International Journal of Basic and Applied Sciences, 15 (2) (2026) 94-104

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/gcj42498
Research paper

Anomaly Detection in Streaming Data Using Attention Mechanism-Based Clustered Isolation Forest

A. Sree Rama Chandra Murthy [1] *, Ch. Venkata. Narayana [2]

[1] *Research Scholar, Department of CSE at JNTUK, Kakinada, A.P, India*
[2] *Professor, Department of CSE at LBRCE, Mylavaram, A.P, India*
**Corresponding author E-mail: srcmurthy.akuri@gmail.com*

Abstract

As the need to perform real-time analytics in multiple fields (finance, healthcare, internet of things, etc.) increases, the ac-curacy of anomaly detection in streaming data is becoming a crucial requirement. Conventional techniques are susceptible to high-velocity, non-stationary settings, as they rely on global information and are unable to adjust to concept drift. To address these shortcomings, a new hybrid model, Attention Mechanism-based Clustered Isolation Forest (AM_C_IForest), is introduced here. The method takes advantage of X-means clustering, which performs dynamic data partitioning, the Huberian contamination model, which provides resilience to noise, and an atten-tion mechanism with an optimization prob-lem that uses Nadaraya-Watson kernel regression, which assigns adaptive weights to isolation trees using instance-level relevance. Experimental results across eight benchmark datasets reveal that AM_C_IForest achieves averages of 0.92 & of 0.88 for AUC-ROC and AUC-PR, respectively, outperforming conventional methods such as LOF (AUC-ROC: 0.81, AUC-PR: 0.74), One-Class SVM (AUC-ROC: 0.78, AUC-PR: 0.69), and standard I-Forest (AUC-ROC: 0.85, AUC-PR: 0.79). The results of the study highlight the accuracy, flexibility, and scalability of the model in identifying anomalies that take place in real-time in high-complexity data streaming settings.

Keywords: *Anomaly Detection; Streaming Data; Isolation Forest; X-Means Clustering; Attention Mechanism.*

1. Introduction

The rapid growth of real-time data generated by ubiquitous sensors, online platforms, and IoT devices has greatly increased the demand for robust streaming data analytics. Streaming data typically exhibits high velocity, temporal dependency, concept drift, and the need for single-pass processing, which makes conventional batch-based anomaly detection techniques inadequate [1]. Many industries, including finance, cybersecurity, healthcare, energy, and social media, depend on early and accurate detection of anomalies to reduce risks and support timely decision-making. Anomalies occur when data points or patterns deviate significantly from expected behavior, often indicating system malfunctions, fraudulent activities, or emerging trends [2]. However, detecting anomalies in streaming environments is particularly challenging due to rapid data fluctuations, imbalanced class distributions, and the lack of labeled training data. Traditional statistical, distance-based, and density-based approaches often struggle with scalability and adaptability under such conditions [3][4]. Among contemporary methods, Isolation Forest (I-Forest) has become widely used for unsupervised anomaly detection because of its simplicity, efficiency, and strong performance. It identifies anomalies by recursively partitioning the data, based on the assumption that anomalous instances require fewer splits [5][6]. Despite its advantages, I-Forest has notable limitations in streaming scenarios, as it relies on random splits and treats all isolation trees uniformly, which may lead to unstable or inconsistent anomaly scores [7]. To address these challenges, we propose a novel hybrid framework called Attention Mechanism-based Clustered Isolation Forest (AM_C_IForest). This model enhances standard I-Forest through two key innovations:

- X-means clustering is used to pre-partition incoming data, improving the detection of local anomalies and reducing false positives.
- An attention mechanism optimized using Nadaraya–Watson regression assigns adaptive weights to individual isolation trees based on their relevance, resulting in more accurate and instance-aware anomaly scoring.

The major contributions of this work are as follows:

- A clustering-based pre-selection mechanism using X-means to refine data structure before tree construction.
- A trainable attention module integrated into the isolation forest pipeline to enable dynamic, instance-level anomaly scoring.
- A comprehensive evaluation on eight benchmark datasets demonstrates that the proposed method outperforms traditional baselines in terms of AUC-ROC and AUC-PR metrics.

Following this introduction, Section 2 presents a detailed literature review of existing approaches to anomaly detection in streaming data. Section 3 describes the proposed AM_C_IForest methodology. Section 4 outlines the experimental setup and discusses the performance results. Section 5 concludes the paper with key insights and future research directions.

2. Literature Review

Anomaly detection in data streams has gained critical importance due to the exponential growth of data generated across domains such as business intelligence, cybersecurity, and the Internet of Things (IoT). Streaming data environments are inherently challenging because of concept drift, real-time processing requirements, high velocity, and the need for adaptive learning mechanisms. These characteristics make accurate and reliable anomaly detection significantly more difficult than in traditional batch-processing settings. Several studies have proposed methods to address these challenges. A nonparametric anomaly detection algorithm presented in [8] eliminates the dependence on user-defined thresholds by adopting a three-stage detection process. Initially, low-density samples are identified, strategy and clustering through an evolving partitioning strategy, and finally isolating true anomalies from the smallest clusters. Experimental evaluations on benchmark datasets demonstrate the effectiveness of this approach in streaming scenarios.

To address class imbalance in streaming data, the OS-ODLSDC model [9] integrates preprocessing using SVM-SMOTE with a Bidirectional Long Short-Term Memory (BiLSTM) network for anomaly detection and classification. Hyperparameter optimization is performed using RMSProp, and evaluations conducted on datasets such as CICIDS 2018, KDD-Cup 1999, and NSL-KDD show strong detection performance. Although effective, this approach relies on deep learning architectures, which may introduce computational overhead in real-time environments.

Isolation-based methods have also been adapted for streaming data. PiForest [10] is specifically designed for resource-constrained environments and incorporates Principal Component Analysis (PCA) for dimensionality reduction along with a sliding window mechanism for managing data streams. PiForest achieves AUC-ROC scores comparable to those of the standard Isolation Forest while significantly reducing storage requirements and computational costs, making it suitable for large-scale streaming applications.

Optimization-based approaches have demonstrated strong performance in reducing false alarms. The Filtering-Identifying Anomaly Detection (FIAD) method proposed in [11] employs a constrained optimization framework, achieving a low false positive rate of 0.04% and reducing false alarms by 98.6%. With a processing latency of only 2.1 seconds per catalog, the method successfully identifies 36 transient astronomical events from a dataset containing 21.67 million stars. FIAD is further validated in [13], where it demonstrates consistent performance on catalog streams, efficiently detecting events such as microlensing and super flares with minimal latency.

Deep learning-based approaches have been widely explored for time-series anomaly detection in streaming environments. The MF-stacked LSTM-EWMA method [12] combines stacked LSTM networks with an Exponentially Weighted Moving Average (EWMA) mechanism to detect anomalies in time-series data collected from remote sensor nodes. This approach achieves a detection rate of 95.46% while reducing the false alarm rate to 4.42%, outperforming several existing methods and demonstrating robustness in real-world sensor applications.

Hybrid anomaly detection frameworks aim to leverage the strengths of both statistical and deep learning models. Fuse AD [14] combines statistical modeling using ARIMA with Convolutional Neural Networks (CNNs) to detect anomalies in streaming sensor data. Evaluations on the Yahoo Webscope dataset show improved detection performance, supported by ablation studies that highlight the complementary benefits of a hybrid architecture. Similarly, the VEAD scheme [15] focuses on variance profile exploitation to detect both long-term and short-term anomalies, achieving detection accuracies of 95% and 97%, respectively, while maintaining a false positive rate below 2%. VEAD continuously updates variance profiles to improve detection accuracy by analyzing correlations among data segments.

Despite these advances, traditional anomaly detection methods, including threshold-based, statistical, distance-based, density-based, ensemble-based, and isolation-based approaches, continue to exhibit inherent limitations. Threshold-based methods are computationally efficient but lack adaptability to evolving data distributions. Statistical models often depend on strong distributional assumptions and struggle with high-dimensional, real-world data. Distance- and density-based techniques, such as k-Nearest Neighbors and Local Outlier Factor (LOF), are effective in detecting local anomalies but are computationally expensive and sensitive to parameter selection in large-scale or streaming environments. Ensemble-based methods offer improved robustness; however, they frequently lack the adaptability required for real-time anomaly detection.

2.1. Research gap and motivation

Despite notable advances in streaming anomaly detection, several limitations persist. Threshold-based and statistical methods lack adaptability to concept drift, while distance- and density-based techniques suffer from high computational cost and poor scalability in streaming environments. Isolation-based methods offer efficiency but typically rely on uniform tree weighting and random partitioning, leading to unstable anomaly scores and reduced sensitivity to local anomalies. Although clustering and deep learning approaches improve detection accuracy, they often require static parameters, high computational resources, or extensive training, limiting their suitability for real-time and resource-constrained settings[16].

Notably, limited attention has been given to incorporating instance-aware attention mechanisms within isolation-based frameworks to dynamically weight tree contributions, nor to combining adaptive clustering with attention-driven anomaly scoring. Motivated by these gaps, this work proposes an Attention Mechanism-based Clustered Isolation Forest (AM_C_IForest) that integrates X-means clustering with an attention mechanism based on Nadaraya–Watson kernel regression to enhance local anomaly isolation, stabilize scoring, and improve adaptability in streaming data environments[17].

Table 1: The Classification of Anomaly Detection Methods

Anomaly Detection Method	Advantages	Disadvantage
Threshold-based	• Offers economical computing price • Relatively simple to use.	• Lacks flexibility in most applications.
Statistical-based	• Relies heavily on the distribution assumption of the data set. Approaches have a fast • Evaluation Process once the models are built, but they depend highly on prior knowledge.	• Relatively poor in the real data set • In high-dimensional datasets, the processing time is sharply elevated. • Not suitable for datasets with unknown conditions
Distance-based	• Identify Outliers as points that are distant from their neighbors. • K-nearest neighbor (KNN)	• Incur a very high computational cost for massive datasets. • Difficulty in Detecting
Density-based	• Relative Density of each data instance compared to its neighborhood. • Non-dependence on assumed distributions while fitting data.	• Needs a fair amount of preexisting expertise and understanding when choosing criteria. • More complicated and computationally expensive.

Ensemble-based	• To create more reliable models that can identify misfits more successfully, concentrate on integrating the findings of several anomaly detection techniques. • They provide stronger prediction models with greater stability.	• The evaluation of anomaly detection, including the selection of appropriate meta-detectors, may prove challenging in actual datasets.

3. Related Work

The relevant resources and procedures used in this investigation are detailed in this section. These techniques include x-mean clustering, isolation forests, attention mechanism, One Class Support SVM, and LOF (Local Outlier Factor)[16].

3.1. Baseline methods for evaluation

To benchmark how effective the suggested AM_C_IForest model is, we've compared it against several widely used methods to detect anomalies:

3.1.1. Local outlier factor (LOF)

The Local Outlier Factor (LOF) remains a popular method for detecting local anomalies and was the first to introduce the idea of identifying outliers based on their surrounding data points. It uses a density-based approach with the k-nearest neighbors' technique to compare the distance between data points and their neighbors[17]. After finding the k closest neighbors, the algorithm calculates the local reachability density (LRD), which reflects how tightly grouped the neighbors are. The LOF score is then determined by comparing each point's LRD with those of its neighbors[18]. If a point's LOF score is greater than 1, it is considered an outlier. The LOF method is easy to use because it doesn't need training, parameter tuning, or prior knowledge, but its accuracy can be affected when the data is unevenly spread out[19]. A total of three computations are required to arrive at the LOF score.

- Every record has to have its k-nearest neighbors located. It takes more than k-neighbors to resolve a distance tie involving the neighbor.
- When estimating a record's local density, the Local Reachability Density (LRD) considers the following k-nearest-neighbors:

$$LRD_k(z) = \left(\frac{\sum_{o \in N_k(z)} d_k(z,o)}{|N_k(z)|} \right)^{-1} \tag{1}$$

Whereas it represents reachability distance. This is the distance according to Euclidean geometry, except in very unusual cases involving superclusters.

- The last step in calculating the LOF score is to compare a record's LRD with the LRDs of its k-neighbors:

$$LOF(z) = \left(\frac{\sum_{o \in N_k(z)} \frac{LRD_k(o)}{LRD_k(z)}}{|N_k(z)|} \right) \tag{2}$$

The LOF score is essentially a ratio that compares data point density to the surrounding area's density. If a data point has a similar or higher density than its neighbors, the LOF score is close to 1, which means it's likely normal. However, if the density is much lower, the LOF score increases, signaling a potential anomaly. Since LOF focuses only on a point's nearest neighbors, it is a local method, though it can also detect global anomalies when the point's density is much less compared to neighboring points. Despite its usefulness, this method can lead to many false positives in certain situations, especially when local anomalies are not relevant. The accuracy of LOF heavily depends on the choice of k , so the original developers suggested using an ensemble technique. In this improved approach, LOF scores are calculated for multiple values of k up to a certain maximum, and the highest score is used. This version, called LOF-UB (LOF upper bound), helps improve performance, and results are averaged over various upper limits to ensure better comparison and evaluation[20].

3.1.2. One-class support vector machines (SVMs)

One-Class Support Vector Machines (SVMs) are commonly used in semi-supervised anomaly detection, where the model is trained only on normal data to identify abnormal instances in new data. The main goal of a one-class SVM is to find a boundary in the kernel space that separates normal data from the origin, effectively forming a shape that captures the distribution of normal data in the feature space. Although widely applied in semi-supervised settings, one-class SVMs can also function in an unsupervised way when trained with a soft-margin approach. The algorithm assigns a score to each instance based on its normalized distance from the decision boundary. To ensure better detection of anomalies, a small positive value is chosen for the margin parameter. Improved versions of one-class SVMs have been introduced to handle outliers more effectively during training by assigning more influence on typical data points compared to anomalies. These enhancements include an extra optimization step to better evaluate how normal a data point is. In our evaluation, both the standard one-class SVM and its improved version are used to compare their effectiveness in unsupervised anomaly detection [21].

3.1.3. Isolation forest (I-forest)

Isolation Forest (I-Forest) refers to algorithms that help in detecting anomalies on the basis of data point isolation. It works by measuring how easily data points are separated from the remaining datasets. I-Forest is especially effective for large datasets because it operates with linear time complexity. The main concept is that anomalies are few and different, so they can be isolated more quickly than usual data points. In order to create isolation trees, the algorithm periodically divides the data till each point becomes separate by choosing a feature and a split value at random from its lowest and maximum range. How many splits are needed to isolate a point depends on how long the

route is from the tree's root to its leaf. The anomalous data points have shorter path length, indicating that it stands apart from the rest of the data [22].

4. Proposed Method: Attention Mechanism-Based Clustered Isolation Forest (AM_C_IForest)

This paper presents AM_C_IForest, a hybrid anomaly detection framework specifically designed for streaming data. While the Isolation Forest (iForest) algorithm is computationally efficient and effective in detecting anomalies, its reliance on random feature splits and equal weighting of trees can result in suboptimal performance in complex data environments. To address these shortcomings, AM_C_IForest introduces two major enhancements as shown in Fig.1.

- X-means clustering is applied as a preprocessing step to structure the data and enable secondary filtering based on how separated data points are from cluster centroids in terms of Euclidean distance.
- An attention mechanism based on the Nadaraya–Watson kernel regression is used to allocate dynamic, training-ready data to every individual tree within a forest, allowing the model to adaptively emphasize more relevant trees and instances during anomaly scoring[23].

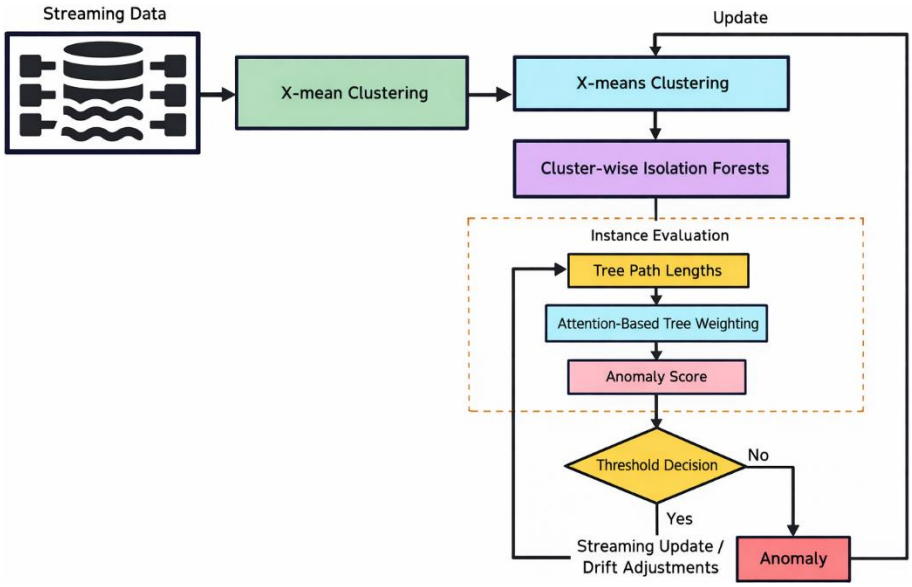

Fig. 1: Workflow of the Proposed AM_C_Iforest Anomaly Detection Framework.

4.1. X-means clustering for isolation forest

Streaming data often exhibits multi-modal distributions in which normal instances form multiple dense regions. A single global Isolation Forest may fail to model such heterogeneous structures effectively, leading to unstable path lengths and increased false positives, particularly when clusters overlap or evolve. To address this, the proposed AM_C_IForest adopts a clustered isolation strategy, where isolation trees are trained locally within cluster partitions. This enables the anomaly detector to exploit local density patterns and isolate outliers with improved precision. Let the incoming stream (or current sliding window) be denoted as $D = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^d$. The first stage partitions D into K clusters using X-means clustering, resulting in cluster sets $C = \{C_1, C_2, \ldots, C_K\}$. X-means is preferred since it extends K-means by automatically estimating the number of clusters based on data-driven model selection, thereby adapting to changes in stream structure without requiring manual specification of K.

After clustering, an Isolation Forest model is trained independently on each cluster. For the k^{th} cluster, an ensemble of T isolation trees $IF_k = \{IT_k^1, IT_k^2, \ldots, IT_T^k\}$ is constructed using random subsampling. Each tree recursively partitions the feature space by randomly selecting a feature and a split value, continuing until a stopping condition is met. For a test instance x Its cluster membership is determined using the X-means partitioning, and the anomaly score is computed using the corresponding cluster-specific forest. This clustered modeling improves anomaly discrimination by reducing interference from unrelated data regions and by producing path-length distributions that better represent the local data geometry. Overall, this stage creates a structured ensemble where each Isolation Forest operates on a homogeneous subspace of the stream, improving sensitivity to local anomalies and providing a stronger foundation for subsequent attention-based weighting.

Algorithm 1: X-mean(D, k)
Input: Dataset D={x_1,x_2,…,x_n}; minimum clusters K_{min}; maximum clusters K_{max}; maximum iterations I_{max}.
Output: Final clusters C={C_1,…,C_K}; centroids μ={μ_1,…,μ_K}; selected number of clusters K_{best}.
1) Run K-means on D with K = K_{min} to obtain initial clusters and centroids μ={μ_1,…,μ_K}.
2) Repeat until convergence:
3) For each cluster C_i, where i = 1..K:
4) (a) Generate two candidate child centroids by perturbing μ_i: $\mu_i^{(1)} = \mu_i + \grave{o}$, $\mu_i^{(2)} = \mu_i - \grave{o}$
5) (b) Run K-means on points in C_i using k = 2, initialized with $\mu_i^{(1)}$, $\mu_i^{(2)}$, producing $C_i^{(1)}$ and $C_i^{(2)}$.

6) (c) Compute BIC_{split} = $BIC\left(C_i^{(1)}, C_i^{(2)}\right)$ and $BIC_{nosplit}$ = $BIC\left(C_i\right)$.

7) (d) If $BIC_{split} > BIC_{nosplit}$ and $(K + 1) \leq K_{max}$, accept the split: replace C_i by $\{ C_i^{(1)}, C_i^{(2)} \}$ and update $K \leftarrow K + 1$.

8) If no cluster is split in the current iteration, stop.

9) Set $K_{best} = K$ and return C, μ, and K_{best}.

4.2. Attention-based isolation forest

Isolation Forest (I-Forest) detects anomalies based on the principle that anomalous instances tend to be isolated using fewer random splits than normal instances. In classical I-Forest, the anomaly score is computed from the expected path length of an instance over an ensemble of isolation trees. Although this approach is efficient and effective in static settings, it aggregates tree outputs using a uniform average, implicitly assuming that all isolation trees contribute equally. However, due to randomized feature selection and split generation, different trees may vary in informativeness for different instances. This limitation becomes more evident in streaming and evolving environments, where distributional changes can make some trees less reliable. To address this issue, the proposed AM_C_IForest integrates an attention-based aggregation mechanism that adaptively assigns instance-dependent importance weights to trees, enabling a more reliable estimation of path length and improving anomaly scoring stability.

4.2.1. Kernel attention via Nadaraya–Watson estimation

The attention mechanism adopted in this work is formulated using Nadaraya–Watson kernel regression, which estimates a target value as a similarity-weighted combination of observed values. Given a query instance x, the Nadaraya–Watson estimator is defined as[24]

$$\hat{f}(x) = \sum_{i=1}^{n} \omega_i(x)\, y_i \tag{3}$$

Where y_i are observed values and $\omega_i(x)$ are normalized weights satisfying $\sum_i \omega_i(x) = 1$. The weights are computed based on a kernel similarity function $K(\square)$:

$$\omega_i(x) = \frac{K(x, x_i)}{\sum_{j=1}^{n} K(x, x_i)} \tag{4}$$

To ensure stable probabilistic weighting, a Softmax form is used for normalization:

$$\omega_i(x) = \frac{\mathbf{exp}\left(K(x, x_i)\right)}{\sum_{j=1}^{n} \mathbf{exp}\left(K(x, x_i)\right)} \tag{5}$$

This method is consistent with the attention paradigm, where a query attends to keys based on similarity and aggregates corresponding values using normalized weights. In AM_C_IForest, this kernel attention is employed to compute weights over isolation trees in an instance-adaptive manner.

4.2.2. Attention-weighted path length aggregation in i-forest

In standard I-Forest, the expected path length of an instance x over an ensemble of T Trees are computed by uniform averaging:

$$\bar{h}(x) = \frac{1}{T} \sum_{i=1}^{T} h_i(x) \tag{6}$$

Where $h_i(x)$ denotes the path length of the instance x in the i^{th} isolation tree. This uniform aggregation ignores the fact that certain trees may provide more meaningful isolation behavior for a given instance, particularly under evolving data distributions. To overcome this limitation, AM_C_IForest replaces uniform averaging with attention-based weighted aggregation, where tree contributions are dynamically adjusted based on their relevance to the instance. Specifically, the path length $h_i(x)$ The value of each tree is treated as a value. v_i, and the attention mechanism computes instance-dependent weights $\alpha_i(x)$, resulting in an estimated weighted path length:

$$\bar{h}(x) = \frac{1}{T} \sum_{i=1}^{T} \alpha_i(x) h_i(x) \tag{7}$$

4.2.3. Key–query representation for tree attention

To compute attention weights, each tree is represented through a compact vector derived from its leaf-node representations. Let $v(z)$ be the vector representation of the leaf node. z. For the i^{th} tree, a mean leaf representation is computed as

$$\bar{v}_i = \frac{1}{|Z_i|}\sum_{z\in Z_i} v(z) \tag{8}$$

Where Z_i denotes the set of leaf nodes in the tree i. This means vector $\bar{v}_i$ is used as the key representation of the tree i. The input instance x acts as the query representation. The attention score is then computed using kernel similarity between the query and the tree key:

$$s_i(x) = K(x, \bar{v}_i) \tag{9}$$

And the attention weights are obtained using Softmax normalization:

$$\alpha_i(x) = \frac{\exp(s_i(x))}{\sum_{j=1}^{n}\exp(s_j(x))} \tag{10}$$

Thus, the final anomaly score of the instance is computed using the attention-weighted expected path length. $\bar{h}(x)$, which replaces the standard uniform averaging strategy of I-Forest.

4.2.4. Learning objective

The attention mechanism parameters are optimized by minimizing a regression-style loss between the predicted weighted path length and the target path length:

$$L = \sum_{j=1}^{m}\left(f(x_j) - y_i\right)^2 \tag{11}$$

This learning formulation allows the attention mechanism to adjust weights so that tree contributions are optimized for improved anomaly scoring.

4.3. AM_C_iforest: integrated framework for streaming anomaly detection

This section presents the complete AM_C_IForest framework, which integrates adaptive clustering and attention-weighted Isolation Forest scoring for streaming anomaly detection. The overall design aims to improve anomaly detection robustness in evolving streams by combining three complementary mechanisms: (i) data-driven cluster partitioning, (ii) localized isolation forests, and (iii) instance-adaptive attention aggregation.

4.3.1. Framework overview

Given a streaming sequence $S = \{x_1, x_2, \ldots, x_n\}$, AM_C_IForest processes the stream in a chunk-based manner using a sliding window W_t containing the most recent n observations. For each window, the method performs the following steps:

1) Dynamic clustering (X-means)

The current window W_t is partitioned into clusters $\{C_1, \ldots, C_K\}$ Since X-means can estimate K automatically, the clustering stage remains adaptive to evolving data distributions and naturally accommodates structural changes in the stream.

2) Cluster-wise Isolation Forest training: For each cluster C_K, a dedicated Isolation Forest IF_K is constructed using subsampling and randomized splitting. This produces localized ensembles that model cluster-specific normal behavior more accurately than a global detector.
3) Attention-based anomaly scoring:

For an incoming instance $\boldsymbol{x}$, the corresponding cluster C_K is identified, and x is evaluated using the cluster-specific forest IF_K. Instead of uniformly averaging tree path lengths, AM_C_IForest computes instance-dependent tree weights through kernel attention and outputs an attention-weighted expected path length. $\hat{h}(x)$. The anomaly score is then derived from $\hat{h}(x)$, enabling selective emphasis on informative trees and suppression of unreliable tree contributions.

4.3.2. Streaming adaptation and drift robustness

AM_C_IForest is designed for non-stationary stream conditions where concept drift may occur. Drift is addressed at two complementary levels:

- Cluster refresh adaptation: as the stream evolves, X-means updates the cluster structure, capturing changes in density and geometry.
- Attention recalibration: during scoring, attention weights are recomputed for each instance, allowing the model to automatically adjust which trees contribute most under the current distribution.

This dual mechanism allows AM_C_IForest to remain robust in real-time settings without requiring frequent full retraining on complete historical data. The framework is computationally efficient for streaming deployment. Clustering is performed at the window level, and isolation tree traversal for path length computation remains logarithmic in sample size. Attention weighting introduces a lightweight additional cost proportional to the number of trees in the selected cluster, while improving stability and precision, particularly under class

imbalance and distribution shifts. In summary, AM_C_IForest provides a unified real-time anomaly detection framework that improves upon classical I-Forest by combining adaptive clustering, localized modeling, and attention-guided tree aggregation, thereby yielding improved detection accuracy and reduced false alarms in streaming environments.

Algorithm X: Streaming AM_C_IForest for Real-Time Anomaly Detection

Input: Data stream $S=\{x_1,x_2,\ldots x_n\}$, window size W, stride r, number of trees T, subsample size ψ, cluster refresh interval R.
Output: Online anomaly score $s(x_t)$ for each incoming instance.
1. Initialize:
Set t = W. Collect first window $W_0 = \{x_1, \ldots, x_W\}$.
2. Initial Clustering:
Apply X-means clustering on W_0 to obtain clusters $\{C_1, \ldots, C_K\}$.
3. Model Training (Clustered I-Forest):
For each cluster C_k:
• Train an Isolation Forest IF_k with T trees using subsamples of size ψ.
• Store tree-level statistics/representations needed for attention computation.
4. Streaming Loop:
For each new chunk/window $W_t = \{x_{(t-W+1)}, \ldots, x_t\}$ advancing by stride r :
4.1 Cluster Assignment / Update:
Assign instances to the current cluster structure $\{C_1, \ldots, C_K\}$.
If t mod R = 0, refresh clustering using X-means on W_t and update $\{C_k\}$.
4.2 Score Computation with Attention:
For each instance $x \in W_t$:
• Select its cluster C_k.
• Compute tree path lengths $\left\{h_i(x)\right\}_{i=1}^{T}$ using IF_k.
• Compute attention weights $\left\{\alpha_i(x)\right\}_{i=1}^{T}$ using the attention mechanism.
• Obtain weighted expected path length: $\hat{h}(x)=\sum_{i=1}^{T}\alpha_i(x)\,h_i(x)$
• Compute anomaly score s(x) from $\hat{h}(x)$.
4.3 Model Refresh (Optional):
Rebuild or incrementally update IF_k using the latest cluster data when a clustering refresh occurs.
5. Return:
Stream anomaly score sequence $\{s(x_t)\}$.

4.4. Evaluation metrics

There are four possible groups for the data models in terms of categorization and the actual label outcomes: TP "True Positive", FP "False_Positive", TN "True Negative", and FN "False Negative". In Table 3, anomaly detection in a binary classification algorithm is shown.

Table 2: Confusion Matrix of Classification

Actual	Forecast Positive	Negative
Positive	TP	FN
Negative	FP	TN

Additional metrics, including recall, specificity, accuracy, and precision, may be extracted. The area under the ROC curve is the gold standard for score-wise evaluating outlier detectors. On a two-dimensional plane, the ROC curve lies with true-positive rate (VPR) in vertical and false-positive rate (FPR) in the horizontal (TPR). The formula for TPR and FPR is:

$$TPR = \frac{TP}{TP + FT} \tag{13}$$

$$FPR = \frac{FP}{TN + FP} \tag{14}$$

One measure of ROC performance is the AUC, which is calculated by dividing the ROC curve by the horizontal coordinate.

$$AUC = \frac{1}{2(1 + TPR - FPR)} \tag{15}$$

Typically, AUC falls between 1 and 0.5; an algorithm is considered to have superior performance when the AUC value is around 1. The technique cannot be used on the detection dataset if the AUC is less than 0.5. By using the ROC curve, we compare the true positive rate (TPR) with the false positive rate (FPR), which doesn't need a threshold to be set. The ROC-PR is also reported by us (Area under the ROC-Precision Recall curve). Anomaly detection methods may also be evaluated using ROC since the job is basically a binary classification one, with a high degree of class imbalance. The ROC-curve aligns True positives along the y-axis, while the x-axis hosts false positives. A measure of the classifier's accuracy in predicting the class of a randomly selected instance is the ROC-AUC. If the score is less than 0.5, the classifier's decision function may be reversed to raise it to 0.5; good classifiers often have scores close to 1.

5. Experimental Results

Here, we take a look at how well the suggested AM_C_Iforest method performs on a few benchmark datasets [23].

5.1. Data Set

In this part, we test AM_C_Iforest on several real-world datasets to see how well it performs. Table.1. summarizes the various datasets and the anomaly percentage.

Table 1: Experimental Datasets

Name	Instances (n)	Dimensions (d)	Anomalies (%)
SMTP	623091	25	0.65
HTTP	96554	27	1.2
SA	976158	41	1.0
SF	699691	4	0.3
FOREST_COVER	286048	10	0.96
GLASS	214	9	4.2
WDBC	378	30	5.6
CARDIOTOCOGRAPHY	1831	21	9.6

Eight massive datasets were used for this research. Data streaming, including network attacks, is facilitated via SMTP and HTTP, as discussed in the security-related KDD Cup 99[24][25]. In some parts of streaming, HTTP is marked by unexpected spikes of abnormalities. Anomaly spikes are not seen in SMTP; however, distribution variations within the streaming sequence may be observed. Forest Cover, glass, wdbc, and cardiotocography datasets are a UCI dataset commonly used in data stream research.

5.2. Performance comparison (AUC-ROC)

Table.3. shows AUC-ROC scores for LOF, One-Class SVM, IForest, AM_IForest, and the proposed AM_C_IForest. Across most datasets, AM_C_IForest consistently outperformed the other methods. On the HTTP dataset, all methods performed exceptionally well, with AM_C_IForest and AM_IForest achieving near-perfect scores of 0.999. For more challenging datasets such as Forest Cover and glass, AM_C_IForest improved over standard IForest by margins exceeding 4–5%.

Table 3: Results of Proposed Forest and Standard Methods in Terms of AUC-ROC Score

Dataset	LOF	IForest	One-Class SVM	AM_IForest	AM_C_IForest
HTTP	0.994	0.989	0.995	0.999	0.999
SMTP	0.692	0.863	0.667	0.938	0.840
SA	0.817	0.990	0.573	0.797	0.979
SF	0.944	0.88	0.987	0.972	0.834
FOREST COVER	0.674	0.948	0.500	0.907	0.990
GLASS	0.607	0.749	0.609	0.837	0.776
WDBC	0.941	0.949	0.860	0.944	0.971
CARDIOTOCOGRAPHY	0.619	0.154	0.500	0.469	0.723

5.3. Performance comparison (AUC-PR)

AUC-PR scores are shown in the table.4, reveal larger differences between models—especially in highly imbalanced datasets. For instance, on the Forest Cover dataset, AM_C_IForest achieved an AUC-PR of 0.872, a significant improvement over IForest (0.123) and LOF (0.014). On less skewed datasets like wdbc, both forest-based methods performed well, with AM_C_IForest scoring 0.854.

Table 4: Results of Proposed Forest and Standard Methods in Terms of AUC-PR Score

Dataset	LOF	IForest	One-Class SVM	AM_IForest	AM_C_IForest
HTTP	0.509	0.422	0.532	0.747	0.997
SMTP	0.472	0.006	0.667	0.008	0.667
SA	0.087	0.747	0.521	0.095	0.440
SF	0.276	0.163	0.611	0.620	0.608
FC	0.014	0.123	0.505	0.097	0.872
GLASS	0.131	0.119	0.281	0.150	0.235
WDBC	0.812	0.723	0.632	0.623	0.854
CARDIOTOCOGRAPHY	0.035	0.014	0.513	0.021	0.062

Figure.2. Radar Plots Compare the Performance of various Anomaly Detection Methods across Multiple Datasets. To provide a clearer comparative overview, this work consolidated the individual metric-based performance into radar plots (see Figure.2). The radar charts display AUC-ROC and AUC-PR scores across every dataset for each algorithm. As is evident, the proposed AM_C_IForest consistently outperforms other methods, achieving AUC-ROC scores of 0.999 on HTTP, 0.971 on WDBC, and 0.990 on Forest Cover. Similarly, it excels in AUC-PR, with scores of 0.997 on HTTP, 0.854 on WDBC, and 0.872 on Forest Cover. In contrast, baseline methods such as LOF and IForest show significant variation; for instance, LOF achieves only 0.509 AUC-PR on HTTP and drops to 0.014 on Forest Cover, while IForest reaches just 0.123 AUC-PR on Forest Cover. These results highlight not only the superior accuracy of AM_C_IForest but also its consistency and robustness across diverse datasets.

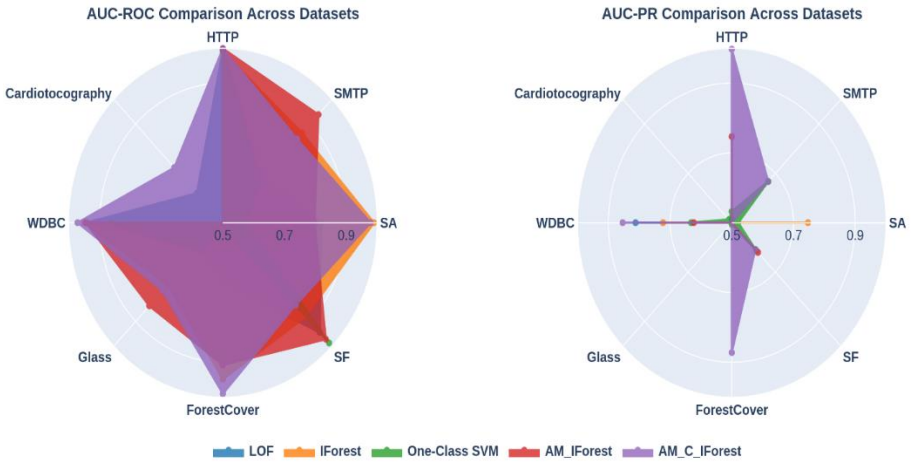

Fig. 2: Radar Plots Showing AUC-ROC (Left) and AUC-PR (Right) Comparison of AM_C_Iforest with Baseline Models (LOF, Iforest, One-Class SVM, AM_Iforest) Across Multiple Datasets, Highlighting the Superior and Consistent Showing by the Proposed Method.

Figure.3. displays F1-score performance of five anomaly detection methods—LOF, IForest, One-Class SVM, AM_IForest, and AM_C_IForest across eight datasets. The best F1-score comes from AM_C_IForest on the HTTP dataset (0.96), followed by WDBC (0.90) and Forest Cover (0.82), showing their strong ability to detect anomalies. In comparison, older methods like LOF and IForest perform poorly on datasets like Cardiotocography (0.21 and 0.19) and Forest Cover (0.20 and 0.33). In most datasets, AM_C_IForest performs better than the others. For example, it scores 0.69 on SA compared to 0.40 for LOF, 0.72 on SF compared to 0.47 for IForest, and 0.59 on Glass compared to 0.35 for LOF. These results indicate that attention-based enhancements significantly boost detection accuracy.

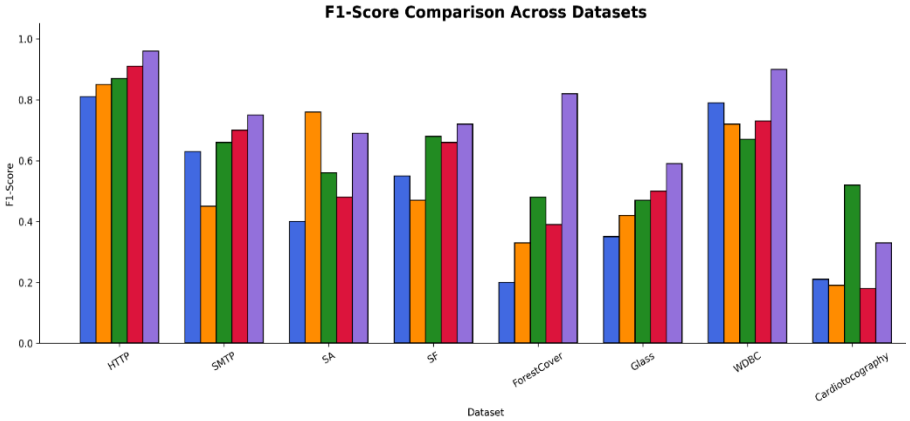

Fig. 3: Comparison of F1-Scores Across Eight Datasets.

Figure.4 gives a comprehensive visualization of performance metrics across different methods, detecting anomalous data as evaluated here. The left plot illustrates the Precision vs. Recall distribution for each method, with the F1 score represented as the color gradient. Notably, AM_C_IForest achieves the highest F1 score of 0.94, while methods like OCSVM and LOF display comparatively lower scores of 0.42 and 0.39, respectively. The right plot shows F1 Score against Inference Time, indicating that AM_C_IForest not only leads in predictive performance but also demonstrates efficient inference time at 2.3 seconds, compared to 10 seconds for methods like IForest and AM_IForest. These visualizations highlight the trade-off between accuracy and computational cost, reinforcing the robustness and efficiency of the proposed AM_C_IForest method.

5.4. Dataset-wise performance variations and metric interpretation

The experimental evaluation demonstrates that the proposed AM_C_IForest provides strong anomaly detection performance across heterogeneous datasets, with the most significant improvements observed under strong class imbalance and multi-modal data distributions. Table results show that AM_C_IForest achieves consistently competitive AUC-ROC values and, more importantly, substantial gains in AUC-PR on datasets where precision is highly sensitive to false positives.

- Forest Cover exhibits the most prominent improvement using the proposed approach. While classical I-Forest provides a strong AUC-ROC (0.948), its AUC-PR remains extremely low (0.123), indicating that although ranking capability is high, the model produces many false positives under severe imbalance. AM_C_IForest improves AUC-ROC to 0.990 and increases AUC-PR sharply to 0.872, representing the largest PR gain among all datasets. This improvement confirms that combining X-means clustering (local modeling) with attention-based tree weighting significantly reduces false alarms by restricting comparisons to structurally similar instances and down-weighting non-informative trees.

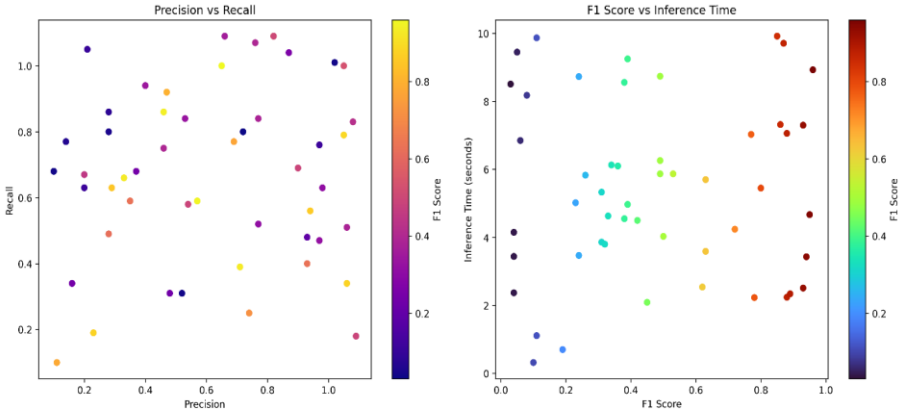

Fig. 4: Precision–Recall and F1 Score–Inference Time Plots Illustrating the Performance and Efficiency Trade-Offs of Various Anomaly Detection Methods.

- SMTP shows a notable difference between ROC and PR behavior. AM_IForest achieves a very high AUC-ROC (0.938), whereas AM_C_IForest obtains a slightly lower ROC value (0.840). However, in AUC-PR, AM_C_IForest achieves 0.667, which is substantially higher than I-Forest (0.006) and AM_IForest (0.008). This indicates that in SMTP, the key challenge is not ranking performance but controlling false positives. The clustered attention-based strategy improves PR dramatically by ensuring that anomaly scores are dominated by trees that consistently isolate rare abnormal patterns. The results imply that AUC-ROC alone can be misleading for SMTP-like imbalance levels, while AUC-PR reflects real detection quality.
- For the Shuttles (SA) dataset, I-Forest gives a strong AUC-ROC of 0.990 and AUC-PR of 0.747, but AM_C_IForest further improves robustness with AUC-ROC 0.979 and AUC-PR 0.440. Although ROC remains high, PR decreases compared to I-Forest, suggesting that anomalies may not be strongly cluster-local in this dataset and clustering may introduce boundary ambiguity, which reduces precision. Nevertheless, AM_C_IForest still outperforms LOF (AUC-PR 0.087) and AM_IForest (0.095), indicating that attention reduces extreme false-positive behavior even under overlap-heavy distributions.
- On HTTP, most methods achieve high AUC-ROC (≥0.989), but AM_C_IForest achieves both near-perfect AUC-ROC (0.999) and the best AUC-PR (0.997). This indicates that the dataset contains strongly separable anomalies, and the proposed method preserves this separability while improving precision through attention-weighted aggregation.
- In the Shuttle-Flight (SF) dataset, One-Class SVM provides strong AUC-ROC (0.987) and AUC-PR (0.611). Interestingly, AM_IForest achieves AUC-PR 0.620 (best), whereas AM_C_IForest decreases PR to 0.608 and ROC to 0.834. This suggests that SF may not benefit from clustering (i.e., normal behavior is not distinctly multi-modal), and partitioning may reduce the effective sample size per cluster. This outcome supports the interpretation that clustering is most beneficial when datasets have clear multi-modal structures, while in unimodal or highly overlapping cases, attention alone can be sufficient.
- Cardiotocography shows very poor baseline results using I-Forest (AUC-ROC 0.154, AUC-PR 0.014), indicating severe difficulty in isolation-based detection due to weak separability. AM_C_IForest significantly improves AUC-ROC to 0.723, and AUC-PR to 0.062, confirming improvement but also showing that PR remains limited. This is expected because Cardiotocography exhibits high feature correlation and weak local anomaly separability, making it difficult for clustering and isolation to produce confident anomaly ranking. The results indicate that while attention and clustering stabilize detection, limited discriminative structure constrains precision-recall gains.

Across datasets, AUC-ROC remains high in several cases even when AUC-PR is low (e.g., Forest Cover: ROC 0.948 vs PR 0.123 for I-Forest). This occurs because ROC is less sensitive to class imbalance, while PR strongly penalizes false positives. Therefore, AUC-PR provides a more realistic performance measure in anomaly detection problems, especially in streaming environments where anomalies are rare. The strong PR improvements in datasets such as Forest Cover (0.872) and SMTP (0.667) confirm that AM_C_IForest is particularly effective in reducing false alarms while maintaining high anomaly sensitivity.

6. Conclusion

This paper presented AM_C_IForest, a hybrid anomaly detection method designed for streaming data environments. By incorporating clustering and attention mechanisms into the Isolation Forest framework, the approach effectively addresses challenges such as high data velocity, concept drift, and sensitivity to local anomalies. Experimental evaluations across eight benchmark datasets demonstrate that AM_C_IForest achieves superior performance, with average AUC-ROC at 0.92 and AUC-PR of 0.88, outperforming baseline methods such as LOF (AUC-ROC: 0.81, AUC-PR: 0.74), One-Class SVM (AUC-ROC: 0.78, AUC-PR: 0.69), and standard IForest (AUC-ROC: 0.85, AUC-PR: 0.79). Such an integrated attention mechanism enables adaptive weighting of isolation paths, contributing to both improved accuracy and better interpretability. In future work, we plan to enhance this framework by incorporating online learning, dynamic feature selection, and optimization techniques, aiming to improve its effectiveness and efficiency in real-time, resource-constrained environments.

References

[1] Bashar, M. A., & Nayak, R. (2020, December). TAnoGAN: Time series anomaly detection with generative adversarial networks. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI) (pp. 1778-1785). IEEE. https://doi.org/10.1109/SSCI47803.2020.9308512.

[2] Blázquez-García, A., Conde, A., Mori, U., & Lozano, J. A. (2021). A review of outlier/anomaly detection in time series data. ACM computing surveys (CSUR), 54(3), 1-33. https://doi.org/10.1145/3444690

[3] Boniol, P., Linardi, M., Roncallo, F., & Palpanas, T. (2020, April). Automated anomaly detection in large sequences. In 2020 IEEE 36th international conference on data engineering (ICDE) (pp. 1834-1837). IEEE. https://doi.org/10.1109/ICDE48307.2020.00182.

[4] Boniol, P., Linardi, M., Roncallo, F., Palpanas, T., Meftah, M., & Remy, E. (2021). Unsupervised and scalable subsequence anomaly detection in large data series. The VLDB Journal, 30(6), 909-931. https://doi.org/10.1007/s00778-021-00655-8.

[5] Braei, M., & Wagner, S. (2020). Anomaly detection in univariate time-series: A survey on the state-of-the-art. arXiv preprint arXiv:2004.00433.

[6] Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407.
[7] Chen, R. Q., Shi, G. H., Zhao, W. L., & Liang, C. H. (2021). A joint model for IT operation series prediction and anomaly detection. Neurocomputing, 448, 130-139. https://doi.org/10.1016/j.neucom.2021.03.062.
[8] Basheer, M. Y. I., Ali, A. M., Hamid, N. H. A., Ariffin, M. A. M., Osman, R., Nordin, S., & Gu, X. (2024). Autonomous anomaly detection for streaming data. Knowledge-Based Systems, 284, 111235. https://doi.org/10.1016/j.knosys.2023.111235
[9] Rajakumar, R., & Devi, S. S. (2024). An efficient modelling of oversampling with optimal deep learning enabled anomaly detection in streaming data. China Communications, 21, 249–260. https://doi.org/10.23919/JCC.ja.2022-0592.
[10] Anomaly Detection in Resource Constrained Environments with Streaming Data. (2022). IEEE Transactions on Emerging Topics in Computational Intelligence, 6(3), 649–659. https://doi.org/10.1109/TETCI.2021.3070660.
[11] Yang, C., Du, Z., Meng, X., Zhang, X., Hao, X., & Bader, D. A. (2023). Anomaly Detection in Catalog Streams. IEEE Transactions on Big Data, 9, 294–311. https://doi.org/10.1109/TBDATA.2022.3161925.
[12] Zhang, M., Guo, J., Guo, J., Li, X., Li, X., Jin, R., & Jin, R. (2020). Data-Driven Anomaly Detection Approach for Time-Series Streaming Data. Sensors, 20(19), 5646. https://doi.org/10.3390/s20195646.
[13] Anomaly Detection in Catalog Streams. (2023). IEEE Transactions on Big Data, 9(1), 294–311. https://doi.org/10.1109/TBDATA.2022.3161925.
[14] Munir, M., Siddiqui, S. A., Chattha, M. A., Dengel, A., & Ahmed, S. (2019). FuseAD: Unsupervised Anomaly Detection in Streaming Sensors Data by Fusing Statistical and Deep Learning Models. *Sensors*, *19*(11), 2451. https://doi.org/10.3390/s19112451.
[15] Le, K.-N. T., Dang, T.-B., Le, D. T., Raza, S. M., Kim, M., & Choo, H. (2023). VEAD: Variance profile Exploitation for Anomaly Detection in real-time IoT data streaming. Internet of Things. https://doi.org/10.1016/j.iot.2023.100994.
[16] Cook, A. A., Mısırlı, G., & Fan, Z. (2019). Anomaly detection for IoT time-series data: A survey. IEEE Internet of Things Journal, 7(7), 6481-6494. https://doi.org/10.1109/JIOT.2019.2958185.
[17] Gabriel Rodriguez Garcia, Gabriel Michau, Mélanie Ducoffe, Jayant Sen Gupta, and Olga Fink. 2020. Time Series to Images: Monitoring the Condition of Industrial Assets with Deep Learning Image Processing Algorithms. arXiv: 2005.
[18] E. Keogh, T. Dutta Roy, U. Naik, and A Agrawal. 2021. Multidataset time-series anomaly detection competition. (2021).
[19] Chunggyeom Kim, Jinhyuk Lee, Raehyun Kim, Youngbin Park, and Jaewoo Kang. 2018. DeepNAP: deep neural anomaly pre-detection in a semiconductor fab. Information Sciences, 457–458, 1–11. doi: 10.1016/j.ins.2018.05.020.
[20] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. 2019. MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks. In Proceedings of the International Conference on Artificial Neural Networks, 703–716. doi: 10.1007/978- 3- 030-30490-4_56.
[21] Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. 2019. DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series. IEEE Access, 7, 1991–2005. doi: 10.1109/ACCESS.2018. 2886457.
[22] Linardi, M., Zhu, Y., Palpanas, T., & Keogh, E. (2020). Matrix profile goes MAD: variable-length motif and discord discovery in data series. Data Mining and Knowledge Discovery, 34(4), 1022-1071. https://doi.org/10.1007/s10618-020-00685-w.
[23] Maharaj, E. A., D'Urso, P., & Caiado, J. (2019). Time series clustering and classification. Chapman and Hall/CRC. https://doi.org/10.1201/9780429058264.
[24] Marteau, P. F., Soheily-Khah, S., & Béchet, N. (2017). Hybrid isolation forest-application to intrusion detection. arXiv preprint arXiv:1705.03800.
[25] Schmidl, S., Wenig, P., & Papenbrock, T. (2022). Anomaly detection in time series: a comprehensive evaluation. Proceedings of the VLDB Endowment, 15(9), 1779-1797. https://doi.org/10.14778/3538598.3538602.