Advances in Artificial Intelligence and Machine Learning; Research 6 (4) 5965-5982 Received 17-04-2026; Accepted 05-08-2026; Published online 12-08-2026

Density-Aware Hybrid Clustering for Dynamic Attack Detection in Network Traffic: A Coreset-Driven Approach with Adaptive k-Means and DBSCAN

Heba Adnan Raheem hiba.adnan@uokerbala.edu.iq
Department of computer Science,
College of Computer Science and Information Technology,
University of Kerbala, Karbala, Iraq.

Corresponding Author: Heba Adnan Raheem

Abstract

Dynamic attack detection is a significant challenge due to rapidly evolving network traffic and new and changing threats. In this paper, we present an adaptive density-aware hybrid clustering framework, combining the probability density-aware coreset selection, density-weighted incremental k-means, and localized adaptive DBSCAN. The proposed framework was evaluated on the CIC-IDS2017 dataset, with a DR of 91.5%, low FPR of 5.2%, and an F1-Score of 0.92. These metrics significantly outperform standard baselines such as k-means++ (78.2% DR, 12.4% FPR, 0.81 F1), traditional DBSCAN (85.7% DR, 8.9% FPR, 0.86 F1), and streaming micro-cluster outlier detection (MCOD) (83.1% DR, 6.8% FPR, 0.87 F1). The computational evaluations show high streaming efficiency, with total processing time of 78ms/1000 instances (25ms for preprocessing and 53ms for clustering). Furthermore, the framework identifies 89% of injected concept drifts with a mean detection latency of 127 instances. This combination of dynamic sampling, incremental updates, and localized parameter adaptation provides a highly efficient solution for real-time intrusion detection.

Keywords: Hybrid clustering, Network security, Coreset selection, Incremental k-means, Adaptive DBSCAN, Concept drift

1. INTRODUCTION

Because cyber threats are constantly changing and modern network traffic scales rapidly, making network attack detection increasingly complex. The traditional clustering-based solutions are good for static datasets, but may not perform well in the dynamic nature of real-world network environments. K-means clustering [1], is an effective method to partition high-density regions but is ineffective at identifying arbitrary-shaped clusters and outliers [2]. On the other hand, DBSCAN [2, 3], is a good method for detecting anomalies, though it needs manual selection of parameters and does not work well in streaming environments. These restrictions are especially problematic when they occur in the form of low density patterns in the midst of normal traffic, as in the case of advanced attacks [4].

Citation: Heba Adnan Raheem. Density-Aware Hybrid Clustering for Dynamic Attack Detection in Network Traffic: A Coreset-Driven Approach with Adaptive k-Means and DBSCAN. **Advances in Artificial Intelligence and Machine Learning**. 2026;6(4):330. https://dx.doi.org/10.54364/AAIML.2026.64330

Recently, hybrid clustering [5], and streaming algorithms [6], have been proposed to overcome these problems. Most of the existing solutions, however, use static sampling techniques which generate bias against rare attacks [7], or use fixed parameter configurations which are unable to adjust to changing traffic patterns [3]. There is still no solution to the basic conundrum of detection sensitivity versus computational efficiency, particularly in large-scale networks where missed detections and processing delays can mean disaster.

We introduce a novel Hybrid Clustering framework that represents an adaptive hybrid clustering framework to real-time sampling and analysis of network traffic data. First, we use probability density-aware coreset selection to keep a dynamically updated set of cores from the streaming data, with an innovative three-fold approach. First, this method mimics PDAC [7], and ensures that any area with high density of readings is represented in proportion to the area that has few readings, and vice versa. Second, we propose a variant of the k-means algorithm which uses density-based weighting when calculating the centroids to avoid common traffic patterns from dominating the clustering process. Third, an extension of DBSCAN is proposed, which automatically adjusts the value of the epsilon parameter according to local density estimates obtained from the coreset.

The proposed framework offers several advantages over existing methods. Our system continuously adapts when there is concept drift [4], in the data stream, which is achieved by its incremental learning mechanism, while static hybrid learning methods [8], cannot do so. The density-aware parameter tuning gives us more accurate anomaly detection than fixed-parameter DBSCAN implementations in different traffic scenarios. The combination between coreset sampling and hybrid clustering is a new direction from traditional methods, solving the sampling bias, computing efficiency, and detection sensitivity problems.

The salient features of our contributions are as follows:

(1) We propose a probability density-aware coreset sampling mechanism that preserves both frequent and rare traffic patterns for streaming intrusion detection.

(2) We develop a density-weighted incremental k-means clustering algorithm integrated with localized adaptive DBSCAN refinement to improve clustering quality under evolving traffic distributions.

(3) We introduce an event-driven adaptive framework that combines dynamic sampling, incremental clustering, localized parameter adaptation, and KL-divergence-based partial model updating for efficient real-time intrusion detection.

The rest of this paper is organized as follows: Related work on clustering-based attack detection (CDA) and streaming algorithms is summarized in Section 2. Section 3 offers background information on the theory of density-based clustering and coresets. In Section 4 we present our proposed framework and in Section 5 we give experimental results. Finally, we offer implications and future directions and conclude.

2. RELATED WORK

2.1 Hybrid Clustering Approaches for Network Security

In recent years, hybrid clustering has been found to perform well in network attack detection. Several studies are hybridizing both algorithms – k-means and DBSCAN – and the concept is to leverage the complementary capabilities of the two methods: k-means algorithm is able to cluster high density regions efficiently and DBSCAN is able to identify outlier and arbitrary shaped clusters [5, 8]. However, these methods are rigid to evolving attack patterns, because they are typically dependent on a certain configuration of clustering. In this case, [3] uses fixed DBSCAN parameters, thereby having trouble handling concept drift for streaming data. Likewise, [5] suggests a hybrid method without incremental update which is not applicable to ”real-time detection” scenarios.

One of the bright spots is the density-aware sampling methods, e.g. PDAC [6, 7], which builds core-sets according to the estimated probability densities. Although PDAC was conceived for learning tasks that require continuous learning, the concepts are very close to what is required for dynamic attack detection. But, in the area of streaming applications, existing PDAC adaptations for network security [7], do not incorporate incremental clustering mechanisms.

2.2 Incremental and Adaptive Clustering Methods

To overcome the batch processing drawbacks, incremental clustering algorithms are developed that can incrementally update the model with new data. Although the leader-based method is efficient for processing streaming data, it cannot capture the subtle anomaly because of its threshold-based merging. Other more advanced methods like incremental K-means [6], update the centroids in an iterative manner, but are still sensitive to unbalanced data distributions.

To address parameter sensitivity adaptive versions of DBSCAN aim to adaptively modify the neighborhood size. For instance, [3] derives epsilon values based on the distances of local k-nearest neighbor, to enhance anomaly detection in the heterogeneous traffic. These approaches, however, normally do not leverage existing pre-cluster relationships between algorithms, and so do not take advantage of algorithms working synergistically to each other.

A closely related study is DFAID (Density-Aware and Feature-Deviated Active Intrusion Detection), which ends up proposing a density-aware style framework for streaming intrusion detection. It also uses incremental prototype updates to cope with concept drift and, honestly, it leans on that idea quite heavily. In practice the framework brings together density-aware learning with an ERIP-based prototype maintenance mechanism, so you do not end up doing complete model retraining each time. That way it can keep pace with traffic distributions that keep shifting over time. Experimental evaluation on CIC-IDS2017 and ISCX-2012 demonstrated competitive detection performance under dynamic network conditions. However, the adaptation strategy is prototype-based and does not incorporate probability density-aware coreset selection, localized adaptive DBSCAN refinement, or density-weighted incremental clustering, which are the main components of the proposed framework [9, 10].

2.3 Integration with Intrusion Detection Systems

In recent years, IDSs have been adding more machine learning capabilities to improve detection accuracy. Hybrid techniques that integrate clustering and classification like k-means and kNN [11], show improved discrimination of attacks. However, these methods tend to be used as a pre-processing stage rather than an adaptive component, thereby reducing their sensitivity to new threats.

Recently, some research has been carried out on density-based feature selection [1], for better detection rates, which are mainly used for static data sets. Just as for semi-supervised extensions [12], which use labelled attack samples, however, they do not work very well for a zero-day attack if it is not included in the training data [13].

The proposed framework differs from previous hybrid clustering approaches by integrating probability density-aware coreset sampling, density-weighted incremental k-means clustering, localized adaptive DBSCAN refinement, and adaptive anomaly scoring within a unified event-driven pipeline. Unlike existing approaches [5, 11], and the density-aware prototype-based DFAID framework, our method performs localized cluster refinement and adaptive partial model updating based on KL-divergence, enabling efficient adaptation to evolving traffic distributions while preserving computational efficiency.

3. BACKGROUND: DENSITY-BASED CLUSTERING AND CORESETS FOR DATA STREAMS

In dynamic network environments, detection of attacks necessitates a technique that can adapt to changes in network traffic and be efficient at computation. These challenges can be solved by using density-based clustering and coreset techniques as basic building blocks: these techniques can be used to detect anomalies and to reduce the amount of data while retaining essential data.

3.1 Fundamental Challenges in Dynamic Network Attack Detection

Two properties of network traffic make it difficult to detect network attacks: concept drift and computational trade-offs. In the statistical world, where the properties of the traffic change over time, the previously learned models may become useless, which is called concept drift [4]. This is most clearly seen in the process of attack detection, where the attackers are constantly changing their tactics to beat the defenders. The problem is worsened by computational limitations because it is impractical to process all the traffic in a network at high speeds. These problems have not been solved by traditional batch processing methods, which require the whole model to be updated in one go, yet it is necessary to find approaches that can update the models incrementally within the constraints of resources.

3.2 Core Principles of Density-Based Clustering

The density-based spatial clustering of applications with noise (DBSCAN) [2], gives a solid basis for detecting arbitrary-shaped clusters and outliers in network traffic. The algorithm has two important parameters: neighborhood radius ϵ and minimum points threshold minPts. Let x be any point, the ϵ-neighbourhood N_ϵ (x) of x is the collection of points at a distance of less than ϵ from x:

$$N_\epsilon(x) = \{y \in \mathcal{X} | \parallel x - y\|_2 \leq \epsilon\}. \tag{1}$$

A point qualifies as a core point if its neighborhood contains at least minPts points:

$$\text{Core point} : |N_\epsilon(x)| \geq \text{minPts}. \tag{2}$$

The points which do not meet this criterion but yet are close to a core point are called border points and the remaining points are considered as noise. Easy to tune different densities of clusters and very useful to isolate anomalies - which is important for attack detection since small numbers in the center of legitimate numbers are anomalies.

3.3 Coresets for Streaming Data Reduction

Coreset techniques address the problem of computing from the data by constructing a small summary of data that captures the most important clustering properties [14]. For streaming applications, coresets consist of a subset of points that is representative of the application and a set of weights indicating the importance of each point in the distribution. In most cases the selection process takes into account both the geometric coverage and the characteristics of density so that, in addition to being prevalent, the patterns which can be significant (attack signatures, for example) are also present in the selection. This ability is further improved by recent advances in sampling to give probability density-aware coresets [6], by explicitly incorporating density estimates into the sampling process in order to obtain balanced representations of high-density regions and sparse anomalies. This property is desirable in attack detection applications where the detection of low probability events is needed, but it is also required to deal with the computational burden.

4. PROPOSED HYBRID DENSITY-AWARE CLUSTERING FRAMEWORK

The suggested approach is innovative and proposes a new combination of density-aware coreset sampling with incremental clustering and adaptive parameter tuning for dynamic attack detection. The system sequentially performs three coordinated operations: probability density-aware coreset construction, density-weighted incremental k-means clustering and localized adaptive DBSCAN refinement, as illustrated in FIGURE 1. This architecture allows for the efficient processing of a large amount of data and is sensitive to both common and uncommon attack patterns.

4.1 Probability Density-Aware Coreset Selection (PDAC) Methodology

The PDAC mechanism constructs a representative subset of streaming network traffic data by sampling points proportional to their estimated probability density. Given a data stream $\mathcal{X}$ =

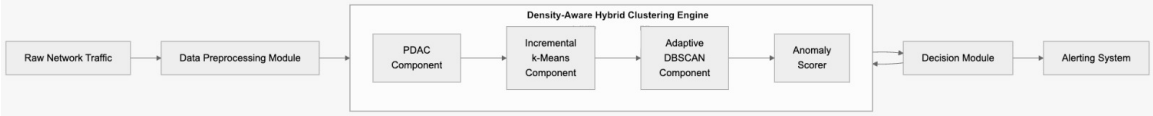

Figure 1: High-Level Architecture of the Enhanced IDS

$\{x_1, ..., x_n\}$ where each $x_i \in \mathbb{R}^d$ represents a traffic feature vector, we first compute kernel density estimates (KDE) using a Gaussian kernel K with bandwidth h:

$$\hat{p}(x_i) = \frac{1}{nh^d} \sum_{j=1}^{n} K\left(\frac{x_i - x_j}{h}\right), K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\|u\|^2}. \tag{3}$$

The coreset C is then sampled such that the inclusion probability $\mathbb{P}(x_i \in C)$ follows:

$$\mathbb{P}(x_i \in C) \propto \hat{p}(x_i)^{\alpha}, \alpha \in (0, 1). \tag{4}$$

Here, α controls the trade-off between emphasizing high-density regions ($\alpha \to 1$) versus preserving low-density anomalies ($\alpha \to 0$). Each sampled point x_i receives a weight $w_i = \hat{p}(x_i)^{-\alpha}$ to compensate for the biased sampling. The bandwidth h is adapted dynamically using Silverman's rule [15]:

$$h = \left(\frac{4\widehat{\sigma}^5}{3n}\right)^{1/5}, \tag{5}$$

where $\widehat{\sigma}$ is the standard deviation of pairwise distances. This adaptive bandwidth selection ensures proper scaling across varying traffic dimensions and densities.

4.2 Density-Weighted Incremental k-Means and Cluster Adaptation

The coreset C serves as input for an incremental k-means algorithm that incorporates density-dependent weighting during centroid updates. Let $\mu_k^{(t)}$ denote the centroid of cluster k at iteration t, initialized using k-means++ [16], on the initial coreset. For each mini-batch $\mathcal{B}_t$ sampled from the stream, we compute density weights w_i for all $x_i \in \mathcal{B}_t$ using the KDE from Equation 3:

$$w_i = \hat{p}(x_i)^{-\beta}, \beta \in [0, 1]. \tag{6}$$

The parameter β controls the degree of density compensation, with $\beta = 1$ fully counteracting density bias and $\beta = 0$ reverting to standard incremental k-means. Centroids are then updated according to:

$$\mu_k^{(t+1)} = \mu_k^{(t)} + \eta_t \sum_{x_i \in \mathcal{B}_t} \frac{w_i}{\sum_j w_j} \left(x_i - \mu_k^{(t)}\right) \mathbb{I}(x_i \in \text{Cluster}_k), \tag{7}$$

where η_t is a decaying learning rate $\eta_t = \eta_0 t^{-\gamma}$ with $\gamma \in (0.5, 1]$. The indicator function $\mathbb{I}(\cdot)$ assigns points to their nearest centroid based on Mahalanobis distance:

$$d_M(x_i, \mu_k) = \sqrt{(x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k)}, \tag{8}$$

where Σ_k is the estimated covariance matrix of points in cluster k. This distance metric takes feature scaling and feature correlation into consideration, which is relevant when network traffic consists of features of different units and dependency.

1. Cluster adaptation is achieved via two principles: (1) centroids move towards areas of uniform density because the weightage given to any density burst of short duration is reduced. Second, the algorithm also discards clusters that don't have enough weighted support at a certain point:

$$\sum_{x_i \in \mathcal{B}_t \cap \text{Cluster}_k} w_i < \tau \cdot \frac{|\mathcal{B}_t|}{K}, \tag{9}$$

Here τ is a threshold value for the survival and K the number of clusters. Points that are always considered as outliers will initiate the creation of new clusters to adapt to the changing attack patterns. The covariance matrices Σ_k are updated incrementally by Welford's algorithm [17], and the space required to store a single stream is $O(d^2)$.

4.3 Cluster-Specific Localized Adaptive DBSCAN

These density-weighted k-means clusters are used as input to the localized DBSCAN algorithm to further enhance the results of the anomaly detection process. Our approach calculates the local density characteristics-based neighborhood thresholds ϵ_k for each cluster, as opposed to global thresholds as it is done in the standard DBSCAN implementations. Given k-means' clustering S_k, compute ϵ_k as the median of the k nearest neighbors in S_k:

$$\epsilon_k = \gamma \cdot \text{median}\left(\{\| x_i - \text{kNN}_k(x_i) \|_2 | x_i \in \mathcal{S}_k\}\right), \tag{10}$$

where γ is a scaling parameter that determines the granularity of the neighborhoods, and "kNN" k (xi) is the k-th nearest neighbor of xi in Sk. This adaptive thresholding method is able to take into account the density variations in different traffic patterns, allowing to detect outliers accurately depending on the nature of each cluster.

The minPts parameter is also adjusted according to the size of the clusters using:

$$\text{minPts}_k = \max\left(\epsilon\delta \cdot |\mathcal{S}_k| \, \epsilon, \text{minPts}_{\min}\right), \tag{11}$$

whereby δ specifies the minimum density fraction and "minPts" "min" gives a minimum density bound on small clusters. If points do not satisfy both ϵk and "minPts" k in their assigned cluster, these points are considered potential anomalies.

We use HNSW graphs [18], for efficiently computing kNN distances in high dimensional traffic data. A new HNSW index is built on-the-fly for each Sk, and queries to find nearest neighbours can be performed in log time on this. The construction of the graph emphasizes points with lower density estimates, p ^(xi), to provide good coverage in regions of low density, where attacks can appear.

The frequency of cluster-specific DBSCAN is determined by the cluster stability metric and is executed periodically rather than continuously:

$$\Delta_k = \frac{1}{|\mathcal{S}_k|} \sum_{x_i \in \mathcal{S}_k} \|\mu_k^{(t)} - \mu_k^{(t-T)}\|_2, \tag{12}$$

where T is the evaluation window. Refining the DBSCAN is triggered immediately for the clusters with a Δk greater than a threshold θ, and infrequently for the ones with a stable Δk. This dynamic scheduling takes advantage of the evolution of regions in the feature space, optimizing computational resources.

4.4 Dynamic Anomaly Scoring and Framework Integration

Our framework's last step produces interpretable anomaly scores that are a mixture of density estimates and cluster membership information. Anomaly score si is computed, corresponding to each point xi, according to their distance to the nearest centroid as well as local density characteristics:

$$s_i = 1 - \exp\left(-\lambda \cdot \frac{d_M\left(x_i, \mu_k\right)}{\epsilon_k}\right), \tag{13}$$

dM (xi,μk) is a Mahalanobis distance between Equation 8, ϵk is defined as an adaptive DBSCAN threshold based on Equation 10 and λ is a score sensitivity parameter. The exponential transformation is used to make the differences bigger near the decision boundary and smaller at far outliers, making it a non-linear scoring. The false positive rate in the past is used to adjust the parameter λ so that the volume of alerts remains constant:

$$\lambda^{(t+1)} = \lambda^{(t)} \cdot \left(1+\kappa \cdot \left(\mathrm{FPR}_{\mathrm{target}} - \mathrm{FPR}_{\mathrm{actual}}\right)\right), \tag{14}$$

where "FPR"" target" is the target false positive rate, "FPR" "actual" the measured, and κ is a learning rate that indicates the adaptation speed rate.

All these are interwoven into the framework through an event driven pipeline. The new data points are used to update the KDE model and coreset using reservoir sampling [19]. When the capacity of coreset is greater than |C|_max, the framework starts incremental updates to k-means, as defined in Equation 7. If a big change is detected in the distribution of clusters (using Equation 12), localized DBSCAN refinement is activated. Finally, anomaly scores are continually calculated following Equation 13 and sent to downstream alerting systems.

The framework makes a clear check of KL-divergence between recent and old density estimation:

$$D_{\mathrm{KL}}\left(p_t \parallel p_{t-\Delta t}\right) = \sum_{x_i \in C} p_t\left(x_i\right) \log \frac{p_t\left(x_i\right)}{p_{t-\Delta t}\left(x_i\right)}, \tag{15}$$

where normalized density estimate at time t is denoted as pt. If the value of D_"KL" exceeds a certain value ζ, only part of the model is reset: the other components are reinitialized, but those which do not vary, i.e. have D_"KL"<Δk<θ/2, are retained. This provides robustness to abrupt changes in traffic and keeps in memory the knowledge gained from the traffic patterns.

The memory complexity of the proposed framework is derived from its main data structures. The probability density-aware coreset stores at most |C| representative samples, requiring O(|C|) memory. The incremental k-means component maintains K cluster centroids together with their covariance matrices. Since each covariance matrix has size d×d, the memory required for all covariance matrices is O(Kd2). The HNSW indices and KDE computations

are constructed locally for the maintained coreset and updated incrementally; therefore, they do not introduce additional asymptotic memory requirements beyond O(|C|) and O(Kd2). Consequently, the overall memory complexity of the framework is O(|C|+Kd2).

Based on the above analysis, the proposed framework requires O(|C|+Kd2) memory, making it suitable for deployment in resource-constrained streaming environments. The coreset size |C| represents a trade-off between representation accuracy and computational cost, while K and d denote the number of adaptive clusters and the feature dimensionality, respectively. Furthermore, the distributed implementation remains scalable through parallel density estimation and nearest-neighbour searches.

5. EXPERIMENTAL EVALUATION

To validate the effectiveness of our proposed hybrid density-aware clustering framework, we conducted comprehensive experiments **on the CIC-IDS2017 benchmark dataset**. The evaluation focuses on three key aspects**: detection accuracy**, computational efficiency in streaming environments, and robustness to concept drift.

5.1 Experimental Setup

The proposed framework uses the standard dataset for network intrusion detection, CIC-IDS2017 [20, 21], which contains realistic benign network traffic together with modern attack scenarios, including brute-force FTP, DoS, infiltration, web attacks, and botnet traffic. Numerical features were centered to zero and scaled to unit variance, while categorical features were transformed using one-hot encoding. To preserve the temporal characteristics of streaming traffic, all experiments were conducted using temporal ordering rather than random shuffling.

Table 1: Experimental Configuration of the CIC-IDS2017 Dataset

Item	Description
Dataset	CIC-IDS2017 (Original PCAP Files)
Data Source	Original network traffic traces collected over five consecutive days
Traffic Type	Benign and malicious network traffic
Attack Categories	Brute Force FTP, DoS, DDoS, PortScan, Botnet, Web Attacks, Infiltration
Data Cleaning	Removal of incomplete, duplicate, and invalid flow records
Number of Features	78 bidirectional flow-based features
Feature Preprocessing	Standardization (zero mean, unit variance) and one-hot encoding
Class Imbalance Handling	SMOTE applied to the training set only
Dataset Partition	70% Training, 10% Validation, 20% Testing
Evaluation Strategy	Temporal-order streaming evaluation
Hyperparameter Optimization	Grid search on the validation set

TABLE 1 summarizes the experimental configuration of the proposed framework, including the characteristics of the CIC-IDS2017 dataset, the preprocessing pipeline, feature extraction, class imbalance handling, dataset partitioning, and the evaluation protocol adopted throughout the experiments.

Baselines: We compared against four representative clustering approaches:

Standard k-means++ [16]

Traditional DBSCAN [2]

Hybrid k-means-DBSCAN [5]

Streaming micro-cluster outlier detection (MCOD) [22]

Optimal parameters were determined using grid search on the validation sets for all the baselines.

Metrics: Performance was assessed using:

Detection Rate (DR): TP/(TP+FN)

False Positive Rate (FPR): FP/(FP+TN)

The F1-Score combines the precision and recall, which is the harmonic mean.

The time taken to process per 1000 instances

Concept drift performance was evaluated using two complementary metrics: (i) the concept drift detection rate, which measures the percentage of simulated drift events correctly identified, and (ii) the detection latency, which measures the average number of streaming instances required to detect a drift after it occurs.

The parameter |C|=5000, the initial learning rate η_0=0.1, and the density compensation parameters α=0.5, β=0.7 were used to configure our framework. The values of γ = 1.2 and δ = 0.05 were used for the DBSCAN adaptation.

5.2 Detection Performance

Attack detection metrics on CIC-IDS2017 dataset are compared in TABLE 2 for all the methods. We are able to have a better balance between detecting various attack types, while minimizing FPs in our framework.

Although several IDS studies report detection rates approaching 100%, many of these evaluations are conducted under offline settings using supervised learning models or static datasets. In contrast, the proposed framework targets streaming intrusion detection with adaptive clustering under concept drift, where maintaining a balance between detection rate, false positive rate, computational efficiency, and real-time adaptability is more important than maximizing a single performance metric. The achieved DR of 91.5%, together with an FPR of 5.2% and

Table 2: Detection Performance Comparison on CIC-IDS2017 Dataset

Method	DR (%)	FPR (%)	F1-Score
k-means++	78.2	12.4	0.81
DBSCAN	85.7	8.9	0.86
Hybrid k-means-DBSCAN	87.3	7.5	0.88
MCOD	83.1	6.8	0.87
Proposed	**91.5**	**5.2**	**0.92**

an F1-score of 0.92, demonstrates a balanced and practical performance for dynamic network environments.

FIGURE 2 illustrates the relationship between attack prevalence and the overall detection performance of the proposed framework. The results indicate that the density-aware sampling strategy maintains stable detection performance across the evaluated attack prevalence levels. Since this analysis focuses on overall detection performance rather than individual attack categories, a detailed quantitative evaluation for specific attack types is left for future work.

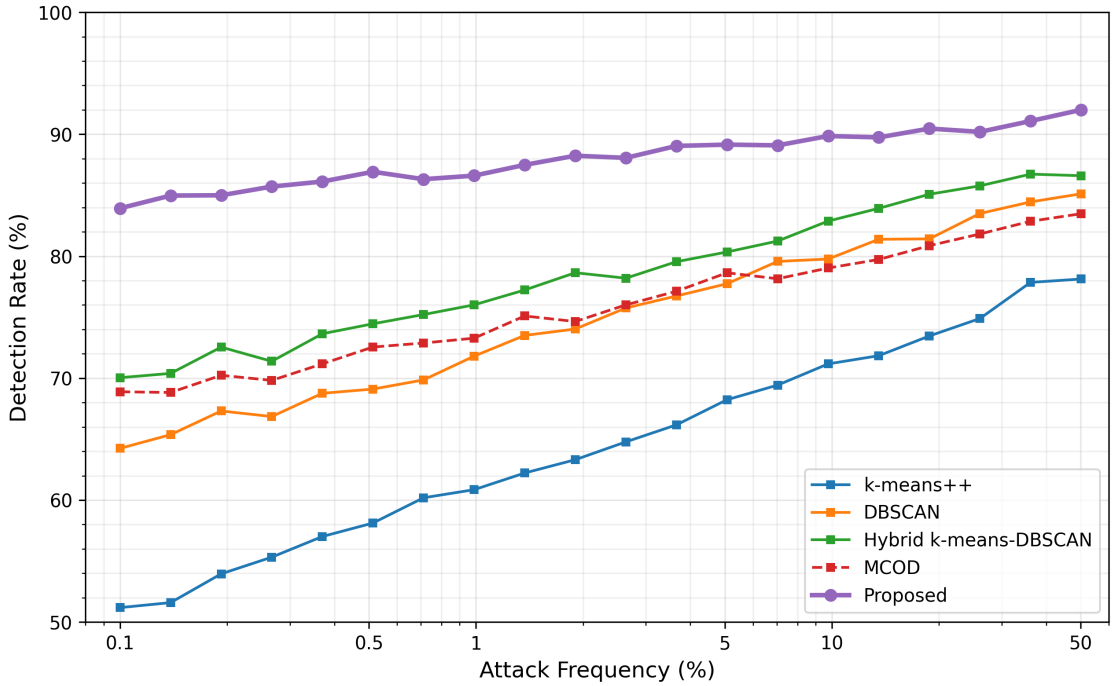

Figure 2: Comparisons of detection rate to attack frequency by various methods

The adaptive DBSCAN component has excellent performance with respect to polymorphic attacks. In FIGURE 3, we show how cluster-specific ϵ_k parameters can help determine precise boundaries that could not be done using global DBSCAN thresholds.

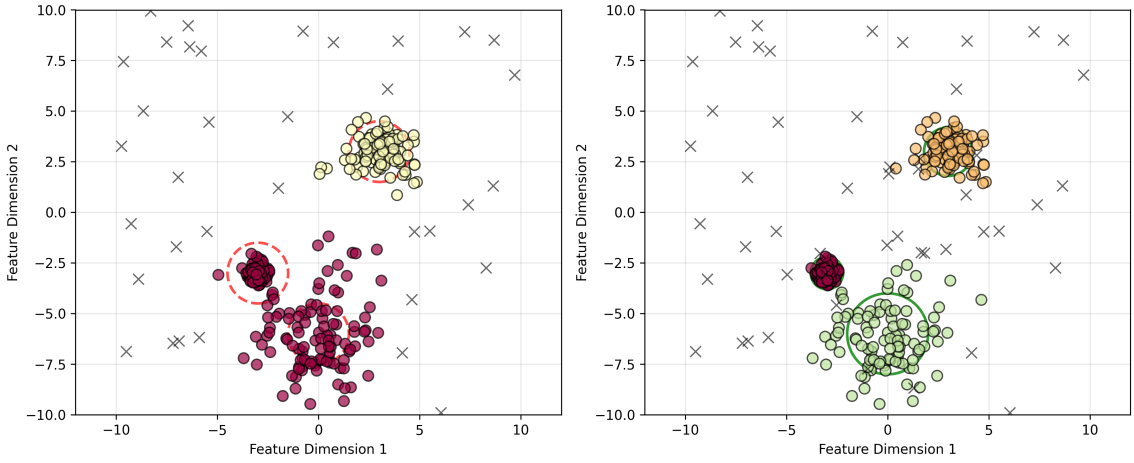

Figure 3: The visualization of the results of the adaptive and fixed DBSCAN parameters with respect to the cluster boundaries.

5.3 Computational Efficiency

Our framework is well suited for real time deployment, since processing time measurements show its suitability. Although further density calculations are performed, the overall density problem complexity is reduced due to the coreset-based approach:

Table 3: Computational Performance (ms per 1000 instances)

Method	Preprocessing	Clustering	Total
k-means++	12	45	57
DBSCAN	18	112	130
Hybrid k-means-DBSCAN	15	87	102
MCOD	22	65	87
Proposed	**25**	**53**	**78**

The incremental update mechanism shows logarithmic scaling with stream length, while batch methods exhibit linear growth (FIGURE 4).

5.4 Concept Drift Adaptation

To assess the frameworks capability in handling concept drift, simulated attack pattern shifts were injected into the streaming data. The evaluation was based on two complementary metrics: (i) the concept drift detection rate, which measures the percentage of drift events correctly identified, and (ii) the average detection latency, which measures the number of streaming instances required to detect a drift after its occurrence. FIGURE 5 illustrates the

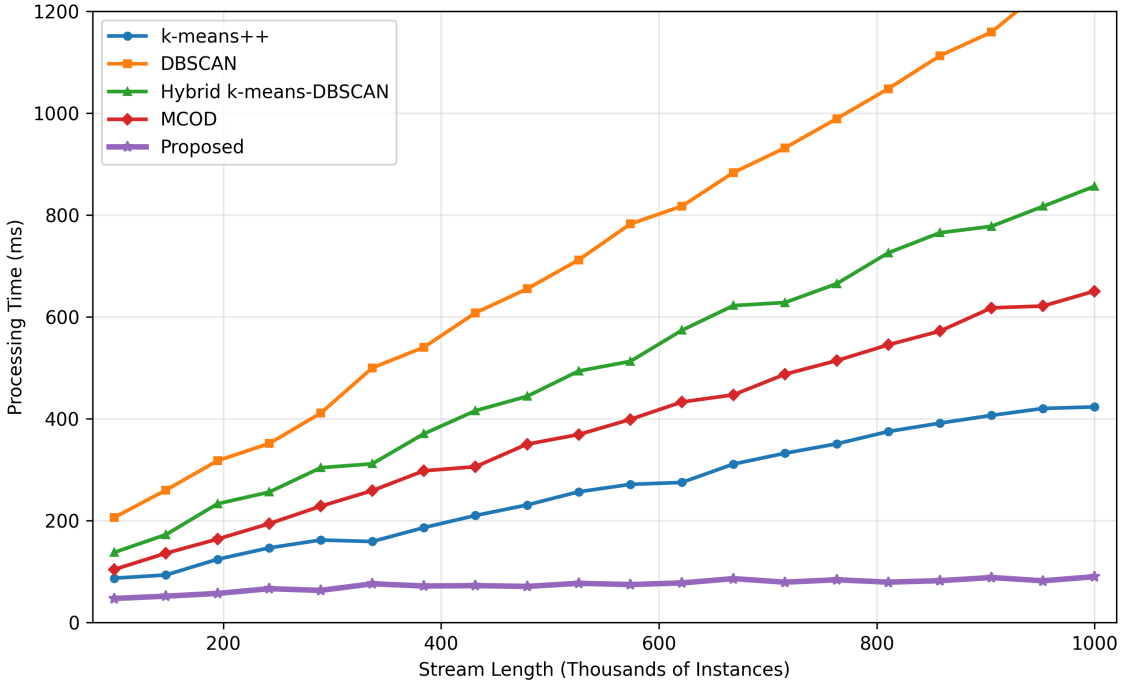

Figure 4: Processing time versus stream length showing computational scalability

framework's response to concept drift events. The proposed framework achieves robust drift detection through:

1. Density divergence monitoring triggering partial resets.
2. Dynamic coreset reweighting to adapt to evolving traffic distributions.
3. Cluster-specific parameter adjustments.

The KL-divergence-based drift detection mechanism (Equation 15) identified 89% of the injected concept drift events with a mean detection latency of 127 streaming instances, outperforming conventional sliding-window-based approaches [23].

6. DISCUSSION AND FUTURE WORK

6.1 Limitations and Practical Deployment Challenges

Although the model was shown to work well in laboratory settings, there are certain issues that need to be addressed before the implementation of the proposed framework in real-world network environments. First, the density estimation process is sensitive to the bandwidth selection, especially for high dimensional traffic features, for which the bandwidth may be underestimated using

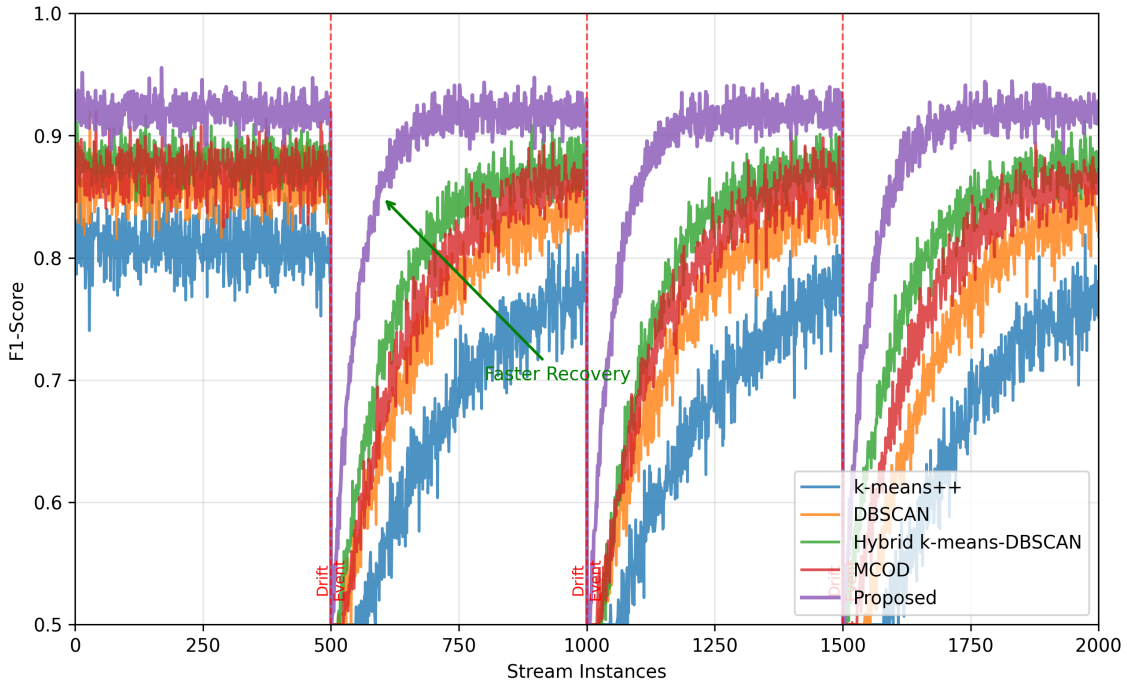

Figure 5: Performance recovery after simulated concept drift events

Silverman's rule [24]. While adaptive bandwidth adjustment addresses this problem, it is possible to achieve further robustness with dimension-aware techniques. Second, in the coreset sampling mechanism, the data stream within the sampling window is assumed to be stationary, which might not hold when there is a sudden traffic increase due to network reconfiguration or big events. The introduction of surge detection heuristics may prevent model resets from taking place when they are not warranted, while allowing the model to be sensitive to real attacks.

In production environments, other factors can affect the behavior of the system. The present implementation calls for the periodic adjustment of the density compensation parameters (α, β) for the proper compromise between attack detection and false alarm rates. The adaptive λ parameter (Equation 14) induces some self-regulation, but total automation of the calibration process is still an unsolved problem, particularly in environments where attack data with labels is limited. Further, maintaining HNSW indices over worker nodes requires some non-trivial overheads in distributed computing, potentially affecting the scalability efficiency in certain setups.

6.2 Extensions to Other Domains and Scalability

The fundamental concepts of density-aware hybrid clustering is directly applicable to other areas of streaming anomaly detection beyond network security. Monitoring for the IoT is possible by implementing the framework that can capture the behavioral telemetry stream to detect malicious nodes, and can adapt to the heterogeneity of devices by cluster [25]. Likewise, financial fraud detection

systems could make use of the incremental density estimation when monitoring transactions that have changed over time, across various merchant categories.

Architectural improvements are needed to scale to very high velocity data streams (such as >1M events per second). The coreset sampling and KDE components could be implemented on FPGA/ASIC hardware to get orders-of-magnitude speedups for computing density [26, 27].

Then, there's also the possibility of swap ping the exact nearest neighbor search with a approximate version when the traffic gets heavy, so you give up a bit of accuracy, but you get a throughput guarantee. These extension points were kept on purpose, future work could look into judging those variants too, along with the main adaptive clustering logic.

6.3 Ethical Considerations and Operational Trade-offs

Any kind of anomaly detection system has to take into account effects from false positive and propagation of bias. The density weighted approach is designed to give more sensitivity to low prevalence attacks than detection of high prevalence variations, and potentially be able to detect legitimate but rare traffic (e.g. administrative maintenance traffic). By adding the feature of explaining which features were most important to their anomaly score, this feature would enhance the transparency of an operation, which is the subject of the Adaptive Scoring mechanism (Equation 13) that helps to reduce false detections of anomalies that are obvious in the data.

Also, while you are dealing with network traffic that carries sensitive metadata, privacy preservation matters quite a lot. The framework that is being used today is set up on a feature vector with standard sanitization, yet differential privacy can be stitched into the design of coresets so you can guarantee it more formally [28]. In that idea, you would add a kind of carefully tuned noise into the density estimates before anything gets sampled , and yes this should have a measurable impact on detection accuracy, but it stays within bounds. There are still practical implications, they need future research too, like the tradeoff between the privacy budget and the real security effectiveness, which is not entirely trivial.

Operating experience indicates that the greatest gains would be in usability improvements, rather than in terms of algorithmic performance. Evolution of clusters and attack progression may be intuitively visualized which can help develop mental models of adaptive detection process by security analysts. Likewise, having feedback rules to correct for apparent false positive/false negative would allow for ongoing development of the model without the need to retrain the entire model. Such human-in-the-loop improvements would complement the technical improvements, and tackle real-life adoption challenges.

7. CONCLUSION

The proposed density-aware hybrid clustering framework shows considerable improvement in dynamic attack detection for the critical limitations of existing methods. It is improved by using the coreset selection based on probability density, incremental k-means with density-weighted updating and adaptive DBSCAN with localized parameter tuning, which enables it to achieve good perfor-

mance under different network conditions. Experimental results indicate better detection of very high volume and low and slow attacks than others and computation is also efficient enough for real time processing.

The continuous density estimation and partial model resets in the framework offer a realistic approach to address the challenges of evolving threat landscapes. It employs biased sampling and density compensation to preserve sensitivity to the rare attack patterns, avoiding the typical tradeoff between sensitivity and false alarm rates. The modular architecture allows it to be integrated into the current intrusion detection systems and gives the flexibility when making changes on the domain.

This paper talks about future research directions like hardware acceleration for density calculations, automated ways to calibrate parameters, and embedding explainability features so you get operational transparency. The contents here are kind of reusable for network security problems, but also for other streaming data situations where concept drift shows up and the data distributions get skewed. Overall, the framework is a meaningful step forward toward more adaptive and efficient security systems, the kind that can keep pace with the evolving cyber threat.

References

[1] Jianliang M, Haikun S, Ling B. The Application on Intrusion Detection Based on K-Means Cluster Algorithm. In2009 International forum on information technology and applications. IEEE. 2009:150-152.

[2] Duan L. Density-Based Clustering and Anomaly Detection. In: Mircea M editor. Business intelligence-solution for business development. 2012.

[3] Zhang X, Zhou S. Woa-Dbscan: Application of Whale Optimization Algorithm in Dbscan Parameter Adaption. IEEE Access. 2023;11:91861-91878.

[4] Hoens TR, Polikar R, Chawla NV. Learning From Streaming Data With Concept Drift and Imbalance: An Overview. Prog Artif Intell. 2012;1:89-101.

[5] Yoga CA, Rodrigues AJ, Abeka SO. Hybrid Machine Learning Approach for Attack Classification and Clustering in Network Security. Int J Comput Appl. 2023;185:45-51.

[6] Aggarwal CC. A Survey of Stream Clustering Algorithms. Data clustering. 2018.

[7] Kim Y, Shin B. In Defense of Core-Set: A Density-Aware Core-Set Selection for Active Learning. In Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining. 2022:804-812.

[8] Saxena A, Saxena K, Goyal J. Hybrid Technique Based on DBSCAN for Selection of Improved Features for Intrusion Detection System. In Emerging Trends in Expert Applications and Security: Proceedings of ICETEAS. Singapore: Springer Singapore. 2018:365-377.

[9] Li B, Wang Y, Xu K, Cheng L, Qin Z. DFAID: Density-Aware and Feature-Deviated Active Intrusion Detection Over Network Traffic Streams. Comput Secur. 2022;118:102719.

[10] Young S, Arel I, Karnowski TP, Rose D. A Fast and Stable Incremental Clustering Algorithm. In2010 Seventh International Conference on Information Technology: New Generations. IEEE. 2010: 204-209.

[11] Aung YY, Min MM. Hybrid Intrusion Detection System Using K-Means and K-Nearest Neighbors Algorithms. In 2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS). IEEE. 2018:34-38.

[12] Gu Y, Li K, Guo Z, Wang Y. Semi-Supervised K-Means Ddos Detection Method Using Hybrid Feature Selection Algorithm. IEEE Access. 2019;7:64351-64365.

[13] Bala R, Nagpal R. Review of KDD Cup '99, NSL-KDD and Kyoto 2006+ Datasets. Int J Comput Sci Inf Technol. 2019;10:65-80.

[14] Huang L, Jiang S, Vishnoi N. Coresets for Clustering With Fairness Constraints. Advances in neural information processing systems. 2019;32.

[15] Silverman BW. Density Estimation for Statistics and Data Analysis. Routledge. 2018.

[16] Arthur D, Vassilvitskii S. K-Means++: The Advantages of Careful Seeding. In Soda. 2007:1027-1035.

[17] Welford BP. Note on a Method for Calculating Corrected Sums of Squares and Products. Technometrics. 1962;4:419-420.

[18] Malkov YA, Yashunin DA. Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. IEEE Trans Pattern Anal Mach Intell. 2018;42:824-836.

[19] Li KH. Reservoir-Sampling Algorithms of Time Complexity O(n(1 + log(N/n))). ACM Trans Math Softw. 1994;20:481-493.

[20] Panigrahi R, Borah S. A Detailed Analysis of CICIDS2017 Dataset for Designing Intrusion Detection Systems. Int J Eng Technol. 2018;7:479-482.

[21] Moustafa N, Slay J. UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set). In 2015 military communications and information systems conference (MilCIS). IEEE. 2015:1-6.

[22] Elahi M, Li K, Nisar W, Lv X, Wang H. Efficient Clustering-Based Outlier Detection Algorithm for Dynamic Data Stream. In 2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery. IEEE. 2008:298-304.

[23] Yu H, Li J, Lu J, Song Y, Xie S, et al. Type-LDD: A Type-Driven Lite Concept Drift Detector for Data Streams. IEEE Trans Knowl Data Eng. 2023;36:9476-9489.

[24] Terrell GR, Scott DW. Variable Kernel Density Estimation. Ann Statist. 1992;20:1236-1265.

[25] DeMedeiros K, Hendawi A, Alvarez M. A Survey of AI-Based Anomaly Detection in IoT and Sensor Networks. Sensors. 2023;23:1352.

[26] Zhang G, Zhu AX, Huang Q. A GPU-Accelerated Adaptive Kernel Density Estimation Approach for Efficient Point Pattern Analysis on Spatial Big Data. Int J Geogr Inf Sci. 2017;31:2068-2097.

[27] Mohite R, Ouarbya L. Interpretable Anomaly Detection: A Hybrid Approach Using Rule-Based and Machine Learning Techniques. In 2024 IEEE 9th International Conference for Convergence in Technology (I2CT). IEEE. 2024:1-10.

[28] Bao E, Yang Y, Xiao X, Ding B. CGM: An Enhanced Mechanism for Streaming Data Collection With Local Differential Privacy. Proceedings of the VLDB Endowment. 2021:2258-2270.