Leakage-Aware Evaluation of Intrusion Detection Systems for Rare and Unseen Attacks under Temporal Distribution Shift

Ameen Shaheen *, Wael Alzyadat, and Aysh Alhroob
Department of Software Engineering, Faculty of Science and Information Technology,
Al-Zaytoonah University of Jordan, Jordan
Email: a.shaheen@zuj.edu.jo (A.S.); wael.alzyadat@zuj.edu.jo (W.A.); aysh@zuj.edu.jo (A.A.)

Manuscript received April 25, 2026; revised June 21, 2026; accepted July 21, 2026; published September 22, 2026
*Corresponding author

***Abstract*—Intrusion Detection Systems (IDS) must operate under dynamic network conditions where attack distributions change over time and previously unseen threats may emerge after deployment. However, many machine-learning-based IDS studies still rely on random train-test splitting, which can lead to overly optimistic performance estimates. To address this limitation, this paper presents a leakage-aware evaluation framework for intrusion detection using the Communications Security Establishment–Canadian Institute for Cybersecurity Intrusion Detection System 2018 (CSE-CIC-IDS2018) dataset under a time-based train-test protocol. The proposed framework includes dataset curation, temporal partitioning, leakage-free feature cleaning, baseline and class-weighted Extreme Gradient Boosting (XGBoost) classification, confidence-threshold sensitivity analysis, and duplicate-filtered robustness checking. The results show that the dataset represents a severely imbalanced multi-class detection problem. Under the time-based split, the baseline model achieves high overall accuracy but fails on unseen attack classes and remains unstable on several rare-known attacks. Class-weighted training improves detection for rare-known classes represented during training, but does not improve performance on unseen attacks. In addition, confidence-based rejection fails to identify unseen attacks because the classifier remains highly confident even when making incorrect predictions on previously unseen traffic. Threshold-sensitivity analysis further shows that stricter rejection thresholds do not provide a useful trade-off, while duplicate-filtered evaluation confirms that the unseen-attack failure is not explained by exact train-test feature overlap. Overall, the findings show that reliable intrusion detection requires more than strong benchmark accuracy. It requires leakage-aware evaluation, class-sensitive analysis, robustness checks against evaluation artifacts, and explicit mechanisms for handling unseen attacks under realistic temporal shift.**

***Index Terms*—Intrusion Detection Systems (IDS), leakage-aware evaluation, Extreme Gradient Boosting (XGBoost), confidence-based rejection, explainable artificial intelligence**

I. Introduction

Intrusion Detection Systems (IDS) serve as one of the principal security measures in today's complex networks. With the increase in network complexity and interconnectedness, the potential threat level also increases because of expanded attack surfaces [1]. The diversity and sophistication of malicious activity are outpacing traditional signature-based intrusion detection capabilities, thereby creating the need to transition toward Machine Learning (ML) and Deep Learning (DL)-based IDS solutions [2]. Recent reviews also indicate that ML/DL-based IDS has emerged as a leading area of study because these approaches can learn patterns from traffic features and enable more adaptive detection behavior over time compared with traditional static rule-based systems [3].

Public benchmark datasets have been important to the development of IDS research because they make it possible to evaluate models under common experimental settings and compare results across different studies. Among the most widely used examples is Canadian Institute for Cybersecurity Intrusion Detection System 2017 (CICIDS2017)/Communications Security Establishment–Canadian Institute for Cybersecurity (CSE-CIC) [4], which was introduced to address several limitations of earlier benchmarks by providing more realistic traffic, contemporary attack scenarios, and detailed flow-level features for machine-learning-based intrusion detection research [5]. However, good results on benchmark datasets do not necessarily indicate that a model will perform reliably in real deployment. As in many machine-learning applications, the reported performance can be influenced by how the data are preprocessed, split, and evaluated, which may lead to overly optimistic conclusions about generalization [6].

Information leakage is a major threat to the validity of research in this area. In machine learning, leakage occurs when information that would not actually be available at prediction time affects the training or evaluation process, which can make the results look better than they really are [7]. In IDS research, this problem can arise when correlated traffic patterns, temporally related flows, or very similar samples are unintentionally split between the training and test sets. When that happens, the reported performance may appear strong, even though the model

may not behave as reliably in a real deployment setting. More broadly, leakage has long been recognized as a common cause of irreproducibility and misleading performance claims in predictive modeling [8]. Therefore, IDS evaluation should not only report predictive accuracy, but should also control how temporal information, duplicate-like samples, preprocessing steps, and feature construction may influence the reported results.

Beyond leakage, distribution shift [9] is also an important but generally overlooked problem for model generalizability. The majority of models rely on the assumption that the training data and the test data have been drawn from the same probability distribution. In fact, there will typically be significant variation in the ways in which data are generated by each environment as time passes, especially when those environments include network-based systems used for various business operations [10]. For example, patterns of traffic usage, the campaigns that attackers engage in, and the operating conditions of the network can all vary across different times or locations. Therefore, using random splits of a single dataset to estimate the accuracy of a model's performance in a future environment is likely to provide overly optimistic estimates of its generalizability. Furthermore, this method does not reflect how a model would perform during actual real-world deployments [11].

Another major issue in IDS is class imbalance. Real-world intrusion datasets contain much more benign traffic than malicious traffic, and as a result, some attack types appear only rarely in these datasets. Previous studies have shown that conventional learning algorithms tend to favor the majority classes because they are much more prevalent in the data, and that aggregate accuracy can still indicate good overall performance even when minority classes are detected poorly [12]. This is particularly problematic in IDS, since the classes that are most difficult to detect may also be among the most important from an operational perspective. Therefore, evaluations based mainly on overall accuracy may obscure poor recall for minority attacks and may underestimate the real-world security costs associated with missed detections [13]. At the same time, rare attacks and unseen attacks should not be treated as the same problem: rare attacks may still be represented during training, whereas unseen attacks are completely absent from the training label space.

A significant limitation in many IDS studies [14, 15] has been the use of closed-set settings. In a closed-set setting, it is assumed that all attack classes that may appear in the test data are already included in the training data. However, in real-world deployment, an IDS may operate in environments where previously unseen attack types may emerge. This challenge is closely related to Open-Set Recognition (OSR), where a classifier is required not only to distinguish between known classes, but also to identify inputs that do not belong to any of the classes observed during training [16]. Under such conditions, conventional classifiers may still assign unseen samples to one of the known classes with high confidence, resulting in predictions that may appear plausible but are actually incorrect [17].

Confidence scoring becomes an even greater concern when applied in practice. Many ML systems rely on thresholds to reject samples or flag them as unknown based on their confidence levels. However, recent work in out-of-distribution detection has shown that standard-trained ML models can assign high-confidence predictions to samples that fall outside the model's original training distribution [18], which undermines the effectiveness of simple confidence thresholds for rejecting potentially malicious samples. In the context of IDS, this means that classifiers may not only fail to identify new attack types correctly, but may also appear highly confident in those incorrect predictions. This creates a serious trustworthiness challenge for automated security workflows. Explainability is also becoming increasingly important in IDS, because cybersecurity analysts need more than the final label in order to assess alarms and respond appropriately. Recent surveys suggest that Explainable Artificial Intelligence (XAI) is likely to play a larger role in IDS; however, the technical challenges involved in developing and deploying trustworthy XAI-enabled security systems remain largely unresolved [19].

Motivated by these shortcomings, this paper presents a leakage-aware evaluation framework for ML-based IDS and addresses key limitations in prior work. The proposed framework is built around two main concepts: leakage awareness and realistic evaluation. Leakage awareness relates to the fact that many previous studies have relied on random splits between training and test datasets; however, such splits do not accurately reflect how an IDS would perform in practice. In addition, these evaluation settings often produce overly optimistic results and do not adequately account for the possibility of attacks appearing during testing that were not present during training. Therefore, the proposed framework adopts a time-based split between the training and test datasets. Under this setting, the IDS is trained on historical traffic and then evaluated on later traffic that may include previously unseen attacks. The evaluation is further strengthened by removing temporal and identifier-like attributes from the final feature matrix, applying training-set-based preprocessing, analyzing confidence-threshold sensitivity, and conducting an exact train-test feature-overlap robustness check. In this way, the proposed evaluation framework provides a more realistic representation of how an IDS may behave in an operational environment.

The main contributions of this work are as follows:

- A leakage-aware time-based evaluation protocol for intrusion detection that avoids unrealistic random-split optimism and provides a more deployment-relevant estimate of model generalization.
- A leakage-free feature construction and robustness analysis that removes temporal and identifier-like attributes from model training and examines the effect of exact train-test feature overlap.
- A rare-class-centered performance analysis showing why aggregate accuracy is insufficient under severe class imbalance in IDS.
- An empirical demonstration that conventional closed-set IDS models can fail completely on unseen attack categories under realistic temporal separation.

- A controlled assessment of imbalance-aware training through class weighting, showing that it improves several rare known attacks but remains insufficient for unseen attacks.
- A threshold-sensitivity analysis of confidence-based rejection, showing that simple confidence thresholds do not provide reliable unknown-attack detection under temporal distribution shift.

II. Related Works

Recent research in IDS has been towards ML and DL frameworks that model complex traffic behavior and multiple attack pattern classifications from the signature-driven approaches of earlier years. This shift has also shown a growing focus on operational concerns for transparency, robustness, and interpretability, rather than just raw detection accuracy alone [20]. At the same time, XAI has emerged as an important direction for IDS design because cybersecurity analysts frequently require more than a final label; they need interpretable evidence to help validate alarms, understand model behavior, and support response decisions [21].

The IDS literature has consistently identified one of the most significant challenges as the extreme class imbalance found in real-world network traffic. In this type of environment, most flows are either benign or belong to a small number of dominant attack types, which greatly outnumber minority and rare attacks. More recent large-scale evaluations have shown that imbalance-aware learning strategies can improve minority-class detection; however, this improvement depends heavily on both the data distribution and the specific mitigation strategy used [22]. This issue becomes even more critical in the case of extremely underrepresented attack types, where an IDS can achieve very strong overall metrics while having almost no ability to detect the rarest classes, thereby making overall accuracy or weighted average measures insufficient for assessing the true effectiveness of an IDS [23]. More recent ensemble-based approaches developed to address imbalance in intrusion detection have shown promise for improving minority-class detection; however, like earlier methods, they still operate largely under traditional closed-set assumptions and therefore do not fully address the problem of generalization beyond previously known attack categories [24].

Recent hybrid and deep-learning-based IDS studies have also attempted to improve detection accuracy by combining spatial and temporal feature learning. For example, the unified deep-learning approach in Ref. [25] integrates spatial and temporal features for efficient IDS modeling, while the hybrid MapReduce-based optimized Conv-LSTM framework in Ref. [26] uses large-scale processing and hybrid optimization to improve intrusion detection performance. These studies demonstrate the value of advanced DL architectures for improving benchmark-level detection. However, their primary emphasis remains on improving predictive performance within the available benchmark setting, whereas the present study focuses on how IDS performance changes under leakage-aware temporal separation, rare-known attacks, and completely unseen test-only attack categories.

A second line of IDS research has focused on making IDS models more understandable and trustworthy. Recent research on explainable IDS frameworks has shown that post-hoc interpretation methods can help reveal which traffic attributes contribute most to a detector's decisions, thereby improving analyst trust and helping determine whether the model relies on security-relevant features or on spurious correlations [27]. Although this is helpful for increasing trust in an IDS model, it does not address the problem of evaluation bias or deployment mismatch. In particular, explainability can help interpret model decisions after training, but it cannot by itself correct leakage, temporal drift, class imbalance, or unseen-attack failure.

These limitations point to a broader gap in the current literature. There is considerable research focusing on each of these areas individually, including explainable IDS, imbalance-aware detection, deep-learning-based IDS architectures, and benchmark-level performance optimization. However, there is limited research that focuses specifically on how IDS models perform under deployment-oriented evaluation settings, where the test set is temporally separated from the training set and may include attack categories that are not included in the training data. This issue is especially important because real-world network environments are non-stationary, and attack distributions change over time. While recent evaluation-focused studies have started to draw attention to the gap between benchmark performance and realistic security behavior, a systematic treatment that combines leakage-aware temporal evaluation, rare-class analysis, unseen-attack failure analysis, confidence-threshold sensitivity, and duplicate-overlap robustness remains limited [28]. To address this gap, the present study evaluates IDS performance under a time-based protocol that reduces temporal and preprocessing leakage, focuses on rare attack detection, and directly analyzes the failure of conventional models on unseen attacks, including the inadequacy of simple confidence-based rejection under overconfident predictions.

III. Proposed Leakage-Aware Evaluation Framework

This study adopts a leakage-aware, deployment-oriented framework to evaluate an IDS in terms of its actual performance, as opposed to idealized random-split datasets. The proposed method contains four distinct phases: first, dataset curation and verification; second, leakage-controlled temporal partitioning; third, feature cleansing followed by supervised learning; and fourth, post-hoc evaluation through imbalance-aware training, confidence-based rejection, and explanation. This framework provides two primary goals: quantification of IDS predictive accuracy and examination of whether a well-performing IDS will remain accurate during temporally shifted traffic that may contain rare or previously unknown types of attack. Fig. 1 describes the overall workflow of the proposed framework, from dataset curation and time-based partitioning to model training,

confidence-based rejection, and final evaluation and analysis.

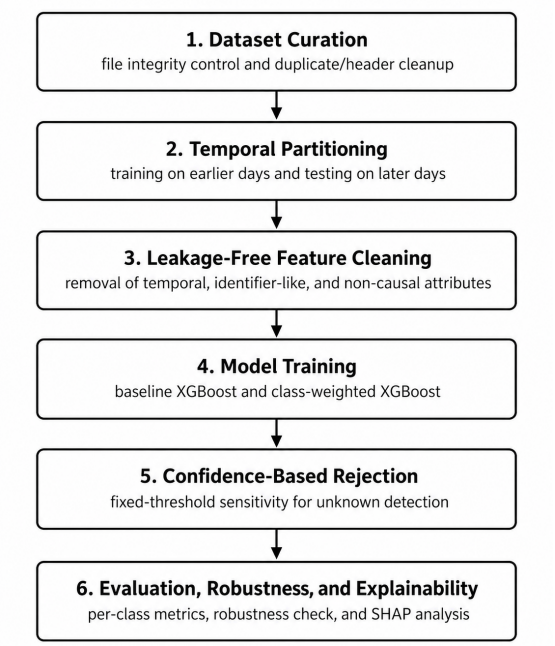

Fig. 1. The proposed leakage-aware evaluation framework.

A. Dataset and Data Integrity Control

The experiments were conducted on the CSE-CIC-IDS2018 benchmark, which was introduced to provide realistic traffic traces and contemporary attack scenarios for machine-learning-based intrusion detection research [29]. Before any model development, the dataset was audited at the file level to ensure structural consistency across daily Comma-Separated Values (CSV) files. Files with incompatible schemas were excluded to prevent mixing heterogeneous feature spaces. The retained files were then merged into a unified dataset; duplicate rows were removed, and anomalous rows caused by header leakage inside the merged table were discarded. After this integrity-control step, nine structurally consistent daily files were retained for the final experiments, while the incompatible daily file was excluded. The retained files correspond to the acquisition days used later for the leakage-aware temporal partition.

B. Leakage-Aware Temporal Partitioning

A central design choice in this work is the use of a time-based split instead of a random train-test split. In network intrusion datasets, temporally adjacent flows may share attack context, traffic behavior, or environmental characteristics. Random partitioning may therefore place highly related samples in both training and testing subsets, artificially inflating performance.

To prevent this issue, each sample was linked to its acquisition day, and the training and testing sets were created from disjoint temporal partitions. The adopted temporal split used five acquisition days for training and four later acquisition days for testing, with no overlap between the two partitions, as summarized in Table I. Let d_i denote the acquisition day associated with sample i. A sample i is assigned to the training set if $d_i \in T_{\text{trian}}$ and to the testing set if $d_i \in T_{test}$. The two acquisition-day sets are required to be non-overlapping, as expressed by:

$$T_{\text{trian}} \cap T_{\text{test}} = \emptyset \quad (1)$$

where T_{trian} is the set of days assigned to training, T_{test} is the set of days assigned to testing, and $\emptyset$ denotes the empty set, meaning that no acquisition day appears in both partitions.

Under this protocol, samples collected on days belonging to T_{trian} were used only for training, whereas samples collected on days belonging to T_{test} were reserved only for testing. This design reduces temporal leakage and simultaneously creates a more realistic scenario in which some attack categories may appear only in the test partition, thereby exposing the detector to previously unseen attacks. As shown in Table I, Bot and Infiltration appear only in the test partition and are therefore treated as unseen attack classes, while Brute Force-Web, Brute Force-XSS, and SQL Injection are treated as rare-known attacks because they appear in both partitions with very low support.

The per-class support further clarifies the distinction between unseen and rare-known attacks under the adopted temporal split. Bot and Infiltration have no training samples and appear only in the testing partition, with 284,219 and 161,897 testing samples, respectively. By contrast, Brute Force-Web, Brute Force-XSS, and SQL Injection appear in both partitions but with very low support, with 249/362, 79/151, and 34/53 training/testing samples, respectively. These support counts explain why Bot and Infiltration are treated as unseen attacks, whereas Brute Force-Web, Brute Force-XSS, and SQL Injection are treated as rare-known attacks.

TABLE I: TEMPORAL SPLIT SUMMARY AND CLASS AVAILABILITY

Item	Training partition	Testing partition
Acquisition days	02-14-2018, 02-15-2018, 02-16-2018, 02-21-2018, 02-22-2018	02-23-2018, 02-28-2018, 03-01-2018, 03-02-2018
Number of samples	4,860,763	3,037,111
Acquisition-day overlap	None	None
Classes present in both partitions	Benign, Brute Force-Web, Brute Force-XSS, SQL Injection	Benign, Brute Force-Web, Brute Force-XSS, SQL Injection
Classes present only in this partition	DDOS attack-HOIC, DDOS attack-LOIC-UDP, DoS Attacks-Golden Eye, DoS Attacks-Hulk, DoS Attacks-Slow HTTP Test, DoS Attacks-Slow Loris, FTP-Brute Force, SSH-Brute Force	Bot, Infiltration

C. Feature Cleaning and Input Construction

Temporal partitioning was followed by feature cleaning for the preparation of the final input model. All non-causal and identifier-like attributes, such as flow Identifiers (IDs),

source and destination Internet Protocol (IP) addresses, and timestamps, were eliminated because they may act as dataset-specific shortcuts rather than traffic-behavior features that are applicable across different settings. Infinite values were replaced with missing values, and unusable columns were discarded after consistency checks. To avoid preprocessing leakage, missing values were imputed using statistics estimated from the training set and then applied to the test set.

The day attribute was retained only as split metadata and removed from the final feature matrix before training. As a result, the model was trained exclusively on cleaned traffic features, while the temporal information was used only to enforce leakage-aware evaluation. Together, the temporal split, removal of timestamp, day, and identifier-like attributes, training-set-based imputation, and fixed evaluation protocol were used to reduce the main leakage sources considered in this study.

D. Intrusion Detection Model

The core supervised detector is based on XGBoost, a scalable gradient-boosted tree ensemble that has demonstrated strong performance on structured tabular data [30]. Given a feature vector x, the classifier predicts the most likely class among the labels observed during training according to:

$$\hat{y} = \arg\max_{k \epsilon y_{\text{trian}}} p(y = k|x) \quad (2)$$

where $\hat{y}$ is the predicted class label, y_{trian} is the set of labels available during training, $p(y = k|x)$ is the posterior probability assigned to class k for sample x. Because this formulation is a closed-set classifier, the model can only assign each test sample to one of the classes observed during training and cannot explicitly output a new attack class unless an additional rejection mechanism is applied. Two model variants were considered: Baseline XGBoost, trained with the natural class distribution, and Weighted XGBoost, trained with class-dependent sample weights to reduce majority-class dominance.

This design enables a controlled comparison of standard learning and imbalance-aware learning under the same temporal evaluation setting.

E. Imbalance-Aware Training

Since the retained dataset exhibits severe class imbalance, a weighted training strategy was introduced for the second model variant. Let n_c denote the number of training samples in class c, N the total number of training samples, and C the number of known training classes. The weight assigned to class c is defined as:

$$w_c = \frac{N}{C\, n_c} \quad (3)$$

where w_c is the weight assigned to class c, N is the total number of training samples, C is the total number of training classes, and n_c is the number of samples belonging to class c.

Each training sample receives the weight of its corresponding class. This formulation increases the penalty assigned to mistakes on minority classes and decreases the dominance of frequent classes during optimization. The purpose of this step is not to rebalance the test distribution, but to determine whether cost-sensitive learning improves rare-attack detection under realistic evaluation. In this study, class weighting is used only as an imbalance-aware strategy for rare-known attacks that are represented in the training set; it is not expected to solve unseen-attack detection, because unseen classes have no training samples from which class weights can be learned.

F. Confidence-Based Rejection for Unseen Attacks

As the time-based split may introduce attack categories that do not appear in the training set, a post hoc confidence-based rejection mechanism was added to test whether low-confidence predictions can be used to flag potentially unknown traffic. For each test sample x, the model outputs a posterior probability vector over the known training classes. The confidence score is defined as:

$$s(x) = \max_{k \epsilon y_{\text{trian}}} p(y = k|x) \quad (4)$$

where $s(x)$ is the confidence score of sample x, $\mathrm{y}_{\text{trian}}$ is the set of known classes observed during training, and $p(y = k|x)$ is the posterior probability assigned to class k.

Given a threshold τ, the final decision rule becomes.

$$\hat{y}_\tau(x) = \begin{cases} \text{Unknown}, & \text{if } s(x) < \tau, \\ \arg\max_{k \in Y_{\text{train}}} p(y = k \mid x), & \text{if } s(x) \geq \tau. \end{cases} \quad (5)$$

where $\hat{y}_\tau(x)$ is the final thresholded prediction, $s(x)$ is the confidence score, τ is the confidence threshold, Y_{train} is the set of classes observed during training, and the label Unknown is assigned when the model confidence falls below the threshold.

This mechanism is intentionally simple. Its role is not to claim a complete open-set detection framework, but rather to test whether confidence thresholding alone is sufficient to mitigate the failure of conventional closed-set classifiers on unseen attacks. To avoid relying on a single arbitrary threshold, the analysis evaluates multiple fixed thresholds and reports threshold sensitivity rather than selecting a test-optimized threshold.

G. Evaluation Protocol

Precision, Recall, F1-Score, and Macro-F1 performance measures were used for evaluation of the performance of the model. The focus was specifically placed upon each category of the rare attacks as the overall Accuracy may still be at a high level while failing in the detection of an intrusion that has a low frequency (and potentially high operational importance). Scalar measures were also supplemented by confusion matrices and recall plots for individual classes to show the systematics of the failure modes.

To avoid mixing rare-class detection with unseen-attack

detection, the evaluation separates rare-known attacks from unseen attacks. Rare-known attacks are defined as attack classes that appear in both the training and testing partitions but represent less than 1% of the training data. Under this definition, Brute Force-Web, Brute Force-XSS, and SQL Injection are treated as rare-known attacks. In contrast, Bot and Infiltration are treated as unseen attacks because they are absent from the training partition and appear only in the temporally later test set.

A class-based-rare analysis was also carried out for the few "attack-classes" that are found under a pre-defined rarity level, and how each of these attack classes' recalls were modified from the baseline to weight models. Thus, this allows us to be able to isolate the difference in the overall performance of the benchmark and the ability of the model to protect against very low-prevalence security-critical attacks. The unseen attack classes are analyzed separately because class weighting cannot learn a class-specific correction for labels that are not represented during training.

H. Explainability Analysis

To interpret the behavior of the tree-based detector, SHAP analysis was applied using Tree Explainer, which provides exact feature attributions for tree ensembles [31]. The SHAP decomposition expresses the prediction as:

$$f(x) = \phi_0 + \sum_{j=1}^{m} \phi_j \quad (6)$$

where $f(x)$ is the model output for sample x, ϕ_0 is the expected model output over the background distribution, m is the number of input features, and ϕ_j is the contribution of feature j to the final prediction.

Explainability was used in two complementary ways. First, global explanation was performed to identify the traffic features that most consistently influenced the model across many samples. Second, local explanation was used to inspect selected rare-attack and failure cases. This allowed the study to connect quantitative performance outcomes with interpretable feature-level behavior, especially in cases where the detector appeared accurate globally but failed on rare or unseen attacks. SHAP was used only as a supplementary interpretability tool and was not used for model training, threshold selection, or unseen-attack detection. Therefore, the XAI component should not be interpreted as a primary detection mechanism, but rather as an explanatory analysis that supports the interpretation of model behavior.

IV. Results and Discussion

This section presents the experimental results of the proposed leakage-aware IDS framework under the adopted time-based evaluation protocol. The results are organized to address the main study questions, including class imbalance, the effect of temporal partitioning, classifier performance, the impact of imbalance-aware training, and the usefulness of confidence-based rejection for unseen attacks. To avoid redundancy, only tables and figures with clear analytical value are included.

A. Dataset Composition and Class Imbalance

Following file-level integrity control, nine structurally consistent files were selected and retained from the CSE-CIC-IDS2018 Corpus (one file was eliminated because it had inconsistent features). The remaining files were then combined to create a single dataset. All duplicates as well as those anomalous rows generated by header contamination were deleted during the data cleaning process. In so doing, the multi-class intrusion structure inherent to the original data was preserved; however, artifacts that could have compromised the validity of subsequent analyses were eliminated. Table II demonstrates that the resultant dataset is severely imbalanced. Specifically, it indicates that benign traffic accounts for 77.33 percent of all observations. Additionally, Table II demonstrates that several operationally relevant attack categories are represented by very few samples; specifically, Brute Force-Web, Brute Force-XSS, and SQL Injection represent some of these attack categories. These figures demonstrate that the classification problem is both multi-class and skewed toward majority traffic. As such, aggregate accuracy does not provide a sufficient measure of practical IDS effectiveness.

Fig. 2 graphically depicts this long-tailed distribution by illustrating class frequency on a log-scale, thus maintaining visual representation of extremely rare attacks that would otherwise be collapsed onto a linear x-axis. Collectively, Table II and Fig. 2 establish that the dataset presents a genuine challenge to the task of detecting intrusions. Therefore, evaluation of performance must explicitly account for minority-class behavior instead of merely relying upon global summary metrics.

TABLE II: CLASS DISTRIBUTION OF THE CLEANED DATASET AND IDENTIFICATION OF RARE ATTACK CLASSES

Class	Count	Percentage	Category Type
Benign	6,107,566	77.33%	Majority
DDOS attack-HOIC	668,478	8.46%	Common attack
DoS attacks-Hulk	434,964	5.51%	Common attack
Bot	284,219	3.60%	Unseen attack
Infiltration	161,897	2.05%	Unseen attack
SSH-Bruteforce	117,324	1.49%	Common attack
DoS attacks-GoldenEye	41,455	0.52%	Common attack
FTP-BruteForce	39,355	0.50%	Rare attack
DoS attacks-SlowHTTPTest	29,673	0.38%	Rare attack
DoS attacks-Slowloris	10,285	0.13%	Rare attack
DDOS attack-LOIC-UDP	1,730	0.02%	Rare attack
Brute Force-Web	611	0.008%	Rare attack
Brute Force-XSS	230	0.003%	Rare attack
SQL Injection	87	0.001%	Rare attack

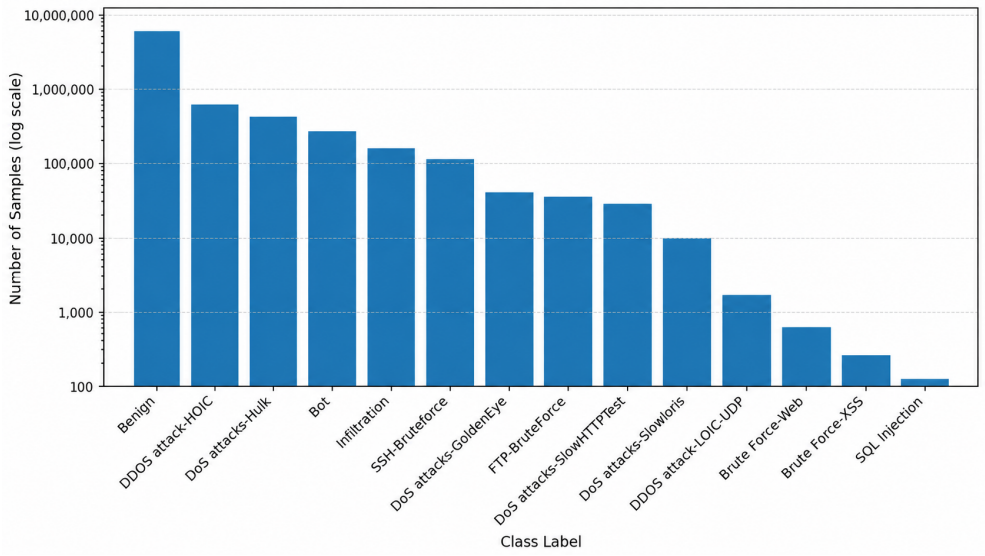

Fig. 2. Class distribution of the cleaned CSE-CIC-IDS2018 dataset on a logarithmic scale.

B. *Temporal Split and Realistic Evaluation Setting*

To minimize temporal leakage and better reflect real deployment conditions, the dataset was partitioned according to acquisition day rather than by random sample-level splitting. This produced temporally disjoint training and testing subsets with no overlap in collection time, as reported in Table I. A particularly important consequence of this protocol is that it creates a more demanding and realistic evaluation scenario in which some attack categories appear only in the test set and are therefore entirely unseen during training.

As shown in Table I, the detector is not only required to generalize to new instances from classes observed during training, but is also confronted with Bot and Infiltration, which are completely absent from the training partition. This shifts the task beyond standard multi-class classification toward a more realistic intrusion detection setting characterized by both temporal drift and class mismatch between training and testing data. In this study, Brute Force-Web, Brute Force-XSS, and SQL Injection are treated as rare-known attacks because they appear in both training and testing partitions with very low training support, whereas Bot and Infiltration are treated as unseen attacks because they appear only in the temporally later test partition. Consequently, the experiments that follow assess not only classification performance on known attacks, but also the model's ability to operate under a deployment-oriented scenario in which previously unseen attack categories emerge at test time.

C. *Baseline Performance Under Time-Based Testing*

The baseline XGBoost model was first evaluated under the leakage-aware temporal split in order to establish a realistic reference point before introducing any imbalance-aware intervention. The baseline model was evaluated using the leakage-free feature set described in Section III, after removing temporal and identifier-like attributes from the final feature matrix. Although the model achieved an overall accuracy of 0.8530, the class-wise results show that this value is highly misleading when interpreted in isolation. The apparently strong aggregate performance is driven primarily by the dominant benign class, whereas the detector behaves much less reliably on rare and previously unseen attacks.

Table III shows a pronounced discrepancy between global and class-sensitive performance. While the detector almost perfectly recognizes benign traffic, it completely fails to detect the unseen attack classes Bot and Infiltration, both of which obtain zero recall. The results also reveal unstable behavior even for rare-known attacks that are represented during training. In particular, Brute Force-Web, Brute Force-XSS, and SQL Injection are only partially recovered. These findings demonstrate that a high overall accuracy does not translate into robust intrusion detection under realistic temporal separation, and that the baseline model remains strongly biased toward dominant traffic patterns.

TABLE III: BASELINE XGBOOST PERFORMANCE ON THE TIME-BASED TEST SET

Class	Precision	Recall	F1-Score	Support
Benign	0.853	0.9999	0.9207	2,590,429
Bot	0.000	0.0000	0.0000	284,219
Brute Force-Web	0.758	0.5414	0.6323	362
Brute Force-XSS	0.956	0.7285	0.8302	151
Infiltration	0.000	0.0000	0.0000	161,897
SQL Injection	0.875	0.5283	0.6588	53
Accuracy	–	–	0.8530	–
Macro-F1	–	–	0.2535	–
Weighted-F1	–	–	0.7854	–

Fig. 3 shows the structural error patterns for the baseline model. Most correct predictions are concentrated in the benign class, while unseen attack samples are forced into one of the known training labels rather than being identified as new or anomalous. This behavior is expected from a closed-set classifier, because the model can only predict classes observed during training. Therefore, the confusion matrix provides a clear visual explanation of why the baseline model performs well on dominant known traffic but fails under unseen-attack conditions.

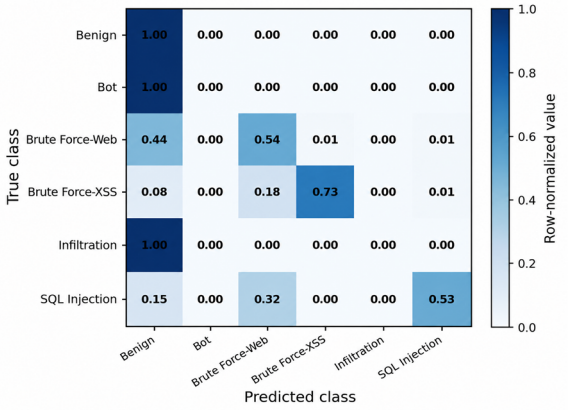

Fig. 3. Row-normalized confusion matrix of the baseline XGBoost model under the leakage-free temporal split.

The same results shown in Fig. 4 are viewed from a class-specific perspective and make it more evident that failures occur with the minority and unseen attacks. As before, recall of benign traffic remains very close to perfect; however, for the lower-prevalence attack classes, the performance drops unevenly and then totally collapses for Bot and Infiltration. SQL Injection is only partially recovered, with a recall of 0.5283, as also reported in Table III. Thus, as expected, this clearly demonstrates that the baseline model has limited practical application in a deployment-oriented environment, since the types of attacks which are most difficult to detect are the ones most often overlooked when overall performance is assessed using only general metrics.

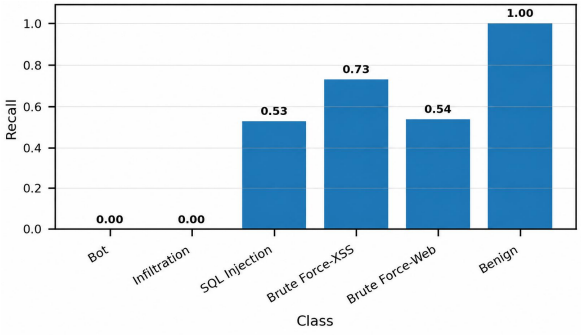

Fig. 4. Per-class recall of the baseline XGBoost model under the leakage-free temporal split.

D. *Effect of Imbalance-Aware Training*

To determine if the poor results from minority attacks were due to class imbalance, an inverse-frequency weighted version of XGBoost was run using the same leakage-free temporal evaluation protocol and test distribution as the baseline. This experiment isolates the effect of using an imbalance-aware learning strategy and enables a direct comparison with the baseline under equivalent conditions.

Table IV indicates that class-weighted training improves recall for several rare-known attack classes, but the improvement is selective and does not necessarily improve F1-score for every class. The most notable recall gains are observed for Brute Force-XSS, where recall increases from 0.7285 to 0.8013, and SQL Injection, where recall increases from 0.5283 to 0.6604. Brute Force-Web shows only a small recall improvement, increasing from 0.5414 to 0.5525. These results illustrate that an imbalance-aware loss can allocate more learning attention toward underrepresented classes, thereby reducing part of the bias toward typical traffic flows.

TABLE IV: COMPARISON OF BASELINE AND WEIGHTED XGBOOST ON THE TIME-BASED TEST CLASSES

Class	Recall (Base)	Recall (Weighted)	F1 (Base)	F1 (Weighted)
Benign	0.9999	0.9998	0.9207	0.9206
Bot	0.0000	0.0000	0.0000	0.0000
Brute Force-Web	0.5414	0.5525	0.6323	0.6260
Brute Force-XSS	0.7285	0.8013	0.8302	0.6994
Infiltration	0.0000	0.0000	0.0000	0.0000
SQL Injection	0.5283	0.6604	0.6588	0.5833
Macro-F1	–	–	0.2535	0.2176

However, Table IV also shows that there is no improvement for the two unseen attack classes, Bot and Infiltration, both of which remain at zero recall under both baseline and weighted training. This confirms that rare-known attacks and unseen attacks represent different evaluation problems: class weighting may improve detection for underrepresented classes observed during training, but it cannot learn a correction for attack categories that are completely absent from the training partition. While training methods may be developed to address imbalances in datasets relative to how often certain events occur, they do not provide a solution to the limited applicability of closed-set classifiers under unseen-attack conditions.

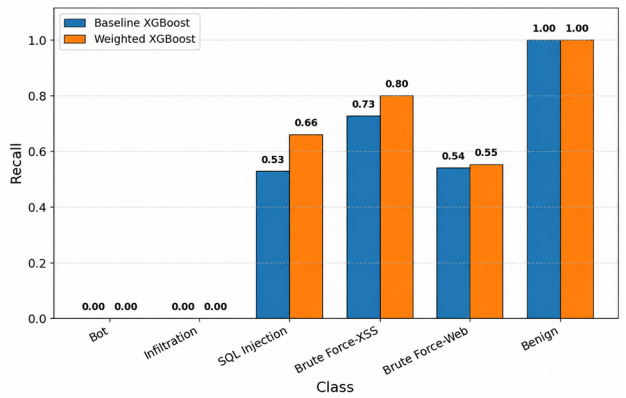

Fig. 5. Recall comparison between the baseline and weighted XGBoost models under the leakage-free temporal split.

Fig. 5 illustrates the selective nature of the improvement achieved by weighted training. The gain is concentrated in rare-known classes, especially Brute Force-XSS and SQL Injection, whereas the unseen classes remain completely unaffected. This visual contrast reinforces the central interpretation of this experiment: the weighted model is more effective at recovering selected underrepresented attacks within the known label space, but it still fails to generalize to attack categories absent from training. Accordingly, class weighting should be regarded as a mitigation for rare-known class imbalance, but not as a solution to unseen-attack generalization under realistic

temporal evaluation.

E. Confidence-Based Rejection and Reliability Failure on Unseen Attacks

To investigate whether simple confidence thresholding could mitigate the failure of the classifier on previously unseen attacks, a post hoc rejection mechanism was introduced in which low-confidence predictions were reassigned to an Unknown category. This experiment was intended to test whether unseen or out-of-distribution traffic would naturally produce lower predictive confidence, which in turn would make threshold-based rejection a practical safeguard under realistic deployment conditions. Instead of relying on a single rejection threshold, multiple fixed thresholds were evaluated to examine whether the conclusion changes under different confidence cutoffs.

Table V shows that confidence-based rejection does not recover unseen attacks in any meaningful way. Even when the threshold is increased from 0.500 to 0.999, the overall rejection rate remains extremely small, increasing only from approximately 0.0000 to 0.0010, while the unseen rejection rate reaches only 0.0010 at the most aggressive threshold. This confirms that the failure is not simply caused by choosing an inappropriate threshold.

Also, Table V shows that aggressive thresholding increasingly suppresses rare-known detections. The mean rare-class rejection rate increases as the threshold becomes stricter, while mean rare-class recall after rejection decreases from 0.5822 at threshold 0.500 to 0.0705 at threshold 0.999. Therefore, naive rejection does not provide a useful trade-off: it fails to identify unseen attacks while reducing the already fragile detection of rare-known attacks.

TABLE V: CONFIDENCE-REJECTION THRESHOLD SENSITIVITY

Threshold	Unknown Count	Overall Rej.	Unseen Rej.	Rare Rej.	Rare Recall
0.500	12	0.0000	0.0000	0.0257	0.5822
0.700	140	0.0000	0.0001	0.1399	0.5303
0.800	212	0.0001	0.0001	0.1670	0.5194
0.900	409	0.0001	0.0002	0.2793	0.4658
0.950	505	0.0002	0.0002	0.3548	0.4296
0.990	936	0.0003	0.0003	0.4740	0.3473
0.995	1,209	0.0004	0.0004	0.5581	0.2761
0.999	3,138	0.0010	0.0010	0.7637	0.0705

Fig. 6 illustrates that the mean confidence deficit remains very low for benign traffic and for the unseen attack classes Bot and Infiltration. Although Infiltration has a slightly larger confidence deficit than the other groups, all values remain close to zero, indicating that the classifier produces high-confidence predictions even when the prediction is wrong. This explains why threshold-based rejection is ineffective against unknown traffic: the model is not only wrong on unseen attacks, but also confident in those wrong assignments.

Fig. 7 further supports this interpretation by showing the threshold-sensitivity behavior across the tested cutoff values. The rejection rate remains negligible even under very strict thresholds, whereas rare-known recall deteriorates as the rejection threshold increases. Taken together, Table V, Fig. 6, and Fig. 7 demonstrate that the baseline model lacks meaningful uncertainty awareness under temporal distribution shift. Therefore, effectively addressing unknown attacks requires a more formal open-set or uncertainty-aware strategy beyond simply applying a posterior confidence threshold.

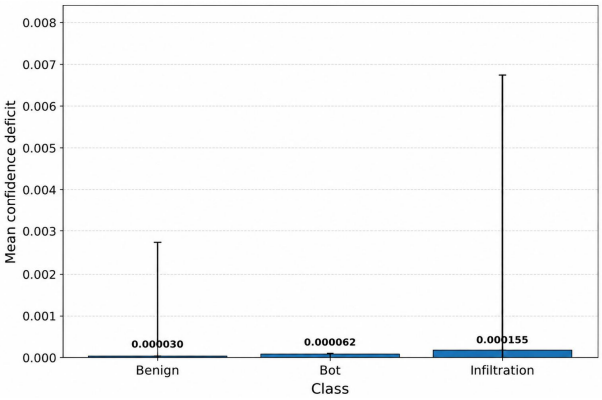

Fig. 6. Mean confidence deficit for benign and unseen attack classes under the leakage-free temporal split.

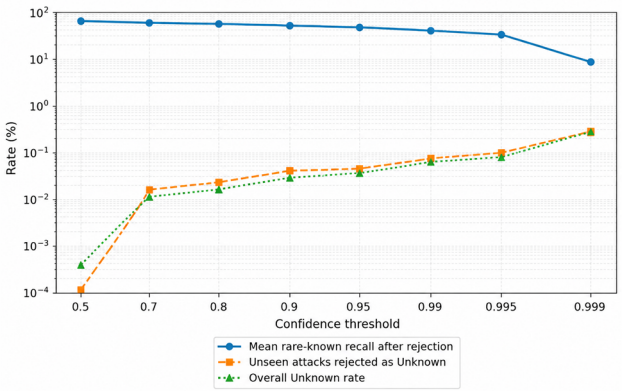

Fig. 7. Threshold-sensitivity analysis of confidence-based rejection under the leakage-free temporal split.

F. Robustness Check against Exact Train-Test Feature Overlap

To further examine whether the conclusions were affected by exact train-test feature overlap, an additional robustness check was conducted after leakage-free feature cleaning. Exact feature-vector hashes from the training set were compared against the test set, and overlapping test samples were removed before re-evaluating the trained models. This audit removed 294,820 test samples, corresponding to 9.71% of the original test partition, leaving 2,742,291 test samples for the duplicate-filtered evaluation.

As shown in Table VI, the duplicate-filtered results confirm the main conclusion of the study. The baseline XGBoost model achieved an accuracy of 0.8374, Macro-F1 of 0.5096, and Weighted-F1 of 0.7633, while the weighted XGBoost model achieved an accuracy of 0.8373, Macro-F1 of 0.4777, and Weighted-F1 of 0.7633. More importantly, Bot and Infiltration still obtained zero recall after duplicate filtering. Therefore, the observed failure on unseen attacks is not explained by exact train-

test feature overlap, but reflects the limitation of closed-set learning under temporal class mismatch.

TABLE VI: DUPLICATE-FILTERED ROBUSTNESS CHECK

Metric	Baseline XGBoost	Weighted XGBoost
Test samples after filtering	2,742,291	2,742,291
Accuracy	0.8374	0.8373
Macro-F1	0.5096	0.4777
Weighted-F1	0.7633	0.7633
Bot recall	0.0000	0.0000
Infiltration recall	0.0000	0.0000

G. Limitations and Validity Considerations

Although the adopted protocol reduces several major sources of evaluation leakage, the results should be interpreted within the limits of the retained CSE-CIC-IDS2018 files and their acquisition conditions. The temporal split provides a more realistic evaluation than random sample-level splitting, but it may still reflect the specific order and composition of the available acquisition days. Therefore, the reported results should be viewed as evidence of temporal generalization failure under the studied setting, rather than as a universal estimate of IDS performance across all IoT or edge-network environments.

Another limitation is that the detector is evaluated as a closed-set classifier. Consequently, unseen attacks can only be assigned to one of the known training labels unless a rejection mechanism is applied. The confidence-based rejection experiment shows that simple thresholding is not sufficient in this setting, but it does not replace more advanced open-set or uncertainty-aware methods. In addition, the duplicate-filtered robustness check controls exact train-test feature overlap, but it does not guarantee removal of all possible near-duplicate or campaign-level dependencies.

V. CONCLUSION

This study presented a leakage-aware and realism-driven evaluation framework for intrusion detection using the CSE-CIC-IDS2018 dataset under a time-based train-test protocol. By preventing temporal overlap between training and testing data and removing temporal and identifier-like attributes from model training, the adopted setup provided a more realistic assessment of IDS generalization than conventional random-split evaluation. The results showed that the dataset represents a severely imbalanced multi-class detection problem in which benign traffic dominates, while several attack categories are extremely rare. Under this setting, the baseline XGBoost model achieved high overall accuracy, but failed completely on unseen attack classes and remained unstable on several rare-known attacks, demonstrating that aggregate metrics alone are insufficient for realistic IDS evaluation. The weighted model improved recall for several rare-known attacks that were represented during training, confirming the value of imbalance-aware learning. However, no improvement was observed for unseen attacks, indicating that class imbalance and attack novelty are distinct challenges. In addition, the confidence-based rejection analysis showed that simple thresholding failed to identify unseen attacks, as the classifier remained highly confident even when making incorrect predictions on traffic absent from the training set. The threshold-sensitivity analysis further confirmed that stricter rejection thresholds did not provide a useful trade-off, since unseen-attack rejection remained negligible while rare-known recall deteriorated. The duplicate-filtered robustness checks also confirmed that the observed failure on Bot and Infiltration was not explained by exact train-test feature overlap. Overall, the findings show that reliable intrusion detection requires more than strong benchmark accuracy. It requires leakage-aware evaluation, class-sensitive analysis, robustness checks against evaluation artifacts, and explicit mechanisms for handling unseen attacks under realistic temporal shifts. Future work may therefore explore dedicated open-set intrusion detection, uncertainty-aware modeling, and cross-dataset validation strategies for more robust handling of previously unseen attacks.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

AUTHOR CONTRIBUTIONS

Ameen Shaheen carried out the research; Wael Alzyadat performed data analysis; Aysh Alhroob contributed to methodology development and manuscript preparation. All authors reviewed and approved the final version.

ACKNOWLEDGMENT

All authors express their gratitude to their institutes, colleagues, and individuals who cooperated in conducting this work and provided valuable support.

REFERENCES

[1] A. Hnaif, K. Jaber, M. Alia, and M. Daghbosheh, "Parallel scalable approximate matching algorithm for network intrusion detection systems," *The International Arab Journal of Information Technology*, vol. 18, no. 1, pp. 77–84, 2021. doi: 10.34028/iajit/18/1/9

[2] M. Abduljawad and A. Ahmad, "Comparative analysis of machine learning algorithms for intrusion detection in IoT networks," *International Journal of Advances in Soft Computing and Its Applications*, vol. 17, no. 3, pp. 260–275, 2025. doi: 10.15849/IJASCA.251130.15

[3] H. M. R. U. Rehman *et al*., "A systematic literature study of machine learning techniques-based intrusion detection: datasets, models, challenges, and future directions," *Journal of Big Data*, vol. 12, 264, 2025. doi: 10.1186/s40537-025-01323-2

[4] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Information Systems Security and Privacy (ICISSP)*, 2018, pp. 108–116. doi: 10.5220/0006639801080116

[5] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on CSE-CIC-ids2018 big data," *Journal of Big Data*, vol. 7, 104, 2020. doi: 10.1186/s40537-020-00382-x

[6] D. Ha Thanh, "A transfer-aware, deployment-oriented evaluation framework for NetFlow-Based Intrusion Detection Systems (TAN-IDS)," *PLOS ONE*, vol. 21, no. 4, e0346801, 2026. doi: 10.1371/journal.pone.0346801

[7] H. Al-Mimi, N. A. Hamad, M. M. Abualhaj, M. Sh. Daoud, A. A.

Dahoud, and M. Rasmi, "An enhanced intrusion detection system for protecting HTTP services from attacks," *Int. J. Adv. Soft Comput. Appl.*, vol. 15, no. 2, pp. 68−83, 2023. doi: 10.15849/IJASCA.230720.05

[8] A. Apicella, F. Isgrò, and R. Prevete, "Don't push the button! Exploring data leakage risks in machine learning and transfer learning," *Artificial Intelligence Review*, vol. 58, no. 11, 339, 2025. doi: 10.1007/s10462-025-11326-3

[9] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodriguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern Recognition*, vol. 45, no. 1, pp. 521–530, 2012. doi: 10.1016/j.patcog.2011.06.019

[10] L. L. Guo, S. R. Pfohl, J. Fries, A. E. W. Johnson, J. Posada, C. Aftandilian, N. Shah, and L. Sung, "Evaluation of domain generalization and adaptation on improving model robustness to temporal dataset shift in clinical medicine," *Scientific Reports*, vol. 12, no. 1, 2726, 2022. doi: 10.1038/s41598-022-06484-1

[11] M. A. Shyaa, N. F. Ibrahim, Z. Zainol, R. Abdullah, M. Anbar, and L. Alzubaidi, "Evolving cybersecurity frontiers: A comprehensive survey on concept drift and feature dynamics aware machine and deep learning in intrusion detection systems," *Engineering Applications of Artificial Intelligence*, vol. 137, 109143, 2024. doi: 10.1016/j.engappai.2024.109143

[12] G. J. de Bruin *et al*., "Experimental evaluation of train and test split strategies in link prediction," in *Proc. International Conference on Complex Networks and their Applications*, 2020, pp. 79−91. doi: 10.1007/978-3-030-65351-4_7

[13] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009. doi: 10.1109/TKDE.2008.239

[14] F. M. Megahed, Y.-J. Chen, A. Megahed, Y. Ong, N. Altman, and M. Krzywinski, "The class imbalance problem," *Nature Methods*, vol. 18, no. 11, pp. 1270–1272, 2021. doi: 10.1038/s41592-021-01302-4

[15] W. J. Scheirer, A. Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, pp. 1757–1772, 2013. doi: 10.1109/TPAMI.2012.256

[16] F. Gao, L. Yang, and H. Li, "A survey on open set recognition," *Journal of Nanjing University (Natural Sciences)*, vol. 58, no. 1, pp. 115–134, 2022. doi: 10.13232/j.cnki.jnju.2022.01.012

[17] Y. Zhang, J. Niu, D. Guo, Y. Teng, and X. Bao, "Unknown Network Attack Detection Based on Open Set Recognition," *Procedia Computer Science*, vol. 174, pp. 387–392, 2020. doi: 10.1016/j.procs.2020.06.114

[18] S. Yang, Z. Zhang, H. Lin, and Y. Chen, "Generalized out-of-distribution detection: A survey," *International Journal of Computer Vision*, vol. 132, no. 12, pp. 5635–5662, 2024. doi: 10.1007/s11263-024-02117-4

[19] V. Z. Mohale and I. C. Obagbuwa, "A systematic review on the integration of explainable artificial intelligence in intrusion detection systems to enhance transparency and interpretability in cybersecurity," *Frontiers in Artificial Intelligence*, vol. 8, 1526221, 2025. doi: 10.3389/frai.2025.1526221

[20] A. Hozouri, A. Mirzaei, and M. Effatparvar, "A comprehensive survey on intrusion detection systems with advances in machine learning, deep learning and emerging cybersecurity challenges," *Discover Artificial Intelligence*, vol. 5, 314, 2025. doi: 10.1007/s44163-025-00578-1

[21] S. Neupane, J. Ables, W. Anderson, S. Mittal, S. Rahimi, I. Banicescu, and M. Seale, "Explainable intrusion detection systems (X-IDS): A survey of current methods, challenges, and opportunities," *IEEE Access*, vol. 10, pp. 112392–112415, 2022. doi: 10.1109/ACCESS.2022.3216617

[22] V. Shanmugam, R. Razavi-Far, and E. Hallaji, "Addressing class imbalance in intrusion detection: A comprehensive evaluation of machine learning approaches," Electronics, vol. 14, no. 1, 69, 2025. doi: 10.3390/electronics14010069

[23] M. S. Milosevic and V. M. Ciric, "Extreme minority class detection in imbalanced data for network intrusion," *Computers and Security*, vol. 123, 102940, 2022. doi: 10.1016/j.cose.2022.102940

[24] H. Ren, Y. Tang, W. Dong, S. Ren, and L. Jiang, "DUEN: Dynamic ensemble handling class imbalance in network intrusion detection," *Expert Systems with Applications*, vol. 229, 120420, 2023. doi: 10.1016/j.eswa.2023.120420

[25] P. rajesh Kanna and P. Santhi, "Unified deep learning approach for Efficient intrusion detection system using integrated spatial-temporal features," *Knowledge-Based Systems*, vol. 226, 107132, 2021. doi: 10.1016/j.knosys.2021.107132

[26] P. Rajesh Kanna and P. Santhi, "Hybrid intrusion detection using mapreduce based black widow optimized convolutional long short-term memory neural networks," *Expert Systems with Applications*, vol. 194, 116545, 2022. doi: 10.1016/j.eswa.2022.116545

[27] O. Arreche, T. Guntur, and M. Abdallah, "XAI-IDS: Toward proposing an explainable artificial intelligence framework for enhancing network intrusion detection systems," *Applied Sciences*, vol. 14, no. 10, 4170, 2024. doi: 10.3390/app14104170

[28] S. Layeghy, M. Gallagher, and M. Portmann, "Benchmarking the benchmark−Comparing synthetic and real-world network IDS datasets," *Journal of Information Security and Applications*, vol. 80, 103689, 2024. doi: 10.1016/j.jisa.2023.103689

[29] Canadian Institute for Cybersecurity. CSE-CIC-IDS2018 on AWS. [Online]. Available: https://www.unb.ca/cic/datasets/ids-2018.html

[30] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting System," in P*roc. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794. doi: 10.1145/2939672.2939785

[31] S. M. Lundberg, G. G. Erion, H. Chen, A. D. Grave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S. I. Lee, "From local explanations to global understanding with explainable AI for trees," *Nature Machine Intelligence*, vol. 2, no. 1, pp. 56–67, 2020. doi: 10.1038/s42256-019-0138-9

Ameen Shaheen received his B.Sc. and M.Sc. degrees in computer information systems in 2010 and 2012, respectively, and his Ph.D. degree in computer science in 2018. He is currently an assistant professor at the Department of Software Engineering, Faculty of Science and Information Technology, Al-Zaytoonah University of Jordan, Amman, Jordan. His research interests include databases, algorithms, systems analysis, software engineering, artificial intelligence, cloud computing, parallel computing, and distributed systems. He has participated in research related to software engineering, mobile learning applications, wearable technologies, and computing systems.

Wael Alzyadat is an associate professor in the Department of Software Engineering, Faculty of Science and Information Technology, at Al-Zaytoonah University of Jordan, Amman, Jordan. He currently serves as the head of the software engineering department. His research interests include software engineering, big data analytics, intelligent systems, data mining, knowledge discovery in databases, networks, and decision-support systems. His research work includes big data models for decision making, drug chemical structure analysis, data classification and clustering, MapReduce-based big data processing, and software requirements engineering.

Aysh Alhroob is a full professor of software engineering and currently serves as the dean of the Faculty of Science and Information Technology at Al-Zaytoonah University of Jordan, Amman, Jordan. He received his Ph.D. in software engineering from the University of Bradford, United Kingdom. His research interests include software engineering, software testing, software requirements engineering, data and text mining, big data analytics, data engineering, and intelligent software systems. He has published several research papers in the areas of requirements analysis, automated software testing, machine learning-based software engineering, and big data systems.