International Journal of Computer Networks and Applications (IJCNA)
DOI: 10.22247/ijcna/2026/10 Volume 13, Issue 1, January – February (2026)

REVIEW ARTICLE

Unsupervised One-Class Learning for IoT Intrusion Detection: A Review of Methods and Prospects

Thavavel Vaiyapuri
College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, Al Kharj, Saudi Arabia.
✉ t.thangam@psau.edu.sa

Karthiyayini Murugesan
Department of Computer Science and Engineering, Sethu Institute of Technology, Tamil Nadu, India.
karthiyayinisaravanan5@gmail.com

Received: 15 October 2025 / Revised: 30 January 2026 / Accepted: 12 February 2026 / Published: 26 February 2026

Abstract – The rapid proliferation of the Internet of Things (IoT) has heightened the need for intrusion detection systems capable of operating within stringent computational, memory, and energy constraints. One-Class Learning (OCL) has emerged as a promising approach for IoT security, as it models only benign behavior and remains effective when attack samples are scarce, irregular, or evolving. Despite increasing interest in OCL, existing studies focus primarily on detection accuracy and offer limited insight into the practical deployability of these models across heterogeneous IoT environments. This review provides a deployment-oriented synthesis of OCL techniques for IoT intrusion detection system (IDS). It introduces a structured taxonomy spanning shallow learning (SL), representation learning (RL), and deep one-class neural architectures (OC-DNN), and consolidates evidence from widely used IDS datasets. A unified evaluation framework is proposed, assessing computational complexity, inference latency, memory footprint, and suitability for edge deployment. The findings presented establish a clear foundation for developing OCL-based intrusion detection systems that are both computationally efficient and operationally viable for real-world IoT ecosystems.

Index Terms – Cybersecurity, IoT Security, Anomaly Detection, Unsupervised Learning, One-Class Classification, Lightweight Edge Computing, Sustainable IDS.

1. INTRODUCTION

The rapid expansion of the IoT is reshaping critical sectors such as healthcare, smart cities, industrial automation, and intelligent transportation by enabling pervasive connectivity and continuous data exchange [1], [2], [3], [4]. At the same time, IoT ecosystems introduce security risks that differ fundamentally from traditional enterprise networks due to device heterogeneity, lightweight protocols and irregular traffic patterns. These constraints are especially pronounced in distributed deployments spanning device-, edge-, fog-, and cloud-level infrastructures, where intrusion detection must operate under real-time requirements and limited computational budgets [5]. Consequently, IDS designed for resource-rich environments often become impractical or unreliable when applied directly to IoT settings [6], [7].

Traditional IDS research is largely grounded in supervised learning. It assumes the availability of large labeled traffic covering known attack classes. In operational IoT networks, these assumptions rarely hold and malicious samples are often scarce. Also, class distributions are heavily skewed and traffic behavior changes over time [8], [9]. Further, new attacks appear without warning and Zero-day threats are a common example. This gap between supervised requirements and real-world conditions limits practical reliability OCL addresses this limitation. It models the normal IoT profile using benign data only. Events that deviate from this learned profile are flagged as malicious activity [10], [11]. As a result, OCL reduces dependence on exhaustive attack labels and offers stronger adaptability to previously unseen intrusions.

Although OCL is increasingly adopted, the research landscape is still scattered. Existing work spans shallow models, representation learning, and deep neural architectures. However, studies vary widely in the datasets used, the evaluation setup, and the assumptions made [12]. This makes direct comparison difficult. More importantly, most surveys focus on detection accuracy and give limited attention to deployment realities in IoT systems. An IDS cannot be judged by accuracy alone. It must also meet practical constraints. These include compute cost, memory footprint, inference delay, and energy usage. The constraints differ across devices, edge nodes, fog layers, and cloud backends [13], [14], [15]. At present, there is no review that systematically compares OCL approaches across paradigms under these deployment constraints. This motivates a unified, IoT-oriented review. The goal is to synthesize OCL methods using both methodological and deployment-focused perspectives.

To address the identified gap, this review offers a unified synthesis of unsupervised OCL for IoT intrusion detection. It

REVIEW ARTICLE

goes beyond cataloging methods by summarizing how OCL approaches are formulated and evaluated, and by explaining why reported results often differ across IoT settings. The discussion is grounded in practical IoT conditions such as device heterogeneity, evolving traffic patterns, severe class imbalance, and constrained compute and memory, so that findings are interpreted in context rather than as standalone benchmark scores. The review also clarifies the evaluation assumptions that commonly limit cross-study comparability. These include treating training data as predominantly benign, expecting attacks to appear as detectable deviations from normality, and assuming benign behavior remains stable despite traffic evolution. Differences in feature engineering, preprocessing, threshold calibration, and train/test splitting further contribute to inconsistent reporting and evaluation outcomes. By making these factors explicit and structuring the literature accordingly, the review enables more meaningful comparisons and helps identify priorities for improving the robustness and deployability of OCL-based IoT IDS [16]. The key contribution of this study is as follows,

a. This review provides a comprehensive and deployment-aware survey of unsupervised OCL methods for IoT intrusion detection.

b. A structured taxonomy is proposed, covering SL, RL, and OC-DNN, with a clear comparison of their foundations and IoT suitability.

c. A deployability-focused analysis is presented, discussing training and inference cost, memory footprint, and feasibility across device, edge, fog, and cloud layers.

d. Evidence is synthesized across major IDS benchmarks, highlighting how dataset properties affect OCL performance.

e. Finally, the review summarizes open challenges and future research directions to support resource-efficient and scalable OCL-based IDS for next-generation IoT systems.

The remainder of this paper is organized as follows: Section 2 presents background on IoT IDS and the OCL paradigm. Section 3 reviews existing surveys and highlights unresolved gaps. Section 4 introduces the proposed taxonomy of OCL approaches. Section 5 provides comparative analysis and evaluation perspectives, followed by Section 6 outlining challenges and future prospects. Finally, Section 7 concludes the paper.

2. BACKGROUND

2.1. Overview of IoT IDS

An IDS monitors network traffic and host activity to identify malicious behavior, policy violations, and abnormal operational patterns [17]. In IoT environments, traffic is observed through lightweight messaging and low-overhead exchanges rather than long, continuous sessions. Protocols such as MQTT, CoAP, and ZigBee are widely used and often generate short-lived, intermittent communications [18]. This operating profile limits the richness and stability of observable evidence and increases sensitivity to changes in device behavior and network conditions. Consequently, IDS approaches developed for enterprise networks—where traffic is more persistent and monitoring resources are plentiful—often require redesign and careful tuning to remain reliable in IoT settings [6], [7].

IoT IDS approaches are commonly categorized by detection strategy and deployment location. Signature-based IDS detects known threats using predefined rules and patterns, but it provides limited protection against novel and evolving attacks. Anomaly-based IDS models normal behavior and flags deviations, making it more suitable for dynamic IoT conditions and zero-day threats. Hybrid IDS combines both strategies to improve coverage, typically at the cost of higher complexity. From a deployment perspective, IDS can operate at the device, edge gateway, fog layer, or cloud. Device-level detection offers low latency but faces strict resource limits, while edge/fog deployments provide better near-real-time capability with moderate resources. Cloud-based IDS supports heavy analytics and global visibility but may introduce latency and privacy concerns [19], [20].

An effective IoT IDS must therefore balance three critical design objectives: accuracy, efficiency, and robustness [21]. Accuracy is essential for detecting malicious activity within noisy, incomplete, and highly imbalanced traffic. Efficiency ensures that detection algorithms operate within the tight computational and energy budgets characteristic of edge devices. Robustness enables the IDS to withstand evolving attack patterns, adversarial behaviors, and the diverse operating conditions inherent in IoT ecosystems [22].

Meeting these objectives at the same time is difficult. Approaches that prioritize accuracy typically demand substantial computation, which can be impractical for IoT deployments. In contrast, lightweight models reduce resource usage but may weaken detection performance, particularly for subtle or evolving attacks. These constraints have driven growing interest in IDS designs that are adaptive and resource-aware, delivering reliable generalization while maintaining within the operational limits of IoT devices [23].

2.2. One-Class Learning (OCL) Paradigm

OCL is a machine learning paradigm used when normal behavior is well represented during training, while abnormal instances are rare, highly diverse, or unavailable [24], [25]. Unlike conventional multi-class classifiers, OCL does not require labeled examples for every attack type. Instead, it builds a model of normality from benign-dominant data and uses this model to detect deviations during operation [26].

REVIEW ARTICLE

Formally, given training samples $x_i \in R^d$ that represent normal traffic, an OCL method learns a decision function that assigns an abnormality score $s(x)$ to a test sample. A sample is flagged as anomalous when the score exceeds a threshold τ, $s(x) > \tau$ [27], [28]. The score may be derived from different learning objectives, such as measuring distance from a learned normal region, reconstruction error relative to a normal pattern, or low likelihood under a normal data model. Regardless of the objective, the detection outcome depends on both the quality of the learned normal profile and the calibration of τ [29]. This paradigm is well aligned with IoT intrusion detection because IoT traffic is dominated by routine device communications, whereas intrusions are infrequent and may appear as novel behaviors [2], [24]. In such settings, obtaining complete and up-to-date labels for all attack types is difficult, and supervised IDS models can become outdated as threat patterns evolve. OCL mitigates this dependency by focusing on normal traffic and treating departures from normality as potential intrusions, which supports detection of emerging and zero-day threats. However, OCL performance in practice is sensitive to several factors that directly affect IDS reliability. The benign training set may be imperfect and can contain undetected attacks or noisy samples, which distorts the learned normal profile. Normal behavior may also change over time due to device updates, changes in network configuration, or shifts in usage patterns, leading to concept drift. Threshold selection is another key issue, as thresholds calibrated on one dataset split or operating condition may not transfer to another. Differences in feature extraction and preprocessing can further influence the learned normal representation and the resulting scores. These considerations highlight that OCL effectiveness should be assessed not only by accuracy-oriented metrics, but also by stability, false-alarm behavior, and suitability under realistic IoT operating conditions.

3. RELATED WORKS

Prior review articles have examined OCL and anomaly detection from multiple perspectives, including classical one-class classifiers, autoencoder-based methods, and recent deep formulations. These surveys differ in scope. Some focus on specific model families or data modalities, while others provide broader conceptual discussions. Table 1 summarizes representative reviews and contrasts their coverage in terms of methods, evaluation emphasis, and application focus. The main limitations observed across these reviews are summarized below.

3.1. Lack of IoT-Specific IDS Perspective

Although [16] and [13] are positioned in the IDS domain, the reviews in [30] is discussed in a big-data context, and in [31] are largely domain-agnostic rather than IoT IDS. As a result, intrusion-focused requirements—such as operational false-alarm constraints, evolving threat behavior, and deployment-layer considerations are not treated consistently across the surveyed literature.

3.2. Limited Coverage of OCL Families

The survey in [16] evaluates a small set of classical and reconstruction-based models (OC-SVM, iForest, AE, VAE) and is therefore limited in breadth, while [13] focuses exclusively on AE variants which prevents cross-family comparison. Even large-scale empirical studies do not fully cover deep OCL diversity; for example, [32] includes only one deep one-class formulation Deep SVDD alongside one-class SL, leaving other deep variants unaddressed. In contrast, [33] lists a wider set of OCL methods, including OC-DNN, but remains conceptual and does not provide empirical comparison.

3.3. Inconsistent Evaluation Practices

The table shows that some works are empirical but differ substantially in evaluation design: [13] emphasizes dataset analysis and model evaluation across datasets, [16] proposes a unified evaluation framework within the AE family, and [32] performs statistical comparison across 50 datasets. Meanwhile, several reviews are primarily conceptual and provide little or no empirical benchmarking [31], [33]). This mix of evaluation styles, datasets, and reporting practices makes it difficult to draw consistent conclusions across reviews.

3.4. Lack of Deployment-Feasibility Analysis

As reflected in the table, the reviewed works mainly concentrate on model comparison or conceptual organization, with limited attention to operational factors such as inference cost, memory footprint, latency, energy implications, and deployment suitability across device/edge/fog/cloud layers. This omission is particularly important for IoT IDS, where feasibility constraints can be as decisive as detection performance.

Table 1 Comparison of Existing Review Studies on OCL Techniques

Ref	OCL Discussed	Contribution	Strength	Limitation	Domain
[16]	OC-SVM, iForest, AE, VAE	dataset analysis + model evaluation	Compares the OCL on wide range of datasets	Very limited OCL models	IDS
[13]	SAE, SSAE, DAE,	unified model	Strong empirical finding	restricted only to AE	IDS

REVIEW ARTICLE

	ContAE, CAE	evaluation framework	that ContAE outperforms	family models	
[30]	OC-SVM, SVDD, GMM PCA, ELM	Identifies imbalance, & noise as unresolved OCL challenges.	Compares a wide range of OCL algorithms	No empirical comparison; no taxonomy; do not cover deep OCL	big-data context
[31]	OCC for feature, text, & image data	Provides conceptual evaluation framework	Reports that many OCL violate OC constraints.	No empirical comparison	No specific domain
[32]	SVDD, iForest, Deep SVDD, AE	Identifies the best OCL statistically.	Evaluates on across 50 datasets	Under deep OCL, only Deep SVDD is compared	No specific domain
[33]	OC-SVM, SVDD, decision-tree OCC, PCA, ensemble OCC, graph OCC, OC-DNN	Provides unified taxonomy based on methodology type, & application domain.	Covers a wide range of OCL algorithms	No empirical comparison; focuses on conceptual review only.	No specific domain

4. TAXONOMY OF OCL APPROCHES

IoT intrusion detection involves heterogeneous data behaviors, which has led to the development of diverse OCL approaches with different modelling objectives and computational profiles. As summarized in the taxonomy presented in Figure 1, these approaches are broadly organized into three families corresponding to increasing levels of representational depth. This taxonomy provides a structural overview of the field and serves as the organizing framework for the detailed discussions in the subsequent subsections.

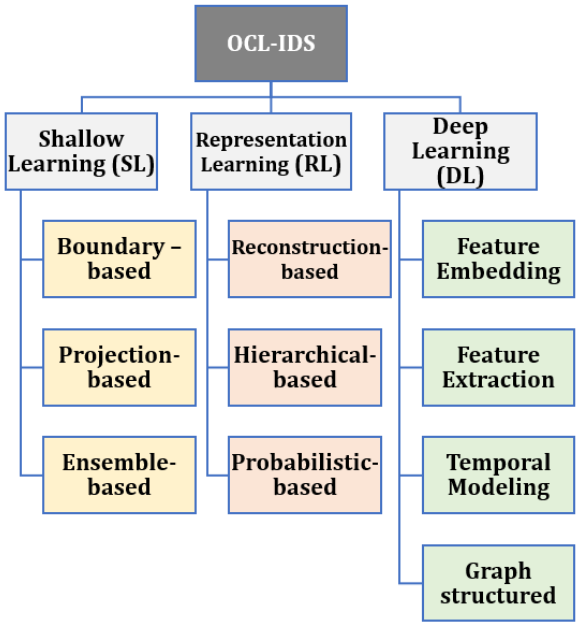

Figure 1 Structured Taxonomy of OCL-Based IDS

4.1. Shallow Learning (SL)

Shallow OCL approaches represent the earliest category of methods applied to IoT intrusion detection. These techniques are generally lightweight, interpretable, and computationally efficient, which makes them well-suited for resource-constrained IoT environments [34]. However, their limited capacity to capture non-linear or dynamic patterns restricts their effectiveness in complex traffic scenarios. Broadly, shallow OCL approaches can be divided into three subcategories:

4.1.1. Boundary-Based Methods

Boundary-based methods define a decision function that encloses normal data and isolates potential anomalies. A well-established representative is the OC-SVM [11], [35], which learns a hyperplane in a high-dimensional feature space that maximizes the margin between the origin and the set of normal data points as shown in Figure 2. Formally, given training samples $x_i \in R^d$ mapped into a feature space via $\phi(x_i)$, the OC-SVM solves the following optimization problem [11]:

$$\min_{w,\rho,\xi} \frac{1}{2}\|w\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho \quad (1)$$

Subject to:

$$\left(w \cdot \phi(x_i)\right) \geq \rho - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n$$

Where w is the weight vector, ρ is the offset that controls the boundary, ξ_i are slack variables allowing soft violations, n is the number of samples, and $\nu \in (0,1]$ controls the trade-off between the fraction of outliers and support vectors. The resulting decision function for a test sample x is[36], [37]:

$$f(x) = \text{sgn}\left(\left(w \cdot \phi(x)\right) - \rho\right) \quad (2)$$

Where f(x)=+1 denotes normal data and f(x)=−1 indicates an anomaly. OC-SVM has been widely adopted in IoT intrusion detection due to its ability to handle high-dimensional and imbalanced data. However, its performance depends heavily on kernel and parameter selection, and it exhibits scalability

REVIEW ARTICLE

challenges when applied to large-scale IoT traffic datasets. The overall OC-SVM workflow used in boundary-based detection is summarized in Algorithm 1.

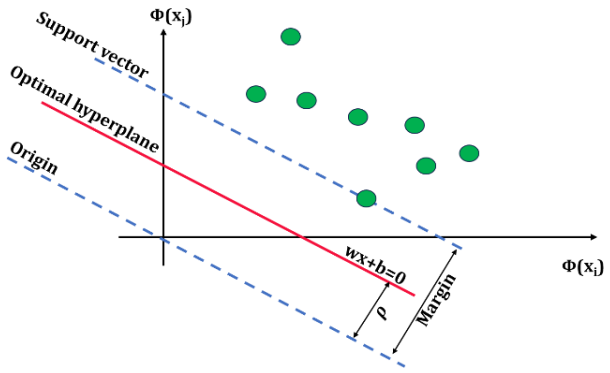

Figure 2 Decision Boundary Illustration of OC-SVM

Input: Normal training samples $X = \{x_i\}_{i=1}^{n}$

Output: Decision function f(x)

Feature mapping: Compute $\phi(x_i)$ for all $x_i \in X$.

Boundary optimization: Learn the parameters w, ρ, ξ_i by solving the OC-SVM optimization problem given in Equation (1).

Decision function: Form the classifier using the decision rule defined in Equation (2).

Classification: For any test sample x:

$f(x) > 0 \Rightarrow$ normal and $f(x) \leq 0 \Rightarrow$ Intrusion

Algorithm 1Boundary-Based OCL

4.1.2. Statistical and Projection-Based Methods

Statistical and projection-based approaches are inherently unsupervised techniques that operate without labeled anomalies. In the context of intrusion detection, they align with the OCL paradigm because they are trained primarily on normal data distributions and identify significant deviations as potential intrusions. Their simplicity, low computational overhead, and suitability for resource-constrained devices make them attractive for IoT environments, although their reliance on linear assumptions and stable distributions limits their adaptability to dynamic traffic conditions. A representative technique in this category is PCA. Given training data $X \in R^{nxd}$, PCA constructs the covariance matrix [38], [39]:

$$C = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T \quad (3)$$

Where x_i is the i^{th} sample and μ is the mean vector. By solving the eigenvalue decomposition of C, the top-k eigenvectors form a projection matrix P_k that captures the dominant variance. A new data point x is reconstructed as:

$$\hat{x} = P_k + P_k^T(x - \mu) \quad (4)$$

The reconstruction error,

$$e(x) = \|x - \hat{x}\|^2 \quad (5)$$

Serves as the anomaly score, where larger values of *e(x)* indicate potential intrusions. Normal IoT traffic generally aligns with the principal subspace and yields low reconstruction error, whereas anomalies exhibit significant residual variance. PCA is computationally efficient and interpretable, making it a suitable candidate for lightweight intrusion detection on IoT devices. Nevertheless, its linearity assumption restricts its ability to capture non-linear correlations, and its performance deteriorates when traffic distributions evolve over time. The overall procedure followed by projection-based methods is summarized in Algorithm 2.

Input: Normal training samples $X = \{x_i\}_{i=1}^{n}$

Output: anomaly score function e(x)

Covariance Estimation: Compute the covariance matrix C using Equation (3).

Eigenvalue Decomposition: Solve the eigenvalue decomposition of Cand select the top-k eigenvectors to form the projection matrix P_k.

Reconstruction of Input Sample: For a new data point x, compute its reconstruction using Equation (4)

Anomaly Score Computation: Compute the reconstruction error using Equation (5)

Classification: For any test sample x:

$e(x) \leq \tau \Rightarrow$ normal and $e(x) > \tau \Rightarrow$ intrusion

τis threshold chosen from training statistics

Algorithm 2 Projection-Based OCL

4.1.3. Ensemble-Based Methods

Ensemble and partitioning-based methods detect anomalies by leveraging multiple weak learners or by recursively dividing the feature space to distinguish normal from abnormal instances. These methods are particularly attractive for IoT intrusion detection due to their scalability and efficiency in handling large datasets without requiring explicit distributional assumptions.

A prominent technique in this category is the iForest, which isolates anomalies by constructing an ensemble of randomly generated binary trees[36]. The key intuition is that anomalous samples are easier to isolate since they typically lie in sparse or irregular regions of the feature space, requiring fewer partitions to separate them from normal data. For a given sample *x*, the expected path length across all trees is defined as [40][41].

REVIEW ARTICLE

$$E[h(x)] = \frac{1}{t}\sum_{j=1}^{t} h_j(x) \quad (6)$$

Where t is the total number of trees and $h_j(x)$ is the path length of x in the j^{th} tree. The anomaly score is then defined as:

$$s(x,n) = 2^{-\frac{E[h(x)]}{c(n)}} \quad (7)$$

Where n is the number of samples used to construct the forest, and c(n) is the average path length of unsuccessful searches in a binary search tree, approximated as:

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \quad (8)$$

With *H(i)* denoting the i^{th} harmonic number. Anomalous points generally exhibit shorter path lengths, leading to higher anomaly scores, whereas normal points require longer paths for isolation. iForest is computationally efficient, scales well to high-dimensional data, and avoids the explicit calculation of distance or density functions. However, its reliance on random partitioning may yield elevated false-positive rates in heterogeneous IoT traffic. Overall, ensemble and partitioning-based approaches, exemplified by iForest, offer a scalable and interpretable solution for anomaly detection in IoT IDS. Nonetheless, careful calibration is often required to balance detection performance with false alarm reduction in complex environments. The complete procedure followed by iForest is summarized in Algorithm 3.

Input**:** Normal training samples $X = \{x_i\}_{i=1}^{n}$

Output: anomaly score function s(x,n)

Forest Construction: Build an ensemble of tisolation trees, each trained on a subsample of X.

Path Length Computation: For a given test sample x, compute its path length $h_j(x)$in each tree $j = 1, \ldots, t$.

Expected Path Length: Estimate the average path length across the forest using Equation (6)

Normalization Term: Compute the normalization factor c(n) using Equation (8).

Anomaly Score: Compute the anomaly score using Equation (7)

Classification: For any test sample x:

$s(x,n) \leq \tau \Rightarrow$ normal and $s(x,n) > \tau \Rightarrow$ intrusion

τ is threshold chosen from training statistics

Algorithm 3 Ensemble-Based OCL

4.2. Representation Learning (RL)

RL extends the OCL paradigm by enabling models to automatically extract informative, non-linear representations of data rather than relying on handcrafted features. For IoT intrusion detection, this capability is especially critical, as traffic patterns are heterogeneous, dynamic, and difficult to model with shallow techniques [42]. By transforming raw inputs into compact latent spaces that preserve the essential structure of normal behavior, RL methods enhance the detection of subtle deviations that may indicate intrusions. This paradigm leverages reconstruction-based learning to model normal IoT traffic and identify anomalies through deviations in reconstruction quality.

4.2.1. Latent Reconstruction-based Approach

Latent reconstruction-based approaches employ reconstruction fidelity as the principal criterion for anomaly detection. The basic AE serves as the foundational model in this category, designed to learn compact latent representations that capture the intrinsic regularities of normal IoT traffic [11], [43]. By reconstructing inputs from these latent embeddings, the AE minimizes reconstruction loss for benign patterns while producing larger errors for anomalous or unseen behaviors. This unsupervised learning paradigm enables OC discrimination without requiring explicit anomaly labels and provides a scalable baseline for intrusion detection in resource-constrained IoT environments. Formally, an AE consists of two components as shown in Figure 3, an encoder function *f(·)* that maps an input $x \in R^d$ into a latent representation $z \in R^k$, and a decoder function *g(·)* that reconstructs the input from this latent space [13]:

$$\hat{x} = g\big(f(x)\big) \quad (9)$$

The anomaly score for a sample x is then computed as the reconstruction error:

$$e(x) = \|x - \hat{x}\|^2 \quad (10)$$

With higher values of *e(x)* indicating potential intrusions.

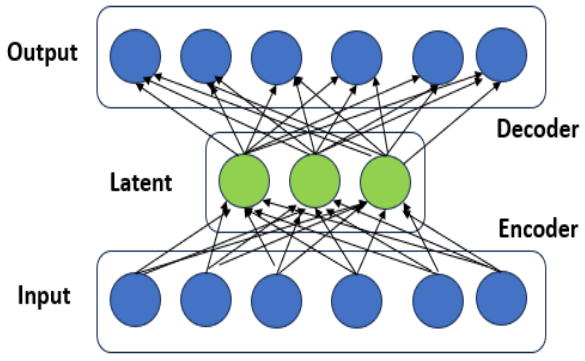

Figure 3 Conceptual Architecture of AE

AEs capture non-linear dependencies in IoT traffic, scale to high-dimensional inputs, and operate in an unsupervised OC setting. Their latent representations can also serve as compact features for downstream analytics. The workflow of AE is

REVIEW ARTICLE

summarized in Algorithm 4. However, basic AEs may overfit, fail to generalize in dynamic environments, or lack sufficient capacity to model hierarchical traffic patterns. Overall, AEs represent an important step beyond SL methods, laying the foundation for advanced variants such as stacked, sparse and contractive AEs that provide greater adaptability and robustness.

Input: Normal training samples $X = \{x_i\}_{i=1}^{n}$

Output: anomaly score function e(x)

Latent Encoding: For each training sample x_i, compute its latent representation using $z_i = f(x_i)$,

Reconstruction: Reconstruct each input using the decoder according to Equation (9)

Anomaly Score Computation: Compute the reconstruction error using Equation (10)

Classification: For any test sample x:

$e(x) \leq \tau \Rightarrow$ normal and $e(x) > \tau \Rightarrow$ intrusion

τis threshold chosen from training statistics

Algorithm 4 Latent Reconstruction-Based OCL

4.2.2. Hierarchical-Based Approach

Basic AEs provide a useful mechanism for anomaly detection, but their shallow structure often lacks the representational power to capture the intricate, hierarchical dependencies in IoT traffic. The SAE address this limitation extending the latent reconstruction framework into a hierarchical architecture that learns progressively abstract representations of input data across multiple encoding and decoding layers. This layered feature extraction enhances the model's ability to detect complex anomalies that may remain indistinguishable in raw encoded features. In a deep AE, each encoder layer transforms the input into higher-level representations, while the decoder symmetrically reconstructs data from these hierarchical embeddings. The encoding process are expressed as [13], [17]:

$$z = f^{(L)}\left(f^{(L-1)}\left(\cdots f^{(1)}(x)\right)\right) \quad (11)$$

Where $f^{(L)}(\cdot)$ denotes the non-linear transformation at the l^{th} encoder layer, and the decoder reconstructs the input as:

$$\hat{x} = g^{(1)}\left(g^{(2)}\left(\cdots g^{(L)}(z)\right)\right) \quad (12)$$

As defined in Equation (11) and Equation (12), the hierarchical stacking of AEs enables multi-level representation learning, capturing both local and global dependencies within IoT traffic, thereby improving robustness against noise and subtle intrusion patterns. Variants such as sparse AEs impose sparsity constraints on latent activations to yield compact and interpretable representations, while denoising AEs enhance resilience by reconstructing original inputs from partially corrupted data. Collectively, these hierarchical and variant architectures increase the expressive capacity and generalization capability of the model across heterogeneous IoT traffic scenarios. However, these benefits come at the cost of increased computational demand and more complex training, which may pose challenges for deployment on highly resource-constrained IoT devices.

4.2.3. Probabilistic Latent Modeling-Based Approach

While deterministic AEs learn a fixed latent representation of normal IoT traffic, they provide limited insight into the underlying data uncertainty and variability. Probabilistic latent modeling, most notably realized through the VAE, extends this framework by introducing a generative probabilistic formulation that captures the distribution of normal data rather than single-point embeddings [44]. This probabilistic view improves generalization and provides a more flexible modeling of normal data, which is particularly valuable for IoT intrusion detection where traffic patterns are dynamic and diverse. Formally, in a VAE, the encoder does not map the input x directly to a latent vector but to a distribution parameterized by a mean $\mu(x)$ and variance $\sigma^2(x)$. The latent variable z is obtained through stochastic process defined as [8]:

$$z = \mu(x) + \sigma(x) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \quad (13)$$

The decoder reconstructs the input from this latent representation $\hat{x} = g(z)$. The VAE training objective is to maximize the evidence lower bound (ELBO), which balances reconstruction accuracy with regularization of the latent space:

$$\mathcal{L}_{VAE} = E_{q(z|x)}[\|x - \hat{x}\|^2] + \beta D_{KL}\big(q(z|x)|p(z)\big) \quad (14)$$

Where D_{KL} denotes Kullback–Leibler divergence between the approximate posterior q(z|x) and the prior p(z), and β controls the trade-off between reconstruction fidelity and latent regularization.

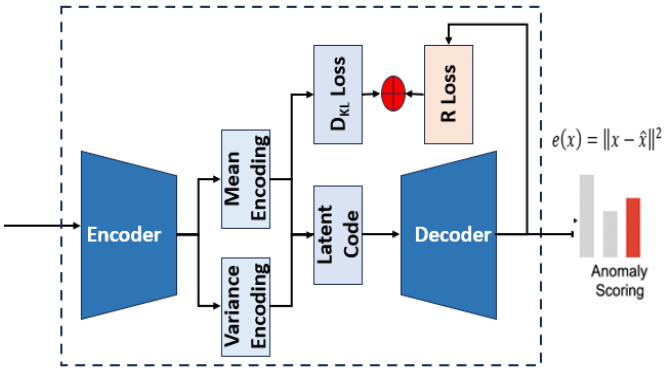

Figure 4 Conceptual Architecture of VAE

REVIEW ARTICLE

For anomaly detection, VAEs leverage both the reconstruction error and the likelihood of a sample under the learned latent distribution. Normal IoT traffic is expected to reconstruct with low error and high likelihood, while anomalies deviate significantly in one or both measures as shown in Figure 4. VAEs offer several advantages over deterministic AEs: they provide smoother latent spaces, better generalization to unseen patterns, and the ability to capture variability in IoT traffic distributions. However, their benefits come with challenges. Training VAEs is computationally more demanding and sensitive to hyperparameter tuning, and the added complexity may limit their deployment on severely resource-constrained IoT devices. The overall procedure used in probabilistic latent modeling–based OCL is summarized in Algorithm 5.

Input: Normal training samples $X = \{x_i\}_{i=1}^{n}$

Output: anomaly score function e(x)

Latent Sampling: For an input x, compute its latent variable using the stochastic process in Equation (13)

Reconstruction: Reconstruct each input using the decoder $\hat{x} = g(z)$.

Anomaly Score Computation: compute the score using the reconstruction and regularization terms of the ELBO using Equation (14)

Classification: For any test sample x:

$\mathcal{L}_{\mathcal{VAE}} \leq \tau \Rightarrow$ normal and $\mathcal{L}_{\mathcal{VAE}} > \tau \Rightarrow$ intrusion τis threshold chosen from training statistics

Algorithm 5 Probabilistic Latent Modeling-Based OCL

4.3. One Class Deep Neural Networks (OC-DNN)

Although AE-based approaches have advanced the state of OCL for IoT intrusion detection, they remain constrained by reconstruction bias and limited ability to capture complex spatio-temporal dependencies inherent in IoT traffic. In particular, reconstruction-driven models may inadvertently reproduce anomalous inputs with low error, while failing to fully exploit structural or sequential characteristics of communication patterns. To address these limitations, OC-DNN have emerged as the next stage of development [38], [45]. These methods leverage the expressive power of deep architectures to learn hierarchical features, model temporal dynamics, and represent relational structures in device interactions. By moving beyond purely reconstruction-based strategies, deep OCL approaches offer enhanced adaptability and scalability, making them well-suited for large-scale and heterogeneous IoT environments.

4.3.1. Feature Extraction Based Approach

Feature extraction-based OC models employ convolutional neural networks (CNNs) to learn spatially structured and discriminative embeddings of benign IoT traffic. Unlike reconstruction-driven frameworks that emphasize input regeneration, these models focus on hierarchical feature extraction to capture spatial correlations and local dependencies in structured traffic representations [46] such as flow matrices or packet feature maps as shown in Figure 5. Formally, let $x \in R^{h \times w}$ denote an input representation of IoT traffic. A CNN applies convolutional filters $W(l)$ with activation functions $\sigma(\cdot)$ across layers [47]:

$$h^{(l)} = \sigma\left(W^{(l)} * h^{(l-1)} + b^{(l)}\right) \quad (15)$$

Where $h^{(0)} = x$, $*$ denotes convolution, and $h^{(l)}$ represents the feature map at layer l. The final latent representation z is obtained through successive convolutions and pooling operations.

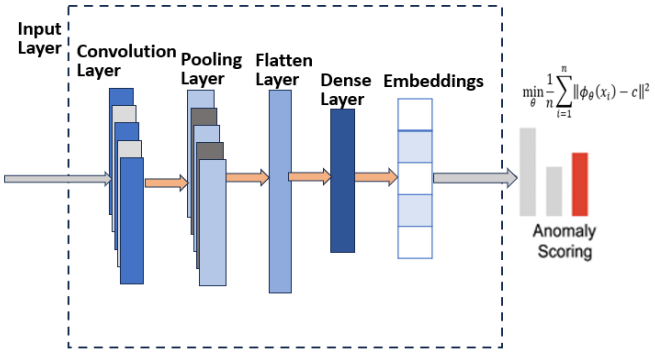

Figure 5 Conceptual Architecture of OC-CNN

Input: Normal training samples $X = \{x_i\}_{i=1}^{n}$

Output: anomaly score function e(x)

Convolutional Feature Extraction: Extract hierarchical features using the CNN update rule in Equation (15)

Latent Representation Obtain the final latent feature vector: $z = \phi(x)$, where $\phi(\cdot)$ represents the CNN feature extraction process

Anomaly Score Computation: For a test sample x, compute its distance to the center using $s(x) = \| \phi(x) - c \|^2$, Classification: For any test sample x:

$s(x) \leq \tau \Rightarrow$ normal and $s(x) > \tau \Rightarrow$ intrusion τis threshold chosen from training statistics

Algorithm 6 Feature Extraction–Based OCL

In the OC setting, embeddings $\phi_\theta(x)$ are optimized with respect to the hypersphere objective introduced in the previous subsection, which enforces clustering around a central point c. For IoT intrusion detection, CNN-based OC networks effectively model local spatial dependencies in traffic data, enabling high-accuracy detection of fine-grained anomalies under moderate computational cost. They are

REVIEW ARTICLE

particularly advantageous in static or short-term traffic scenarios where spatial correlations dominate. However, due to limited receptive fields, CNNs are less effective at capturing long-range temporal dynamics that characterize sequential IoT behaviors. Moreover, deeper hierarchical structures increase memory and energy requirements, presenting deployment challenges for real-time inference on resource-constrained edge nodes. Despite these trade-offs, feature extraction-based OC-DNNs serve as a crucial bridge between compact embedding methods and more advanced temporal or graph-structured anomaly detection frameworks. The overall workflow of this approach is summarized in Algorithm 6.

4.3.2. Feature Embedding Based Approach

Feature embedding-based OC models aim to learn a compact latent representation of normal IoT behavior by directly optimizing the distribution of embeddings, rather than reconstructing inputs. The core objective is to project normal samples into a bounded region of the feature space, typically a hypersphere such that deviations from this region indicate anomalous activity [48]. This compactness principle enforces intra-class similarity and inter-class separability without requiring labeled anomalies.

A representative example is Deep SVDD, which extends the classical SVDD formulation with DNN. Instead of reconstructing inputs, a DNN $\phi_\theta(\cdot)$ maps each sample x_i into a latent space, where embeddings are encouraged to cluster around a center c. The optimization problem is defined as [27]:

$$\min_\theta \frac{1}{n}\sum_{i=1}^{n}\|\phi_\theta(x_i) - c\|^2 \quad (16)$$

With $\phi_\theta(x_i)$ denotes the learned embedding of sample x_i and c is the hypersphere center. At inference, samples that lie far from the center are flagged as anomalies. In the IoT context, hypersphere-based embeddings enable the network to learn compact and noise-resilient representations of benign traffic while mitigating the influence of outliers. This approach is particularly advantageous when abnormal samples are scarce or absent during training. Recent developments have introduced enhancements such as adaptive center estimation, margin regularization, and feature normalization to improve stability under evolving traffic conditions.

Nevertheless, compactness-based methods can be sensitive to feature collapse and often assume a single homogeneous cluster of normal data. This assumption limits their ability to capture the multi-modal and hierarchical structures characteristic of large-scale IoT environments. Despite these challenges, feature embedding-based OC-DNNs remain a theoretically grounded and computationally efficient paradigm, forming the foundation for subsequent advances in temporal and graph-based anomaly detection frameworks. The workflow of the feature embedding-based approaches is summarized in Algorithm 7.

Input**:** Normal training samples $X = \{x_i\}_{i=1}^{n}$

Output: anomaly score function e(x)

Feature Embedding: Map each input xinto the latent space using the embedding network: $h = \phi(x)$.

Center Estimation: Compute the hypersphere center cof the normal embeddings and Equation (16)

$$c = \frac{1}{n}\sum_{i=1}^{n}\phi(x_i).$$

Anomaly Score Computation: For a test sample x, compute s(x) its distance to the center: $s(x) = \| \phi(x) - c \|^2$,

Classification: For any test sample x:
$s(x) \le \tau \Rightarrow$ normal and $s(x) > \tau \Rightarrow$ intrusion
τis threshold chosen from training statistics

Algorithm 7 Feature Embedding–Based OCL

4.3.3. Temporal Modeling Based Networks

Temporal modeling-based OC networks extend feature learning to sequential domains, where IoT traffic exhibits strong temporal dependencies across packets, sessions, or device interactions. Unlike convolutional architectures that primarily capture spatial correlations, these models leverage recurrent neural networks (RNNs) and their gated variants long short-term memory (LSTM) and gated recurrent unit (GRU) networks to model temporal evolution and context-dependent behavior in benign communication patterns [49]. Formally, given a sequence of input vectors $x_1, x_2, \ldots, x_T$ a recurrent model updates its hidden state h_t at each time step *t* according to [46]:

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b) \quad (17)$$

Where W_h and W*x* are recurrent and input weight matrices, b is a bias vector, and $\sigma(\cdot)$ is a non-linear activation function. LSTMs and GRUs extend this formulation by introducing gating mechanisms that regulate how much past information is retained or forgotten, improving stability when modeling long sequences.

In a OC setting, the final hidden representation or aggregated sequence embedding is compared against the learned distribution of normal traffic. Anomalies are identified when the sequence representation deviates significantly from this learned normal profile, either through distance-based objectives or thresholds on prediction errors.

Input**:** Normal training samples $X = \{x_i\}_{i=1}^{n}$

Output: anomaly score function e(x)

Hidden State Initialization: h_0=0.

REVIEW ARTICLE

Temporal Feature Extraction: For each time step t update the hidden state using the recurrence in Equation (17).

Sequence Representation: Obtain the final temporal representation: $z = h_T$,
Anomaly Score Computation: For a test sample x, compute its distance to the center using $s(x:T) = \| z - c \|^2$,
Classification: For any test sample x:

$s(x:T) \leq \tau \Rightarrow$ normal and $s(x:T) > \tau \Rightarrow$ intrusion
τis threshold chosen from training statistics

Algorithm 8 Temporal Modeling–Based OCL

Recurrent and sequence-based models offer key advantages in IoT intrusion detection by capturing temporal dependencies and adapting to the sequential nature of communication flows. They are particularly effective in scenarios where attacks manifest as subtle timing irregularities, delayed responses, or unusual sequence patterns that static models may overlook. However, their practical deployment is constrained by high computational and memory demands, which limit applicability on resource-constrained IoT devices. Training stability, sensitivity to sequence length, and latency in real-time detection further remain challenges that require careful optimization. The algorithmic flow for temporal OCL is provided in Algorithm 8.

4.3.4. Graph Structured Neural Networks

IoT environments can naturally be represented as graphs, where devices act as nodes and their communication flows or protocol interactions form edges. Unlike CNNs and RNNs that focus primarily on spatial or sequential dependencies, GNNs are designed to capture relational structures, making them particularly effective for modeling device-to-device interactions and heterogeneous network topologies.

In a OCL setting, GNNs are trained exclusively on normal communication graphs, and anomalies are identified as deviations in node, edge, or graph-level embeddings from the learned normal patterns [50] as shown in Figure 6. A typical GNN layer updates the representation of a node v by aggregating information from its neighbors $N(v)$ [51]:

$$h_v^{(l+1)} = \sigma\left(W^{(l)} \cdot \mathrm{AGG}\{h_v^{(l)}, h_u^{(l)} : u \in N(v)\}\right) \quad (18)$$

Where $h_v^{(l)}$ is the representation of node v at layer l, AGG denotes an aggregation function (e.g., mean, sum, attention), $W^{(l)}$ is a learnable weight matrix, and $\sigma(\cdot)$ is a non-linear activation function. After several layers, the final embeddings $h_v^{(L)}$ capture both the local and global structural context of each node within the IoT graph. For OC anomaly detection, these embeddings can be constrained to remain within a compact region of the latent space, or evaluated through reconstruction objectives that attempt to recover node features or adjacency structures. Anomalous devices or flows manifest as embeddings deviate from the learned distribution of normal traffic, or as graph regions with high reconstruction error. GNN-based OCL methods provide several advantages for IoT intrusion detection. They are capable of modeling device heterogeneity, capturing both direct and indirect relationships among nodes, and enabling anomaly detection at multiple levels (node, edge, or global graph). However, they also face notable challenges: scalability issues when applied to large-scale IoT deployments, high computational overhead for training and inference, and sensitivity to evolving network structures in dynamic environments. The procedure underlying graph-based one-class learning is outlined in Algorithm 9.

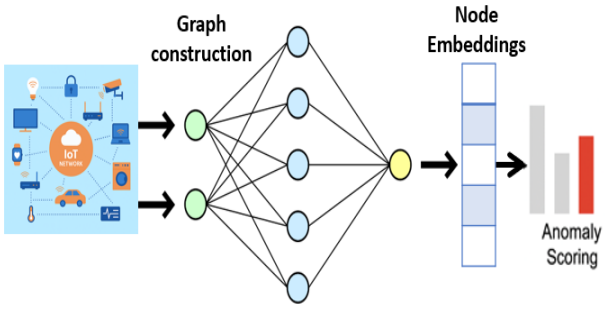

Figure 6 Conceptual Architecture of OC-GNN

Input: Normal training samples $X = \{x_i\}_{i=1}^n$

Output: anomaly score function e(x)

Graph Construction: From normal samples X, G=(V, E)

Neighborhood Aggregation (GNN Update): For each layer l and node v, update its representation using Equation (18)

Final Node Embedding: After L layers, obtain the latent embedding: $z_v = h_v^{(L)}$.

Anomaly Score Computation: Compute anomaly score for each node using $s(v) = \| z_v - c \|^2$

Classification: For any test sample x:

$s(v) \leq \tau \Rightarrow$ normal and $s(v) > \tau \Rightarrow$ intrusion
τ is threshold chosen from training statistics

Algorithm 9 GNN–Based OCL

5. COMPARATIVE ANALYSIS

5.1. Datasets Used Across OCL IDS Studies

OCL-based intrusion detection studies are typically evaluated using a limited set of benchmark datasets that vary in scale, feature dimensionality, class distribution, and attack diversity. These variations affect how consistently normal traffic patterns can be captured and how reliably a threshold can be tuned. When normal traffic is sparse, noisy, or heterogeneous, threshold selection becomes more delicate and controlling false alarms becomes more difficult under changing

REVIEW ARTICLE

conditions. OCL-based intrusion detection studies are typically evaluated using a limited set of benchmark datasets that vary in scale, feature dimensionality, class distribution, and attack diversity. These variations affect how consistently normal traffic patterns can be captured and how reliably a threshold can be tuned. When normal traffic is sparse, noisy, or heterogeneous, threshold selection becomes more delicate and controlling false alarms becomes more difficult under changing conditions [13], [16]. The difficulty level reported in Table 2 is interpreted using three practical indicators: the proportion of normal traffic (imbalance severity), the number of attack categories (attack diversity), and the overall dataset scale. Together, these factors affect how reliably an OCL model can learn normal behavior and how stable its threshold-based decisions remain in practice.

Table 2 Summary of Benchmark IDS Datasets Used in OCL Research

Dataset	# Samples	# Features	% Normal	% Attacks	Attack types	Difficulty Level
BoT-IoT[52]	37,763,497	46	0.36%	99.64%	4	Very Hard
NSL-KDD[53]	148516	41	51.88	48.11%	5	Medium
UNWSNB15[54]	257673	42	36	64	9	Hard
ToN-IoT[55]	22,339,021	42	3.5%	96.45	10	Very Hard
CIDIDS 2017[56]	230092	79	26.75%	73.24	14	Very Hard

As shown in Table 2, IoT-oriented datasets such as BoT-IoT [52], ToN-IoT [55], and CIDDS-2017 [56] exhibit extreme skew and broader attack coverage. Under these conditions, overall accuracy can be misleading, while false-alarm behavior becomes a central performance concern. Moreover, the very large scale of some datasets increases training time and memory requirements, which directly influences feasibility for edge and fog deployments. By contrast, NSL-KDD [53] and UNSW-NB15 [54] are widely used baselines with smaller scale and less extreme skew, making them convenient for controlled experimentation. However, they may not fully capture the volume and diversity observed in contemporary IoT traffic. These differences limit meaningful cross-study comparison. Reported performance often changes with the dataset and evaluation setup, so results from different papers are not always directly comparable. For fair benchmarking, studies should explicitly report the definition of the normal class and any potential contamination in training data, the procedure used to set decision thresholds, and key preprocessing steps such as scaling and feature selection. Performance reporting should also include detection rate and false-alarm rate as primary indicators, supported by F1-score and PR curves. if IoT deployment is stated goal, latency, memory footprint, and training cost should also be reported.

5.2. OCL Evaluation Framework

A systematic evaluation framework is essential to ensure consistency, and comparability in OCL research for IoT intrusion detection. As OCL models increasingly diverge in architectural design, learning paradigms, and deployment contexts, the absence of standardized evaluation practices often leads to fragmented or incomparable results [16], [27]. To address this gap, we introduce a unified evaluation framework that supports both methodological assessment and practical feasibility analysis for heterogeneous IoT environments. The proposed framework introduces a four-layer hierarchical structure (Figure 7) that spans the full lifecycle of OCL model development and evaluation. Each layer captures a distinct yet complementary dimension of assessment:

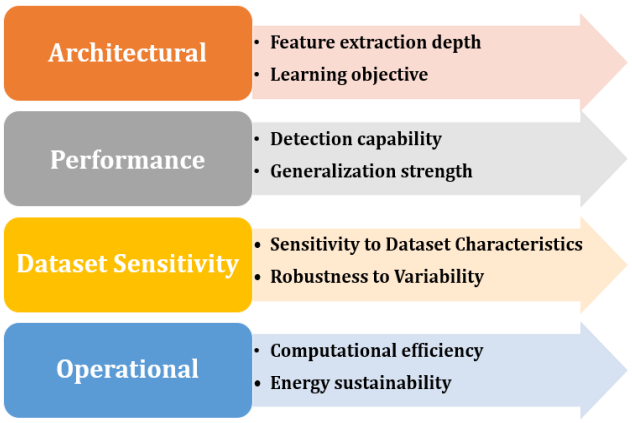

Figure 7 Conceptual Evaluation Framework for OCL

5.2.1. Architectural Evaluation Layer

This layer examines the intrinsic design of the model, including feature extraction depth, latent dimensionality, and the underlying learning objective whether reconstruction-based, compactness-based, or hybrid [33]. This layer ensures that architectural complexity is consistent with the computational and deployment constraints of IoT systems.

REVIEW ARTICLE

5.2.2. Performance Evaluation Layer

This layer defines the quantitative criteria used to assess the detection capability and generalization strength of OCL model. Common metrics such as precision, recall, F1-score, and ROC-AUC are employed to assess sensitivity to anomalies, false-alarm behavior, and detection robustness under varying network conditions. These metrics provide a consistent and comparable basis for evaluating OCL models across different datasets and attack scenarios.

5.2.3. Dataset Sensitivity Evaluation Layer

This layer interprets how dataset characteristics shape the performance of OCL models. Drawing on empirical results, it analyzes how factors such as imbalance severity, feature heterogeneity, traffic modality, and variability in normal-class behavior influence model outcomes. By linking dataset difficulty to observed performance patterns, this layer explains why certain OCL methods succeed while others fail under specific data conditions.

5.2.4. Operational Evaluation Layer

This layer evaluates the practical deployability of OCL models in resource-constrained IoT environments. Key operational factors include computational overhead, inference latency, memory footprint, energy efficiency, and scalability [57]. These attributes determine whether a model can operate reliably on edge or fog devices, and whether its real-time responsiveness meets the sustainability and performance requirements of IoT systems.

These four layers form a comprehensive evaluation methodology that moves beyond accuracy-centric reporting. The framework promotes balanced assessment, capturing both methodological robustness and deployment practicality, while acknowledging trade-offs uch as deeper models offering richer representations but incurring higher latency, and lightweight models providing efficiency at potential cost to precision.

5.3. Architectural Evaluation

Building upon the evaluation hierarchy depicted in Figure 7, this subsection concentrates on the architectural evaluation layer, which forms the foundation of the proposed framework. This layer assesses how model structure, representation depth, and learning objectives collectively influence the efficiency and effectiveness of OCL models for IoT intrusion detection.

As summarized in Table 3, OCL architectures can be grouped into three tiers. SL methods rely on predefined feature mappings and statistical boundary formation, leading to transparent and computationally light decision mechanisms; however, their fixed representations limit their ability to capture complex non-linear correlations and dynamic traffic variations that characterize large-scale IoT environments [48].

In contrast, RL models learn compact, non-linear latent spaces via reconstruction or probabilistic objectives, enabling improved adaptability and anomaly discrimination by modeling the regularities of benign traffic [16], [51]. While RL improves representational flexibility, it also increases architectural depth and parameter complexity; therefore, practical edge–fog adoption depends on lightweight variants and optimization to balance accuracy with efficiency [57][58].

Table 3 Comparative Analysis of Architectural Efficiency Across OCL Paradigm

Dimension	SL	RL	OC-DNN
Learning Principle	Relies on predefined features and statistical or geometric boundaries	Learns non-linear latent representations via reconstruction or probabilistic modeling.	Learns hierarchical, spatial, or temporal embeddings through DNN [16], [51].
Feature Handling	Handcrafted or kernel-transformed features.	Automatically extracts compact latent features.	Learns discriminative, high-level features optimized for compactness or sequence modeling [48].
Objective Function	Boundary or density-based separation.	Reconstruction or likelihood maximization.	Embedding compactness or structure-preserving optimization.
Adaptability	Low-static and sensitive to traffic drift.	Moderate-handles non-linearity but prone to reconstruction bias.	High-captures spatial, temporal, and structural variations.[46]
Interpretability	High-transparent decision boundaries.	Moderate-latent features partially explainable.	Low-black-box nature; improved via XAI [22].
Deployment Suitability	Ideal for edge devices.	Suitable for edge-fog with lightweight variants.	Preferable for edge-fog deployment.[57][58]
Representative Models	OC-SVM, SVDD, PCA, iForest	AE, Stacked AE, VAE	Deep SVDD, CNN-OCL, LSTM-OCL, GCN-OCL

REVIEW ARTICLE

At the upper end of the complexity spectrum, OC-DNN architectures extend one-class representation learning to hierarchical, temporal, and relational domains through deep embeddings [16], [51]. This added capacity yields stronger robustness and generalization across heterogeneous IoT traffic, particularly by capturing spatial/temporal/structural variations [46]. However, these benefits come with higher computational and energy overheads and lower inherent interpretability, often requiring explainability mechanisms to support practical adoption [22]. Consequently, effective deployment of OC-DNNs is best aligned with tiers that can support their resource demands with edge–fog settings when paired with hardware-aware optimization and model-efficiency techniques [57][58].

Overall, the architectural evaluation highlights a clear trade-off between representational capacity and efficiency: deeper architectures improve feature abstraction and adaptability [46], but increase computational burden and energy consumption. Therefore, architectural selection should be guided by the deployment tier with lightweight SL models for constrained edge devices, optimized RL models for edge–fog nodes, and high-capacity OC-DNNs for fog–cloud intrusion analytics [57][58].

5.4. Performance Evaluation Across IDS Studies

This subsection synthesizes performance results reported in existing OCL-based IoT intrusion detection studies, rather than introducing new experimental findings. Table 4 consolidates accuracy, precision, recall, F1-score, and AUROC values published across multiple studies evaluating representative OCL models on benchmark datasets such as NSL-KDD, UNSW-NB15, ToN-IoT, BoT-IoT, and CICIDS2017. Presenting these results in a unified format enables a coherent comparison of how different OCL paradigms perform under varied IoT traffic conditions.

Table 4 Comparative Analysis of Performance Efficiency Across OCL Paradigm

OCC	Dataset	Accuracy	Precision	Recall	F1	AUROC
OCSVM [11], [35], [59]	NSL-KDD	0.89	0.85	0.81	0.83	83.86
	UNSW-NB15	0.95	0.93	0.92	0.92	
	ToN-IoT	0.87	0.84	0.79	0.81	
	BoT-IoT	88.65	83.58	99.96	90.84	
iForest [35], [41], [42], [48]	NSL-KDD	0.86	0.83	0.78	0.80	94.59
	UNSW-NB15	0.94	0.92	0.91	0.91	
	ToN-IoT	0.85	0.82	0.77	0.79	
	BoT-IoT	66.29	84.37	46.55	59.26	
AE [13][2], [35], [42], [43]	NSL-KDD	0.91	0.89	0.84	0.86	
	UNSW-NB15	0.96	0.94	0.93	0.93	
	ToN-IoT	0.88	0.86	0.82	0.84	
	BoT-IoT	96.42	92.17	98.15	96.04	
SAE[13][60]	NSL-KDD	85.23		85.13	86.79	
	UNSW-NB15	87.16		96.58	91.1	
VAE[16][44]	NSL-KDD	0.90	0.87	0.83	0.85	
	UNSW-NB15	0.96	0.94	0.92	0.93	
	ToN-IoT	0.88	0.85	0.81	0.83	
GAN-AE[10], [44], [61]	NSL-KDD	91.12	87.27	98.81	92.68	
Deep SVDD[33], [35]	NSL-KDD					96.11
	UNSW-NB15	0.832		0.711	0.721	

REVIEW ARTICLE

	BoT-IoT	95.36	92.21	99.98	95.94	
OC-LSTM [45], [49]	NSL-KDD					96.86
	CICIDS2017					99.24

Across the surveyed literature, SL models such as OC-SVM [11], [35], [59] and iForest [35], [41], [42], [48] show moderate performance on balanced datasets but degrade on imbalanced IoT traffic. OC-SVM shows reasonable F1-scores on NSL-KDD and UNSW-NB15 but becomes less reliable on ToN-IoT and BoT-IoT. iForest is particularly impacted by imbalance, with results showing competitive AUROC on NSL-KDD but substantially reduced F1-score (≈0.59) on BoT-IoT. In contrast, RL models such as AE [13][2], [35], [42], [43] achieves strong F1-scores on both classical datasets and modern IoT benchmarks, including approximately 0.86 on NSL-KDD, 0.93 on UNSW-NB15, and 0.95 on BoT-IoT. VAE [16][44] exhibits similarly stable behavior across NSL-KDD, UNSW-NB15, and ToN-IoT, underscoring the advantage of latent representation learning in OC anomaly detection compared to SAE models [13][60]. GAN-AE [10], [44], [61] demonstrate the most consistent and competitive performance. Deep OC models display strong yet dataset-dependent performance. Deep SVDD [33], [35] achieves competitive results on BoT-IoT (F1 ≈ 0.96) and high AUROC on NSL-KDD but performs more weaker on UNSW-NB15 (F1 ≈ 0.72), reflecting limited robustness across heterogeneous datasets. Sequential models such as OC-LSTM [45], [49], evaluated in flow-based settings, report very high AUROC values (above 0.96 on NSL-KDD and 0.99 on CICIDS2017), highlighting the value of modeling temporal dependencies in modern IoT traffic. Overall, the evidence from existing IoT IDS studies indicates that reconstruction-based OCL methods offer the strongest cross-dataset reliability, shallow OCC models are suitable primarily for simpler or balanced datasets, and deep compactness-based approaches can achieve state-of-the-art results but with inconsistent stability. These findings provide the foundation for the dataset sensitivity analysis presented in Section 5.5.

5.5. Dataset Sensitivity Analysis

The comparative evidence drawn from existing IoT IDS studies indicates that OC learning performance is strongly shaped by the intrinsic characteristics of each dataset. Table 5 summarizes the best and worst performing OCC models across NSL-KDD, UNSW-NB15, ToN-IoT, and BoT-IoT, and reveals consistent patterns linking dataset difficulty to model behavior. On NSL-KDD, where the traffic distribution is relatively balanced and feature interactions are less complex, AE-based models achieve the highest F1-scores, while tree-based methods such as iForest perform comparatively worse due to their limited ability to capture fine-grained feature dependencies.

Table 5 Sensitivity Analysis of OCL Methods Across IDS Datasets

Dataset	Best OCC	Worst OCC	Sensitivity Analysis Remarks
NSL-KDD	AE (0.86)	iForest (0.80)	Balanced dataset → deep feature reconstruction models perform better; tree models lose performance due to lack of complex boundary representation.
UNSW-NB15	AE, VAE (0.93)	Deep SVDD (0.721)	Heterogeneous modern traffic → AE, VAE learn richer latent representations; Deep SVDD struggles with multi-feature diversity and contamination.
ToN-IoT	AE (0.84)	iForest (0.79)	Multi-sensor + sparse abnormalities → reconstruction models generalize better; iForest splits fail under mixed IoT modalities.
BoT-IoT	AE (0.95)	iForest (0.59)	Extremely imbalanced dataset (0.36% normal). AE reconstructs benign traffic well; iForest fails because tree partitioning collapses under ultra-imbalance and sparse normal class.

A similar trend is observed in UNSW-NB15, although the dataset introduces more heterogeneous traffic and diverse attack behaviors. Here, AE and VAE again provide the strongest performance, benefiting from robust latent representation learning, whereas compactness-based Deep SVDD exhibits reduced effectiveness, likely due to contamination and structural variability in the normal class. Sensitivity becomes more pronounced in datasets with multi-modality or extreme imbalance. In ToN-IoT, which integrates multi-sensor IoT streams with sparse abnormal events, reconstruction-based models maintain performance advantages by generalizing well to mixed traffic modalities.

In contrast, iForest's hierarchical partitioning is less effective when normal samples are dispersed across heterogeneous feature spaces. The effect of imbalance is most evident in

REVIEW ARTICLE

BoT-IoT, where the normal class constitutes less than 1% of the dataset. Under such conditions, AE achieves the highest reported F1-score by accurately modeling benign patterns, while iForest deteriorates sharply as its partitioning strategy fails to separate extremely sparse normal samples from dense attack traffic.

Overall, the analysis demonstrates that dataset properties including imbalance severity, feature heterogeneity, and modality diversity substantially influence the relative success of OCL methods. Reconstruction-based models consistently exhibit resilience across varying dataset conditions, while shallow boundary models and compactness-based deep architectures show greater sensitivity to difficult data regimes. These observations highlight the necessity of incorporating dataset-aware evaluation when interpreting OCL performance in IoT intrusion detection.

5.6. Operational Evaluation

A major limitation of existing research on OCL for IoT intrusion detection is the absence of a systematic assessment of operational feasibility. While most studies benchmark OCL models based on detection accuracy, far fewer consider the practical constraints imposed by IoT device heterogeneity, resource limitations, and runtime execution costs. In IoT environments where memory budgets may be restricted to a few kilobytes, processors operate at sub-GHz frequencies, and energy availability varies widely the ability of a model to execute efficiently can be as important as its detection capability. The present study addresses this gap by presenting Table 6, which provides a detailed comparison of OCL families in terms of training complexity, inference cost, memory footprint, and deployment feasibility. This analysis serves as a practical decision-making tool that has been largely missing in prior literature.

Table 6 Comparative Analysis of Operational Efficiency Across OCL Paradigm

OCC Model	Training Complexity	Inference Complexity	Memory Requirements	Deployment Consideration (IoT/Edge)
OC-SVM	$O(n^2)$ to $O(n^3)$ for kernel training	O(s) where s = #support vectors	Stores support vectors	Practical only when dataset is small; memory grows with s
iForest	O(n · t) for t trees	O(tree depth)	Stores t trees	Efficient for IoT gateways;
PCA	Projection $O(d^3)$	Projection $O(d^3)$	Stores projection matrix	Ideal for low-power IoT due to deterministic structure
AE	O(n · (dh + $2h^2$) · E)	O(dh + $2h^2$)	Parameters approx O(dh + h^2)	Deployable on mid-tier edge devices; compressible
SAE	O(n · (dh + $2h^2$) · E · L)	O(dh + $2h^2$)	parameter O(L h^2)	Requires edge gateway; too heavy for microcontrollers
VAE	O(2n · (dh + $2h^2$) · E)	O(dh + $2h^2$)	Encoder + decoder + latent params	Suitable for GPU-enabled fog nodes; heavy for IoT
GAN-AE	O(2n · (dh + $2h^2$) · E)	O(dh + $2h^2$)	Parameters of both G and D	Not suitable for IoT; edge-fog recommended
Deep SVDD	O(n · (dh + $2h^2$) · E)	Forward only: O(dh + $2h^2$)	Based on CNN/ LSTM structure	Feasible on fog nodes; high cost for IoT devices
OC-LSTM	O(n · T · h^2) where T = sequence length	O(T · h^2)	Recurrent weights + hidden states	Very effective for time-series IoT but requires edge gateway hardware

The comparison in Table 6 reveals clear stratification across OCL model classes. Shallow statistical and boundary-based models, such as OC-SVM, offer mathematically elegant solutions but face challenges when scaled. Kernel-based OC-SVM training scales between $O(n^2)$ and $O(n^3)$, which is prohibitive even for moderate-sized traffic datasets. At inference time, OC-SVM remains lightweight, but storing the support vectors becomes the dominant memory cost—making this approach unsuitable for memory-constrained devices. By contrast, iForest exhibits linear training complexity with respect to the number of trees, and its inference cost is limited to tree traversal. This efficiency makes it appealing for IoT gateways; however, the number of trees and depth of each tree directly influence memory usage and latency.

The projection-based models, such as PCA, stand out for their deterministic computational pattern and minimal storage requirements. PCA's training cost is dominated by a matrix

REVIEW ARTICLE

decomposition step $O(d^3)$, but once the projection matrix is obtained, inference reduces to a simple matrix multiplication independent of dataset size. This makes PCA one of the few OCL techniques that can be deployed even on microcontroller-class IoT hardware. Its main limitation lies in its linearity assumption, which restricts detection performance when IoT traffic exhibits nonlinear or multi-modal structure. Under the RL category, AEs provide richer feature encoding capabilities but also introduce higher computational and memory demands compared to SL approaches. AEs offer a practical balance: although training involves iterative gradient updates over encoder–decoder parameters, inference can be executed through a single forward pass, making them feasible for edge devices when combined with pruning and quantization. SAEs and GAN-based AE hybrids, which increase representational depth to capture more complex patterns, incur substantially larger parameter footprints. Consequently, these models are better suited for edge nodes where memory and energy availability are less restrictive.

In the OC-DNN category, temporal models such as OC-LSTM and GRU-derived architectures specialize in capturing sequential and temporal dependencies that static or feed-forward models overlook. Their computational cost grows with sequence length T and hidden-state dimension h, resulting in $O(nTh^2)$ training overhead and noticeable inference latency. These models are particularly effective for detecting timing anomalies, burst patterns, and sequence-level irregularities in IoT communication flows. However, their runtime and memory requirements preclude deployment on low-tier IoT devices; instead, they are typically executed on IoT gateways or network-edge servers that offer moderate processing capability.

Feature embedding–based OC-DNNs, exemplified by Deep SVDD, aim to reduce reconstruction overhead by learning compact hypersphere-constrained embeddings. Their operational footprint depends strongly on the backbone network: lightweight CNN-based implementations can be executed on edge devices with limited resources, while deeper CNN or LSTM backbones offer better representation quality at the cost of significantly increased computation and memory demand, making them more appropriate for deployment at the fog tier. These models provide a favorable trade-off between expressiveness and efficiency, positioning them between reconstruction-based RL models and more computationally intensive temporal or generative architectures. Taken together, the deployment-oriented analysis in Table 6 highlights a critical insight: OCL models differ as much in their computational profile as in their detection capability, and these operational differences dictate where each model can realistically be deployed within a heterogeneous IoT infrastructure. Shallow and linear models align with device-level constraints; compact deep models suit the edge tier; and resource-intensive generative or sequential models align with fog and cloud environments. By explicitly linking computational complexity, memory cost, and deployability, this review offers a detailed operational perspective missing in prior OCL surveys. This analysis equips researchers and practitioners with practical guidance for selecting intrusion detection models that balance accuracy, latency, energy constraints, and hardware capabilities, ensuring realistic and efficient deployment in real-world IoT systems.

5.7. Recent Advances in OC-IDS (2024-2026)

To address the need for broader and more recent coverage, this subsection synthesizes recent studies that formulate intrusion detection as an OCL problem during the year 2024-2026. Consistent with the taxonomy-driven comparison in Section 5, the recent literature can be organized into shallow, representation-learning, and deep OCL. Overall, the new evidence largely corroborates the earlier comparative trends while adding clearer findings on robustness under adversarial manipulation, deployment feasibility on constrained platforms, and federated training.

5.7.1. SL-based OCL

Recent IoT studies reaffirm that classical one-class baselines remain competitive when carefully engineered for practical constraints. For example, an IoT anomaly detection framework based on OC-SVM and Isolation Forest reports strong results on ToN_IoT and evaluates reliability through cross-validation, feature importance analysis, and hyperparameter tuning, while explicitly testing robustness under label-flip poisoning and improving transparency via LIME [62].

Complementary OC-SVM-oriented work introduces online adaptation and enhanced feature generation to better handle traffic fluctuations and diverse attacks without requiring anomalous samples during training, often combining OC-SVM with statistical components and explanation tools to improve operational trust [63].

In parallel, recent advances in bounded one-class support vector classification target instability under contaminated normal data by enforcing a more stable decision boundary and incorporating density-aware weighting to reduce outlier influence[64]. Ensemble-style shallow OCL is also being revisited; random neighborhood aggregation diversifies local one-class learners and improves robustness compared to single-classifier OCC in several benchmark settings[65].

Collectively, these studies support the comparative finding that shallow OCL remains attractive for lightweight IDS due to simplicity and low inference overhead; however, literature evidence indicates that strong performance increasingly depends on robust boundary learning, systematic tuning, and explainability, especially under noisy or adversarial conditions.

REVIEW ARTICLE

5.7.2. RL-Based OCL

Recent studies reinforce that reconstruction-based OCL often provides stable performance across heterogeneous datasets while enabling deployment-oriented optimizations [27], [66], [67]. Some recent experimental studies evaluate OCL-based IDS and validates runtime behavior in a real-time prototype tested on a Raspberry Pi, showing that one-class IDS can achieve strong detection with low latency when designed as a lightweight modular pipeline [16], [68], [69]. In addition, an IoT smart-home framework couples a one-class asymmetric stacked autoencoder with an edge–fog–cloud architecture, treating dimensionality reduction and distributed inference as first-class design elements and reporting both detection effectiveness and processing-time measurements on BoT-IoT and IoT-23[70]. Multi-dataset benchmarking further highlights that although supervised DL typically achieves higher raw accuracy, certain OCL methods can be competitive; however, cross-dataset consistency remains weaker than for supervised models, and OC-SVM is frequently among the more time-intensive choices, whereas alternatives such as DROCC can be more efficient and stable in some regimes [71]. Overall, the recent literature strengthens the earlier comparative observation that AE-based OCL offers a favorable accuracy–efficiency trade-off under heterogeneous traffic, and it shifts the discussion from model-only comparisons toward system-level deployability, supported by evidence from edge deployments and distributed processing pipelines.

5.7.3. Deep OCL

Beyond reconstruction-only approaches, recent studies increasingly adopt deep one-class designs to improve generalization to unseen threats and address domain-specific constraints [72]. In SCADA contexts, attack-aware one-class frameworks combine protocol-normalizing transforms with deep representation learning and adversarial training to tighten anomaly boundaries and improve robustness against stealthy unknown intrusions [73]. For zero-day web security, one-class ensemble strategies aggregate multiple reconstruction models and report strong detection performance with short training/testing times, aiming to reduce false alarms while maintaining sensitivity to unknown attacks [74]. A recent direction is also federated one-class IDS, motivated by privacy and cross-domain data silos: FOCC-style approaches mitigate Non-IID degradation by synthesizing threat-specific samples using VAEs and aggregating latent representations, while exploiting parallel inference to reduce computational delays [75]. However, recent work also highlights a new attack surface in federated OCL, where one-class FL models can be vulnerable to backdoor behaviors, motivating defenses such as median-based client selection and machine unlearning mechanisms to remove attacker influence [76]. Overall, this evidence extends the comparative analysis beyond offline accuracy by showing that deep OCL is increasingly adopted for domain-specific and zero-day settings, while federated OCL is promising for privacy but requires explicit robustness evaluation against Non-IID effects and training-time attacks.

6. CHALLENGES AND FUTURE PROSPECTS

Despite the notable progress of OCL for IoT intrusion detection, translating research outcomes into practical deployments remains an open problem. Current approaches often demonstrate strong detection capability in experimental settings but face difficulties when confronted with the realities of heterogeneous IoT infrastructures, evolving traffic, and constrained computing environments [77]. Bridging this gap requires addressing a series of challenges that simultaneously define future research directions, as summarized in Figure 8.

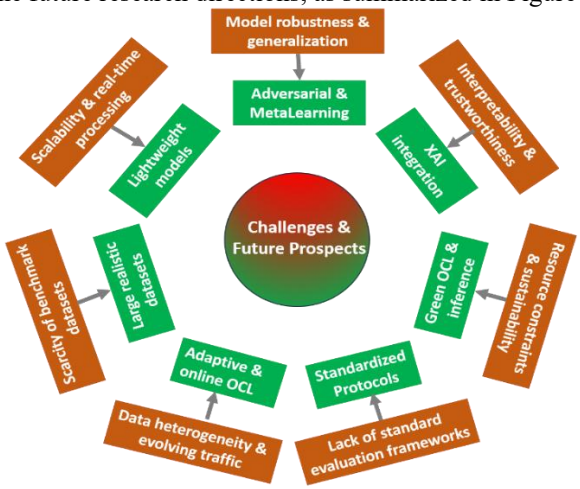

Figure 8 Illustration of Challenges and Future Prospects for OCL Based IoT IDS

- Data Heterogeneity and Evolving Traffic

IoT networks comprise devices with diverse capabilities, communication protocols, and traffic characteristics, where normal behavior evolves continuously due to firmware updates, user mobility, and changing operational contexts. Such non-stationarity often results in concept drift, undermining the reliability of static OCL models [44]. Addressing this issue calls for adaptive and online frameworks that incorporate incremental learning, transfer learning, or continual adaptation to ensure sustained robustness under evolving traffic conditions.

- Scarcity of Benchmark Datasets

The development of OCL models depends heavily on benchmark datasets, yet most available IoT intrusion datasets are outdated, synthetic, or insufficiently representative of real-world environments [16]. This limitation restricts both the generalizability and the reproducibility of experimental results across diverse IoT scenarios. Progress in this area will require

REVIEW ARTICLE

the creation of large-scale, realistic, and continuously updated datasets, potentially supported by federated and privacy-preserving data collection, to provide a stronger foundation for benchmarking and model validation.

- Scalability and Real-Time Processing

Massive IoT traffic streams combined with strict latency requirements pose significant scalability challenges for deep OCL models such as CNNs, RNNs, and GNNs. Although these architectures provide strong representational power, they often come with high training and inference costs that hinder real-time deployment [2]. Addressing this issue will require the design of lightweight model and hardware-aware optimizations to make OCL intrusion detection feasible on edge and fog devices.

- Model Robustness and Generalization

OCL models often overfit to specific training distributions, which restricts their ability to generalize to unseen devices, traffic types, or protocols [13]. This vulnerability is further exacerbated by adversarial manipulations that can exploit model weaknesses and cause false negatives. Enhancing resilience in such settings calls for the integration of adversarial robustness, domain generalization, and meta-learning approaches to ensure consistent performance under distributional shifts and malicious evasion attempts.

- Interpretability and Trustworthiness

Deep OCL approaches often operate as black boxes, providing limited insight into why a particular traffic instance is flagged as anomalous. This lack of interpretability restricts their adoption in critical domains such as healthcare and industrial IoT, where accountability and transparency are essential [22]. Incorporating explainable AI (XAI) into OCL frameworks offers a promising direction to enhance trust, allowing stakeholders to understand anomaly decisions, strengthen system auditing, and comply with regulatory requirements.

- Resource Constraints and Sustainability

IoT devices and edge nodes operate under strict limitations in energy, memory, and computational power, yet many state-of-the-art OCL methods demand substantial resources, raising concerns about deployment feasibility and energy footprint [43], [58]. Advancing this area will require the development of green OCL methods that integrate energy-efficient training, edge-friendly inference, and sustainable hardware design, ensuring large-scale adoption while aligning with global sustainability objectives.

- Lack of Standard Evaluation Frameworks

Current studies often rely on inconsistent datasets, metrics, and validation protocols, making it difficult to perform reliable cross-comparisons [48]. Variability in threshold selection for anomaly scores and differences in performance reporting further complicate reproducibility. To overcome these limitations, future efforts should focus on establishing standardized benchmarks, open-source toolkits, and cross-dataset evaluation protocols that promote transparency and comparability across the field.

In summary, these challenges highlight that advancing OCL for IoT intrusion detection requires a shift from accuracy-oriented model development toward deployment-aware, adaptive, and trustworthy solutions. Addressing data heterogeneity, scalability, interpretability, and sustainability will be essential for transitioning OCL models from controlled laboratory settings to large-scale, operational IoT infrastructures. Furthermore, progress will depend on the establishment of unified evaluation standards and the availability of realistic datasets that reflect the complexity and dynamism of modern IoT ecosystems. By confronting these limitations through collaborative research across machine learning, networking, and systems engineering, the community can pave the way for OCL-based IDS that are not only accurate but also robust, efficient, explainable, and ready for real-world deployment.

7. CONCLUSION

This review has presented a structured and deployment-aware analysis of OCL approaches for IoT intrusion detection, spanning SL, RL, and deep OCL models. Central to this work is a multilayer evaluation framework that jointly considers architectural design, detection performance, dataset sensitivity, and operational factors. This framework shows that OCL effectiveness is not determined by algorithmic choice alone, but by the interaction between model structure, data characteristics, and deployment constraints. Across benchmark IDS datasets, RL models, particularly AE variants, exhibit the most consistent behavior, whereas SL methods are more vulnerable to imbalance, feature heterogeneity, and mixed IoT modalities. These findings underscore the influences of data characteristics on the model generalizability and the need for evaluation practices that account for operational factors such as latency, memory, and resource usage along device–edge–fog–cloud tiers.

Future progress will depend on developing adaptive, and energy-efficient OCL models, supported by deployment-centric benchmarks. Integrating methodological, empirical, and multilayer operational perspectives, this review provides a foundation for designing practical and scalable OCL-based IDS for next-generation IoT environments.

ACKNOWLEDGEMENT

The authors extend their appreciation to Prince Sattam bin Abdulaziz University for funding this research work through the project number (PSAU/2025/01/33512).

REVIEW ARTICLE

REFERENCES

[1] F. Zayas-Gato et al., "A hybrid one-class approach for detecting anomalies in industrial systems," Expert Syst., vol. 39, no. 9, p. e12990, 2022.

[2] A. G. Ayad, M. M. El-Gayar, N. A. Hikal, and N. A. Sakr, "Efficient Real-Time anomaly detection in IoT networks using One-Class autoencoder and deep neural network," Electronics, vol. 14, no. 1, p. 104, 2024.

[3] A. K. Silivery, K. R. M. Rao, and R. Solleti, "Dual-path feature extraction-based hybrid intrusion detection in IoT networks," Computers and Electrical Engineering, vol. 122, no.1, p. 109949, 2025.

[4] V. Harit, R. Dahiya, and U. Garg, "Machine Learning Approaches for IoT Botnet Detection and Mitigation: A Comprehensive Review," International Journal of Computer Networks and Applications, vol. 12, no. 6, pp. 936–952, 2025.

[5] S. Gupta and B. Singh, "Lightweight ensemble learning based intrusion detection framework with explainable artificial intelligence," Eng. Appl. Artif. Intell., vol. 163, no. 1, p. 112936, 2026.

[6] S. Jamshidi et al., "Carbon-Aware Intrusion Detection: A Comparative Study of Supervised and Unsupervised DRL for Sustainable IoT Edge Gateways," arXiv preprint arXiv:2511.18240, 2025.

[7] Y. Xin et al., "LUFT-CAN: A lightweight unsupervised learning based intrusion detection system with frequency-time analysis for vehicular CAN bus," Journal of Systems Architecture, vol.198, no.1, p. 103567, 2025.

[8] T. Vaiyapuri, W. H. Elashmawi, W. Asiedu, and others, "VAE-CNN: Deep Learning on Small Sample Dataset Improves Hydrogen Yield Prediction in Co-gasification," Journal of Computational and Cognitive Engineering, vol. 4, no. 3, pp. 332–342, 2025.

[9] K. Prathapchandran, "BioTrust-IoT: A Biologically Inspired Adaptive Trust System for Detecting Man-in-the-Middle Attacks," International Journal of Computer Networks and Applications, vol. 12, no. 6, pp. 912–935, 2025.

[10] T. Kim and W. Pak, "Early detection of network intrusions using a GAN-based one-class classifier," IEEE Access, vol. 10, pp. 119357–119367, 2022.

[11] A. Binbusayyis, T. Vaiyapuri, "Unsupervised deep learning approach for network intrusion detection combining convolutional autoencoder and one-class SVM," Applied Intelligence, vol. 51, no. 10, 2021.

[12] J. Suganya, M. Prakash, S. Akilandeswari, and P. Anitha, "Enhancing Security in Underwater Wireless Sensor Networks using a Hybrid Approach of Beetle Antenna Search and Genetic Algorithms," International Journal of Computer Networks and Applications, vol. 12, no. 6, pp. 953–972, 2025, DOI: 10.22247/ijcna/2025/56.

[13] T. Vaiyapuri and A. Binbusayyis, "Application of deep autoencoder as an one-class classifier for unsupervised network intrusion detection: a comparative evaluation," PeerJ Comput. Sci., vol. 6, pp. 1–26, 2020.

[14] C. Qiu et al., "Mining Multi-Scale Spatial-Frequency Clues for Unsupervised Intrusion Detection," IEEE Transactions on Information Forensics and Security, vol. 20, pp. 11447 – 11461, 2025.

[15] Q. Jiang, K. Wang, Y. Wei, H. Liu, and B. Wang, "XIPHOS: Adaptive In-Vehicle Intrusion Detection via Unsupervised Graph Contrastive Learning," IEEE Transactions on Information Forensics and Security, vol. 20, pp. 10419–10433, 2025.

[16] D. Paolini, P. Dini, E. Soldaini, and S. Saponara, "One-Class Anomaly Detection for Industrial Applications: A Comparative Survey and Experimental Study," Computers, vol. 14, no. 7, p. 281, 2025.

[17] R. Gangula, M. M. Vutukuru, and M. R. Kumar, "Stacked autoencoder with weighted loss function for intrusion detection in IoT application," Multimed. Tools Appl., vol. 84, no. 21, pp. 23501–23529, 2025.

[18] T. K. Boppana and P. Bagade, "GAN-AE: An unsupervised intrusion detection system for MQTT networks," Engineering Applications of Artificial Intelligence, vol. 119, p. 105805, 2023.

[19] B. S. Ali et al., "ICS-IDS: application of big data analysis in AI-based intrusion detection systems to identify cyberattacks in ICS networks," J. Supercomput., vol. 80, no. 6, pp. 7876–7905, 2024.

[20] J. Zhang, Y. Li, and L. Zhang, "Heterogeneous network intrusion detection via domain adaptation in IoT environment," Internet Technology Letters, vol. 8, no. 1, p. e531, 2025.

[21] S. Kaushik et al., "Robust machine learning based Intrusion detection system using simple statistical techniques in feature selection," Sci. Rep., vol. 15, no. 1, p. 3970, 2025.

[22] S. K. R. Mallidi and R. R. Ramisetty, "Advancements in training and deployment strategies for AI-based intrusion detection systems in iot: A systematic literature review," Discover Internet of Things, vol. 5, no. 1, p. 8, 2025.

[23] A. C. Turcato, G. Serpa Sestito, A. L. Dias, and R. Andrade Flauzino, "Unsupervised Machine Learning-Based Intrusion Detection in PROFINET Networks," Journal of Control, Automation and Electrical Systems, vol. 36, pp. 766–777, 2025.

[24] W. Xu, J. Jang-Jaccard, T. Liu, F. Sabrina, and J. Kwak, "Improved bidirectional gan-based approach for network intrusion detection using one-class classifier," Computers, vol. 11, no. 6, p. 85, 2022.

[25] R. Kumar and M. Swarnkar, "QuIDS: A quantum support vector machine-based intrusion detection system for IoT networks," Journal of Network and Computer Applications, vol. 234, p. 104072, 2025.

[26] S. M. Hashemi Jouybari, A. Janbaz, and A. Esfandiari, "A multivariate filter feature selection and stacking-based ensemble learning algorithm for network intrusion detection," J. Supercomput., vol. 81, no. 10, p. 1129, 2025.

[27] P. Bountzis, D. Kavallieros, T. Tsikrika, S. Vrochidis, and I. Kompatsiaris, "A Deep One-Class Classifier for Network Anomaly Detection Using AutoEncoders and One-Class Support Vector Machines," Front. Comput. Sci., vol. 7, p. 1646679, 2025.

[28] X. Wang, L. Qi, X. Wei, W. Zhu, H. Jiang, and Z. Guan, "AED: A novel approach for intrusion detection without abnormal samples in big data environment," ACM J. Data Inf. Qual., vol. 17, no. 3, pp. 1–20, 2025.

[29] A. A. Zadeh, Advancing One-Class Classification: A Comprehensive Analysis From Theory to Novel Applications. Ph.D. dissertation, Florida Atlantic University, 2024.

[30] N. Seliya, A. Abdollah Zadeh, and T. M. Khoshgoftaar, "A literature review on one-class classification and its potential applications in big data," J. Big Data, vol. 8, no. 1, p. 122, 2021.

[31] T. Hayashi, D. Cimr, H. Fujita, and R. Cimler, "Critical Review for One-class Classification: recent advances and the reality behind them," arXiv preprint arXiv:2404.17931, 2024. https://doi.org/10.48550/arXiv.2404.17931.

[32] H. O. Marques, L. Swersky, J. Sander, R. J. G. B. Campello, and A. Zimek, "On the evaluation of outlier detection and one-class classification: a comparative study of algorithms, model selection, and ensembles," Data Min. Knowl. Discov., vol. 37, no. 4, pp. 1473–1517, 2023.

[33] S. S. Khan and M. G. Madden, "One-class classification: taxonomy of study and review of techniques," Knowl. Eng. Rev., vol. 29, no. 3, pp. 345–374, 2014.

[34] A. Kumar and T. K. Das, "ORADS: One Class Rule-Based Anomaly Detection System in Autonomous Vehicles," IEEE Sens. J., vol. 25, no. 5, pp. 8988 – 8997, 2025.

[35] C. Wang, Y. Sun, S. Lv, C. Wang, H. Liu, and B. Wang, "Intrusion detection system based on one-class support vector machine and gaussian mixture model," Electronics, vol. 12, no. 4, p. 930, 2023.

[36] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, and A. Alazab, "Hybrid intrusion detection system based on the stacking ensemble of c5 decision tree classifier and one class support vector machine," Electronics., vol. 9, no. 1, p. 173, 2020.

[37] R. Krebs, S. S. Bagui, D. Mink, and S. C. Bagui, "Applying Multi-CLASS Support Vector Machines: One-vs.-One vs. One-vs.-All on the UWF-ZeekDataFall22 Dataset," Electronics, vol. 13, no. 19, p. 3916, 2024.

[38] B. Hu et al., "A deep one-class intrusion detection scheme in software-defined industrial networks," IEEE Trans. Industr. Inform., vol. 18, no. 6, pp. 4286–4296, 2021.

REVIEW ARTICLE

[39] M. K. U. Ahamed and A. Karim, "Cascaded intrusion detection system using machine learning," Systems and Soft Computing, vol. 7, p. 200182, 2025.
[40] G. Giacinto, R. Perdisci, M. Del Rio, and F. Roli, "Intrusion detection in computer networks by a modular ensemble of one-class classifiers," Information Fusion, vol. 9, no. 1, pp. 69–82, 2008.
[41] A. Binbusayyis and T. Vaiyapuri, "Identifying and benchmarking key features for cyber intrusion detection: an ensemble approach," IEEE Access, vol. 7, pp-106495 - 106513, 2019.
[42] B. Lewandowski and R. Paffenroth, "Autoencoder Feature Residuals for Network Intrusion Detection: One-Class Pretraining for Improved Performance," Mach. Learn. Knowl. Extr., vol. 5, no. 3, pp. 868–890, 2023.
[43] W. Yao, L. Hu, Y. Hou, and X. Li, "A lightweight intelligent network intrusion detection system using one-class autoencoder and ensemble learning for IoT," Sensors, vol. 23, no. 8, p. 4141, 2023.
[44] F. Kang, T. Feng, and J. Lin, "VAE-GAN-Guided Cross-Class Generation: A Class Imbalance Data Augmentation Method for Network Intrusion Detection," Electronics (Basel)., vol. 14, no. 11, p. 2103, 2025.
[45] A. Halbouni, T. S. Gunawan, M. H. Habaebi, M. Halbouni, M. Kartiwi, and R. Ahmad, "CNN-LSTM: hybrid deep neural network for network intrusion detection system," IEEE Access, vol. 10, pp. 99837–99849, 2022.
[46] T. Vaiyapuri and A. Binbusayyis, "Deep self-taught learning framework for intrusion detection in cloud computing environment," IAES International Journal of Artificial Intelligence, vol. 13, no. 1, pp. 747–755, 2024.
[47] E. U. H. Qazi, A. Almorjan, and T. Zia, "A one-dimensional convolutional neural network based deep learning system for network intrusion detection," Applied Sciences, vol. 12, no. 16, p. 7986, 2022.
[48] A. Binbusayyis and T. Vaiyapuri, "Comprehensive analysis and recommendation of feature evaluation measures for intrusion detection," Elsevier, vol. 6, no. 7, p. e04262, 2020.
[49] Y. Li et al., "One-class LSTM network for anomalous network traffic detection," Applied Sciences, vol. 12, no. 10, p. 5051, 2022.
[50] E. Caville, W. W. Lo, S. Layeghy, and M. Portmann, "Anomal-E: A self-supervised network intrusion detection system based on graph neural networks," Knowl. Based. Syst., vol. 258, p. 110030, 2022.
[51] T. Bilot, N. El Madhoun, K. Al Agha, and A. Zouaoui, "Graph neural networks for intrusion detection: A survey," IEEE Access, vol. 11, pp. 49114–49139, 2023.
[52] J. Ashraf et al., "IoTBoT-IDS: A novel statistical learning-enabled botnet detection framework for protecting networks of smart cities," Sustain. Cities Soc., vol. 72, p. 103041, 2021.
[53] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of KDD CUP 99 data set," IEEE symposium on computational intelligence for security and defense applications, pp. 1–6, 2009.
[54] N. Moustafa and J. Slay, "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), 2015, pp. 1–6.
[55] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets". Sustain. Cities Soc. Vol. 72, no.2, pp. 102994, 2021.
[56] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, and others, "Toward generating a new intrusion detection dataset and intrusion traffic characterization.," ICISSp, vol. 1, no. 2018, pp. 108–116, 2018.
[57] A. Alotaibi and M. A. Rassam, "Enhancing the sustainability of deep-learning-based network intrusion detection classifiers against adversarial attacks," Sustainability, vol. 15, no. 12, p. 9801, 2023.
[58] H. Alazzam, A. Sharieh, and K. E. Sabri, "A lightweight intelligent network intrusion detection system using OCSVM and Pigeon inspired optimizer," Applied Intelligence, vol. 52, no. 4, pp. 3527–3544, 2022.
[59] W. Zhang, Y. Miao, Q. Wu, L. Yu, and X. Shi, "Intrusion detection of industrial control system based on double-layer one-class support vector machine," IFAC-PapersOnLine, vol. 53, no. 2, pp. 2513–2518, 2020.
[60] K. Yang, Y. Shi, Z. Yu, Q. Yang, A. K. Sangaiah, and H. Zeng, "Stacked one-class broad learning system for intrusion detection in industry 4.0," IEEE Trans. Industr. Inform., vol. 19, no. 1, pp. 251–260, 2022.
[61] P. F. de Araujo-Filho, M. Naili, G. Kaddoum, E. T. Fapi, and Z. Zhu, "Unsupervised gan-based intrusion detection system using temporal convolutional networks and self-attention," IEEE Transactions on Network and Service Management, vol. 20, pp. 4951–4963, 2023.
[62] A. Zahoor, W. Abbasi, M. Z. Babar, and A. Aljohani, "Robust iot security using isolation forest and one class svm algorithms," Sci. Rep., vol. 15, no. 1, p. 36586, 2025.
[63] K. Kerimov, S. Kurbanov, and Z. Azizova, "Methods for detecting anomalies in network traffic based on one-class SVM technology," International Scientific Technical Journal" Problems of Control and Informatics", vol. 70, no. 2, pp. 91–98, 2025.
[64] J. Ye, Z. Yang, Y. Hu, and Z. Zhang, "C-parameter version of robust bounded one-class support vector classification," Sci. Rep., vol. 15, no. 1, p. 975, 2025.
[65] S. Park and J. Yu, "Random neighborhood ensemble-based one-class classification algorithm for anomaly detection," Applied Intelligence, vol. 56, no. 1, p. 27, 2026.
[66] A. Liuliakov, A. Schulz, L. Hermes, and B. Hammer, "Noise Robust One-Class Intrusion Detection on Dynamic Graphs," arXiv preprint arXiv:2508.14192, 2025. https://doi.org/10.48550/arXiv.2508.14192
[67] B. Min and D. Park, "Network Intrusion Detection Using One-Class Models," Convergence Security Journal, vol. 24, no. 3, pp. 13–21, 2024.
[68] T. Shi, R. A. McCann, Y. Huang, W. Wang, and J. Kong, "Malware detection for internet of things using one-class classification," Sensors, vol. 24, no. 13, p. 4122, 2024.
[69] H. H. Al-Khshali, M. Ilyas, F. Sohrab, and M. Gabbouj, "Malware detection with subspace learning-based one-class classification," IEEE Access, vol. 12, pp. 81017–81029, 2024.
[70] A. G. Ayad, N. A. Sakr, and N. A. Hikal, "Fog-empowered anomaly detection in IoT networks using one-class asymmetric stacked autoencoder," Cluster Comput., vol. 28, no. 8, p. 550, 2025.
[71] S. Golestani and D. Makaroff, "Exploring Unsupervised One-Class Classifiers for Lightweight Intrusion Detection in IoT Systems," in 2024 20th International Conference on Distributed Computing in Smart Systems and the Internet of Things, 2024, pp. 234–238.
[72] K. Suresh, K. J. Velmurugan, R. Vidhya, and others, "Deep anomaly detection: A linear one-class SVM approach for high-dimensional and large-scale data," Appl. Soft Comput., vol. 167, p. 112369, 2024.
[73] W. Li, Y. Yao, C. Sheng, N. Zhang, and W. Yang, "ALOC: Attack-Aware by Utilizing the Adversarially Learned One-Class Classifier for SCADA System," IEEE Internet Things J., vol. 11, no. 13, pp. 23444–23459, 2024.
[74] V. Babaey and H. R. Faragardi, "Detecting Zero-Day Web Attacks Using One-Class Ensemble Classifiers," 2025. https://doi.org/10.20944/preprints202503.0257.v1.
[75] S. H. A. Kazmi, F. Qamar, R. Hassan, and K. Nisar, "FOCC: A Synthetically Balanced Federated One-Class-Classification for Cyber Threat Intelligence in Software Defined Networking," IEEE Open Journal of the Computer Society, vol. 6, no. 3, pp. 701 – 713, 2025.
[76] A. H. Bondok, M. Badr, M. Mahmoud, M. Alsabaan, M. M. Fouda, and M. Abdallah, "Securing one-class federated learning classifiers against trojan attacks in smart grid," IEEE Internet Things J., vol. 12, no. 4, pp. 4006–4021, 2024.
[77] S. Naciri, O. Gamache, J. Ouellet-Boudreault, A. Abusitta, M. Bellaiche, and T. Halabi, "Learning-enabled Intrusion Detection in IoT: Current Challenges and Future Directions," in 2025 IEEE Conference on Smart Internet of Things, 2025, pp. 404–411.

REVIEW ARTICLE

Authors

Dr. Thavavel Vaiyapuri is a faculty member College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University, Saudi Arabia. She received her Ph.D. degree in Computer Science and has over a decade of academic and research experience. Her research interests include Artificial Intelligence of Things (AIoT), sustainable and green computing, low-carbon AI frameworks with applications in healthcare. She is a member of the IEEE Computer Society and a Fellow of the Higher Education Academy (HEA), U.K. Dr. Vaiyapuri is dedicated to advancing interdisciplinary AI research that supports the UN SDGs.

Mrs. Karthiyayini Murugesan received her M.Tech. degree in Computer Science and Engineering from Sethu Institute of Technology, India, in 2019. She is currently preparing for doctoral research in AI with a focus on sustainable and intelligent systems. Her research interests include sustainable edge intelligence for healthcare. With a strong interest in interdisciplinary applications of AI, she aspires to develop socially impactful solutions aligned with UN SDGs.

How to cite this article:

Thavavel Vaiyapuri, Karthiyayini Murugesan, "Unsupervised One-Class Learning for IoT Intrusion Detection: A Review of Methods and Prospects", International Journal of Computer Networks and Applications (IJCNA), 13(1), PP: 179-199, 2026, DOI: 10.22247/ijcna/2026/10.