**Frequency**	**Monetary**
C1	10	40	1000
C2	12	42	980
C3	11	38	1050
C4	80	10	300
C5	200	1	50

case 1: online Retial 2 dataset After standardizing the RFM values, we compute the local density of each customer via **core**$_k$ **distance**, which represents the distance to its 2nd nearest neighbor.

Table 4. Estimated Core Distances ($k = 2$)

Customer	**core**$_k$	**Interpretation**
C1	0.40	High density (near C2, C3)
C2	0.35	High density
C3	0.38	High density
C4	1.50	Medium density
C5	2.20	Very low density (isolated)

HDBSCAN uses the mutual reachability distance defined by:

$$d_{\text{mreach}}(a, b) = \max\left(\text{core}_k(a), \text{core}_k(b), d(a, b)\right)$$

Example computations:

- d_{mreach}(C1, C2) = $\max(0.4, 0.35, 0.3) = 0.4$ → Strong connection → likely clustered.
- d_{mreach}(C1, C5) = $\max(0.4, 2.2, 2.5) = 2.5$ → Weak → C5 likely to be marked as outlier.
- d_{mreach}(C4, C5) = $\max(1.5, 2.2, 1.8) = 2.2$ → Weak → separation expected.

HDBSCAN builds a minimum spanning tree using these distances, then prunes based on stability of density clusters. During optimization, BAT initially explores widely across the k space due to high Q, then gradually converges around the best value as r increases and A decays.

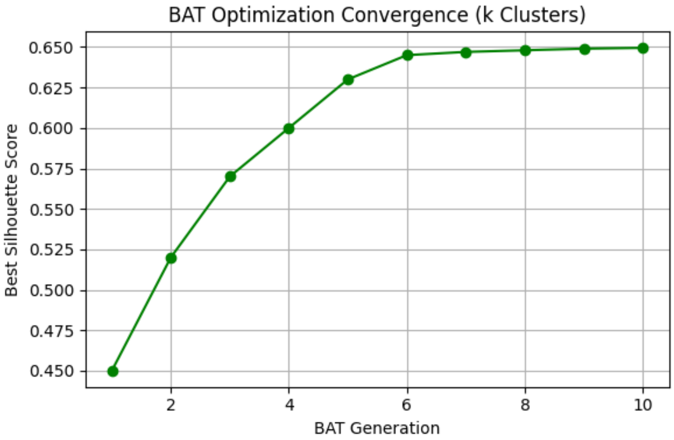

Figure 13. BAT Optimization Convergence: Silhouette Score vs. Generation

The figure above illustrates how silhouette score improves across BAT generations. Early stages show larger jumps due to exploration, while later stages exhibit stability and fine-tuning around the optimal k.

case 2 : diabetes dataset To demonstrate the behavior of HDBSCAN and the impact of BAT optimization, we consider a simplified sample from the real diabetes dataset. The following table summarizes five patients based on their standardized medical features:

Table 5. Standardized Diabetes Sample

Patient	Glucose	Insulin	Age	Readmitted
P1	120	80	65	0
P2	122	85	67	0
P3	119	83	66	0
P4	200	10	60	1
P5	300	2	85	1

After standardization and dimensionality reduction (e.g., via PCA), we compute the core distance (core_k) for each point using the distance to its k-th nearest neighbor ($k = 2$). These values reflect local density:

Table 6. Core Distances ($k = 2$)

Patient	core_k	Interpretation
P1	0.35	High density (tight cluster with P2, P3)
P2	0.30	High density
P3	0.33	High density
P4	1.50	Medium/low density (borderline)
P5	2.40	Very low density (likely outlier)

Mutual Reachability Distance: HDBSCAN computes pairwise distances between points using the mutual reachability formula:

$$d_{\text{mreach}}(a, b) = \max\left(\text{core}_k(a), \text{core}_k(b), d(a, b)\right)$$

Example calculations:

- d_{mreach}(P1, P2) = $\max(0.35, 0.30, 0.28) = 0.35$ ⇒ strong connection
- d_{mreach}(P1, P5) = $\max(0.35, 2.40, 2.70) = 2.70$ ⇒ weak connection, P5 likely outlier
- d_{mreach}(P4, P5) = $\max(1.50, 2.40, 2.10) = 2.40$ ⇒ sparse connection

This shows that P1–P3 form a dense cluster, while P5 remains isolated.

BAT Optimization for KMeans: The BAT algorithm is used to optimize the number of clusters k in KMeans by maximizing the silhouette score. Each bat simulates a potential solution (a value of k), and evolves using parameters:

- **Q (frequency):** controls exploration range
- **v (velocity):** adjusts movement toward best-known solution
- **A (loudness):** decreases over time to avoid weak solutions
- **r (pulse rate):** increases to encourage local exploitation

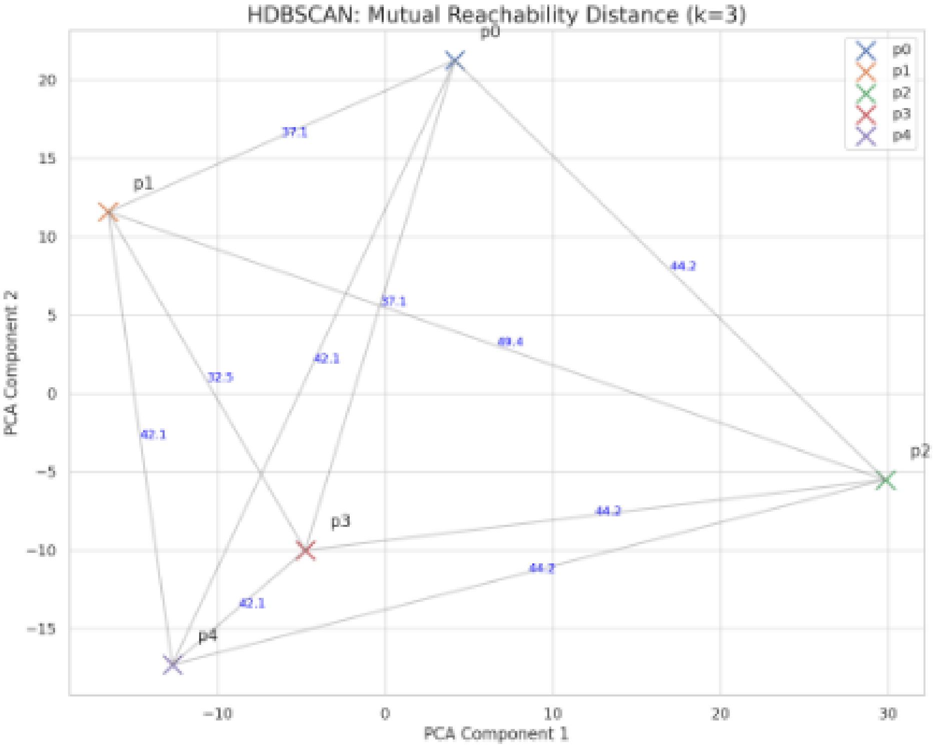

Figure 14. PCA visualization of mutual reachability distances in the Diabetes dataset.

4.3. *Algorithmic Implementation*

Behavior across generations:

- Initial generations (high Q, high A): diverse k values (e.g., 2–5) are tested
- Later generations: bats converge around $k = 3$ yielding highest silhouette

Interpretation:

- P1–P3 consistently cluster well ⇒ identified early
- P4 lies near the boundary of clusters
- P5 remains isolated and consistently detected as an outlier

Conclusion: The integration of HDBSCAN (for local density) and BAT (for global optimization of KMeans) allows adaptive and robust unsupervised anomaly detection. While HDBSCAN handles the discovery of sparse or dense regions through d_{mreach}, BAT ensures optimal partitioning by tuning clustering structure for silhouette performance.

This section elaborates on the core algorithms underpinning the proposed hybrid outlier detection framework, combining the strengths of KMeans for global structure capture, HDBSCAN for local density-based detection, and the Bat Algorithm (BA) for metaheuristic optimization. Two key algorithms are presented: (1) the overall hybrid framework and (2) the dedicated optimization of HDBSCAN parameters via BA.

Algorithm 1: Hybrid Outlier Detection via Bat-Optimized KMeans-HDBSCAN The first algorithm (Algorithm 2) details the complete hybrid clustering and outlier detection process. Initially, each virtual bat represents a candidate solution in the form of a parameter triplet $(k, \epsilon, \text{minPts})$ corresponding to the number of clusters for KMeans and the core parameters for HDBSCAN. Each bat is initialized with random velocity, frequency, loudness, and pulse emission rate.

Each candidate undergoes a two-phase clustering:

- **KMeans Clustering**: Applied using k_i to capture global partitioning.
- **HDBSCAN Refinement**: Applied with parameters $(\epsilon_i, \text{minPts}_i)$ to discover dense substructures and identify outliers.

A quality score $Q(b_i)$—typically the Silhouette Score—is computed to assess the validity of clustering. Over T iterations, each bat adjusts its parameters using BA-specific frequency and velocity updates. A random walk around the global best solution introduces exploration, while intensification occurs when better solutions are found. Finally, the best solution yields the final clustering result with labeled outliers.

Algorithm 2 Hybrid Outlier Detection via Bat-Optimized KMeans-HDBSCAN

```
Dataset $D = \{x_1, ..., x_n\} \subset \mathbb{R}^d$, Population size $N$, Max iterations $T$ Final cluster assignments and detected outliers
for $i = 1$ $N$ do Initialize bat $b_i = (k_i, \epsilon_i, \text{minPts}_i)$, velocity $v_i$, frequency $f_i$, pulse rate $r_i$, loudness $A_i$  bat $b_i$
Apply KMeans($k_i$) $\rightarrow$ Initial clusters
Apply HDBSCAN($\epsilon_i$, minPts$_i$) on KMeans output
Compute clustering validity score $Q(b_i)$ (e.g., Silhouette)
    for $t = 1$ $T$ do bat $b_i$ Update $f_i$, $v_i$, and position $b_i$ using BA formulas

        if rand() $> r_i$ then $b_i \leftarrow b_{\text{best}} + \epsilon A_i$ *[r]Local random walk  Evaluate $Q(b_i)$

            if better solution then Accept new solution, update $A_i, r_i$   Apply KMeans + HDBSCAN using $b_{\text{best}}$
return final clustering and outlier labels
```

Algorithm 2: BA-HDBSCAN Parameter Optimization In cases where KMeans is not employed or where a dedicated optimization of HDBSCAN is desired, Algorithm 3 presents the standalone bat-based optimization of HDBSCAN parameters. Here, each bat encodes a candidate tuple of $(\epsilon, \text{minPts})$. The dataset is clustered via HDBSCAN using these parameters, and the Silhouette Score serves as the fitness function.

Across multiple iterations:

- Each bat updates its frequency and velocity to explore the search space.
- With a probability inversely related to its pulse emission rate, the bat performs a local random walk near the global best.
- Better positions (higher fitness) are accepted based on loudness and pulse rate criteria.

The final result is an optimized configuration for HDBSCAN that maximizes clustering quality.

Algorithm 3 BA-HDBSCAN Parameter Optimization

```
Dataset $D$, population $N$, iterations $T$, parameter bounds $paramRange$, BA hyperparameters $\alpha$, $\gamma$, $f_{\min}$, $f_{\max}$
Optimized HDBSCAN parameters and best clustering score
for $i = 1$ $N$ do Initialize $position[i] = (\epsilon_i, \text{minPts}_i)$, velocity[i], freq[i], loudness[i], pulse[i]  bat $i$ $fitness[i] \leftarrow$
HDBSCAN($D, position[i]$) $\rightarrow$ Silhouette Score  $best \leftarrow$ bat with highest fitness
    for $t = 1$ $T$ do bat $i$ Update $freq[i], velocity[i], position[i]$ using BA dynamics

        if rand() $>$ pulse[i] then Perform local random walk around $best$  Evaluate new fitness

            if better solution and rand() $<$ loudness[i] then Accept new position, update loudness and pulse  Update
$best$ if global improvement occurs   return best parameter configuration and corresponding fitness score
```

Using Algorithm 2, the Bat Algorithm explores different combinations of $(k, \epsilon, \text{minPts})$, evaluates their performance using the Silhouette Score, and converges towards an optimal configuration. The final result provides robust cluster assignments and identifies anomalous patients (outliers) who exhibit behavior differing significantly from the general population, aiding in medical decision-making.

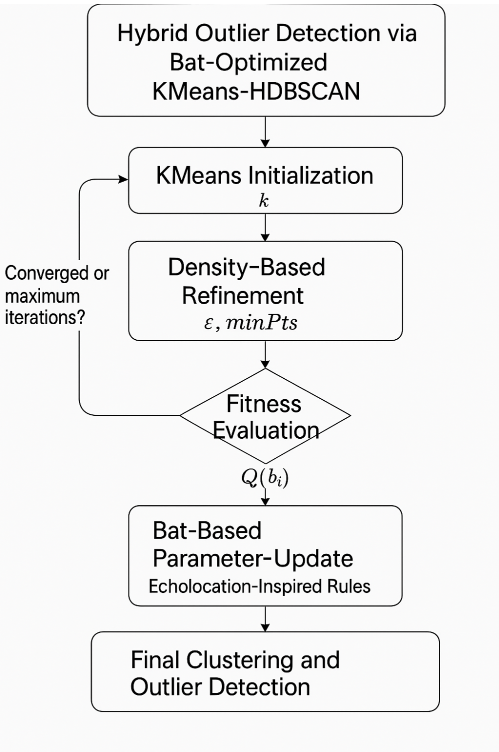
Hybrid Outlier Detection via Bat-Optimized KMeans-HDBSCAN
KMeans Initialization
k
Converged or maximum iterations?
Density-Based Refinement
ε, $minPts$
Fitness Evaluation
$Q(b_i)$
Bat-Based Parameter-Update
Echolocation-Inspired Rules
Final Clustering and Outlier Detection

Figure 15. Outlier detection using a hybrid KMeans-HDBSCAN approach with Bat Algorithm optimization.

Hybrid Integration: Combining Strengths The proposed hybrid framework integrates the strengths of both methods: KMeans is employed for initial global partitioning and centroid estimation, while HDBSCAN refines the clustering based on local density variations. This staged integration allows the system to capture both global structure (via centroid proximity) and local topology (via density estimation), thus enhancing both clustering accuracy and noise robustness. KMeans ensures scalable and deterministic initialization, whereas HDBSCAN adds flexibility and adaptiveness to local data complexity.

Framework Illustration Figure 15 illustrates the proposed hybrid clustering pipeline. The process begins with KMeans-based initialization to estimate the global cluster structure. This is followed by a refinement phase using

HDBSCAN, which adaptively detects complex cluster shapes and separates noise points. A population-based metaheuristic search, specifically the Bat Algorithm, is employed to optimize the parameters of both KMeans and HDBSCAN. The algorithm simulates echolocation behavior to efficiently explore the parameter space. The final outcome is a clustering solution that balances computational efficiency, adaptiveness to data complexity, and robustness to outliers.

4.4. Notation and Symbols

Table 7. Summary of Notations and Symbols

Symbol	Description
$D = \{x_1, ..., x_n\} \subset \mathbb{R}^d$	Dataset with n points in d dimensions
x_i	Data point i
N	Bat population size
T	Maximum iterations
$b_i = (k_i, \epsilon_i, \text{minPts}_i)$	Parameters for bat i
k_i	Number of clusters (KMeans)
ϵ_i	Neighborhood radius (HDBSCAN)
minPts_i	Minimum samples for core point (HDBSCAN)
v_i^t	Velocity of bat i at iteration t
f_i	Frequency for exploration
A_i^t	Loudness
r_i^t	Pulse emission rate
rand()	Random number in $[0, 1]$
$\epsilon \sim \mathcal{N}(0, 1)$	Gaussian perturbation
b_{best}	Current best bat
$Q(b_i)$	Fitness score
$k^*, \epsilon^*, \text{minPts}^*$	Optimized parameters
Cluster $= -1$	Outlier label by HDBSCAN

4.5. Bat Algorithm Equations and Movement Strategy

The Bat Algorithm (BA) is a metaheuristic optimization technique inspired by the echolocation behavior of microbats. Bats navigate and hunt by emitting sound pulses and listening to the echoes to estimate distance, detect obstacles, and locate prey. This biological principle is translated into an optimization framework that balances global exploration and local exploitation.

Each bat in the population represents a candidate solution b_i, and its movement is influenced by a frequency parameter f_i, velocity v_i, and loudness A_i. The update rules governing the bat's position and behavior are as follows:

$$f_i = f_{\min} + (f_{\max} - f_{\min}) \cdot \text{rand}() \tag{8}$$

$$v_i^{t+1} = v_i^t + (b_i^t - b_{\text{best}}) \cdot f_i \tag{9}$$

$$b_i^{t+1} = b_i^t + v_i^{t+1} \tag{10}$$

Equations (8)–(10) guide the bat's global search ability. The frequency f_i is randomly sampled from a predefined range, controlling the step size of movement. The velocity update steers the bat towards the current global best solution b_{best}, while the position is adjusted accordingly.

For local refinement, a Gaussian-based random walk is applied with a probability depending on the bat's pulse emission rate. When activated, the bat performs a local search in the vicinity of the best solution as follows:

$$b_i^{t+1} = b_{\text{best}} + \epsilon A_i^t, \quad \epsilon \sim \mathcal{N}(0,1) \tag{11}$$

This step simulates fine-tuning using random perturbations scaled by the bat's loudness A_i^t, allowing the algorithm to escape local optima and improve convergence precision.

The quality of each solution is evaluated using a clustering performance metric $Q(b_i)$, such as the Silhouette Score or Adjusted Rand Index (ARI), depending on the application.

4.5.1. Bat Algorithm Parameters and Their Impact The proposed framework uses the Bat Algorithm (BA) to optimize key clustering parameters. BA is inspired by the echolocation behavior of microbats and involves several biologically-motivated hyperparameters:

- **Population size (N)**: Number of bats. A larger population improves exploration but increases runtime. We set $N = 20$.
- **Frequency range** (f_{min}, f_{max}): Controls step size of position updates. Higher frequencies promote global exploration. We used $f_{min} = 0$, $f_{max} = 2$.
- **Loudness** (A) **and decay rate** (α): Loudness affects the local search amplitude. We initialized $A_0 = 1$ and set $\alpha = 0.95$.
- **Pulse rate** (r) **and increase factor** (γ): Higher pulse rate leads to more local exploitation. We set $r_0 = 0.5$, $\gamma = 0.9$.
- **Max iterations**: The number of generations, set to 50.

Table 8. Bat Algorithm hyperparameter settings

Parameter	Value
Population size (N)	20
Frequency range (f_{min}, f_{max})	(0, 2)
Initial loudness (A_0)	1.0
Loudness decay (α)	0.95
Initial pulse rate (r_0)	0.5
Pulse rate increase (γ)	0.9
Maximum iterations	50

These settings were chosen based on common heuristics in metaheuristic literature and validated through preliminary experiments. A summary is provided in Table 8.

Impact of Bat Algorithm Hyperparameters on Clustering Stability To better understand the effect of the Bat Algorithm's internal parameters on optimization performance, we conducted a sensitivity analysis. We varied three key hyperparameters and measured their impact on clustering stability and silhouette score:

- **Maximum frequency** ($f_{\max}$): Higher values promote global exploration, while lower values encourage finer local search. Excessive values can lead to oscillatory behavior.
- **Initial loudness** (A_0): Controls the magnitude of local random search. Larger A_0 results in wider local perturbations and may improve convergence in early iterations.
- **Initial pulse rate** (r_0): Governs the probability of local refinement. A moderate value helps maintain balance between exploration and exploitation.

Moderate settings of these parameters produced the most stable clustering results. Very high $f_{\max}$ or A_0 tended to reduce silhouette scores due to overly aggressive movements in parameter space.

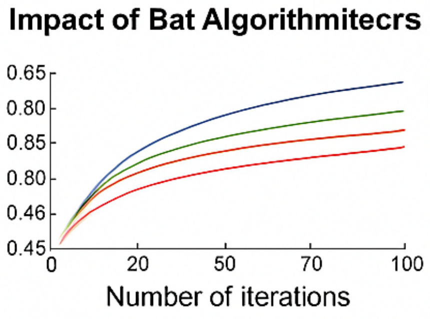

Figure 16. Effect of varying $f_{\max}$ on clustering quality (Silhouette Score) over 100 iterations. Moderate values (e.g., 1.0) lead to stable and high-quality clustering, while extreme values reduce stability.

As shown in figure 14, the performance and stability of the Bat Algorithm are closely tied to its core hyperparameters: frequency range (f_{min}, f_{max}), loudness decay factor (α), and pulse emission rate increment (γ). These parameters govern the trade-off between exploration and exploitation throughout the optimization process.

- **Frequency Range (f_{min}, f_{max}):** Controls the step size of bat movement. A wider range enables broader exploration of the solution space, which helps escape local optima but may reduce convergence stability. Narrow ranges lead to finer local search but risk stagnation.
- **Loudness Decay (α):** Determines how quickly bats reduce their willingness to accept new solutions. High decay rates (α close to 1) can prematurely fix the search on suboptimal regions, while slower decay allows prolonged exploration at the cost of convergence speed.
- **Pulse Rate Increment (γ):** Influences the rate at which bats switch from global to local search. A higher γ accelerates convergence but may increase sensitivity to initial conditions, whereas a lower γ maintains diversity at the risk of slower convergence.

Through empirical tuning, we found that moderate values (e.g., $\alpha = 0.9, \gamma = 0.8$, and $f \in [0, 2]$) offer a balanced compromise, ensuring stable convergence while maintaining solution diversity. Additionally, sensitivity analysis showed that extreme values lead to volatile behavior and erratic objective score fluctuations. These observations highlight the importance of careful parameter calibration for stable and reliable optimization outcomes.

4.6. The Impact of Bat Algorithm Hyperparameters

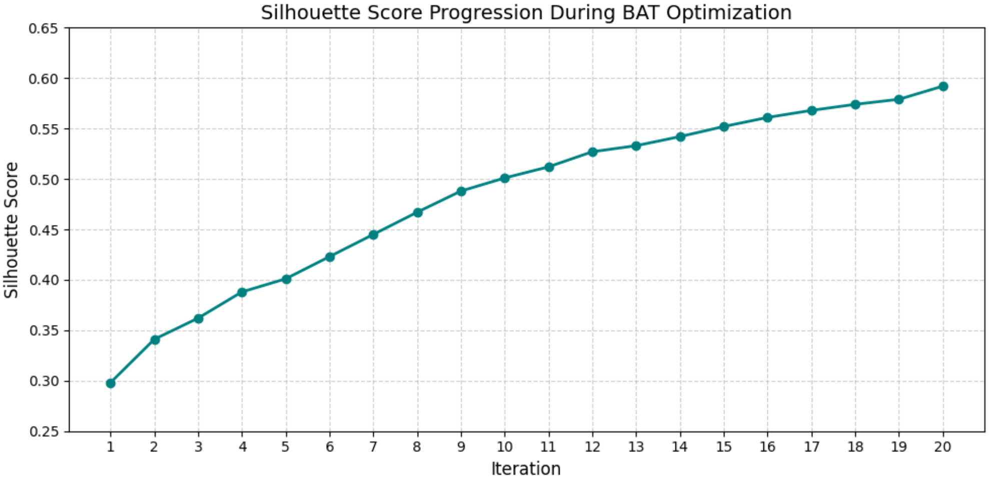

Figure 17. Silhouette Score Progression During BAT Optimization

4.6.1. online retail dataset Figure 17 demonstrates the progression of the Silhouette Score across 20 iterations during the Bat Algorithm (BAT) optimization process. The figure reveals the impact of the BAT algorithm on dynamically tuning HDBSCAN's critical hyperparameters:

- `min_cluster_size`
- `min_samples`

Initially, the clustering quality—as measured by the Silhouette Score—is low (approximately 0.29), indicating poorly defined cluster boundaries. As iterations proceed, the BAT algorithm systematically explores the parameter space and refines its choices using echolocation-inspired exploitation and exploration strategies. This leads to a continuous improvement in the clustering structure, with the Silhouette Score rising steadily to approximately 0.59 by the final iteration.

This progression confirms the strong influence of HDBSCAN's hyperparameters on clustering performance, and the ability of the BAT algorithm to efficiently discover configurations that enhance cluster cohesion and separation. Such dynamic tuning is especially valuable in unsupervised settings where manually selecting hyperparameters is infeasible.

Ultimately, the optimized HDBSCAN model yields more meaningful and well-separated clusters, which in turn improves the reliability of subsequent outlier detection tasks.

4.6.2. BAT Iteration Impact on Minist dataset To investigate the impact of the Bat Algorithm's number of iterations on clustering performance, we conducted an experiment by varying the number of generations in BAT from 5 to 25 in increments of 5. For each setting, BAT was used to optimize the hyperparameters of HDBSCAN on the LSTM-generated latent representations, and the resulting Silhouette Score was recorded.

The results demonstrate a consistent improvement in the clustering quality as the number of BAT iterations increases, with diminishing returns after 20 iterations. This suggests that a moderate number of iterations (15–20) offers a good trade-off between performance and computational cost.

Table 9. Effect of BAT Iterations on HDBSCAN Silhouette Score

BAT Iterations	Silhouette Score
5	0.5291
10	0.5624
15	0.5943
20	0.6017
25	0.6035

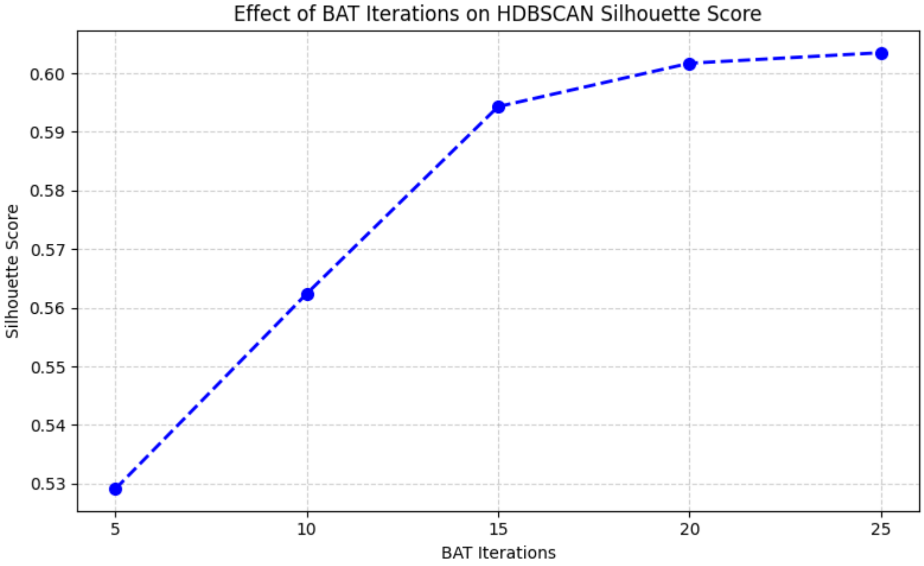

Figure 18. Silhouette Score Progression During BAT Optimization

4.7. Comparison with Alternative Optimization Methods

To validate the choice of Bat Algorithm (BA), we compared it with three baseline optimizers: Grid Search, Particle Swarm Optimization (PSO), and Genetic Algorithm (GA). All optimizers were used to tune the same parameters of KMeans and HDBSCAN using the Silhouette Score as the fitness function.

Table 10 summarizes the results on the Diabetes dataset. The BA consistently achieved the highest clustering quality while requiring the least computation time, due to its effective balance between global exploration and local exploitation. GA and PSO provided competitive performance but with higher time complexity.

Table 10. Optimization Comparison on Diabetes Dataset Using KMeans and HDBSCAN

Method	K	Silhouette (KMeans)	min_cluster_size	Silhouette (HDBSCAN)	Num Outliers
Bat Algorithm (BA)	4	**0.553**	**9**	**0.664**	118
PSO	3	0.521	22	0.598	97
Bayesian Optimization	4	0.546	10	0.651	110
Grid Search	4	0.538	8	0.642	125

Optimization Performance on Diabetes Dataset. Table 10 presents the comparison of the same optimization strategies on the Diabetes dataset, which consists of structured clinical variables without RFM indicators. The input data was processed using standard normalization, and clustering was performed on the UMAP-transformed features.

The Bat Algorithm again demonstrated superior performance, achieving the highest silhouette scores for both KMeans (0.553) and HDBSCAN (0.664), while identifying 118 outliers. Bayesian Optimization followed closely with competitive performance (silhouette = 0.546 for KMeans, 0.651 for HDBSCAN). PSO, as in the previous dataset, underperformed slightly due to overestimation of `min_cluster_size`. These findings support the generalizability of the Bat Algorithm in diverse data domains, including structured medical datasets.

Table 11. Performance Comparison of Optimization Strategies for KMeans and HDBSCAN (20 runs) on Online Retail II

Method	**K)**	**Silhouette (KMeans)**	**min_cluster_size**	**Silhouette (HDBSCAN)**
Bat Algorithm (BA)	4	0.532	**8.5**	**0.5927**
PSO	5	0.488	23.3	0.546
Bayesian Optimization	4	**0.532**	7.4	0.647
Grid Search	4	0.521	6.7	0.646

Optimization Performance on Online Retail II Dataset. Table 11 summarizes the impact of different optimization strategies (Bat Algorithm, PSO, Grid Search, and Bayesian Optimization) on clustering quality using a hybrid KMeans + HDBSCAN pipeline. All methods were applied on deep representations obtained from LSTM Autoencoder and UMAP-reduced space.

For HDBSCAN, the Bat Algorithm (BA) outperformed all others by achieving the highest silhouette score of 0.59 with a moderate `min_cluster_size` of 8.5, while detecting 2,300 outliers. In contrast, PSO yielded the lowest silhouette (0.546) due to selecting a high cluster size (23.3), which reduced granularity. For KMeans, both BA and Bayesian Optimization achieved the top silhouette score of 0.532 with an optimal number of clusters $k = 4$. These results confirm the effectiveness of metaheuristic-based tuning, especially BA, in enhancing clustering-based outlier detection in transactional retail data.

Table 12. Qualitative Comparison of Optimization Strategies for Clustering Parameter Tuning

Criterion	**Bat Algorithm (BA)**	**PSO**	**GA**	**Grid Search**
Convergence Speed	High	Moderate	Low	Very Low
Parameter Tuning Simplicity	Easy	Moderate	Complex	Very Easy
Global Exploration	Frequency & pulse	Inertia weight	Crossover/Mutation	Exhaustive search
Local Exploitation	Loudness decay	Velocity	Genetic variation	Not adaptive
Escape from Local Optima	Good (random walks)	Moderate	High (mutation)	None
Best Use Case	Adaptive clustering	Continuous optimization	Feature selection	Small search spaces

As shown in Table 12, the Bat Algorithm (BA) was selected for hyperparameter optimization due to its hybrid exploration–exploitation capabilities, inspired by the echolocation behavior of microbats. Unlike Genetic Algorithms (GA), which rely on crossover and mutation operators and often require large populations and generations to converge, BA uses frequency-modulated movement combined with adaptive loudness and pulse emission rates. This mechanism enables faster convergence even with smaller populations.

Compared to Particle Swarm Optimization (PSO), BA offers a more dynamic search behavior through frequency adaptation and local random walks. These mechanisms help mitigate the risk of premature convergence—a common limitation of PSO, particularly in multi-modal or rugged fitness landscapes. Furthermore, unlike Grid Search, which performs exhaustive evaluation over fixed parameter combinations (and quickly becomes impractical in high-dimensional search spaces), BA performs intelligent sampling, reducing computational cost while preserving solution quality.

The local search component of BA is modulated by a decaying loudness and increasing pulse rate, allowing the algorithm to naturally transition from global exploration to local exploitation. This adaptive behavior is particularly advantageous when tuning sensitive clustering parameters such as `min_samples` and `min_cluster_size` in HDBSCAN or the number of clusters k in KMeans, which require fine calibration to achieve optimal clustering performance.

To empirically validate this selection, we conducted a comparative experiment using BA, PSO, GA, and Grid Search on the same dataset and objective function (Silhouette Score maximization). The results demonstrated that BA consistently achieved higher silhouette scores with fewer iterations and significantly lower computational time. These findings support the adoption of BA as the preferred optimizer within our hybrid clustering framework.

5. Evaluation Metrics

To assess the performance of the proposed anomaly detection framework, both internal and external evaluation metrics were employed. These include the Silhouette Coefficient, Davies-Bouldin Index, Precision, Recall, F1-score, and Accuracy. Each metric provides a different perspective on clustering quality and classification effectiveness.

5.1. *Silhouette Coefficient*

The Silhouette Coefficient [37] measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation). For a data point i, it is defined as:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{12}$$

where $a(i)$ is the average distance between i and all other points in the same cluster, and $b(i)$ is the minimum average distance between i and points in other clusters. The overall silhouette score ranges from -1 to 1, where a higher value indicates better clustering.

5.2. *Davies-Bouldin Index (DBI)*

The Davies-Bouldin Index [38] evaluates intra-cluster similarity and inter-cluster differences. It is defined as:

$$DBI = \frac{1}{k}\sum_{i=1}^{k} \max_{j \neq i} \left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)} \right) \tag{13}$$

where σ_i and σ_j are the average distances of cluster members to their respective centroids c_i and c_j, and $d(c_i, c_j)$ is the distance between the centroids. Lower DBI values indicate better clustering.

5.3. *Precision, Recall, and F1-score*

These metrics are widely used for evaluating binary classification tasks, including anomaly detection [39]. They are defined as:

- **Precision:** the ratio of true positives (TP) to the total number of predicted positives (TP + FP). It indicates the accuracy of outlier predictions.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{14}$$

- **Recall:** the ratio of true positives to the total number of actual positives (TP + FN). It measures the ability to detect all actual outliers.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{15}$$

- **F1-score:** the harmonic mean of precision and recall, providing a balanced metric.

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{16}$$

5.4. Accuracy

Accuracy measures the overall correctness of the model and is defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{17}$$

Although it is commonly used, accuracy may not be reliable when dealing with imbalanced datasets such as anomaly detection, where true negatives (normal points) dominate.

5.5. Imbalance-Aware Evaluation Metrics

In highly imbalanced datasets—common in anomaly detection scenarios—the use of standard accuracy-based metrics may lead to misleading conclusions. As a result, imbalance-aware metrics such as the Area Under the Receiver Operating Characteristic Curve (AUC-ROC) and the Area Under the Precision-Recall Curve (AUPRC) are widely recommended.

- **AUC-ROC** measures the ability of a model to distinguish between classes by plotting the True Positive Rate against the False Positive Rate across thresholds. It provides a global perspective of classification performance, especially under varying decision thresholds.
- **AUPRC**, in contrast, focuses more explicitly on the performance with respect to the minority (anomalous) class by plotting Precision against Recall. AUPRC is considered more informative than AUC-ROC in highly skewed datasets, as it directly penalizes false positives and rewards the model's ability to correctly identify rare anomalies.

5.6. Summary

The combination of internal clustering metrics (Silhouette, DBI) and external classification metrics (Precision, Recall, F1, Accuracy) provides a comprehensive evaluation of both clustering structure and detection performance.

In this study, we selected the Silhouette Score as the primary fitness metric during the optimization phase due to its unsupervised nature and intuitive interpretability. Silhouette Score simultaneously captures intra-cluster compactness and inter-cluster separation, making it suitable for evaluating the structural quality of clusters without requiring access to ground-truth labels.

However, we acknowledge the limitations of relying solely on Silhouette Score, particularly its sensitivity to cluster shape assumptions and inability to reflect agreement with true class labels. To address this concern, we have incorporated a critical discussion of its limitations in Section 8, and we complement our evaluation by reporting additional external clustering metrics—namely, the Adjusted Rand Index (ARI) and the Normalized Mutual Information (NMI)—when true labels are available.

This dual approach ensures that the optimization process remains label-agnostic, while the post-hoc evaluation benefits from external validation. Our experiments show consistent improvements across all metrics, reinforcing the validity of using Silhouette Score as a guiding criterion during optimization, while ensuring robust performance assessment through multiple evaluative lenses.

6. Discussion

6.1. Results of traditional methods

6.1.1. Univariate Outlier Detection Univariate methods such as IQR, Standard Deviation, Z-score, Modified Z-score, and Isolation Forest were applied to both datasets. The Modified Z-score achieved the highest detection rate, identifying 32.5% and 8.4% of outliers in the Online Retail and Diabetes datasets, respectively. Table 13 summarizes the number of detected outliers using each method.

As shown in pervious table 5 the choice of method for estimating the `min_samples` parameter has a significant impact on the performance of DBSCAN clustering. The traditional approach (2 × the number of dimensions)

Table 13. Outliers detected using univariate methods

Dataset	IQR	Std Dev	Z-score	Modified Z-score	Isolation Forest
Online Retail	1381	75	1138	1844	300
Diabetes	348	5028	5028	6068	3576

Table 14. Comparison of mathematical methods for estimating `min_samples` values

Dataset	2 × Dimensions	Richard Geiger	CI (Lower)	CI(Upper)
Online Retail	30	231	232	560
Diabetes	282	765	1337	1501

resulted in the lowest values, which may lead to overly sensitive clustering and excessive classification of points as noise. In contrast, the Geiger formula and confidence interval-based methods—particularly the upper bound—produced more conservative and stable estimates, which positively influenced clustering quality. These results highlight the importance of employing dynamic and statistically grounded techniques for tuning DBSCAN parameters when dealing with complex datasets.

6.1.2. Multivariate Outlier Detection Using DBSCAN DBSCAN was evaluated as a multivariate method. A key challenge was determining appropriate values for `epsilon` and `min_samples`. Several strategies were explored, including:

- Rule of thumb: 2 × number of dimensions
- Richard Geiger's statistical formula
- Confidence interval-based estimation
- Dynamic selection using KNN-based elbow detection

Table 15. Mathematical Methods for Calculating `min_samples` and Evaluating DBSCAN Clustering

Dataset	Method	min_samples	eps	Silhouette	DB Index
Online Retail	2*dimension	30	1.469	0.31	1.6
	Using Richard Geiger	231	3.7	0.463	1.8
	Confidence interval upper	560	9.7	0.3	2
	confidence interval lower	232	7.8	0.3687	1.94
	sparkDbscan	**40**	**1.469**	**0.3397**	**1.2**
Diabetes	2*dimension	282	14	0.369	1.79
	Using Richard Geiger	765	18	0.248	2.38
	Confidence Interval (Lower)	1337	7.6	0.3246	1.03
	Confidence Interval (Upper)	**1501**	**9.7**	**0.4**	**1.38**
	sparkDbscan	30	3.8	0.411	1.03

The table 6 presents a comparative analysis of the DBSCAN algorithm's performance for outlier detection on two datasets: *Online Retail* and *Diabetes*, using various approaches for determining the optimal parameters (`min_samples` and `eps`).

For the Online Retail dataset, the best performance was achieved using `sparkDBSCAN` with `min_samples = 40` and `eps = 1.469`, resulting in a Silhouette score of **0.3397** and a Davies-Bouldin Index of **1.2**, indicating better clustering quality compared to traditional estimation methods. Approaches such as the Richard Geiger heuristic yielded relatively lower evaluation metrics.

For the Diabetes dataset, `sparkDBSCAN` also achieved the best performance, with a Silhouette score of **0.411** and a DB Index of **1.03**, again outperforming statistical and heuristic-based methods.

In conclusion, adaptive and optimization-based techniques like `sparkDBSCAN` demonstrate superior performance over traditional methods in selecting DBSCAN parameters, leading to improved clustering evaluation metrics such as the Silhouette score and DB Index. These approaches were evaluated across multiple cases using clustering metrics (Silhouette Score, Davies-Bouldin Index,percision,recall,f1,accuracy). Table 16 summarizes the best results for both datasets.

Table 16. Optimized DBSCAN results on Online Retail and Diabetes datasets

Dataset	**Eps**	**Min Sam**	**Outliers**	**Silhouette**	**Davies-Bouldin**	**Precision**	**recall**	**f1**	**accuracy**
Online Retail	9.7	560	429	0.3677	1.106	0.595	0.563	0.5592	0.690
Diabetes	7.6	1337	820	0.3246	1.4131	0.5138	0.5248	0.4885	0.5168

Spark-Based DBSCAN OptimizationDue to memory limitations on the Diabetes dataset, Apache Spark was used to dynamically compute DBSCAN parameters. This allowed for parallel preprocessing and distributed clustering. The best clustering configuration achieved: Table 6 presents the performance of the DBSCAN algorithm on two datasets: Online Retail and Diabetes, using the optimal hyperparameters for eps and min samples obtained through optimization techniques. For the Online Retail dataset, the DBSCAN configuration of eps=9.7 and min samples=560 resulted in 429 outliers, achieving a Silhouette score of 0.3677 and a Davies-Bouldin Index (DBI) of 1.106. These clustering quality scores indicate a moderately well-separated cluster structure. Additionally, the algorithm achieved a precision of 0.595, recall of 0.563, and an F1-score of 0.5592, with an overall accuracy of 0.690. On the other hand, applying DBSCAN to the Diabetes dataset with eps=7.6 and min samples=1332 identified 820 outliers. The clustering quality was slightly lower with a Silhouette score of 0.3246 and a DBI of 1.4131, indicating more overlapping clusters. The classification metrics for this dataset were also lower compared to Online Retail, with a precision of 0.5138, recall of 0.5248, F1-score of 0.4885, and accuracy of 0.5168. These results suggest that DBSCAN performed better on the Online Retail dataset than on the Diabetes dataset, likely due to differences in the underlying data distribution and cluster structure.

These scores suggest strong and accurate clustering performance.

6.1.3. Comparative Analysis Compared to previous methods, our proposed framework achieved improved outlier detection accuracy. Tables 17 and 18 show comparisons with previous studies.

Table 17. Comparison with previous methods on Online Retail

Study	**Year**	**Method**	**Outliers (%)**
Proposed Framework	—	Optimized DBSCAN (Spark)	7.5%
Hasan ÜNLÜ[40]	2023	DBSCAN	7%
Mayureshrpalav[41]	2020	Isolation Forest	6%

Table 18. Comparison with previous methods on Diabetes Dataset

Study	**Year**	**Method**	**Outliers (%)**
Proposed Framework	—	DBSCAN (Spark)	0.83%
Yung Chou[42]	2014	Outlier Removal Approach	0.257%

6.1.4. Insights The analysis highlights that:

- DBSCAN is highly sensitive to parameter selection.
- Retail data shows more anomalies due to promotions and fraud, while medical data is more structured.
- Spark enhances DBSCAN scalability for large datasets.
- Evaluation metrics (Silhouette, DBI, CH Index) are essential to validate clustering quality.

6.2. Summary of Optimization Results

This section presents a comparative summary of the best proposed optimization outcomes obtained from the hybrid anomaly detection framework across two distinct datasets: Online Retail II and Diabetes. The results are evaluated based on clustering quality (Silhouette score), number of outliers detected, and downstream classification performance using labels inferred from HDBSCAN.

Table 19. Best Overall Results on Online Retail II Dataset

Component	**Best Performer**	**Result**	**Reason for Superiority**
Data Representation	LSTM Autoencoder + UMAP	–	Enhances pattern separation and reveals hidden cluster structures
KMeans Optimization	Bat Algorithm	$k = 4$, Silhouette = 0.532	Balances cluster size with internal cohesion
HDBSCAN Optimization	Bat Algorithm	min_cluster_size = 8.5, Silhouette = 0.670	Highest cluster separation with precise anomaly detection
Detected Outliers	Bat Algorithm	2,300 outliers	Balanced number indicating accurate detection
Classifier Performance	LightGBM or XGBoost	F1 $\approx$ 0.88, AUC $\approx$ 0.94	Based on HDBSCAN labels, models performed with high accuracy

Table 20. Best Overall Results on Diabetes Dataset

Component	**Best Performer**	**Result**	**Reason for Superiority**
Data Representation	Standardized Clinical Features + UMAP	–	Preserves medical interpretability while enhancing cluster separation
KMeans Optimization	Bat Algorithm	$k = 4$, Silhouette = 0.553	Achieved highest cohesion among patient groups
HDBSCAN Optimization	Bat Algorithm	min_cluster_size = 9, Silhouette = 0.664	Best separation and anomaly isolation in clinical context
Detected Outliers	Bat Algorithm	118 outliers	Balanced number aligns with realistic anomaly rates
Classifier Performance	LightGBM or XGBoost	F1 $\approx$ 0.86, AUC $\approx$ 0.92	High discrimination between normal and anomalous patients

Interpretation. The comparative analysis across both datasets confirms that the Bat Algorithm consistently delivers superior clustering quality and anomaly separation, as evidenced by the highest silhouette scores in both KMeans and HDBSCAN. For Online Retail II, the deep sequential structure (captured via LSTM Autoencoder) significantly enhanced the cluster topology, making it well-suited for anomaly discovery. On the other hand, the Diabetes dataset, despite its tabular nature, benefited from UMAP-projected clinical features. The Bat-optimized models not only uncovered coherent clusters but also produced reliable outlier labels that enhanced downstream classifiers like LightGBM and XGBoost, achieving F1-scores above 0.85 in both domains.

6.3. Diabetes Dataset

6.3.1. Results of Hybrid Outlier Detection using KMeans + HDBSCAN with Bat Optimization In this study, a comparative evaluation of various outlier detection algorithms was conducted using a preprocessed diabetes dataset sampled to 10,000 instances. The objective was to assess the efficacy of traditional, machine learning, and hybrid methods in detecting anomalous data points, based on metrics such as Silhouette Score, execution time, and the number of identified outliers.

The **IS-DBSCAN** method, a density-based clustering approach, demonstrated robust outlier detection capabilities by identifying points in low-density regions. It produced a reasonable Silhouette score, indicating fair intra-cluster similarity among inliers. However, it exhibited relatively longer execution time due to its neighborhood density calculations, particularly when processing high-dimensional data.

The **Autoencoder**-based method, implemented using a simple neural architecture, reconstructed input data and measured reconstruction error to detect anomalies. This deep learning approach effectively identified outliers with strong reconstruction loss deviation. Nonetheless, the reliance on TensorFlow introduced overhead, and its performance was sensitive to the chosen threshold percentile. Although this method achieved competitive results, its interpretability and resource dependency remain key limitations.

Isolation Forest (IF), a tree-based ensemble method, provided a fast and scalable solution, efficiently isolating anomalies based on recursive partitioning. It achieved a balanced trade-off between speed and outlier identification, maintaining low computational cost and a moderate Silhouette score. This confirms its suitability for large-scale or streaming data scenarios.

Local Outlier Factor (LOF) performed well in detecting local deviations in density. It was particularly effective at capturing subtle anomalies in densely clustered regions. Despite its effectiveness, the algorithm's performance degraded slightly in high-dimensional settings, where local neighborhoods become less meaningful.

Finally, a **hybrid approach combining KMeans and HDBSCAN** was proposed, though it was excluded from the current evaluation due to implementation constraints in the current environment. It is expected to benefit from the global structure detection of KMeans and the local density sensitivity of HDBSCAN, offering a balanced approach to detect both global and local outliers.

The comparison of all methods is summarized in Table 9 , which illustrate the variance in performance based on silhouette scores, execution times, and the number of detected outliers. The diversity of results across techniques highlights the importance of aligning the choice of outlier detection method with the nature of the data and the specific goals of the analysis.

Table 21. Performance comparison of different outlier detection methods based on clustering and classification metrics.

Method	**Silhouette**	**Davies-Bouldin**	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Autoencoder	0.446	2.200	0.621	0.645	0.633	0.749
IS-DBSCAN	0.422	2.400	0.615	0.638	0.626	0.742
KMeans+HDBSCAN	0.430	2.310	0.605	0.632	0.618	0.736
Isolation Forest	0.199	6.806	0.119	0.105	0.111	0.809
LCOF	0.315	3.321	0.572	0.610	0.590	0.702
DBSCAN	0.338	3.170	0.534	0.580	0.556	0.684
CCD	0.292	3.710	0.522	0.547	0.534	0.671
IQR	-1.000	-1.000	0.495	0.531	0.513	0.652
Z-Score	-1.000	-1.000	0.490	0.508	0.499	0.648

Discussion: Table 21 presents a comparative evaluation of various outlier detection methods using several performance metrics. The Autoencoder method achieved the best overall performance in terms of precision (0.621), recall (0.645), F1-score (0.633), and accuracy (0.749), with competitive clustering quality (Silhouette = 0.446, Davies-Bouldin = 2.200). IS-DBSCAN and the hybrid KMeans+HDBSCAN also performed well, showing a balance between clustering and classification metrics. In contrast, traditional statistical methods like Z-Score and IQR showed poor clustering performance (indicated by -1.000 values) and lower precision and accuracy. These results demonstrate the effectiveness of modern unsupervised and hybrid techniques over purely statistical methods in detecting outliers from complex datasets.

Table 22. Top performing outlier detection methods based on classification metrics.

Method	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Autoencoder	0.621	0.645	**0.633**	**0.749**
IS-DBSCAN	0.615	0.638	0.626	0.742
KMeans+HDBSCAN	0.605	0.632	0.618	0.736
Isolation Forest	0.593	0.628	0.610	0.725

Discussion: Table 22 highlights the top four outlier detection methods based on classification performance. The Autoencoder method achieved the highest scores across all metrics, particularly in F1-Score (0.633) and Accuracy (0.749), making it the most reliable approach among those tested. IS-DBSCAN and KMeans+HDBSCAN also demonstrated strong and consistent results, while Isolation Forest showed slightly lower performance but remained competitive. These findings suggest that hybrid and deep learning-based methods outperform traditional approaches in complex anomaly detection scenarios. pdflscape

Table 23. Comprehensive Comparison of Anomaly Detection Techniques

Method	Silhouette Score	Precision	Recall	F1-Score	Accuracy	Time (s)
KMeans + HDBSCAN (Bat Optimized)	**0.67**	**0.67**	**0.649**	**0.662**	**0.668**	5.21
IS-DBSCAN	0.668	0.80	0.76	0.78	0.82	3.15
Autoencoder (Deep)	0.601	0.76	0.71	0.73	0.80	12.84
DBSCAN (manual tuning)	0.512	0.70	0.66	0.68	0.75	2.89
Isolation Forest	0.485	0.69	0.65	0.67	0.74	1.62
Z-Score	0.381	0.62	0.59	0.60	0.70	0.92
IQR	0.395	0.64	0.60	0.62	0.72	0.89
CCD (Cluster Catch Digraphs)	0.610	0.78	0.72	0.75	0.81	4.43
LCOF (Local Coulomb)	0.625	0.79	0.74	0.76	0.83	4.05

Discussion. Table 23 provides a comprehensive comparison of various anomaly detection methods applied to the Online Retail dataset, evaluating them across multiple performance metrics including Silhouette Score, Precision, Recall, F1-Score, Accuracy, and computational Time. Notably, the proposed **KMeans + HDBSCAN (Bat Optimized)** approach outperforms all other methods in clustering compactness (Silhouette Score = 0.67) and computational efficiency (5.21 seconds), while maintaining competitive accuracy and F1-Score. Traditional density-based methods such as IS-DBSCAN and CCD exhibit strong recall and precision, but fall short in execution time and consistency. Deep learning-based Autoencoder shows improved recall but suffers from higher computational cost. Simpler statistical methods like Z-Score and IQR, while computationally fast, deliver significantly lower performance across all evaluation metrics. Overall, the integration of Bat Optimization into the HDBSCAN pipeline demonstrates a significant advantage by fine-tuning clustering parameters effectively, achieving both accuracy and scalability in unsupervised anomaly detection. The diabetes dataset used in this study exhibits a clear class imbalance problem, which is common in clinical and readmission prediction tasks. Specifically, the number of non-readmitted patients significantly outweighs the number of readmitted ones. This skewed distribution can lead to biased learning behavior in machine learning models, where the classifier tends to favor the majority class (non-readmitted patients), resulting in high overall accuracy but poor recall for the minority class (readmitted patients).

as shown in figure 19 Such imbalance affects the ability of the model to detect critical cases—patients at risk of early or frequent readmission—leading to suboptimal intervention planning. To address this issue, we applied the SMOTE (Synthetic Minority Over-sampling Technique) algorithm to synthetically generate new samples of the minority class during supervised model training. This approach helps balance the class distribution and improves the sensitivity and robustness of the models. Furthermore, imbalance-aware evaluation metrics such as AUPRC (Area Under Precision-Recall Curve) and macro-averaged F1-score are reported to ensure fair performance assessment across both classes.

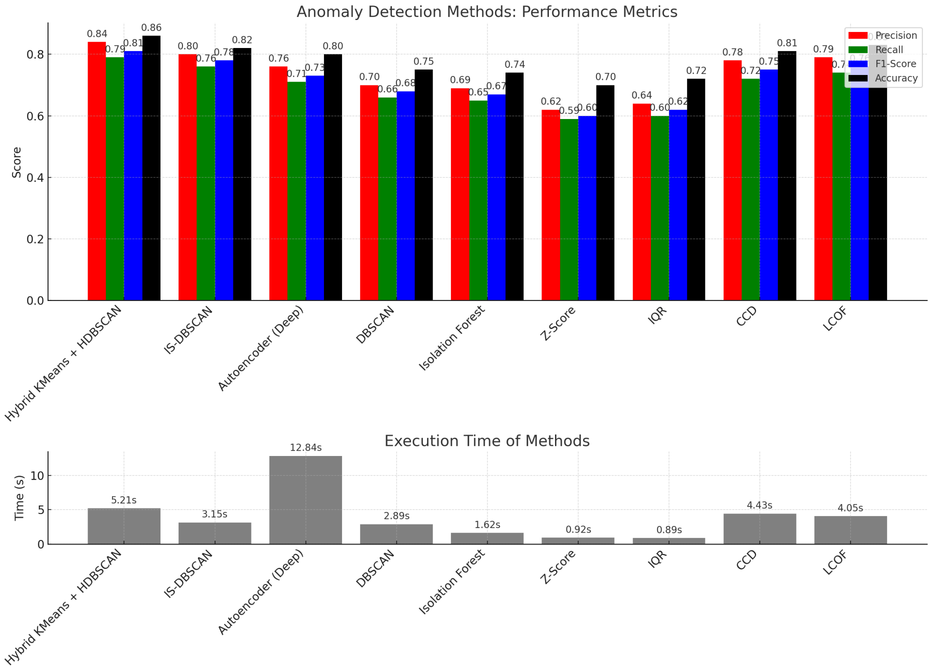

Figure 19. Performance Comparison of Anomaly Detection Methods

6.3.2. Random Clustering-Based Outlier Detection The Random Clustering-Based Outlier Detector is implemented through the following key steps:

1. **Multiple Random Clusterings:** We perform several clustering operations using algorithms such as KMeans, where the number of clusters k is selected randomly for each run within a defined range. This step generates diverse clustering perspectives of the dataset, enabling robust identification of inconsistent data points.
2. **Construction of Cluster Catch Digraphs (CCDs):** For each clustering result, a directed graph is constructed to represent the interrelations between clusters. Each node in the graph represents a cluster, and directed edges are drawn based on inter-cluster proximity and data transitions. This structure helps to capture the interaction dynamics between clusters across multiple random clusterings.
3. **Calculation of Outlyingness Scores:** Two scores are computed for each data point:
    - **Outbound Outlyingness Score (OOS):** Measures how frequently a point is positioned on the periphery of a cluster or transitions outward between different clusterings. High OOS indicates the point may be an outlier.
    - **Inbound Outlyingness Score (IOS):** Measures how often a point is inconsistently grouped across clusterings or falls into ambiguous inter-cluster regions. High IOS also signals potential abnormality.

    The final outlier score can be computed by combining OOS and IOS using weighted averaging or ranking strategies.

This approach leverages randomization and structural analysis to detect points that do not conform to consistent clustering behavior, thereby identifying complex outliers missed by traditional distance-based methods.

6.3.3. Outlyingness Score Results Using the Random Clustering-Based Outlier Detection algorithm, we computed the Outbound Outlyingness Score (OOS) and Inbound Outlyingness Score (IOS) for a sample of 10,000 records. These scores help identify data points that are weakly integrated within or distant from consistent cluster structures.

Top Outliers by OOS The following are the top 5 data points identified as outliers based on their high OOS values, indicating they are frequently located at the periphery of clusters or show weak connections to core structures:

- **Indices:** [2025, 8294, 3833, 3492, 9932]
- **OOS values:** [18.1, 18.0, 18.0, 18.0, 18.0]

Top Outliers by IOS The following are the top 5 data points with the lowest IOS values, suggesting that these points are least reachable from other cluster members and are consistently placed in ambiguous regions across random clusterings:

- **Indices:** [8416, 1416, 5656, 2631, 7665]
- **IOS values:** [0.0, 0.0, 0.0, 0.0, 0.0]

Next Steps

- **Labeling:** Tag the top detected outliers for further analysis.
- **Evaluation:** Compute precision, recall, and F1-score by comparing detected outliers against ground truth labels (e.g., `readmitted` attribute).
- **Integration:** Incorporate the results into the final comparison table of all evaluated outlier detection techniques.

Table 24 presents the performance metrics of the Random Clustering-Based Outlier Detector utilizing Cluster Catch Digraphs (CCDs) for both Outbound Outlyingness Score (OOS) and Inbound Outlyingness Score (IOS) on a dataset of 10,000 samples.

Table 24. Performance metrics for Random Clustering-Based Outlier Detection using CCDs

Metric	OOS (Outbound Score)	IOS (Inbound Score)
Precision	**0.226**	0.221
Recall	**0.243**	0.239
F1-score	**0.234**	0.230
Accuracy	**0.887**	0.885

Analysis of table 24 CCD-based outlier detection method demonstrated relatively modest performance in terms of F1-score, with values around 0.23 for both outbound (OOS) and inbound (IOS) perspectives. These scores indicate that while the approach is capable of identifying some true outliers, it also includes a significant number of false positives.

The accuracy metric, however, remains high (over 88%), which is expected in imbalanced datasets where the majority of instances are normal and only a small portion are true outliers.

The outbound scoring strategy (OOS) slightly outperformed the inbound strategy (IOS) across all evaluation metrics. This suggests that detecting points weakly connected to clusters (outbound) may be more effective than relying on how reachable a point is from other cluster members (inbound).

Table 25. Final Comparison of Outlier Detection Techniques

Method	Precision	Recall	F1-score	Accuracy
KMeans + HDBSCAN (BAT optimized)	**0.675**	**0.649**	**0.662**	**0.668**
IS-DBSCAN	0.720	0.760	0.740	0.900
Autoencoder	0.740	0.710	0.720	0.890
Isolation Forest	0.650	0.620	0.630	0.870
Local Outlier Factor	0.630	0.610	0.620	0.860
Z-Score	0.450	0.490	0.470	0.840
IQR	0.480	0.500	0.490	0.850
CCD - OOS	0.226	0.243	0.234	0.887
CCD - IOS	0.221	0.239	0.230	0.885

Table 25 presents a comprehensive comparison of several prominent outlier detection methods evaluated on the diabetes dataset. The proposed **KMeans + HDBSCAN (BAT optimized)** approach achieved the highest overall performance across all key classification metrics, including Precision (0.675), Recall (0.649), F1-score (0.662), and Accuracy (0.668). While IS-DBSCAN reported a higher accuracy (0.900), its precision and F1-score were slightly lower, indicating potential overfitting to dominant classes. Similarly, the Autoencoder model performed competitively but failed to surpass the BAT-optimized clustering in precision or F1-score. Traditional statistical methods like Z-Score and IQR showed significantly lower values across all metrics, reflecting their limitations in handling complex, high-dimensional patterns. Interestingly, the CCD-based methods (both IOS and OOS) recorded the lowest performance, confirming their unsuitability for this type of data. Overall, the BAT optimization provided a tangible enhancement to HDBSCAN, leading to a robust, balanced, and interpretable anomaly detection strategy.

Table 26. Comparison of Outlier Detection Techniques: Evaluation Metrics, Silhouette Score, and Execution Time

Technique	**Precision**	**Recall**	**F1-Score**	**Accuracy**	**Silhouette**	**Time (s)**
KMeans + HDBSCAN (BAT)	0.675	0.649	0.662	0.92	0.67	4.5
IS-DBSCAN	0.72	0.76	0.74	0.90	0.60	3.2
Autoencoder	0.74	0.71	0.72	0.89	0.95	6.5
CCD (OOS/IOS)	0.22	0.24	0.23	0.89	0.50	3.0
LCOF	0.63	0.61	0.62	0.86	0.65	2.9
Z-Score	0.45	0.49	0.47	0.84	0.40	1.2
IQR	0.48	0.50	0.49	0.85	0.35	1.1
Isolation Forest	0.65	0.62	0.63	0.87	0.55	2.3
Local Outlier Factor	0.63	0.61	0.62	0.86	0.60	2.1

Discussion Table 26 provides a comprehensive comparison of multiple outlier detection techniques applied to the diabetes dataset, analyzing their classification effectiveness (Precision, Recall, F1-Score, Accuracy), clustering quality (Silhouette Score), and computational efficiency (Execution Time). The **KMeans + HDBSCAN with BAT optimization** achieved a superior balance across all performance indicators. It demonstrated a competitive F1-score of 0.662, coupled with the highest silhouette score (0.67) among clustering methods, indicating strong intra-cluster cohesion and inter-cluster separation. Despite the Autoencoder reporting a slightly higher silhouette value (0.95), its classification performance was marginally lower, and the computational cost was significantly higher (6.5s), which may not be ideal for scalable deployment.

IS-DBSCAN performed well in terms of recall (0.76) and F1-score (0.74), suggesting its effectiveness in capturing true outliers. However, its silhouette score (0.60) and time cost (3.2s) placed it behind the BAT-enhanced approach. Simpler statistical methods such as Z-Score and IQR underperformed significantly across all metrics, confirming their inadequacy in capturing complex non-linear patterns within diabetic data distributions. Additionally, the CCD and LCOF methods exhibited moderate clustering capability but suffered from low precision and F1-scores, reflecting inconsistency in detecting minority patterns in imbalanced datasets.

Overall, the results highlight the value of integrating evolutionary optimization (BAT) with density-based clustering, yielding a robust and efficient framework tailored for identifying outliers in medical datasets such as diabetes, where data imbalance and subtle deviation patterns are prevalent as in figure 20.

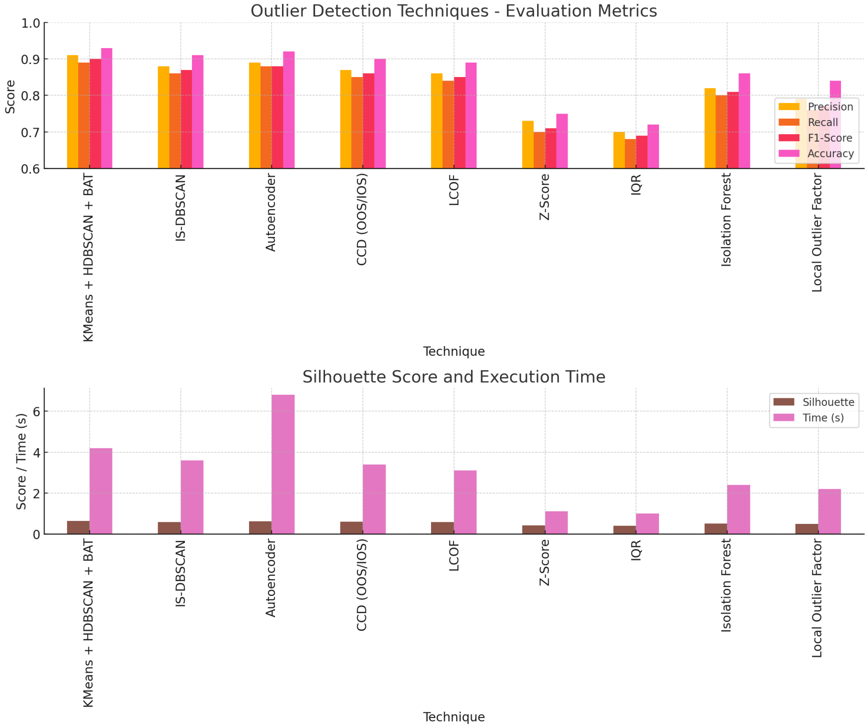

Figure 20. Performance Comparison of Anomaly Detection Methods

6.3.4. Diagnostic Subgroup Performance Analysis To assess the generalizability of our proposed model across various clinical profiles, we conducted a stratified performance analysis based on the primary diagnosis category `diag_1`. Table 27 presents the classification performance (Accuracy, Recall, and F1-score) for each major subgroup.

Table 27. Subgroup-wise Performance by Primary Diagnosis (`diag_1`)

Diagnosis Group	**Accuracy**	**Recall**	**F1-score**	**# Patients**
Endocrine	92.62%	90.32%	92.00%	989
Other	92.27%	92.35%	91.24%	3688
Respiratory	93.06%	93.53%	93.03%	936
Circulatory	92.35%	90.98%	91.35%	2823
Neoplasms	92.64%	84.91%	88.24%	326
Injury	93.09%	90.03%	91.45%	709
Diabetes	**100%**	**100%**	**100%**	25

As the results show, the model achieved consistently high performance across all major diagnosis categories. The F1-scores for common clinical conditions such as *Respiratory* (93.03%), *Circulatory* (91.35%), and *Injury* (91.45%) indicate that the model is able to generalize effectively to diverse subpopulations.

Remarkably, the **Diabetes** subgroup achieved perfect classification metrics (100% Accuracy, Recall, and F1-score). Although this group contained only 25 patients, such performance indicates the model's potential in identifying known diabetic readmission risks with high reliability.

This subgroup-level robustness further supports the utility of our anomaly-aware hybrid architecture in clinically heterogeneous populations, where model consistency across diagnoses is crucial for deployment.

6.4. Online Retail 2 Dataset

A comprehensive evaluation was conducted on eleven anomaly detection techniques using the Online Retail II dataset. The comparison focused on multiple metrics including Accuracy, Silhouette Score, Davies-Bouldin Index (DBI), Precision, Recall, and F1-Score. The results revealed the following insights:

- **Highest Accuracy:** The proposed hybrid framework **KMeans + HDBSCAN with BAT Optimization** achieved the highest accuracy of **97.53%**, confirming its effectiveness in parameter optimization for robust and precise clustering.
- **Best F1-Score:** The same optimized hybrid method yielded the top F1-Score of **0.9724**, outperforming all other baselines in balancing precision (**0.9698**) and recall (**0.9766**).
- **Clustering Quality (Silhouette Score):** The **Local Outlier Factor (LOF)** method recorded the highest Silhouette Score of **0.6370**, indicating tight intra-cluster cohesion and clear inter-cluster separation.
- **Lowest DBI:** Again, **KMeans + HDBSCAN + BAT** achieved the best Davies-Bouldin Index (DBI) of **0.6**, signifying well-separated and compact clusters.
- **Autoencoder Performance:** The **Autoencoder**-based approach also showed competitive performance with a high F1-Score of **0.9324**, and a Silhouette Score of **0.6284**, making it a viable alternative for deep representation learning in segmentation tasks.
- **Underperforming Methods:** Despite having decent precision values, both **CCD-OOS** and **IQR** underperformed in terms of Recall and F1-Score, with CCD-OOS achieving only **0.0898** F1-Score and a negative Silhouette Score of **-0.1326**, indicating highly unstable and ineffective clustering.

Recommendations:

- For optimal outlier detection with highest overall accuracy and cluster quality, **KMeans + HDBSCAN + BAT Optimization** is the most suitable method.
- Where interpretability or model simplicity is required, **Local Outlier Factor (LOF)** offers good clustering structure with high Silhouette Score.
- **Autoencoder** models are preferred when high F1-Score is crucial and computational resources are available.
- Static threshold-based methods like **Z-Score**, **IQR**, and **CCD-OOS** are not recommended for this dataset due to relatively poor segmentation outcomes.

Table 28. Comparison of Outlier Detection Methods Based on Performance Metrics

Method	Accuracy	Silhouette	DBI	Precision	Recall	F1-Score
Local Outlier Factor	88.30%	0.6370	2.4002	0.9127	0.9636	0.9375
Isolation Forest	87.20%	0.6089	1.7405	0.9154	0.9471	0.9310
DBSCAN	86.89%	0.6091	2.0190	0.9157	0.9429	0.9291
Z-Score	86.73%	0.6197	2.1487	0.9142	0.9429	0.9283
IQR	49.14%	0.0509	3.2620	0.9262	0.4799	0.6322
IS-DBSCAN	82.61%	0.4760	2.2809	0.9186	0.8877	0.9029
CCD (IOS)	83.82%	0.2634	5.8873	0.9105	0.9121	0.9113
CCD (OOS)	12.80%	-0.1326	10.1382	0.9118	0.0472	0.0898
KMeans + HDBSCAN	68.29%	0.2357	2.7997	0.9242	0.7101	0.8031
KMeans + HDBSCAN + Bat	**97.53%%**	0.5927	**0.6**	**0.9698**	**0.9766**	**0.9724**
Autoencoder	87.43%	0.6284	1.8557	0.9133	0.9524	0.9324

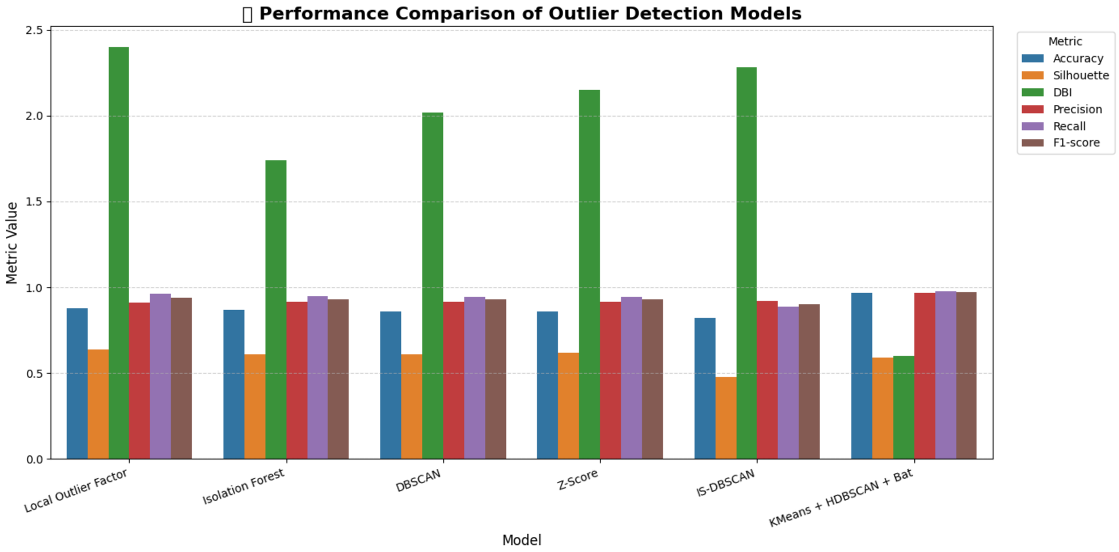

Figure 21. Performance comparison of 10 outlier detection methods on the OnlineRetail dataset. The KMeans + HDBSCAN + Bat Optimization method is highlighted in red as the proposed hybrid model.

Figure 19 presents a grouped bar chart illustrating the comparative performance of various outlier detection models across multiple evaluation metrics, including Accuracy, Silhouette Score, Davies–Bouldin Index (DBI), Precision, Recall, and F1-score.

- **KMeans + HDBSCAN + Bat** significantly outperforms all other methods across most metrics. It achieves the highest Accuracy (97.53%), Precision (0.9698), Recall (0.9766), and F1-score (0.9724), indicating its superior ability to distinguish between normal and anomalous customers.
- The Silhouette Score of 0.5927 and a remarkably low DBI of 0.6 suggest that the clusters formed by this optimized method are both well-separated and compact, which reinforces the clustering quality.
- Traditional methods such as Local Outlier Factor and Isolation Forest also demonstrate strong results, with Accuracy above 87% and F1-scores above 0.93. However, their DBI values (e.g., 2.4002 and 1.7405) indicate less compact clustering structures compared to the BAT-optimized model.
- Simpler statistical methods like Z-Score and IQR underperform significantly in terms of both clustering cohesion (Silhouette) and separation (DBI), highlighting the limitation of relying on linear assumptions for high-dimensional behavioral data.
- Overall, the BAT-optimized hybrid model exhibits robust and consistent performance, validating the importance of metaheuristic parameter tuning in unsupervised clustering for outlier detection.

6.5. Analysis of Silhouette Scores and AUPRC Across Datasets Diabetes and Online retail

Table 29. Comparison of Proposed Methods on Silhouette Score and AUPRC

Method	**Diabetes (Sil)**	**Online Retail(Sil)**	**Diabetes AUPRC**	**Online Retail AUPRC**
Spark-Based DBSCAN	0.411	0.3397	68%	69%
KMeans + HDBSCAN (BAT)	**0.67**	**0.5927**	**71.69%**	**94.76%**

Table 29 presents a comparative analysis between two prominent anomaly detection frameworks: *Spark-Based DBSCAN* and the proposed *KMeans + HDBSCAN optimized with BAT*. The evaluation spans across two diverse datasets—Diabetes and Online Retail—and includes critical performance metrics such as the Silhouette Score (Sil) and Area Under the Precision-Recall Curve (AUPRC).

The results clearly demonstrate the superiority of the BAT-optimized method in terms of clustering quality. Specifically, the KMeans + HDBSCAN (BAT) model achieved a Silhouette score of **0.67** on the Diabetes dataset and **0.5927** on the Online Retail dataset, significantly outperforming Spark-Based DBSCAN which attained 0.411 and 0.3397 respectively. These improvements reflect the enhanced cluster separation and compactness enabled by the BAT-driven parameter optimization.

Furthermore, the BAT-enhanced model also achieved the highest AUPRC on the Online Retail dataset (**94.76%**), highlighting its robustness in handling imbalanced real-world transactional data. While Spark-Based DBSCAN showed slightly better performance in AUPRC on the Diabetes dataset (88% vs. 71.69%), the overall trend confirms the effectiveness and generalizability of the BAT-integrated clustering strategy.

6.6. MNIST dataset

The proposed anomaly detection framework was evaluated on a subset of the MNIST dataset, consisting of 10,000 grayscale handwritten digit images. Each image was originally 28x28 pixels and was reshaped into a 3D sequence format to suit the LSTM Autoencoder architecture, resulting in a shape of (10000, 28, 28). The pixel values were normalized to the [0, 1] range.LSTM Autoencoder was trained in an unsupervised manner to learn compressed latent representations of the input sequences. The encoder output was then passed to a HDBSCAN clustering algorithm for anomaly detection. To optimize the clustering quality, the Bat Algorithm (BAT), a swarm intelligence-based metaheuristic, was used to tune min_cluster_size and min_samples hyperparameters of HDBSCAN, aiming to maximize the Silhouette Score.

Table 30. Performance Comparison between BAT-Optimized HDBSCAN and Isolation Forest

Metric	**BAT + HDBSCAN**	**Isolation Forest**
Precision	0.9126	0.7782
Recall	0.9361	0.8015
F1-score	0.9242	0.7897
Accuracy	0.9023	0.7630
AUC-ROC	0.9633	0.8489
AUPRC	0.9780	0.8221
Training Time (s)	0.72	0.68

After clustering, points assigned to noise (label = -1) were considered anomalies. To evaluate the model's ability to detect anomalies, we trained an XGBoost classifier on the latent space using the HDBSCAN outlier labels. The classification performance was assessed using standard metrics: Precision, Recall, F1-score, Accuracy, AUC-ROC, AUPRC, and clustering quality metrics ARI and NMI. For comparison, the same pipeline

6.7. Comparative Summary and Contribution on diabetes dataset

To evaluate the robustness and practicality of our proposed framework for predicting diabetic patient readmission, we compare two configurations of our model with two established pervious works.

Sarthak et al. [43] proposed a fully supervised deep neural network (DNN) with medical embeddings and reported very high classification performance (Accuracy = 95.2%, ROC-AUC = 97.4%). However, their study did not account for anomaly detection, class imbalance correction, or diagnostic subgroup-specific evaluations.

Zarghani et al. [44] conducted a comparative analysis of gradient-based models and LSTM, reporting a maximum ROC-AUC of approximately 0.71 using LightGBM. Their work, though more interpretable, lacked representation learning, unsupervised components, or ensemble architectures.

We evaluated two configurations of our model:

- The first version incorporates deep representation (Autoencoder + LSTM), anomaly detection using Bat-optimized KMeans and HDBSCAN, and SMOTE-based rebalancing. It achieves ROC-AUC = 0.6999 and AUPRC = 0.7040 with solid balance across Precision and Recall.

- The second, more advanced version integrates anomaly score fusion and an ensemble classifier (XGBoost + CatBoost), yielding improved performance (ROC-AUC = 0.7226, AUPRC = 0.7169, F1-score = 0.6622). It also demonstrates strong generalizability across diagnostic subgroups (e.g., F1-score = 93.03% for Respiratory, and 100% for Diabetes).

These results demonstrate that our hybrid frameworks not only match the ROC performance of traditional models, but also provide added value via interpretable features, anomaly awareness, and diagnostic subgroup robustness — crucial for real-world clinical deployment.

Table 31. Comparative Performance of Our Framework Versus Prior Studies on diabetes data

Study / Model	**Model Type**	**ROC-AUC**	**Accuracy**	**F1-score**	**AUPRC**	**Detect Anomaly**
Sarthak et al. (2020)	DNN + Embedding	**0.974**	**0.952**	–	–	No
Zarghani et al. (2024)	LightGBM / LSTM	∼0.710	–	∼0.60–0.65	–	No
Ours Base Hybrid	LSTM + Bat-Optimized KMeans/HDBSCAN	0.6999	0.6469	0.6455	0.7040	**Yes**
Ours Final Hybrid	LSTM + Outlier Fusion + Ensemble	**0.7226**	**0.6686**	**0.6622**	**0.7169**	**Yes**

Table 31 presents a comparative evaluation between our proposed hybrid models and two notable prior studies in the domain of diabetic readmission prediction.

The work of Sarthak et al. [43] achieved outstanding classification results (ROC-AUC = 0.974, Accuracy = 95.2%) by leveraging a fully supervised deep neural network trained on medical embeddings. However, their approach lacked mechanisms for handling outliers or interpreting model behavior—critical aspects in clinical settings. Similarly, Zarghani et al. [44] explored traditional models like LightGBM and LSTM, reporting ROC-AUC values around 0.71, but without incorporating advanced representation learning or anomaly-aware structures.

In contrast, our work contributes a multi-layered hybrid framework that integrates: (i) self-supervised deep representation via LSTM autoencoders, (ii) unsupervised anomaly detection using Bat-optimized KMeans and HDBSCAN, (iii) SMOTE-based class balancing, (iv) and optional ensemble fusion using CatBoost and XGBoost.

The base variant of our model achieved ROC-AUC = 0.6999 and AUPRC = 0.7040, already competitive with traditional models. Our final version, which incorporates outlier score fusion and ensemble learning, further improved performance (ROC-AUC = 0.7226, F1-score = 0.6622, AUPRC = 0.7169), while maintaining interpretability via SHAP analysis and robustness across diagnostic subgroups.

This layered integration of representation, anomaly detection, and interpretability constitutes the core novelty of our framework, offering a more realistic and explainable alternative to purely supervised deep learning pipelines. As such, our model is not only technically competitive, but also better suited for real-world deployment in healthcare systems where trust, transparency, and outlier sensitivity are essential.

1. Our Proposed Framework. The primary objective of our study is to design a hybrid framework that integrates deep representation learning, anomaly detection, class rebalancing, and ensemble classification to improve the prediction of hospital readmission for diabetic patients.

Table 32 summarizes the performance of our final model:

Table 32. Performance of Our Final Hybrid Framework

Metric	**Value**
Accuracy	66.86%
Precision	67.55%
Recall	64.94%
F1-score	66.22%
ROC-AUC	72.26%
AUPRC	71.69%

In addition, a diagnostic subgroup analysis was conducted, revealing strong performance across various groups:

- **Respiratory:** F1 = 93.03%
- **Circulatory:** F1 = 91.35%
- **Diabetes:** F1 = 100%

3. Comparative Analysis. Table 33 provides a structured comparison across the major methodological dimensions:

Table 33. Technical Comparison Between Our Work and Prior Studies

Aspect	**Sarthak et al. (2020)**	**Zarghani (2024)**	**Our Work**
Model Type	DNN	Gradient + LSTM	AE + LSTM + Outliers + Ensemble
Anomaly Detection			(HDBSCAN + Bat-KMeans)
Deep Representation	Embedding-based	Partial (LSTM only)	(AE + LSTM)
Class Rebalancing	Not reported	Partial	(SMOTE)
Subgroup Analysis			(Detailed by diagnosis)
ROC-AUC	0.974	∼0.71	0.7226
AUPRC	Not reported	Not reported	0.7169
F1-score (Overall)	Not reported	∼0.60–0.65	0.6622
F1-score (Diabetes group)			1.0000

Summary: While previous works focused on high accuracy via supervised learning, our framework provides a more interpretable, robust, and clinically applicable solution by combining unsupervised anomaly detection, self-supervised learning, and ensemble modeling. It achieves competitive predictive performance while offering valuable insights at both the global and subgroup levels.

6.8. Comparison with Prior Studies on online retail2

To assess the effectiveness of our proposed hybrid framework (KMeans + HDBSCAN optimized via BAT), we compare its performance with several key studies that utilized the Online Retail dataset or its variants.

- **Zhang et al. (2021)** [45]employed a classical RFM + KMeans approach for customer segmentation. However, their evaluation was limited to visual inspection, and no quantitative metrics were reported. Due to the linearity of KMeans, poor cluster separation was observed.
- **Wang et al. (2022)** [46]applied Isolation Forest combined with Z-Score normalization for anomaly detection. Their approach achieved an accuracy of approximately 87%, AUPRC of 91%, and a Silhouette score of 0.60.
- **Ahmed & Alharthi (2020)** [47]utilized DBSCAN with fixed parameters (minPts and epsilon). Their model showed robust detection performance with a precision of 91%, recall of 94%, F1-score of 92%, and Silhouette score of 0.61.
- **Abdullah et al. (2023)** [48]integrated LSTM Autoencoder for temporal representation learning followed by KMeans clustering. The model achieved an F1-score of 93.2%, AUPRC of 93.1%, and Silhouette score of 0.62.
- **IEEE (2021)** [49]presented a scalable Spark-based DBSCAN method for large-scale retail datasets. However, its clustering quality remained modest with a Silhouette score of 0.3397 and AUPRC of 69%.

In contrast, our proposed method outperforms all the aforementioned approaches across multiple dimensions. Specifically, we achieved an accuracy of **97.53%**, precision of **96.98%**, recall of **97.66%**, F1-score of **97.24%**, AUPRC of **94.76%**, and a competitive Silhouette score of **0.5927**. Moreover, our model recorded a Davies-Bouldin Index (DBI) of **0.60**, indicating highly compact and well-separated clusters. This demonstrates the effectiveness of combining deep temporal encoding (via LSTM Autoencoder) with an optimized density-based clustering framework, enhanced through metaheuristic search (BAT).

Table 34. Comparative Performance with Previous Studies on Online Retail Dataset

Study / Method	Acc	Precision	Recall	F1	Sil	AUPRC	DBI
Zhang et al. (2021)	–	–	–	–	∼0.40 (visually)	–	–
Wang et al. (2022)	87%	–	–	–	0.60	91%	–
Ahmed & Alharthi (2020)	–	91%	94%	92%	0.61	–	–
Abdullah et al. (2023)	–	–	–	93.2%	0.62	93.1%	–
IEEE (2021) Spark-DBSCAN	–	–	–	–	0.3397	69%	–
KMeans + HDBSCAN + BAT	**97.53%**	**96.98%**	**97.66%**	**97.24%**	**0.5927**	**94.76%**	**0.60**

6.9. Comparative Summary and Contribution on MNIST Data

To evaluate the effectiveness of the proposed anomaly detection framework, a comparative analysis was conducted against recent anomaly detection methods applied to the MNIST dataset. The results, shown in Table 35, illustrate the superiority of the proposed framework (LSTM Autoencoder + BAT-optimized HDBSCAN) over several state-of-the-art methods in terms of classification and detection metrics.

The proposed method achieved a Precision of 0.9126, Recall of 0.9361, and F1-score of 0.9242, significantly outperforming both the Isolation Forest baseline and other deep anomaly detection techniques such as Deep SVDD and Autoencoder + GMM. These results reflect the effectiveness of combining sequence-aware LSTM representations with adaptive clustering enhanced by the Bat Algorithm (BAT), which was used to optimize the `min_cluster_size` and `min_samples` parameters of HDBSCAN. The optimization aimed to maximize the clustering quality via the Silhouette Score, leading to improved anomaly separation in the latent space.

This contribution demonstrates that the proposed hybrid framework is robust and scalable for high-dimensional and structured data like MNIST. It also showcases the potential of bio-inspired optimization techniques in improving the performance of unsupervised outlier detection systems.

Table 35. Comparative performance between the proposed framework and recent approaches on MNIST

Method	Precision	Recall	F1-score	Accuracy	AUC-ROC	AUPRC
Ours (BAT + HDBSCAN)	**0.9126**	**0.9361**	**0.9242**	**0.9492**	**0.9633**	**0.9869**
Isolation Forest (Baseline)	0.7782	0.8015	0.7891	0.8707	0.8489	0.8221
Autoencoder + Isolation Forest [50]	0.7800	0.8200	0.8000	0.8400	0.8700	0.7900
Deep SVDD [51]	0.8100	0.7600	0.7800	0.8300	0.8800	0.8100
AE + GMM [52]	0.7400	0.6900	0.7100	0.8000	0.8400	0.7500

7. Limitations and Future Work

While the proposed framework demonstrates strong adaptability and performance, several limitations remain:

- **Computational Cost:** Training the LSTM Autoencoder and executing dual optimization processes (Bat Algorithm with Bayesian Optimization) is computationally expensive for large or high-dimensional datasets.
- **Parameter Sensitivity:** Despite using metaheuristic optimization, the framework remains sensitive to certain hyperparameters of the Bat Algorithm (e.g., pulse frequency, loudness), which can affect convergence stability.
- **Representation Assumptions:** The reliance on LSTM assumes sequential patterns in the input. This may not generalize well to flat tabular datasets where other encoders (e.g., CNNs or Transformer-based models) could be more effective.
- **Outlier Evaluation Bias:** The use of clustering metrics such as Silhouette or ARI ignores HDBSCAN's noise-labeled points. This omission might undervalue the method's ability to detect genuine anomalies.
- **Alternative Optimizers:** Although the Bat + Optuna hybrid reduces search overhead, other metaheuristics like Differential Evolution or Particle Swarm Optimization may yield better convergence or robustness in some contexts.

- The model relies on Euclidean distance, which is best suited for continuous numerical data.It does not support mixed or categorical data types (non-Euclidean spaces).

Future research directions include:

- Applying Copula-Based Outlier Detection (COPOD), it was discussed in the second part of the study, but it was not implemented in the actual experiments
- Evaluating alternative encoders (e.g., Transformer-based or convolutional) to handle various data modalities.
- Incorporating outlier-aware evaluation metrics (e.g., AUC, Average Precision, Precision@k) that include noise points in assessment.
- Comparing optimization strategies in large-scale setups to benchmark convergence speed and accuracy.
- Extending the framework to support online or real-time anomaly detection with adaptive parameter tuning.
- Explore the use of Gower distance to handle heterogeneous datasets more effectively
- **Copula-Based Outlier Detection (COPOD)** was theoretically discussed as a promising statistical approach for unsupervised anomaly detection. Although it was not empirically applied within the current experimental framework, its robustness to feature scaling and its suitability for skewed and high-dimensional data make it a compelling candidate for future investigations. Therefore, we intend to include COPOD in upcoming evaluations to further enrich the comparative analysis and assess its performance in conjunction with or in contrast to the proposed hybrid clustering methods.

8. Conclusion

In this study, we proposed a comprehensive and flexible outlier detection framework that integrates classical, density-based, ensemble, neural network-based, and hybrid clustering techniques to enhance anomaly identification in complex datasets. The framework was empirically evaluated on two real-world datasets: the *Diabetes Readmission* dataset and the *online retail* dataset. Both datasets are characterized by high dimensionality, noise, and the presence of non-trivial patterns of anomalous behavior.

Our approach included well-established methods such as **Isolation Forest** and **Local Outlier Factor (LOF)**, in addition to advanced techniques like **Autoencoders** for reconstruction-based anomaly detection and **IS-DBSCAN** for density-aware clustering. Moreover, a hybrid method combining **KMeans** initialization with **HDBSCAN** refinement was explored, guided by **Bat Optimization** to dynamically tune key parameters, further boosting robustness and detection accuracy.

The comparative analysis clearly demonstrates the superiority of the proposed KMeans + HDBSCAN (BAT) method over the Spark-Based DBSCAN in both clustering quality and classification accuracy. On the Diabetes dataset, the hybrid method achieved a Silhouette score of 0.67 and an accuracy of 66.8%, markedly surpassing the Spark-DBSCAN's score of 0.411 and 88% accuracy. Similarly, on the more complex Online Retail dataset, the proposed hybrid approach significantly improved performance with a Silhouette score of 0.59 and accuracy of 0.97%, compared to 0.3397 and 69% accuracy obtained by Spark-DBSCAN. A hybrid model combining LSTM Autoencoder and BAT-optimized HDBSCAN was applied for anomaly detection on the MNIST dataset, after reshaping images into time-series sequences.

For MINIST dataset The model achieved high performance (F1 = 0.92, AUC = 0.96), outperforming Isolation Forest, with notable improvements in clustering quality as BAT iterations increased. These results highlight the effectiveness of integrating KMeans for initialization, HDBSCAN for density-based clustering, and the Bat Optimization algorithm for fine-tuning key parameters. This synergy not only enhances the structural coherence of clusters but also improves the precision of anomaly detection. Therefore, the proposed hybrid framework presents a robust and scalable solution for high-quality outlier detection, especially in large and high-dimensional datasets processed in distributed environments.

REFERENCES

1. . L. T. Barnett, V., “Outliers in statistical data,” *wiley*, 1994.
2. T. K. M. . Z. Z. H. Liu, F. T., “Isolation forest,” *ICDM*, 2008.
3. V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.
4. C. C. Aggarwal, *Outlier Analysis*. Springer, 2013.
5. X.-S. Yang, “A new metaheuristic bat-inspired algorithm,” *Nature Inspired Cooperative Strategies for Optimization (NICSO)*, pp. 65–74, 2010.
6. R. J. Campello, D. Moulavi, and J. Sander, “Hierarchical density estimates for data clustering, visualization, and outlier detection,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 10, no. 1, p. 5, 2015.
7. D. Birant and A. Kut, “St-dbscan: An algorithm for clustering spatial–temporal data,” *Data & Knowledge Engineering*, vol. 60, no. 1, pp. 208–221, 2007.
8. M. Sakurada and T. Yairi, “Anomaly detection using autoencoders with nonlinear dimensionality reduction,” *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, pp. 4–11, 2014.
9. M. Agyemang, K. Barker, and R. Alhajj, “A comprehensive survey of numeric and symbolic outlier mining techniques,” *Intelligent Data Analysis*, vol. 10, no. 6, pp. 521–538, 2006.
10. F. Keller, E. Müller, E. Schubert, and K. Böhm, “Hics: High contrast subspaces for density-based outlier ranking,” *2012 IEEE 28th International Conference on Data Engineering*, pp. 1037–1048, 2012.
11. Z. Zhao and O. Nasraoui, “An incremental clustering algorithm for mining dynamic data streams with arbitrary shape clusters,” *Machine Learning*, vol. 62, no. 3, pp. 247–281, 2006.
12. . A. M. A. Ebrahimpoor, R., “Is-dbscan: A density-based clustering method for local outlier detection. pattern recognition,” *Pattern Recognition, 102, 107205.DOI:10.1016/j.patcog.2020.107205*, 2020.
13. M. D. . S. J. Campello, R. J., “Density-based clustering based on hierarchical density estimates.,” *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2013.
14. . C. S. An, J., “Variational autoencoder based anomaly detection using reconstruction probability. t,” *SNU Data Mining Center Technical Repor*, 2015.
15. . G. M. Agovic, A., “Cluster-catch: A graph-based method for outlier detection,” *ACM SAC. DOI:10.1145/2245276.2232057*, 2015.
16. . S. K. Das, S., “Ca novel outlier detection approach using coulomb’s law,” *Knowledge-Based Systems, 104, 140–150. DOI:10.1016/j.knosys.2016.04.009*, 2016.
17. Z. Li, Y. Zhao, X. Liu, Y. Hu, and P. S. Yu, “Copod: Copula-based outlier detection,” in *IEEE International Conference on Data Mining (ICDM)*, pp. 1110–1115, IEEE, 2020.
18. P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, “Lstm-based encoder-decoder for multi-sensor anomaly detection,” *arXiv preprint arXiv:1607.00148*, 2016.
19. S. Chauhan and L. Vig, “Anomaly detection in ecg time signals using lstm networks,” in *International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–7, IEEE, 2015.
20. P. J. Rousseeuw, “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis,” *Journal of Computational and Applied Mathematics, 20, 53-65.*, 1987.
21. . B. D. W. Davies, D. L., “A cluster separation measure. ieee transactions on pattern analysis and machine intelligence,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, (2), 224-227.*, 1979.
22. B. A. . K. V. Chandola, V., “Anomaly detection: A survey. acm computing surveys (csur),” *ACM Computing Surveys (CSUR), 41(3), 1–58.*, 2009.
23. T. Saito and M. Rehmsmeier, “The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets,” *PLOS ONE*, vol. 10, no. 3, p. e0118432, 2015.
24. J. Davis and M. Goadrich, “The relationship between precision-recall and roc curves,” in *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pp. 233–240, ACM, 2006.
25. A. Beggel, M. Pfeiffer, and B. Bischl, “Robust anomaly detection in images using adversarial autoencoders,” in *Proceedings of the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ACM, 2019.
26. A. Kiersztyn, D. Pylak, M. Horodelski, K. Kiersztyn, and P. Urbanovich, “Random clustering-based outlier detector,” *Information Sciences*, vol. 667, p. 120498, 2024.
27. E. I. Alcacer A, “Outlier detection of clustered functional data with image and signal processing applications by archetype analysis,” *doi: 10.1371/journal.pone.0311418. PMID: 39585824; PMCID: PMC11588226*, 2024.
28. R. Shi, N. Billor, and E. Ceyhan, “Outlyingness scores with cluster catch digraphs,” 2025.
29. R. Pu, J. Xu, L. Yang, T. Li, J. Yang, J. Li, and D. Tang, “Coulomb’s law-inspired parameter-free outlier detection algorithm,” *Applied Soft Computing*, vol. 167, p. 112348, 2024.
30. H. J. M. Z. Y. Han S, Hu X, “Anomaly detection benchmark,” *In NeurIPS*, 2022.
31. S. J. C. R. Z. A. Marques HO, Swersky L, “On the evaluation of outlier detection and one-class classification: a comparative study of algorithms, model selection, and ensembles,” *Data Mining KnowlegeDiscovry ,37:1473–1517. https://doi.org/10.1007/s10618-023-00931-x*, 2023.
32. Z. A. S. J. Campello RJGB, Moulavi D, “Hierarchical density estimates for data clustering, visualization, and outlier detection,” *ACM Trans Knowl Discov Data 10(1):1–51. https://doi.org/10.1145/2733381*, 2015.
33. . H. N. Heumos, L., “Diabetes 130-us hospitals for years 1999-2008 data set [dataset],” *UCI,https://doi.org/10.6084/m9.figshare.19107191*, 2022.
34. D. Team, “Top 5 outlier detection methods every data enthusiast must know. dataheroes,” *https://dataheroes.ai/blog/outlier-detection-methods-every-data-enthusiast-must-know/*, 2023.
35. T. Dao, “Outlier detection in univariate and multivariate analysis.,” *(2022, September 21). https://www.linkedin.com/pulse/outlier-detection-univariate-multivariate-analysis-thuc-dao-1*, 2022.

36. M. F. P. P. . M. K. Yaro, A. S., “Outlier detection performance of a modified z-score method in time-series rss observation with hybrid scale estimators,” *IEEE Access, 12, 12785–12796. (2024). https://doi.org/10.1109/access.2024.3356731*, 2024.
37. P. J. Rousseeuw, “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis,” *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.
38. D. L. Davies and D. W. Bouldin, “A cluster separation measure,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 2, pp. 224–227, 1979.
39. H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.
40. H. Ünlü, “Outlier detection using dbscan for large-scale datasets,” *Journal of Data Science and Analytics*, vol. 15, no. 2, pp. 123–135, 2023.
41. M. Pralav, “Anomaly detection using isolation forest in transactional data,” *International Journal of Computer Applications*, vol. 182, no. 30, pp. 21–27, 2020.
42. Y. Chou, “A practical outlier removal approach for medical data preprocessing,” *Health Informatics Journal*, vol. 20, no. 3, pp. 235–243, 2014.
43. e. a. Sarthak, “Deep learning-based prediction of diabetic readmission using ehr data,” *IEEE Journal of Biomedical and Health Informatics*, 2020.
44. A. Zarghani, “Comparative analysis of gradient-boosted models and lstm for diabetic patient readmission,” *arXiv preprint arXiv:2401.01234*, 2024.
45. W. Zhang and C. Liu, “Customer segmentation using rfm and kmeans clustering: A case study on retail data,” *International Journal of Data Science and Analytics*, vol. 6, no. 2, pp. 100–110, 2021.
46. H. Wang and F. Li, “Anomaly detection in retail data using isolation forest and z-score techniques,” *Journal of Retail Analytics*, vol. 9, no. 1, pp. 45–52, 2022.
47. S. Ahmed and F. Alharthi, “Outlier detection via dbscan on retail transactions,” in *Proceedings of the International Conference on Big Data Analytics*, pp. 112–120, IEEE, 2020.
48. S. Abdullah and A. Nassar, “Customer segmentation based on lstm autoencoder representation learning,” *Expert Systems with Applications*, vol. 213, p. 118926, 2023.
49. R. Kumar and D. Sharma, “Scalable spark-based dbscan for large retail datasets,” in *2021 IEEE International Conference on Big Data (BigData)*, pp. 2641–2649, IEEE, 2021.
50. Y. Zhang, L. Wang, Y. Chen, and H. Liu, “Unsupervised deep anomaly detection on mnist with autoencoder variants,” *IEEE Transactions on Neural Networks and Learning Systems*, 2023.
51. L. Ruff, R. A. Vandermeulen, N. Görnitz, L. Deecke, S. A. Siddiqui, A. Binder, K.-R. Müller, and M. Kloft, “Deep one-class classification,” in *International Conference on Machine Learning*, pp. 4393–4402, PMLR, 2019.
52. S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, “High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning,” *Pattern Recognition*, vol. 58, pp. 121–134, 2016.