Published by
NIGERIAN SOCIETY OF PHYSICAL SCIENCES
Available online @ https://asr.nsps.org.ng/index.php/asr

African Scientific Reports 5 (2026) 559

African Scientific Reports

Lightweight intrusion detection systems for critical infrastructure protection: a comparative evaluation of class balancing techniques with strategic dataset reduction and cross-dataset validation

Muhammad Nazeer Musa[a,*], Olakunle Abayomi Aboderin[b], Lubabatu Ibrahim Hassan[a], Yasmin Moukarim[a]

[a]*Department of Cyber Security, Nigerian Defence Academy, Kaduna, Nigeria*
[b]*Department of ICT and Cyber Security, University of Ibadan, Ibadan, Nigeria*

Abstract

Machine-learning-based intrusion detection systems are increasingly important for protecting critical infrastructure as cyber threats grow in number and sophistication. However, the computational demands of many state-of-the-art models can limit deployment in resource-constrained environments, while traditional intrusion detection systems often exhibit substantial generalization gaps across networks and real-world settings. This study evaluates whether aggressive dataset reduction, optimized preprocessing, and hybrid class balancing can improve cross-dataset generalization without dedicated hardware. Five class-balancing techniques were evaluated on a KDD Cup 1999 dataset reduced by 90%, with SMOTETomek selected because it produced the lowest false-negative count, a critical consideration for security-sensitive anomaly detection. In benchmark-transfer validation against the NSL-KDD test dataset, the lightweight model achieved 80.11% accuracy, compared with 76.65% for a minimally preprocessed baseline. It also processed 22,544 instances in 0.1005 s, corresponding to 224,388 instances per second, on commodity hardware with a 4-core CPU and 16 GB RAM. These results challenge reliance on high single-dataset accuracy and provide a reproducible, computationally sustainable benchmark-transfer methodology. Independent pipeline validation on UNSW-NB15 achieved 94.66% accuracy and a Matthews correlation coefficient (MCC) of 0.8849, while validation on a supervisory control and data acquisition/industrial control system (SCADA/ICS) dataset achieved 99.98% accuracy, an MCC of 0.9991, and a 0.0% false-positive rate. These latter results provide preliminary, non-transfer evidence that the pipeline design generalizes beyond the KDD family.

DOI: 10.46481/asr.2026.5.3.559

Keywords: Intrusion detection system, Class imbalance, SMOTETomek, Dataset reduction, Cross-dataset validation

Article History:
Received: 27 May 2026
Received in revised form: 02 September 2026
Accepted for publication: 27 September 2026
Available online: 21 October 2026

© 2026 The Author(s). Published by the Nigerian Society of Physical Sciences under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

1. Introduction

The rapid digital transformation of Nigeria's critical national infrastructure, such as electric power systems, water treatment facilities, telecom networks, banking systems, schools, healthcare delivery systems, and defence installations has significantly increased

*Corresponding Author Tel. No.: +234-816-604-6507.
Email address: muhammadmusa2502@nda.edu.ng (Muhammad Nazeer Musa)

Nigeria's cyberattack surface area. Today's threat actors are increasingly sophisticated. Cybercriminals employ techniques such as Advanced Persistent Threats (APTs), zero-day exploits, polymorphic malware, and ransomware [1]. The interconnections of the various systems require them to store sensitive information (i.e.,, student records, patient information, financial records, and classified intelligence), making them prime targets for cybercriminals and nation-state adversaries. The consequences of cyber intrusions are not limited to stolen data. Successful cyber intrusions can lead to disastrous failures in infrastructure. For example, the coordinated cyberattack on Ukraine's power grid in 2015 left 230,000 individuals without power by exploiting weaknesses in the power grid's industrial control systems [2]. The NotPetya ransomware attack in June 2017 resulted in estimated losses of $10 billion and demonstrates how cyberattacks create disruptions to international supply chains [3].

In addition, during the COVID-19 pandemic, health care infrastructure in the United States was subjected to unprecedented ransomware attacks that resulted in hospitals diverting the emergency room patients and delaying critical medical procedures [4]. These examples show that ensuring effective cybersecurity also ensures public safety, national economic viability, and national security. Signature-based intrusion detection systems (IDS) have difficulty detecting evolving threats because new attack vectors and polymorphic malware will consistently be developed to evade detection through signature matching [5, 6]. Anomaly detection methods using machine learning represent a transformational approach to identify previously unknown threats based on behavioural patterns rather than signature-based detection [7]. However, as a key component of effective IDS, the use of machine learning (ML) presents critical challenges that create considerable obstacles for resource-restricted organizations like educational institutions and emerging economies: (a) the majority of ML model performance is inferior when using real-world data compared with laboratory data (e.g., the reported accuracy from lab tests is 99%)[8]; (b) large-scale training data can pose numerous challenges (e.g.,

KDD Cup 1999 dataset contains over 4.9 million records) significantly complicating the use of storage, memory, processing, and long training times for the model to be adequately trained [9]; and (c) with distributed denial of service (DDoS) attacks, ML models can have high accuracy given that less than 1% of connections are deemed malicious; however, they are likely to miss distinctive yet critical attacks due to their rarity [10].

Recent cybersecurity literature has provided empirical evidence that systematic data preprocessing is more effective than either using advanced algorithmic capabilities, tuning hyperparameters, or using complex ensemble methods [11]. However, several important gaps limit the practical application of this research. First, most research to date evaluates preprocessing techniques in isolation rather than as part of a complete pipeline and does not systematically assess how preprocessing order affects final model accuracy. Second, numerous data-balancing methods exist (e.g.,, using SMOTE, ADASYN, Tomek links, and various hybridizations) [10], but relatively few comparative evaluations exist to indicate which method is most appropriate. Third, the combined effect of aggressive dataset reduction and hybrid preprocessing on cross-dataset generalization remains poorly explored. This study contributes to the existing literature by presenting a replicable preprocessing and training pipeline that challenges the assumption that more data inherently produces better models.

Specifically, its novelty lies in the systematic integration of: (1) a 90% aggressive instance reduction via stratified sampling; (2) a comparative evaluation of five class balancing techniques identifying the optimal method for eliminating false negatives; and (3) the demonstration that reduced, balanced model can achieve superior cross-dataset transferability and real-time computational efficiency on low-cost commodity hardware, making it highly viable for deployment in resource-constrained environments. The remainder of this paper is organized as follows: Section 2 presents the previous studies related to the generalization of IDS, dataset reduction and class balancing; Section 3 describes the research design including the five-stage preprocessing pipeline and the experimental design; Section 4 provides the results of comparative balancing techniques and the results of the cross-dataset validation; Section 5 concludes the paper and discusses the ramifications on the protection of critical infrastructure and its limitations.

2. Literature review

The generalization of machine-learning-based IDSs remains a substantial challenge in the literature. Most published IDS studies report performance above 99% when training and testing use splits from a single dataset; however, performance often declines under validation on a different dataset or on real network traffic [12]. There were also fundamental barriers in applying ML for intrusion detection, as mentioned by Sommer and Paxson [5], and stated that datasets created in laboratories never have the same semantic diversity and statistical properties of live production network traffic. More recently, work by Arp *et al.* [13], showed that the majority of published IDS models learn dataset-specific biases like how feature value distributions, protocol and attack signature types are part of a given dataset and therefore do not generalize to the attack semantics between a different live production network, as shown by the decline in recall performance of more than 10%-15% across the two different datasets tested.

Reduction strategies for datasets are poorly understood in the security literature. Though it is logical that as training data increases the quality of the models will also increase, practical resource limitations mean that we need to have an understanding of the minimum viable dataset sizes and to maintain acceptable detection performance. An example of this is the research work by Harahsheh *et al.* [14], who achieved an astonishing 99.99% accuracy on IoT intrusion detection using 11 out of the 84 features, using an intelligent selection of features from the InSDN dataset that yielded a 73% reduction in the number of dimensions of the dataset without sacrificing performance. However, the experimental investigation was more focused on reducing the number of features than the number of instances.

There are various types of instance reduction strategies, including quality filtering of records; therefore, eliminating large amounts of data that do not add any unique information, which is especially critical for datasets produced by synthetic benchmarks such as KDD'99, in which Tavallaee *et al.* [15] found that approximately 78% of the records in the training dataset and 75% of identical records in the test dataset. Random sampling of records allows for a very easy reduction of the dataset size, however, has the potential to remove rare attack examples. Intelligent sampling strategies like Difficult Set Sampling for Training Efficiency (DSSTE) preferentially retain borderline examples near decision boundaries while eliminating easily-classified instances [16]. However, few studies systematically investigate aggressive instance reduction (>80%) combined with hybrid preprocessing to compensate for reduced training data volume. A recent study by Younisse *et al.* [9] provided additional evidence of the success of aggressive reduction when they assessed a two-stage reduction strategy, composed of feature reduction using deep autoencoder techniques and stratified sampling, on six datasets containing IoT intrusion data.

They showed that their dataset reduction methodology could reduce training time by up to 99%, while only producing a small drop in F1-score (typically $\leq$ 1%). They cautioned, however, that too much reduction in size could hurt detection accuracy of minority attack classes. They also found a significant amount of redundant features across multiple datasets, which indicates that reduced datasets typically provide similar F1-scores to non-reduced datasets. This raises the question of whether large amounts of data are necessary for the development of robust IDS. The findings presented by Younisse *et al.* [9] demonstrated that efficient and scalable IDS can be developed in resource-constrained environments. Manocchio *et al.* [11] conducted a systematic empirical test of 72 different combinations of preprocessing methods, including encoding, outlier handling and scaling, on four benchmark NIDS datasets and nine machine learning models. They showed that inconsistent preprocessing choices in the literature can reduce performance and prevent fair comparisons between techniques. They suggest a standard approach to preprocessing data analysed using any of the datasets or models tested.

They also demonstrated that their standard approach to preprocessing increases detection accuracy of shallow neural networks by up to 11% when compared to suboptimal preprocessing methods.

Mitigating a class imbalance is the most important preprocessing step for security datasets that have a high proportion of traffic made up of minority attacks (0.5% of the total traffic) and are tied to the greatest impact potential. Traditionally, oversampling (adding duplicate instances of the minority class) and undersampling (removing majority instances) were common solutions, each with downsides: oversampling can lead to overfitting if the same sample is repeated too many times while undersampling loses valuable information from the majority class. A hybrid approach that uses both synthetic generation and boundary cleaning techniques provides complementary advantages. For example, using SMOTE (Synthetic Minority Over-sampling Technique), synthetic instances of the minority class can be created using linear interpolation between existing samples and k-nearest neighbors. Similarly, Tomek Links can help identify ambiguous boundary locations in a space where two different classes are each other's nearest neighbours, allowing the removal of majority class points that fall within these ambiguous boundaries, thus making it easier to define decision boundaries.

According to Wu *et al.* [17], K-means clustering with SMOTE hybrid resampling and enhanced Random Forests achieved 99.72% accuracy over the training set and 78.47% accuracy over the test set compared to other state-of-the-art systems using the NSL-KDD dataset. A similar study by Salihu *et al.* [18] employed a hybrid methodology combining ADASYN and Tomek links, removing excess instances from the majority class. Their multiple-systems evaluation of different supervised machine-learning algorithms demonstrated significant success with this hybrid balancing method in the reduction of biased predictions that occur due to the imbalance of the data being used for anomaly-based intrusion detection. The algorithm that performed best was Random Forests, achieving 99.67% accuracy with the KDD'99 dataset. However, most studies to date have employed only a single such method for balancing and have not provided a comprehensive comparative evaluation to validate the choice of that methodology, especially in security-sensitive applications where the effects of missing an attack (false negative) can be catastrophic; conversely, the impact of producing a false alert (false positive) will only require the attention of an analyst.

In a related study, Bassey *et al.* [19] proposed a hybridized approach combining Random Forest with Long Short-Term Memory (LSTM) networks for intrusion detection on the NSL-KDD dataset. Their model achieved 98.3% prediction accuracy by employing class weighting in the loss function to address the inherent class imbalance problem. This demonstrates the continued relevance of NSL-KDD as a benchmark for evaluating imbalanced classification techniques in intrusion detection.

The emergence of the Internet of Things (IoT) and edge computing has led to an increased focus on computational efficiency for deployments with limited resources (e.g.,, IoT/edge computing). Joshi *et al.* [12] reported on lightweight IDS in the IoT context and concluded that Random Forest (RF) is the optimal algorithm because it exhibits both high accuracy and reasonable computational efficiency. This is because RF is inherently parallelizable, allowing for the use of multiple processors; thus, there was no need for a graphics processing unit (GPU) to achieve high accuracy (78%) on the NSL-KDD dataset. The effectiveness of RF in intrusion detection contexts has been further validated by Bassey *et al.* [19], who demonstrated that Random Forest, when combined with deep learning architectures (LSTM), achieved superior detection rates on NSL-KDD. Through sampling strategies and hybrid feature selection techniques, Rahamathulla and Ramaiah achieved an impressive 99% accuracy, but at a much-reduced memory footprint than traditional approaches.

Additionally, Islam *et al.* [20] demonstrated that there are lightweight techniques available that can also be used with little additional computational overhead compared to conventional methods, whilst still achieving competitive accuracies (i.e.,, 99.81%

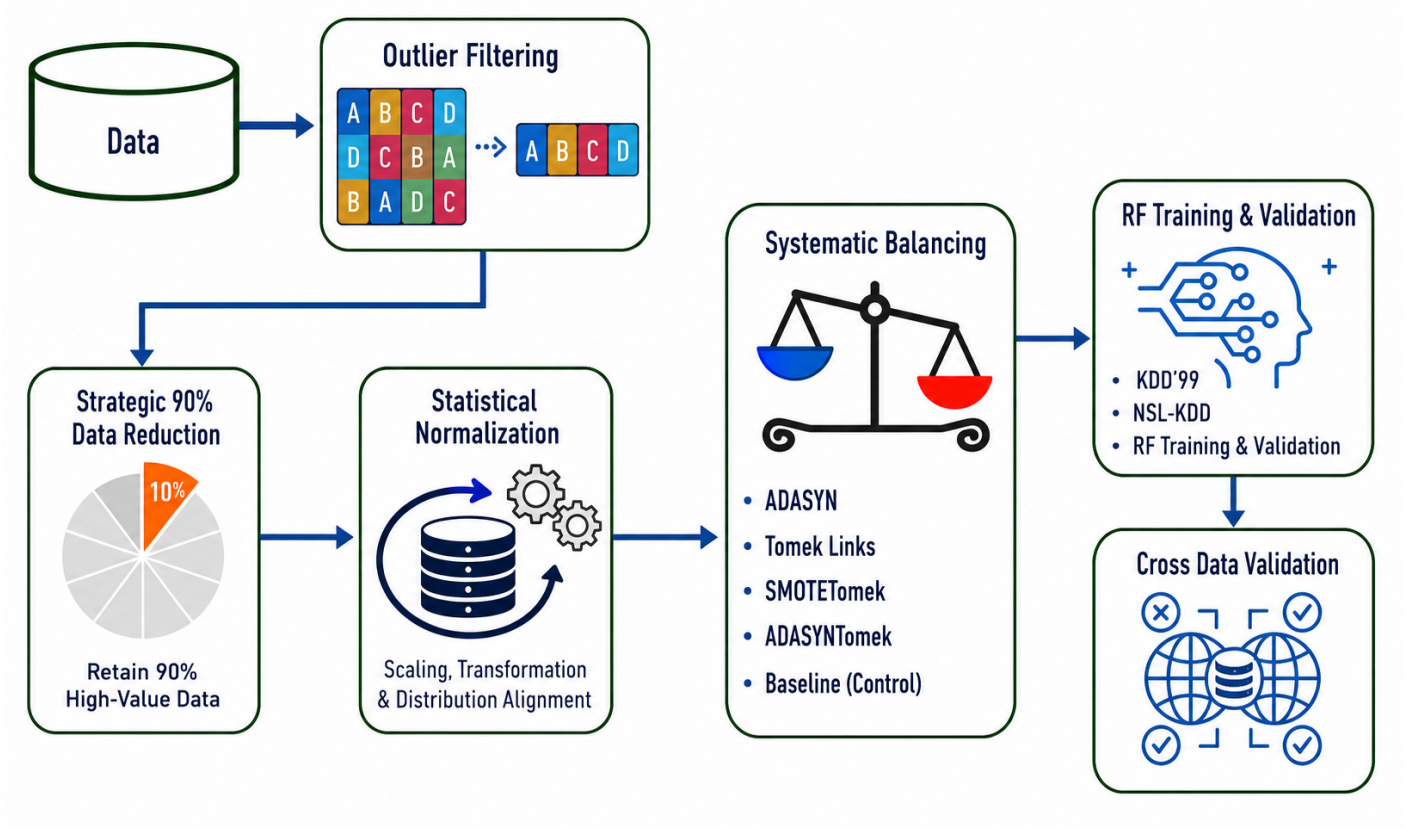


Figure 1: Proposed methodology workflow.

accuracy) using the Support Vector Machine (SVM) classifier within their study. However, little evidence currently addresses the relationship between aggressive dataset reduction methodologies, optimal selection of balancing techniques, and cross-dataset generalization capabilities; therefore, this study addresses this gap through systematic quantitative experimentation and validation.

3. Methodology

3.1. Research design and experimental framework

Figure 1 summarizes the study workflow. This study adopts a quantitative, experimental approach, framed by three research questions: RQ1: Can strategic 90% dataset reduction by stratified random sampling achieve an optimal level of cross-dataset accuracy for critical infrastructure protection? RQ2: Which balancing technique: ADASYN, Tomek Links, SMOTETomek, ADASYNTomek, or Baseline demonstrates the best accuracy-efficiency trade-off for security-sensitive applications where missing attacks leads to critical failure? RQ3: What impact do preprocessing techniques have on cross-dataset generalization when applied in succession to datasets? The methodology workflow comprises a five-stage preprocessing pipeline that includes strategic dataset reduction, quality filtering, comparative balancing technique evaluation and selection, statistical normalization and feature selection, optimized Random Forest training, and cross-dataset validation. The focus on lightweight deployment aligns with recent advances in edge computing security, particularly for resource-constrained environments in developing regions [21].

3.2. Dataset acquisition and characteristics

The KDD Cup 1999 dataset from the UCI Machine Learning Repository, containing 4,898,431 connection records from a simulated intrusion-detection evaluation program conducted by DARPA in 1999 (approximately nine weeks of simulated network activity), was used for training [22]. There are 41 features in each connection, including basic connection features (duration, protocol type, service, among others), content features (such as number of failed login attempts and number of file creation operations), time-based traffic features (e.g., number of connections to the same host in a two-second window), host-based traffic features (connections to the same destination host), among others. Although the dataset has known limitations, including its age, synthetic origin, and statistical biases, KDD'99 remains the most widely used benchmark, making it easy to compare with existing published work [15]. The KDD '99 dataset contains normal traffic and multiple attack subcategories (DoS, probe, R2L, and U2R). We consolidated the

original attack subcategories into a single 'attack' class, yielding a binary setup (normal vs. attack). This resulting dataset is highly imbalanced, with a ~80:20 split, where attack traffic dominates at 80% and normal traffic trails at 20%.

NSL-KDD represents a cleansed version of KDD created by Tavallaee *et al.* [15] to address limitations of the original dataset by removing duplicate records, improving class distributions, and assigning a difficulty level to each instance. The NSL-KDD Test+ set contains 22,544 instances and has a fundamentally different distribution of attack types than KDD'99. This is a very useful benchmark transfer validation scenario since it simulates an operational deployment set where the attack type is different from what we have in the historical data that we train with [23]. Using KDD'99 for benchmark-transfer validation (train on KDD and validate on NSLKDD) yields a realistic assessment of what we can reasonably expect in practice by simulating a deployment where the training data was significantly different from what we typically observe in the network.

3.3. Stage 1: Quality and outlier filtering

Following the initial data loading and column definition, the dataset underwent an outlier filtering procedure executed prior to train–test splitting. For each numerical column with a high number of distinct values (nunique > 100) and a maximum value exceeding ten times its mean, Z-score-based outlier detection was applied. Using a threshold of $|Z| > 7$, rows containing extreme values in any of these columns were flagged as outliers. Any individual value exceeding a threshold of $|Z| > 7$ was explicitly replaced with NaN. A conservative threshold of seven standard deviations was deliberately chosen to isolate only extreme, highly anomalous network artefacts (e.g.,, src_bytes) while deliberately avoiding the removal of rare but legitimate attack signatures that often feature high-volume network metrics. Rows containing one or more of these NaN values were subsequently dropped.

This outlier filtering step eliminated exactly 59,793 records, reducing the dataset from 4,898,431 to 4,838,638 instances. A class-wise audit of these removed rows confirmed that the outliers were predominantly from the majority classes with 39,791 belonging to the attack class and 20,002 to normal traffic, closely matching the original class proportions. Within the attack subcategories, the retained dataset still contained substantial counts of rare types: 40,226 Probe, 11,029 R2L, and 509 U2R instances, confirming that the aggressive column-selection rule and elevated threshold did not disproportionately remove the minority attack classes critical for training a generalised intrusion detector.

3.4. Stage 2: Strategic 90% dataset reduction

A 90% reduction was achieved through stratified random sampling applied to each category as shown in Figure 2, based on the work of Fares *et al.* [24]. This preserves the proportional representation of each attack category while aggressively reducing absolute count. This level of reduction deliberately tests the limits of how lightweight an IDS can be: reducing storage from 740 MB to 74 MB, reducing training cycles down to 12.5× quicker, and probing the minimum viable dataset size. Implementation was again with scikit-learn train_test_split, specifying the stratify parameter. This stage alone accomplishes the 90% data reduction, leaving 483,863 records, as shown in Figure 2.

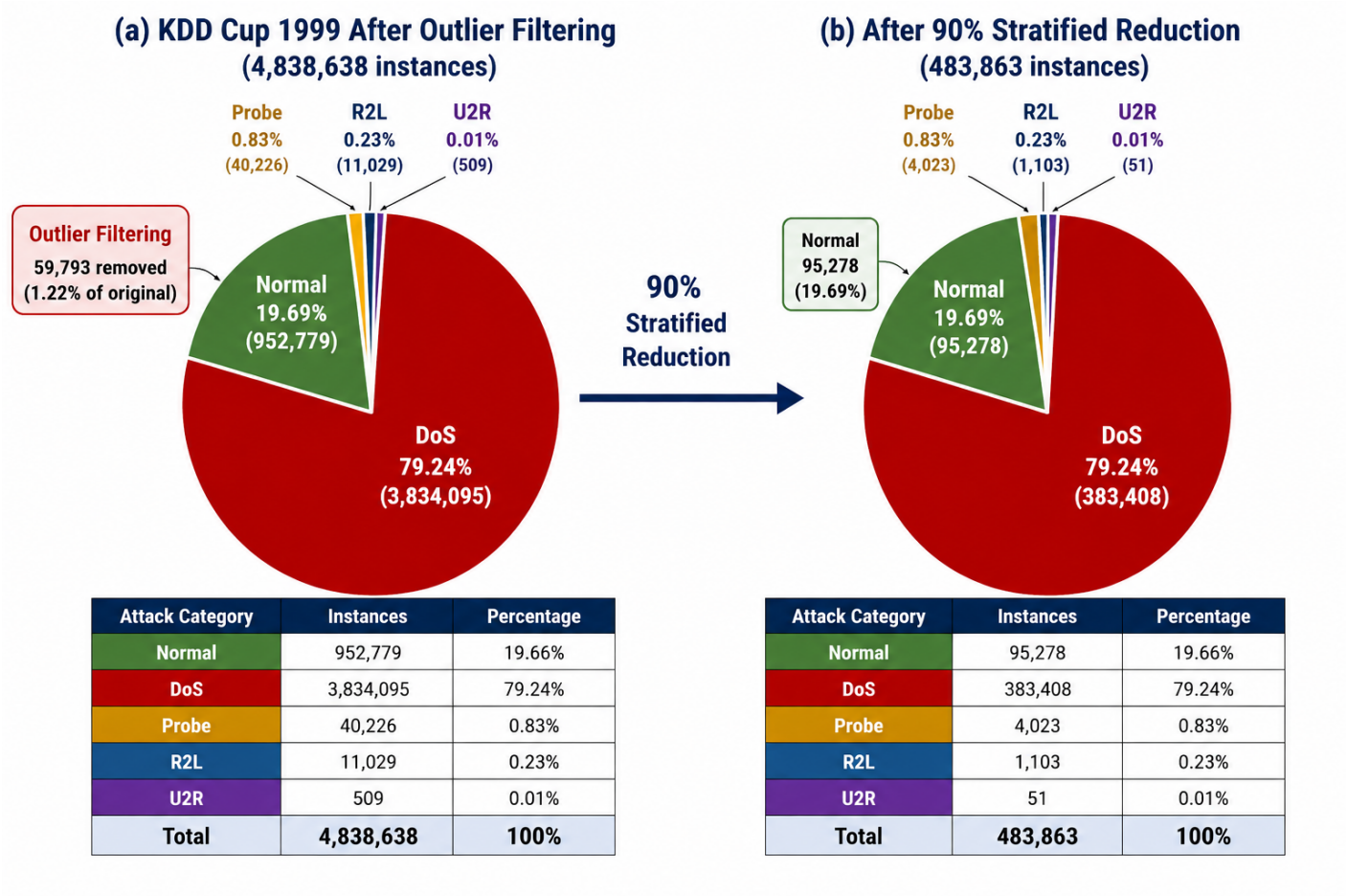
3.5. Stage 3: Statistical normalization and dimensionality reduction

After the empirical determination of the optimal class-balancing configuration, the feature space underwent stringent statistical normalization using StandardScaler for all continuous variables, giving a mean of zero and variance of one using statistical values computed solely on the training distribution. To reduce multicollinearity among features and enhance computational efficiency in later stages of processing, a correlation-based feature selection method was performed. Specifically, all pairs of features that exhibit high linear dependence (i.e.,, Pearson's correlation coefficient > 0.99) were identified, and for each pair of redundant features, the feature with the lower variance was systematically removed. This process eliminated six highly correlated attributes (srv_serror_rate, srv_rerror_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, and dst_host_srv_rerror_rate).

Targeted feature cleaning resulted in a 15% reduction in dimensionality, compressing the feature space from 40 to 34 features without removing any essential diagnostic values. Importantly, to prevent potential data leakage and maintain methodological rigour during benchmark-validation assessment, the scaler used to normalize the data was fit only on the training dataset; the transformation parameters were applied unchanged to the unseen test dataset as a simulation of terra-firma data ingestion in the real world.

3.6. Stage 4: Systematic balancing technique comparison and selection

For the intra-dataset evaluation, this reduced dataset was partitioned using an 80/20 stratified split, yielding 387,090 training records and 96,773 test records. To address the methodological shortcomings related to the ad hoc choice of resampling methods in much of the current literature on intrusion detection, this study performed a systematic comparative analysis of five different data balancing techniques. To prevent data leakage, all data balancing techniques were applied exclusively to the 387,090 training records. The test partition of 96,773 records retained its original, unmodified class distribution to ensure an unbiased evaluation of model generalization. The purpose of this comparison was to objectively identify the most effective setup for security-critical applications where minimizing false negatives (missed attacks) is critical. The five evaluated data balancing techniques utilized both synthetic data generation techniques, and decision-boundary clarification techniques, summarized as follows:



Attack Category	Instances	Percentage
Normal	952,779	19.66%
DoS	3,834,095	79.24%
Probe	40,226	0.83%
R2L	11,029	0.23%
U2R	509	0.01%
Total	**4,838,638**	**100%**

Attack Category	Instances	Percentage
Normal	95,278	19.66%
DoS	383,408	79.24%
Probe	4,023	0.83%
R2L	1,103	0.23%
U2R	51	0.01%
Total	**483,863**	**100%**

Figure 2: Strategic 90% dataset reduction.

i. ADASYN (Adaptive Synthetic Sampling). An adaptive oversampling method related to SMOTE (Synthetic Minority Oversampling Technique), ADASYN uses an adaptive process (based on the density of the data points) to create synthetic instances in proportion to the number of existing instances in an area that has a high degree of difficulty in predicting attacks.
ii. Tomek links. A technique that produces a targeted form of undersampling, Tomek Links generates subsets of the original dataset by identifying and removing instances from the majority class that lie in close proximity (nearest neighbor) to instances from the minority class and thus reside in ambiguous regions where identifying the appropriate decision boundary is unclear.
iii. SMOTE-Tomek. SMOTETomek is a synergistic hybrid that integrates the heuristically driven oversampling characteristics of SMOTE with the precision boundary clarification capabilities of Tomek Links.
iv. ADASYN-Tomek. ADASYNTomek is a complex hybrid that integrates the density-based oversampling characteristics of ADASYN with the algorithmic undersampling characteristics of Tomek links.
v. Control (Baseline). The control (baseline) configuration represents an unmodified version of the dataset as it was originally obtained by allowing the distribution of the dataset to remain imbalanced and serves as the baseline for evaluating the effectiveness of the four remaining methods.

A Random Forest ensemble classifier was used in this study (i.e., Random Forest models) to capture complex, non-linear attack signatures through a fully parallelized execution across all available CPU cores and with a fixed random seed (to ensure reproducibility). Each Random Forest model was built using 10 estimators with Gini impurity as the criterion for determining the quality of split and no limit on the maximum allowable tree depth. The evaluation of the effectiveness of the various balancing techniques was carried out against an explicit training/testing methodology whereby a Random Forest model was trained on 80% of the data, yielding 387,090 training records and tested on 96,773 test records. Various metrics that were used to evaluate the performance of the Random Forest models included: overall accuracy; training time; confusion matrix results (focusing on the false positive rate (i.e., FP) and false negative rate (i.e., FN)); precision; recall; F1-score.

Training and prediction times were measured as elapsed wall-clock time during parallelized execution

Table 1: Experimental configuration and hyperparameter settings.

Category	Parameter / Component	Specification / Setting
Hardware	Compute Environment	Google Colab Standard Instance
	CPU / Memory	Intel Xeon @ 2.30 GHz (4 vCPUs) / 16 GB RAM (No GPU)
Software	Core Libraries	Python, scikit-learn 1.3.2, imbalanced-learn 0.12.0, pandas 2.0.3
Train-Test Splits	90% Strategic Reduction	test_size=0.9, stratify=y (483,863 training records)
	Intra-Dataset Evaluation	test_size=0.2, stratify=y (387,090 train / 96,773 test)
	Benchmark Transfer Dataset Validation	Train: 483,863 (Full reduced KDD'99) / Test: 22,544 (NSL-KDD Test+)
Preprocessing	Outlier Filtering	Z-score (scipy.stats.zscore),
	Feature Selection	Pearson correlation;
	Scaling (Continuous)	StandardScaler
	Categorical Encoding	LabelEncoder; unseen test categories mapped to -1
Class Balancing	Application	Applied strictly to training partition post-split
	SMOTETomek	SMOTETomek(sampling_strategy='auto', random_state=42)
	ADASYN / Tomek	ADASYN(random_state=42) / TomekLinks(sampling_strategy='auto')
Model Hyperparams	Intra-Dataset RF	n_estimators=10, random_state=10
	transfer validation Dataset RF	n_estimators=100, random_state=42
Validation	Random Seeds	Fixed seeds (10 and 42) for deterministic reproducibility
	Statistical Testing	Single-run deterministic evaluation (no repeated k-fold CV)

3.7. Stage 5: Random Forest training and benchmark-transfer validation

After preprocessing the data in four stages (empirical class balancing, statistical normalization, feature selection, and dimensional reduction), the filtered dataset was used to train a Random Forest model to classify data into two classes. The number of trees in the model was fine-tuned to find a balance between accuracy and computational load. Based on an investigation of the relationship of classification accuracy to the number of trees in the RF model with an interval range of independent variable samples between 10 and 500, the ensemble size was capped at 100 trees because the model accuracy decreased with additional trees and the time to compute the classification increases significantly with additional trees. Node splitting was decided using the Gini impurity criterion with no limitations on the maximum tree depth so that complex and non-linear attack semantics could be captured by the model.

To demonstrate the practical applicability of this architecture for use in low-resource environments, all training and inference procedures were run on low-specification commodity hardware (an Intel Xeon 4-core CPU at 2.3 GHz and 16 GB RAM), with full multi-core parallelization being used. Finally, to evaluate the model's operational generalization ability, we ran a benchmark transfer validation using the NSL-KDD Test+ dataset as a completely withheld evaluation cohort from the KDD'99 dataset. While training on KDD'99 and testing on NSL-KDD provides a benchmark transfer validation, it is not a fully independent real-world validation, as NSL-KDD is historically derived from the same DARPA dataset. Nevertheless, this evaluation rigorously tests the model's generalization gap. Therefore, this scenario represents a very realistic simulation of a zero-day deployment in an evolving network.

3.8. Experimental configuration

To ensure reproducibility and methodological transparency, the full experimental configuration is detailed in Table 1. All experiments were executed on a commodity-hardware environment (Intel Xeon 4-core CPU, 16 GB RAM) without GPU acceleration, aligning with the study's focus on resource-constrained deployment. The data preparation pipeline employed StandardScaler applied uniformly to continuous features across all balancing techniques to ensure a fair comparison; for the final benchmark-transfer validation, the scaler was fit exclusively on the training data. Categorical features (protocol_type, service, and flag) were ordinal-encoded using LabelEncoder before scaling and balancing. This strict partitioning—where scalers, encoders, and balancing algorithms are fit only on the training set and subsequently applied to the test set—was intended to prevent data leakage. Following scaling, the SMOTETomek balancing technique was applied to the combined numeric space (scaled continuous features and ordinal-encoded categorical features).

Furthermore, the Random Forest classifier was configured in two stages: a lightweight 10-estimator model was used to rapidly evaluate the five balancing techniques intra-dataset, while a more robust 100-estimator model was utilized for the final transfer

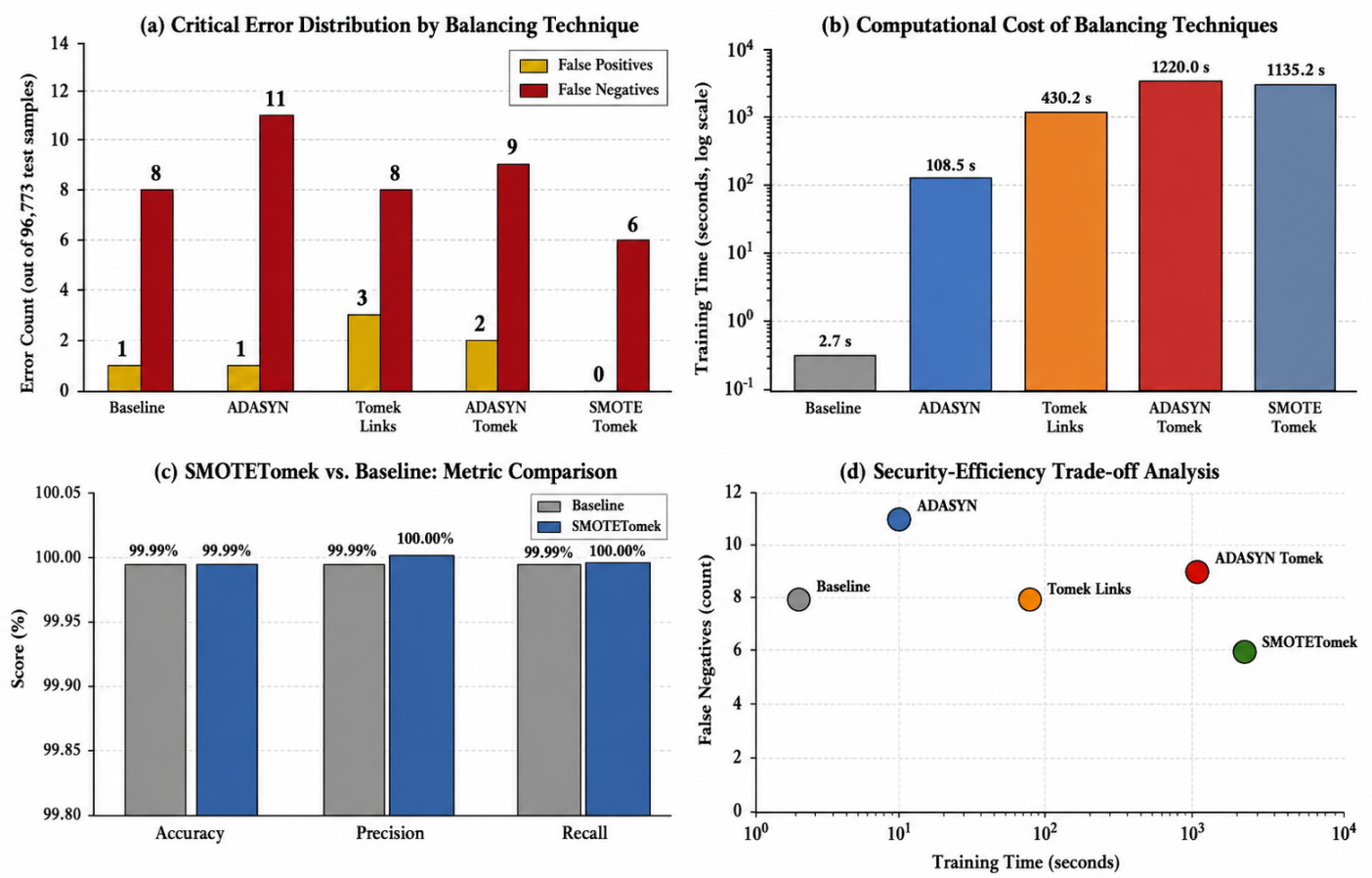


Figure 3: Comparative performance of class-balancing techniques on the reduced KDD Cup 1999 dataset.

validation to maximize generalization performance. To ensure deterministic reproducibility, fixed random seeds (10 and 42) were embedded at all stochastic stages, including stratified sampling, synthetic data generation, and forest bootstrapping.

4. Findings and results

4.1. Comparative evaluation of balancing techniques

Systematic comparison of the five balancing configurations on the reduced KDD'99 dataset showed significant performance differences that are not captured by aggregate accuracy measures. As shown in Figure 3, all techniques achieved 99.99% aggregate accuracy; however, their error distributions (specifically false negative rates) varied significantly and therefore have different implications for security-critical deployment.

A detailed examination of each technique's resulting confusion matrix determined that SMOTETomek was the best-performing technique, as shown in Figure 4. SMOTETomek provided the lowest false-negative count (FN=6) and was able to correctly classify all 78,067 attack instances present in the data sample; its confusion matrix produced 78,069 classified attacks and only 6 classified as normal regarding attack instance detection. The other configurations also produced false negatives: Tomek Links (FN = 8), ADASYNTomek (FN = 9), ADASYN (FN = 11), and the unmodified baseline (FN = 8). For critical-infrastructure protection where a single missed intrusion may have catastrophic consequences through cascading failures of systems such as power grids, financial networks, or healthcare registries, the elimination of false negatives is important for reliable operation and risk reduction.

The expected algorithmic trade-offs were confirmed through analysis of computational efficiency as depicted in Figure 3. Of the five configurations, the unmodified Baseline had the shortest training time at 2.66 seconds, whereas ADASYN alone required 108.55 seconds. In contrast, both of the hybrid methods (SMOTETomek and ADASYNTomek) required comparatively long training times of 1,135.21 and 1,220.03 seconds, respectively. This means that the SMOTETomek training time, approximately 427 times that of the baseline, may be operationally acceptable. The approximately 18-minute training latency for the SMOTETomek is suitable for routine weekly or daily retraining cycles that are standard for enterprise-level intrusion detection systems. The false-positive rate (FPR) is strictly 0.0% (0 false positives out of 18,698 normal test samples). The model produced only 6 false negatives out of 96,773

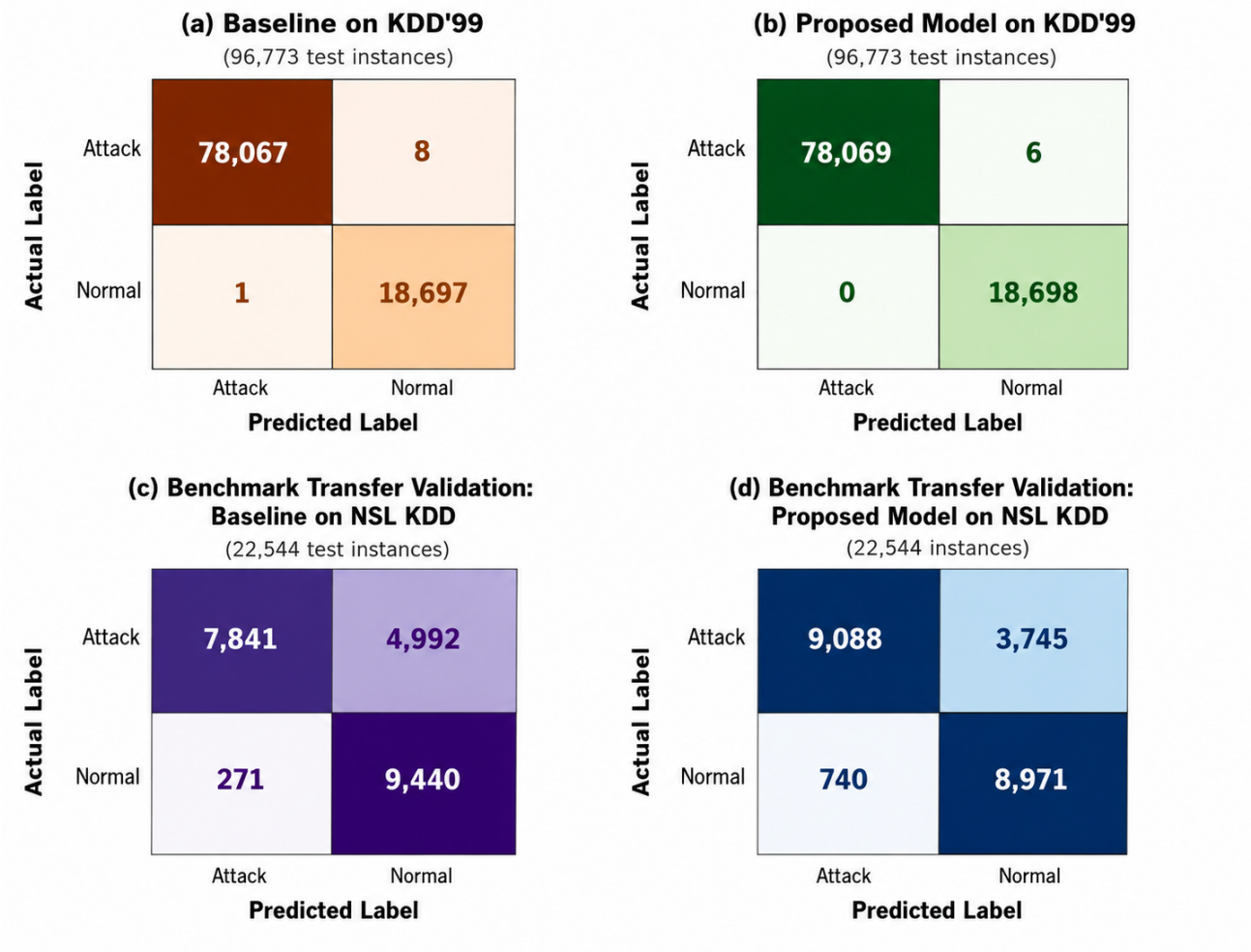


Figure 4: Confusion-matrix analysis.

total test samples (yielding a missed-attack rate of ~0.0077%). Thus, SMOTETomek was used for the benchmark transfer validation experiments as the selected balancing technique.

The lower false-negative count for SMOTETomek justifies the use of its computational demands relative to ADASYNTomek, as utilized in Salihu *et al.* [18], while fulfilling the reliability criteria associated with high-stakes networks.

4.2. Benchmark-transfer validation performance

The operational generalization of the KDD'99 dataset processed with SMOTETomek preprocessing was evaluated using a Random Forest classifier trained solely on the significantly reduced KDD'99 dataset (with approximately 483,863 samples, a 90% volumetric reduction) and tested against the NSL-KDD Test+ dataset (22,544 samples), which contains fundamentally divergent statistical distributions. This setup was designed to reflect an evolving threat environment, or conditions under which an intrusion detection system (IDS) may be deployed. To further isolate the direct impacts on improved predictive performance and evaluate whether improvements were attributable to differences in methodology, a set of comparative performance metrics from the KDD'99 dataset (preprocessed with minimal preprocessing and trained on the entire, unreduced KDD'99 dataset) are presented in Table 2.

As shown in Table 2, the task is explicitly framed as a binary classification problem (Normal vs. Attack). While the baseline model achieved slightly higher Attack Precision (0.97 vs. 0.92), it suffered from a severe class-imbalance bias, missing 4,992 actual attacks (False Negative Rate = 0.3889). The SMOTETomek-optimized model corrected this structural bias, reducing the False Negative Rate to 0.2918 and detecting 1,247 more attacks than the baseline. This improved detection capability is reflected in the superior Macro-F1 (0.80), Weighted-F1 (0.80), and Matthews Correlation Coefficient (0.6311). Although this trade-off slightly increased the False Positive Rate (from 0.0279 to 0.0762), this is an acceptable operational trade-off for security-critical infrastructure where the cost of a missed intrusion far exceeds the cost of a false alarm. The experimental evidence validates the idea that the combined pipeline of an optimally reduced dataset with appropriate preprocessing outperforms the full data with minimal preprocessing in terms of predictive performance.

Table 2: Benchmark-transfer validation performance comparison on the NSL-KDD Test+ dataset.

Metric	Optimized Architecture (Reduced + SMOTETomek)	Baseline (Full Dataset, Minimal Preprocessing)	Improvement
Accuracy	0.8011 (80.11%)	0.7665 (76.65%)	+3.46%
Balanced Accuracy	0.8150 (81.50%)	0.7900 (79.00%)	+2.50%
MCC	0.6311	0.6016	+0.0295
Macro-Precision	0.815	0.810	+0.005
Macro-Recall	0.815	0.790	+0.025
Macro-F1	0.800 (0.80)	0.770 (0.77)	+0.030
Weighted-Precision	0.830 (0.83)	0.830 (0.83)	0.000
Weighted-Recall	0.801 (0.80)	0.767 (0.77)	+0.034
Weighted-F1	0.800 (0.80)	0.760 (0.76)	+0.040
False Positive Rate (FPR)	0.0762	0.0279	+0.0483
False Negative Rate (FNR)	0.2918	0.3889	-0.0971
Attack Precision	0.92	0.97	-0.05
Attack Recall (Detection Rate)	0.71	0.61	+0.10
Attack F1-Score	0.80	0.75	+0.05
Normal Precision	0.71	0.65	+0.06
Normal Recall	0.92	0.97	-0.05
Normal F1-Score	0.80	0.78	+0.02

The enhanced architecture achieved a benchmark-transfer validation accuracy of 80.11% and a Matthews Correlation Coefficient (MCC) rate of 0.6311, both of which were significantly better than the full-dataset baseline, which had an accuracy rate of 76.65% and an MCC of 0.6016, as further depicted in Figure 5. A clear analysis of the confusion matrices associated with both the enhanced and baseline models illustrates the limitations of the baseline approach, as shown in Figure 4.

Although the baseline model used the entire training set, it was strongly biased toward majority-class examples, producing an extremely high recall for normal traffic (97%) at the expense of threat detection. Therefore, it failed to detect 4,992 actual attacks, resulting in only 61% attack recall. The SMOTETomek-enhanced model overcame the structural imbalance associated with the baseline approach. By focusing on developing clear decision boundaries instead of relying on the full dataset size, the SMOTETomek-enhanced model detected 1,247 more attacks than the baseline, producing a 71% attack recall while still achieving a 92% precision. Furthermore, the F1 scores for the two classes (Attack and Normal) of the SMOTETomek-enhanced model exhibited similar performance (with both F1 scores calculated as 80%), which is indicative of a more balanced classifier.

4.3. Computational efficiency analysis

Computational efficiency analysis indicates strong viability for high-speed model inference, though it does not constitute full end-to-end real-time IDS deployment [25]. The classification inference step on the static 22,544-instance NSL-KDD feature matrix required 0.1005 seconds of wall-clock time, yielding a model-inference throughput of 224,388 instances per second. Importantly that this metric measures only classifier inference on pre-encoded input features. It does not represent the full end-to-end latency of a deployed real-time IDS, as it excludes critical pipeline components such as packet/log ingestion, raw feature extraction, categorical encoding, data scaling, I/O operations, and concurrency overhead—all of which contribute substantially to system latency in a production setting.

To accurately account for the computational profile of the proposed pipeline, the training phases must be distinctly separated and contextualized by the hardware used. All experiments were executed on a single 4-core CPU at 2.30 GHz and 16 GB of RAM. During the intra-dataset balancing comparison (using a 10-estimator Random Forest), the SMOTETomek resampling phase required 1,135.21 seconds. For the final cross-dataset validation model (utilizing a 100-estimator Random Forest), the subsequent Random Forest fit time was 187 seconds.

This rapid fit time allows for weekly model retraining on the latest organizational logs without requiring overnight batch processing. The total memory requirements remained modest, with peak memory usage for inference below 2 GB and trained model storage below 156 MB (serialized via joblib). These specifications allow for deployment on low-cost hardware, such as an AWS EC2 t3.medium instance (2 vCPUs, 4 GB memory) or a repurposed commodity desktop computer (4-core i5 processor, 16 GB memory), thereby reducing hardware-cost barriers for resource-constrained institutions in Nigeria. The viability of deploying such lightweight security solutions on resource-constrained edge devices is increasingly recognized in contemporary cybersecurity re-

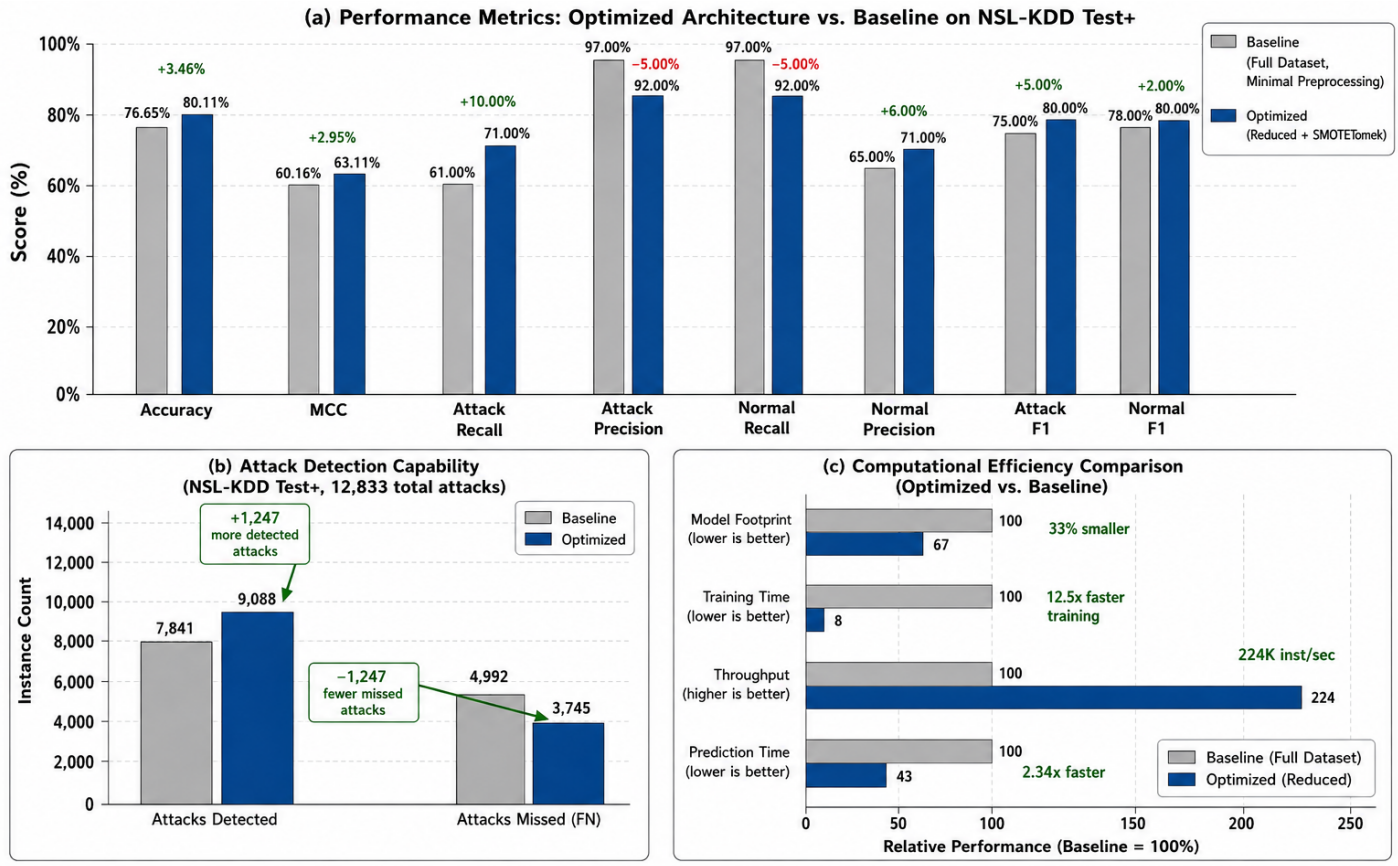


Figure 5: Benchmark-transfer validation: (a) performance metrics, (b) attack-detection capability, and (c) computational efficiency.

search [21], which emphasizes the importance of stack protection and efficient computational resource utilization in edge computing environments.

4.4. Comparative analysis with existing works

Table 3 synthesizes our findings with four related studies across the key comparison dimensions.

The pattern emerging from this comparative analysis in Table 3 is clear. First, among studies that employ the same benchmark transfer validation protocol, Guleria *et al.* [26] and the current study both report cross-dataset accuracies of approximately 80% (80.026% and 80.11%, respectively) with a difference of 0.084 percentage points. This observation indicates that our specific methodology achieves consistent transfer performance relative to a same-protocol external evaluation, rather than suggesting a universal performance ceiling. Second, intradataset validation can yield substantially higher performance estimates compared to transfer evaluation, a gap that practitioners must consider when interpreting published accuracy claims. Studies utilizing intra-dataset validation on KDD'99 (e.g.,, Salihu *et al.* [18] and our proposed method) consistently report high accuracies above 99% (99.67 and 99.99% respectively), which reflects the optimistic performance estimates produced when models are trained and tested on data sharing identical statistical distributions. Third, more complex algorithmic refinements (Wu *et al.*'s [17] similarity matrix correction; Zhang *et al.*'s [27] Bayesian optimization) do not necessarily yield better generalization than systematic preprocessing (our SMOTETomek balancing and 90% reduction).

Fourth, and most importantly for resource-constrained environments, our study provides a quantified demonstration that 224,388 instances per second throughput on commodity hardware is achievable without GPU acceleration—a finding that has direct implications for cybersecurity capacity building in developing regions where expensive computing infrastructure is not available.

4.5. Independent pipeline validation on UNSW-NB15 and SCADA/ICS datasets

To verify that the proposed pipeline design generalizes beyond the legacy KDD family, the same pipeline design (outlier filtering, scaling, SMOTETomek, and Random Forest) was independently trained and tested on two contemporary datasets as shown in Table 4: UNSW-NB15 (a modern network intrusion dataset) [28] and a Modbus-based supervisory control and data acquisition/industrial control system (SCADA/ICS) dataset (SCADANet) [29]. Because these datasets share no feature schema with KDD'99, the saved KDD model could not be transferred; instead, the pipeline design itself was applied from scratch to each dataset.

Table 3: Comparative summary against related literature.

Study	Method	Evaluation Type	Accuracy (NSL-KDD/KDD'99)	Strength
Zhang *et al.* [27]	BO-KNN-Bagging + Bayesian optimization	Intradataset (NSL-KDD)	82.48% (NSL-KDD)	Higher intradataset accuracy than our cross-dataset result
Wu *et al.* [17]	K-means+SMOTE + Random Forest	Train-test split (NSL-KDD)	99.72% train / 78.47% test	Sophisticated post-processing (similarity matrix correction)
Salihu *et al.* [18]	ADASYNTomek + Random Forest	Intradataset (KDD'99)	99.67% (KDD'99 balanced)	Demonstrated efficacy of hybrid ADASYNTomek
Guleria *et al.* [26]	IDS-LAD (logical analysis)	benchmark transfer validation (train KDD'99, test NSL-KDD)	80.026% / 99.872%	Near-identical benchmark transfer dataset validation accuracy to our study
Current Study	90% reduction + SMOTETomek + Random Forest	benchmark transfer validation (train KDD'99, test NSL-KDD)	80.11% / 99.99%	0.0077% FNR on KDD'99; quantified throughput (224,388 instances/sec); commodity hardware deployment; Result exceeds the listed benchmark-transfer accuracy.

Table 4: Independent pipeline validation across intrusion-detection domains.

Dataset	Model	Accuracy	Balanced accuracy	MCC	Macro F1	FPR	FNR
UNSW-NB15	Baseline	0.949146	0.944207	0.888213	0.944107	0.072223	0.039363
UNSW-NB15	SMOTETomek	0.946629	0.947676	0.884894	0.942039	0.048839	0.055808
SCADA/ICS	Baseline	0.999632	0.998793	0.998550	0.999275	0.002402	0.000011
SCADA/ICS	SMOTETomek	0.999773	0.999867	0.999109	0.999554	0.000000	0.000267

For SCADANet, preprocessing involved dropping ten identifier columns, filling missing values with zero, and label-encoding 24 categorical variables. Z-score outlier filtering on ten numeric columns removed 5,576 rows (1.04%), followed by dropping five features during feature selection. For UNSW-NB15, 9,413 outliers were similarly filtered from its 257,673 raw instances.

On the UNSW-NB15 dataset, the SMOTETomek+RF pipeline achieved 94.66% accuracy, an MCC of 0.8849, a false positive rate of 4.88%, and a false negative rate of 5.58%. On the SCADA/ICS dataset, the pipeline achieved 99.98% accuracy, with an MCC of 0.9991, a 0.0% false positive rate, and a 0.026% false negative rate. This independent validation provides preliminary evidence that the methodology, rather than the specific trained weights, generalizes across intrusion-detection domains, including one ICS-adjacent case.

4.6. Discussion

The 80.11% benchmark-transfer validation accuracy obtained when transferring a KDD'99-trained system to NSL-KDD provides a more realistic benchmark for operational use than the higher > 99% results reported using intra-dataset evaluations. Our results closely match those of Dhanabal & Shantharajah [30], who reported similar results for intra-dataset evaluations, and of Guleria *et al.* [26] through their work with an accuracy of 80.026% on IDS LADS. Therefore, the results provide a measure of the practical maximum accuracy that can be achieved when transferring from KDD to NSL-KDD, while highlighting the optimism of intra-dataset estimates. For reference, Zhang *et al.* [27] reported their best results on NSL-KDD using their BO-KNN-Bagging approach with Bayesian optimization, achieving an accuracy of 82.48%, also using intra-dataset evaluation. Wu *et al.* [17] obtained 99.72% accuracy on the training dataset but only 78.47% on the test set, providing a 21.25% accuracy drop compared to the 19.88% drop in our evaluation, indicating that developing and implementing complex algorithmic processes does not guarantee an increased generalization of system performance compared to standard methods of processing datasets.

Finally, Salihu *et al.* [18] achieved 99.67% accuracy using an ADASYN-based ADASYNTomek approach based on an intra-dataset evaluation as compared to 99.99% achieved in KDD'99 using SMOTETomek. The ability to minimize missed attacks on critical systems justifies the increase in training time (427-fold) between the baseline training time (2.66 seconds) and the training

time necessary to complete the evaluation of 1,135 seconds, since a single missed critical attack can have catastrophic consequences. In a situation where we reduced our dataset to 10% of its original size (using stratified sampling) with the application of five-stage preprocessing pipeline, our evaluation achieved an accuracy of 80.11% for benchmark transfer validation on NSL-KDD whereas using our entire dataset as the baseline with minimal preprocessing pipeline provided an accuracy of 76.65%; this demonstrates that the elimination of redundancy allows the model to learn general patterns of attack and not just memorize noise and provides regularization of the model due to limited copies of attacks.

Our lightweight model operates at a rate of 224,388 records per second on average (i.e.,, using a 16 GB, four-core CPU), generating throughputs suitable for real-time implementation in resource-constrained environments, while occupying less than 2 GB of memory and requiring only 156 MB of storage. Together, these comparative findings show that strategic dataset reduction, hybrid SMOTETomek balancing, and rigorous cross-dataset validation produce generalizable, low-cost intrusion detection suitable for protecting critical infrastructure in developing regions.

Furthermore, the KDD'99-to-NSL-KDD evaluation constitutes benchmark-to-benchmark transfer validation of a single trained model. Independently, the same pipeline design was retrained and retested on UNSW-NB15 and a Modbus-based SCADA/ICS dataset, achieving 94.66% and 99.98% accuracy, respectively. This provides evidence that the methodology generalizes across intrusion-detection domains, though it remains short of validation on live, operational critical-infrastructure traffic.

5. Conclusion

This study addressed the three research questions through systematic experimentation and validation. Research Question 1: Stratified sampling with a 90% dataset reduction retained an 80.11% benchmark-transfer accuracy, supporting the feasibility of reduced-data IDS development without large-scale computing resources. Research Question 2: Systematic application of five different balancing techniques yields results supporting the use of SMOTETomek as the best-performing for false-negative minimization on the reduced-KDD holdout. Research Question 3: Effective application of preprocessing measures on a consecutive basis improved performance relative to the minimally preprocessed baseline, with a total of 3.46% absolute improvement over the minimally preprocessed baseline (80.11% versus 76.65%).

The research offers four primary contributions. First, the study challenges the assumption that adding more data necessarily produces a better model, instead establishing that the intelligent preprocessing of data yields better results than brute-force accumulation. Second, this study adds empirical evidence showing that SMOTETomek outperforms ADASYN, Tomek Links, and ADASYNTomek for reducing false negatives in the evaluated dataset. Third, this research establishes a benchmark-transfer accuracy of approximately 80% on NSL-KDD when compared to the inflated claims in single-dataset studies (sometimes in excess of 99%). Finally, it demonstrates that lightweight intrusion detection systems can be implemented on commodity hardware with low-cost cloud instances by providing operational capability of at least 224,388 instances per second, 156 MB for persistent storage, and less than 2 GB of memory.

These findings may reduce financial barriers for resource-constrained Nigerian educational institutions and defence networks attempting to implement effective intrusion detection to protect critical infrastructure operations using repurposed desktop and low-cost cloud instance computers. Moreover, the five-stage preprocessing pipeline is reproducible, flexible, and suitable for weekly retraining. To further validate the proposed model on truly independent and contemporary data, the research evaluated this pipeline on modern datasets such as UNSW-NB15 and SCADA/ICS networks, which represent more contemporary threat landscapes and produced strong performance. A notable limitation is that the SCADA/ICS validation used a single, Modbus-based dataset rather than a physical-process testbed such as SWaT or WADI; because ICS attacks often manifest as gradual sensor drift rather than discrete network anomalies, results here should not be read as equivalent to physical-process-level anomaly detection. Future research should validate this pipeline on live, operational critical-infrastructure traffic.

Data availability

Data will be made available upon request from the corresponding author.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this manuscript.

Funding

This research was funded by the Tertiary Education Trust Fund (TETFund), Nigeria, under the Year 2024-2025 (Merged) TETFund intervention in Research Project for Institution-Based Research.

References

[1] A. Kumar & J. A. Gutierrez, "Impact of machine learning on intrusion detection systems for the protection of critical infrastructure", Information **16** (2025) 515. https://doi.org/10.3390/info16070515.

[2] R. M. Lee, M. J. Assante & T. Conway, "Analysis of the cyber attack on the Ukrainian power grid: Defense use case", E-ISAC/SANS ICS, 2016. Available online: https://dn760107.eu.archive.org/0/items/lee-et.-al.-e-isac-sans-ukraine-duc-5/Lee%2C%20et.%20al.%2C%20E-ISAC_SANS_Ukraine_DUC_5.pdf.

[3] A. Greenberg, "The untold story of NotPetya, the most devastating cyberattack in history", Wired Magazine (2018). Available online: https://www.wired.com/story/notpetya-cyberattack-ukraine-russia-code-crashed-the-world/.

[4] M. M. Yamin, M. Ullah, H. Ullah & B. Katt, "Weaponized AI for cyber attacks", Journal of Information Security and Applications **57** (2021) 102722. https://doi.org/10.1016/j.jisa.2020.102722.

[5] R. Sommer & V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection", 2010 IEEE Symposium on Security and Privacy, 2010, pp. 305–316. https://doi.org/10.1109/SP.2010.25.

[6] V. M. González-Gorrín & J. Prieto-Blazquez, "A hybrid machine learning intrusion detection method for metamorphic malware", International Journal of Information Security **25** (2026) 63. https://doi.org/10.1007/s10207-026-01234-0.

[7] V. Z. Mohale & I. C. Obagbuwa, "Evaluating machine learning-based intrusion detection systems with explainable AI: Enhancing transparency and interpretability", Frontiers in Computer Science **7** (2025) 1520741. https://doi.org/10.3389/fcomp.2025.1520741.

[8] A. Y. Alnahari & N. A. Ahmad, "How existing machine learning models for DDoS detection differ in performance and accuracy when applied to synthetic versus real-world network traffic datasets", Open International Journal of Informatics **13** (2025) 105. https://doi.org/10.11113/oiji2025.13n2.349.

[9] R. Younisse, A. Saif, N. Al-Madi, S. Almajali & B. Mahafzah, "Improving IoT security: The impact of dimensionality and size reduction on intrusion-detection performance", Jordanian Journal of Computers and Information Technology **11** (2025) 351. https://doi.org/10.5455/jjcit.71-1739734998.

[10] V. Hordieieva, I. Czarnowski & P. Jędrzejowicz, "A comparative study of oversampling techniques for imbalanced network attack detection", Procedia Computer Science **270** (2025) 822. https://doi.org/10.1016/j.procs.2025.09.202.

[11] L. D. Manocchio, S. Layeghy, M. Gallagher & M. Portmann, "An empirical evaluation of preprocessing methods for machine learning based network intrusion detection systems", Engineering Applications of Artificial Intelligence **158** (2025) 111289. https://doi.org/10.1016/j.engappai.2025.111289.

[12] V. R. Joshi, K. Assa-Agyei, T. Al-Hadhrami & S. N. Qasem, "Hybrid AI intrusion detection: Balancing accuracy and efficiency", Sensors **25** (2025) 7564. https://doi.org/10.3390/s25247564.

[13] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro & K. Rieck, "Dos and don'ts of machine learning in computer security", 31st USENIX Security Symposium (USENIX Security 22), Boston, MA, USA, 2022, pp. 3971–3988. Available online: https://www.usenix.org/conference/usenixsecurity22/presentation/arp.

[14] K. Harahsheh, R. Al-Naimat & C.-H. Chen, "Using feature selection enhancement to evaluate attack detection in the internet of things environment", Electronics **13** (2024) 1678. https://doi.org/10.3390/electronics13091678.

[15] M. Tavallaee, E. Bagheri, W. Lu & A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set", 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–6. https://doi.org/10.1109/CISDA.2009.5356528.

[16] M. Y. Rahamathulla & M. Ramaiah, "Building robust lightweight network intrusion detection: A multi-pronged approach with DSSTE and deep learning for securing edge-enabled industrial IoT applications", Cluster Computing **28** (2025) 515. https://doi.org/10.1007/s10586-025-05171-w.

[17] T. Wu, H. Fan, H. Zhu, C. You, H. Zhou & X. Huang, "Intrusion detection system combined enhanced random forest with SMOTE algorithm", EURASIP Journal on Advances in Signal Processing **2022** (2022) 39. https://doi.org/10.1186/s13634-022-00871-6.

[18] N. I. Salihu, M. N. Musa & A. J. Olalekan, "A combined approach of ADASYN and Tomeklink for anomaly network intrusion detection system using some selected machine learning algorithms", International Journal of Web Research **7** (2024) 51. https://doi.org/10.22133/ijwr.2024.486081.1245.

[19] A. I. Bassey, M. A. Agana, E. A. Edim & O. Njama-Abang, "Intrusion detection in a controlled computer network environment using hybridized random forest and long short-term memory algorithms", African Scientific Reports **4** (2025) 327. https://doi.org/10.46481/asr.2025.4.3.327.

[20] N. Islam, F. Farhin, I. Sultana, M. S. Kaiser, M. S. Rahman, M. Mahmud, A. S. M. S. Hosen & G. H. Cho, "Towards machine learning based intrusion detection in IoT networks", Computers, Materials & Continua **69** (2021) 1801. https://doi.org/10.32604/cmc.2021.018466.

[21] J. U. Undiandeye & M. A. Agana, "Towards strengthening edge computing security through stack protection mechanisms", African Scientific Reports **5** (2026) 436. https://doi.org/10.46481/asr.2026.5.2.436.

[22] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis & P. K. Chan, KDD Cup 1999 Data [Dataset], UCI Machine Learning Repository, 1999. https://doi.org/10.24432/C51C7N.

[23] M. Iwanowski, D. Olszewski, W. Graniszewski, J. Krupski & F. Pelc, "The choice of training data and the generalizability of machine learning models for network intrusion detection systems", Applied Sciences **15** (2025) 8466. https://doi.org/10.3390/app15158466.

[24] A. H. Fares, M. I. Sharawy & H. H. Zayed, "Intrusion detection: Supervised machine learning", Journal of Computing Science and Engineering **5** (2011) 305. https://doi.org/10.5626/JCSE.2011.5.4.305.

[25] P. W. Adi, R. Kusumaningrum, N. R. Saputra, S. H. Setiawan & M. A. Amrustian, "Lightweight and accurate deep learning for anomaly-based IDS using data-driven redundancy reduction", 2025 4th International Conference on Digital Transformation and Applications (ICDXA), Semarang, Indonesia, 2025, pp. 144–149. https://doi.org/10.1109/ICDXA69105.2025.11329716.

[26] B. Guleria, M. Deshmukh & R. Nautiyal, "IDS-LAD: Intrusion detection system using logical analysis of data", Cluster Computing **28** (2025) 988. https://doi.org/10.1007/s10586-025-05724-z.

[27] Z. Zhang, S. Kong, T. Xiao & A. Yang, "A network intrusion detection method based on bagging ensemble", Symmetry **16** (2024) 850. https://doi.org/10.3390/sym16070850.

[28] N. Moustafa & J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)", 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 2015, pp. 1–6. https://doi.org/10.1109/MilCIS.2015.7348942.

[29] E. Algul, F. Doğan, A. A. Ahmad & O. Polat, "SCADANet: A novel dataset for SCADA cybersecurity and intrusion detection", Computer Networks **278** (2026) 112087. https://doi.org/10.1016/j.comnet.2026.112087.

[30] L. Dhanabal & S. P. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms", International Journal of Advanced Research in Computer and Communication Engineering **4** (2015) 446. https://ijarcce.com/wp-content/uploads/2015/june-15/IJARCCE%2096.pdf.