Available online at website: https://jurnal.iaii.or.id/index.php/RESTI

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 10 No. 5 (2026) 1412 - 1422 e-ISSN: 2580-0760

HyRes-Boost: Hybrid Resampling for Sensitivity–False-Alarm Trade-offs in Network Intrusion Detection

Achmad Affandi[1], Eko Setijadi[2], Endroyono[3], Sri Rahayu[4], Prasetiyono Hari Mukti[5], Suwadi[6], Fransiska Sisilia Mukti[7*]
[1,2,3,4,5,6]Department of Electrical Engineering, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
[7]Department of Informatics, Faculty of Technology and Design, Institut Teknologi dan Bisnis Asia, Malang, Indonesia
[*]*ms.frans@asia.ac.id*

Abstract

Class imbalance in network intrusion detection systems (NIDS) creates a trade-off between intrusion sensitivity and false-alarm control. While hybrid resampling methods often improve minority class representation, their effectiveness varies depending on imbalance severity and operational needs. This study presents HyRes-Boost, a Generate-then-Clean hybrid resampling framework that integrates Adaptive Synthetic Sampling (ADASYN), Edited Nearest Neighbours (ENN), and XGBoost to examine how post-generation editing affects the sensitivity and false-alarm balance. HyRes-Boost is evaluated on UNSW-NB15, CICIDS2018, and CICIoT2023 using a leakage-safe protocol with predictive metrics, operational indicators, asymmetric error costs, and fold-level statistical analysis. Results indicate that HyRes-Boost produces context-dependent changes in operating points rather than universal improvements. In the attack-minority UNSW-NB15 scenario, HyRes-Boost achieves a recall of 99.74%, increasing intrusion sensitivity but also raising the false-alarm rate. In CICIoT2023, where benign traffic is the minority class, HyRes-Boost lowers the false alarm rate to 2.23% from 13.72% in the unbalanced baseline, though attack recall drops to 98.75%. For the highly separable CICIDS2018 dataset, additional resampling yields minimal improvement, with the F1-score remaining near 99.90%. These results suggest that hybrid resampling should not be evaluated solely by aggregate accuracy or ranking. The suitability of HyRes-Boost depends on class prevalence, boundary ambiguity, and the relative costs of false positives and false negatives.

Keywords: cost-sensitive evaluation; false alarm rate; hybrid resampling; imbalanced learning; intrusion detection system; XGBoost

How to Cite: A. . Affandi, "HyRes-Boost: Hybrid Resampling for Sensitivity–False-Alarm Trade-offs in Network Intrusion Detection", *J. RESTI (Rekayasa Sist. Teknol. Inf.)* , vol. 10, no. 5, pp. 1412–1422, Oct. 2026, doi: 10.29207/resti.v10i5.7579.
Permalink/DOI: https://doi.org/10.29207/resti.v10i5.7579

Received: February 12, 2026
Accepted: September 19, 2026
Available Online: October 12, 2026

This is an open-access article under the CC BY 4.0 License
Published by Ikatan Ahli Informatika Indonesia

1. Introduction

Network Intrusion Detection Systems (NIDS) must detect malicious traffic with high sensitivity while maintaining manageable false alarm rates. This balance is essential in Security Operations Centers (SOCs), where even a low false-alarm rate can generate significant alert volumes, increase analyst workload, and delay genuine threat investigations [1]. Conversely, suppressing too many false alarms may result in missed intrusions. NIDS performance should therefore be assessed not only by overall accuracy, but also by the balance between attack sensitivity and false-positive errors under actual traffic conditions [2], [3].

Class imbalance complicates this balance because models trained on skewed distributions may not adequately represent less frequent traffic patterns [4], [5]. The impact of imbalance also depends on its severity, classifier behavior, and resampling configuration [6-8]. Learning-level approaches adjust the influence or cost of classification errors, while data-level methods modify the training distribution through oversampling, undersampling, or both [4]. SMOTE increases minority representation by synthetic interpolation [9], and ADASYN focuses generation on minority observations that are harder to learn [10], [11]. While these strategies can improve minority recognition, altering the training distribution may also create ambiguous decision regions. As a result, higher

recall or accuracy does not always lead to better precision or false-alarm control.

Hybrid resampling combines data generation with cleaning to refine the training distribution. SMOTE-ENN and similar methods apply neighborhood-based editing after class balancing [12], [13], while recent frameworks integrate oversampling, undersampling, noise removal, and ensemble learning for imbalanced network traffic [8], [14], [15]. Sequence-aware designs may also apply noise handling before adaptive generation [16]. These approaches demonstrate that both the sampling method and its order in the process affect the data presented to the classifier. When generation and cleaning are evaluated only as a complete pipeline, their individual effects can become conflated. Changes in recall or false-alarm rate may then be attributed to the hybrid method overall, even if driven mainly by one stage. This can obscure the reasons for changes in detector behavior and may lead to method selection based solely on aggregate performance, without understanding the error trade-offs introduced by each intervention.

Interpretation is also influenced by the experimental environment. IDS performance and the effects of imbalance handling vary with dataset characteristics, imbalance conditions, and learning algorithms [7], [15], [17]. Hybrid-sampling studies report different outcomes across network datasets [8], [18]. Evaluating only a single benchmark can make behavior specific to one traffic distribution appear to be a general property of the resampling strategy. In this study, UNSW-NB15 [19], CICIDS2018 [20], and CICIoT2023 [21] are treated as distinct experimental regimes rather than as additional benchmarks. Their traffic compositions and class prevalence differ: attacks are the minority in UNSW-NB15 and CICIDS2018, while benign traffic is the minority in CICIoT2023. These differences enable examination of the same resampling intervention under attack-minority, benign-minority, and varying baseline-separability conditions, revealing whether its effect is stable or context dependent.

Based on this rationale, this study introduces HyRes-Boost as a controlled Generate-then-Clean framework. Here, ADASYN-based adaptive minority generation is followed by post-generation neighborhood editing using Edited Nearest Neighbours (ENN) before XGBoost classification. XGBoost remains the fixed primary classifier to minimize classifier-tuning effects when assessing changes from resampling [22]. The novelty is in the controlled interpretation of this ordered intervention, not in introducing new forms of ADASYN, ENN, or XGBoost. Comparisons among no balancing, ADASYN-only, ENN-only, and the full HyRes-Boost sequence distinguish changes from adaptive generation and those from post-generation editing. Applying the same sequence across diverse traffic regimes further reveals whether the resulting recall–false-alarm shift is consistent or varies with minority-class identity and baseline classification conditions.

Accordingly, this study offers a sequence-level, cross-regime, and operational evaluation of hybrid resampling for imbalanced NIDS. The analysis clarifies how generation and post-generation editing affect the detector's operating point, reducing the risk of mistaking dataset-specific behavior for a general method property. Conventional predictive metrics are supplemented by recall–FAR analysis, fold-level statistical testing, asymmetric false-positive/false-negative cost analysis, and classifier-robustness checks. This evaluation addresses not only whether hybrid resampling changes predictive performance, but also how those changes occur, whether they persist across different traffic regimes, and whether the resulting sensitivity–false-alarm trade-off is operationally suitable. HyRes-Boost is therefore presented as a controlled framework for interpreting context-dependent resampling behavior, not as a universally superior intrusion-detection solution.

2. Methods

HyRes-Boost uses a controlled Generate-then-Clean framework to assess how sequential changes to the training distribution impact the sensitivity and false-alarm rates of a network intrusion detector. The process involves adaptive minority generation with ADASYN, neighborhood-based editing with ENN, and classification using XGBoost. The primary focus is on the effect of ordered resampling interventions on classifier behavior, not on altering the underlying sampling or classification algorithms. As shown in Figure 1, preprocessing, resampling, and model training are conducted independently within each outer cross-validation training fold. The held-out fold is transformed using preprocessing parameters from the training data and is never resampled. This approach prevents synthetic or removed observations from affecting evaluation data.

The Generate-then-Clean sequence applies neighborhood editing after adaptive minority generation, allowing ENN to operate on the augmented distribution instead of the original training data. ADASYN assigns synthetic minority observations to regions that are more difficult to learn, as defined by Equation 1 [10].

$$r_i = \frac{\Delta_i}{K}, \quad \hat{r}_i = \frac{r_i}{\sum_{j=1}^{N_m} r_j}, \quad g_i = \text{round}(\hat{r}_i G) \tag{1}$$

In Equation 1, i indexes a minority-class observation in the current training fold, and r_i is its local difficulty ratio. Δ_i is the number of majority-class observations among the K nearest neighbors of the i-th minority observation, where K is the ADASYN neighborhood size. The normalized difficulty weight, $\hat{r}_i$, is calculated by dividing r_i by the sum of all local difficulty ratios r_j across the N_m minority observations. Here, j is the

summation index and N_m is the number of minority-class observations in the training fold. G is the total synthetic-sample budget set by the sampling strategy, and g_i is the number of synthetic observations assigned to the local region of the i-th minority observation. The operator $\mathrm{round}(\cdot)$ converts this allocation to an integer. In all experiments, $K = 5$, sampling_strategy = auto, and random_state = 42. The minority class is determined by the class distribution in each training fold, not by a predefined semantic label.

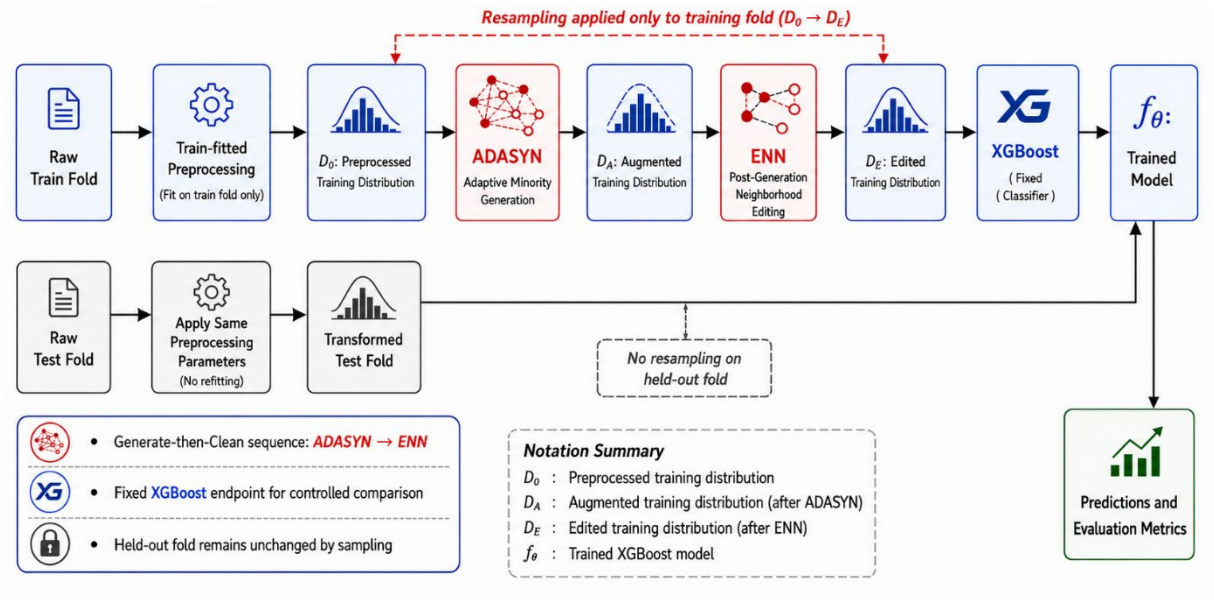


Figure 1. Overview of the HyRes-Boost framework

Following adaptive generation, the augmented distribution is edited using the standard ENN consistency criterion defined in Equation 2 [12].

$$(\mathrm{x}_i, y_i) \in D_E \quad \text{if} \quad y_j = y_i, \ \forall j \in \mathcal{N}_3(\mathbf{x}_i) \tag{2}$$

Here, x_i denotes the feature vector of the i-th targeted observation, y_i its class label, and $\mathcal{N}_3(\mathrm{x}_i)$ the set of its three nearest neighbors under the ENN rule. The symbol y_j denotes the label of the j-th neighbor. Equation 2 shows that a targeted observation remains in the edited distribution D_E only if all three neighbors have the same class label. The innovation is not in the ENN rule, but in applying neighborhood editing to the distribution generated adaptively.

Here, x_i denotes the feature vector of the i-th targeted observation, y_i denotes its class label, and the pair (x_i, y_i) represents the corresponding labeled observation. The symbol D_E represents the edited distribution after ENN, while $\mathcal{N}_3(\mathrm{x}_i)$ denotes the set of three nearest neighbors of x_i under the selected ENN configuration. The index j refers to an observation within this neighborhood, y_j denotes the class label of the j-th neighbor, and $\forall$ indicates that the consistency condition must hold for every neighbor $j \in \mathcal{N}_3(\mathrm{x}_i)$. With n_neighbors = 3, kind_sel = all, and sampling_strategy = auto, a targeted observation is retained in D_E only when all three neighboring labels agree with y_i. Applying this criterion after ADASYN allows neighborhood editing to act directly on local structures produced or modified during synthetic generation.

HyRes-Boost is evaluated as an intervention on classifier operating behavior, rather than as a method expected to improve all predictive metrics at once. For a resampling configuration s, the principal operating point is defined by Equation 3.

$$\mathcal{O}_s = (R_s, F_s) \tag{3}$$

Here, s identifies the evaluated resampling configuration, $\mathcal{O}_s$ denotes its two-dimensional operating point, R_s denotes the attack recall achieved under configuration s, and F_s denotes the corresponding false alarm rate (FAR). Because attack traffic is treated as the positive class throughout the experiments, R_s represents intrusion-detection sensitivity, whereas F_s represents the proportion of benign observations incorrectly classified as attacks.

The incremental effect of post-generation neighborhood editing is quantified relative to adaptive generation alone using Equation 4.

$$\Delta\mathcal{O}_{\mathrm{edit}} = \mathcal{O}_{\mathrm{HyRes}} - \mathcal{O}_{\mathrm{ADASYN}} = (\Delta R, \Delta F) \tag{4}$$

$\Delta\mathcal{O}_{\mathrm{edit}}$ denotes the change in operating point attributable to adding ENN after ADASYN. The term $\mathcal{O}_{\mathrm{HyRes}}$ denotes the operating point obtained from the complete HyRes-Boost sequence, whereas $\mathcal{O}_{\mathrm{ADASYN}}$ denotes the operating point obtained from the ADASYN-only condition. The terms ΔR and ΔF represent the corresponding changes in recall and FAR, respectively. A positive ΔR indicates increased intrusion-detection sensitivity, whereas a negative ΔF indicates improved false-alarm control. This formulation distinguishes an operating-point shift from

a general improvement in predictive performance, since an increase in recall may occur together with an increase in FAR and vice versa.

To isolate the effects of the resampling sequence, XGBoost is used as the primary classifier across all main experimental conditions. The primary ablation compares no balancing, ADASYN only, ENN only, and the full HyRes-Boost sequence, all under the same XGBoost configuration. This approach minimizes the risk that observed differences result from sampler-specific classifier optimization rather than changes in the training distribution. Additional resampling strategies are included for broader benchmarking. Random Forest and Convolutional Neural Network (CNN) models serve as secondary robustness checks to assess whether the resampling effect is unique to XGBoost or consistent across other classifier architectures.

2.1 Dataset

The experimental evaluation uses three publicly available network-security datasets: UNSW-NB15 [19], CICIDS2018 [20], and CICIoT2023 [21]. Each dataset applies the same controlled Generate-then-Clean sequence to varying traffic compositions and class-prevalence conditions. Table 1 summarizes the observations used in the cross-validation experiments.

Table 1. Experimental sample size and class distribution

Datasets	Total Samples	Benign (%)	Attack (%)
UNSW-NB15	2,540,044	87.35	12.65
CIC-IDS2018	1,048,575	72.71	27.29
CIC-IoT2023	3,197,250	2.35	97.65

In all datasets, attack traffic is designated as the positive class. Recall measures the proportion of attacks correctly detected, while FAR indicates the proportion of benign observations incorrectly classified as attacks. The datasets differ in class imbalance: attack traffic is the minority in UNSW-NB15 and CICIDS2018, but benign traffic is the minority in CICIoT2023. This variation enables evaluation of the Generate-then-Clean intervention under different minority-class conditions and helps determine whether its impact on sensitivity and false-alarm rates changes with the traffic regime.

2.2 Data Preprocessing

Preprocessing is conducted independently within each outer cross-validation fold to prevent information leakage. Once the raw data are split into training and held-out test partitions, preprocessing parameters are estimated only from the training partition. Numerical features are imputed with the median from the training fold and standardized using StandardScaler. Categorical features are imputed with the most frequent category in the training fold and transformed using OrdinalEncoder; categories found only in the held-out partition are assigned a value of −1. These steps are implemented with a ColumnTransformer, which is fitted on the training partition and then applied unchanged to the held-out partition.

Identifier-related variables that could reveal session identity or reduce cross-flow generalizability are removed deterministically before model training. For UNSW-NB15, the excluded variables are attack_cat, source and destination IP addresses, source and destination ports, and Stime/Ltime. For CICIDS2018, Timestamp, Flow ID, source and destination IP addresses, source and destination ports, and Protocol are removed. No additional deterministic feature-removal rule is applied to CICIoT2023.

2.3 Experimental Setup

All experiments use stratified 10-fold cross-validation with shuffling enabled and random_state set to 42. In each outer fold, approximately 90% of observations are assigned to the training partition and 10% to the held-out test partition, with class proportions maintained. Preprocessing is fitted only on the training partition, and all resampling is applied exclusively to the transformed training data. The held-out partition is not oversampled or undersampled and is used solely for evaluation within its fold. This approach ensures that information from the evaluation partition does not influence preprocessing or synthetic sample generation. A common random seed is set for Python, NumPy, PyTorch, CUDA, XGBoost, Random Forest, and supported stochastic resampling procedures. For the CNN robustness experiment, an internal stratified validation subset is created from the outer training fold for early stopping. The outer held-out partition is not used for early stopping, classifier selection, or model optimization.

Table 2. Resampling and Classifier Configurations

Component	Role	Configuration
ADASYN	Adaptive minority generation	sampling_strategy = auto; n_neighbors = 5; random_state = 42
ENN	Post-generation neighborhood editing	sampling_strategy = auto; n_neighbors = 3; kind_sel = all
XGBoost	Primary classifier	n_estimators = 300; max_depth = 6; learning_rate = 0.05; subsample = 0.90; colsample_bytree = 0.90; reg_lambda = 1.0; reg_alpha = 0; min_child_weight = 1; objective = binary:logistic; eval_metric = logloss; tree_method = hist
Random Forest	Classifier-robustness check	n_estimators = 300; max_depth = None; min_samples_split = 2; min_samples_leaf = 1; class_weight = None
CNN	Classifier-robustness check	Conv1D layers: 32 and 64 filters; Batch Normalization; ReLU; Max Pooling; fully connected layers: 256 and 128 units; dropout = 0.30; sigmoid output; Adam optimizer; learning_rate = 0.001; binary cross-entropy loss; batch_size = 4096

Resampling and classifier configurations are predefined and applied consistently across all datasets and experimental conditions. XGBoost is the primary classifier for controlled resampling analysis, while Random Forest and CNN are used as secondary robustness checks. Table 2 summarizes the complete resampling and classifier settings.

2.4 Evaluation Protocol and Reproducibility

Predictive performance is evaluated using Accuracy, Precision, Recall, F1-score, Balanced Accuracy, Matthews Correlation Coefficient (MCC), ROC-AUC, AUC-PR, and False Alarm Rate (FAR). Attack traffic is always considered the positive class. Recall measures intrusion-detection sensitivity, and FAR indicates the proportion of benign traffic incorrectly classified as malicious. Balanced Accuracy and MCC address class imbalance, while AUC-PR offers a ranking-based assessment that is useful when class frequencies differ [23], [24]. Equation 5 uses the asymmetric error-cost formulation to evaluate the operational consequences of false-positive and false-negative errors.

$$C_r = FP + rFN,\ r \in \{5,10,20\} \quad (5)$$

Here, C_r is the total error cost for penalty ratio r. FP is the number of false positives, or benign observations incorrectly classified as attacks, and FN is the number of false negatives, or attacks missed by the detector. The parameter r indicates the penalty assigned to one false negative relative to one false positive, with values of 5, 10, and 20, reflecting increasing consequences of missed intrusions. Costs are normalized per 1,000 held-out flows and represent operational sensitivity scenarios, not monetary values. This approach enables assessment of the same recall–FAR operating point under different priorities for false positives and false negatives.

Statistical validation uses matched results from the 10 outer cross-validation folds. For each dataset and evaluation metric, the Friedman test is applied across all 14 resampling configurations to assess overall differences, with Kendall's W reported as the effect-size measure. If the omnibus test indicates differences at $\alpha = 0.05$, HyRes-Boost is compared with relevant baseline configurations using paired Wilcoxon signed-rank tests, followed by Holm adjustment to control the family-wise error rate [25], [26]. Here, α is fixed at 0.05 and Kendall's W represents the concordance-based effect size for the Friedman comparison. Statistical evidence is considered alongside the direction of observed metric changes, so statistical significance is not automatically interpreted as an operationally favorable outcome.

Computational performance is measured as decision-engine prediction latency on transformed held-out folds. Each latency measurement includes three warm-up passes and 10 timed repetitions, with CUDA synchronization for GPU execution. Latency is reported as the mean prediction time in microseconds per sample. XGBoost and Random Forest are evaluated on the complete transformed test array, while CNN inference uses batches of 4,096 observations. A classification threshold of 0.5 is applied throughout. The timing boundary includes only classifier inference and excludes preprocessing, resampling, feature transformation, and model training. Therefore, the reported values represent decision-engine latency, not end-to-end packet-to-alert latency.

Experiments are conducted on a system with an NVIDIA RTX 1000 GPU and 64 GB DDR5 RAM, using Python 3.10, XGBoost 2.0, PyTorch, and imbalanced-learn. XGBoost and CNN use CUDA when available, while Random Forest runs on the CPU. Run-specific details, including operating platform, processor, CPU resources, memory, library versions, CUDA availability, PyTorch CUDA version, GPU model, and GPU count, are recorded in environment.json. This documentation links hardware-sensitive measurements to the execution environment and supports reproducibility of computational results.

3. Results and Discussions

3.1 Feature Space Transformation Analysis

Figure 2 illustrates the evolution of the feature space following adaptive generation and neighborhood editing. ADASYN enhances representation in challenging minority regions, though it may also increase sample density in overlapping areas. ENN then refines the distribution by removing locally inconsistent observations, which alters the neighborhood structure established after adaptive generation.

The ablation results in Table 3 clarify the role of each component. ADASYN drives the primary shift in minority-class representation, increasing attack sensitivity in UNSW-NB15 and significantly reducing false alarms in the benign-minority CICIoT2023 dataset. ENN offers a secondary adjustment, with its impact depending on the local structure formed after adaptive generation. In CICIoT2023, ENN reduces FAR by about 14% compared to ADASYN alone. In UNSW-NB15, ENN slightly increases recall but also raises FAR. In CICIDS2018, both methods yield only minor changes since baseline performance is nearly saturated. These results show that post-generation editing does not have a uniform effect; its contribution depends on how the augmented distribution interacts with the underlying class structure.

Figure 2 demonstrates the distributional changes from the Generate-then-Clean sequence, while Table 3 links these changes to classification outcomes. Adaptive generation sets the main direction of the operating-point shift, and ENN offers a smaller post-generation adjustment. This adjustment should be evaluated based on the resulting balance between sensitivity and false alarms, rather than as an isolated improvement in a single metric.

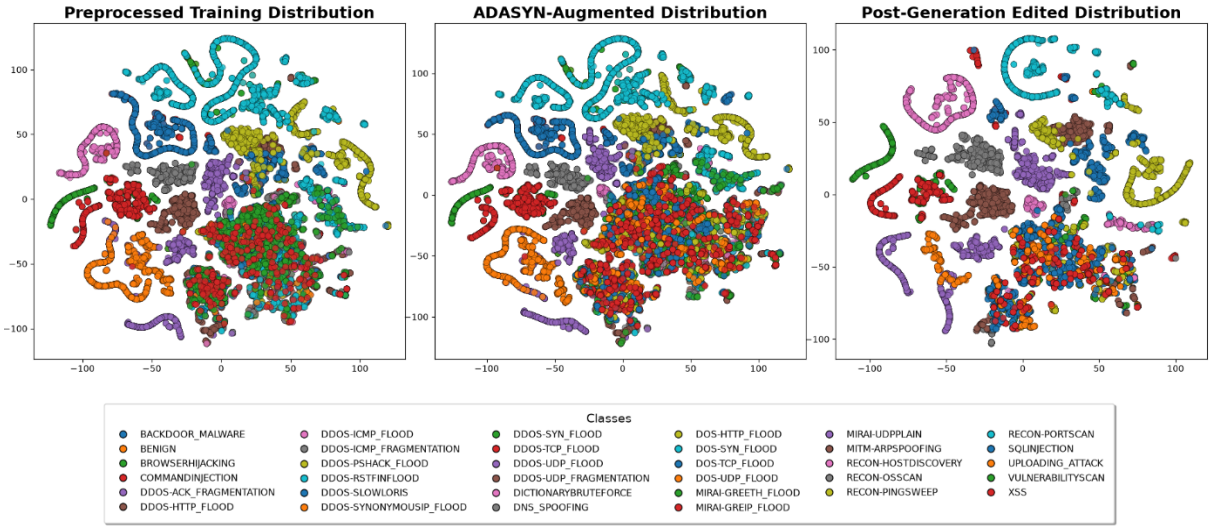


Figure 2. t-SNE visualization of the training distribution before resampling, after ADASYN-based adaptive generation, and after post-generation ENN editing (CICIoT2023 case study)

Table 3. Sequential Impact of Framework Components (Mean ± SD over 10 folds)

Dataset	Configuration	Precision	Recall	F1-Score	FAR	Bal. Acc	MCC	AUC-PR	Latency (µs)
UNSW-NB15	No Balancing	0.9681 ± 0.0018	0.9606 ± 0.0024	0.9644 ± 0.0011	0.00179 ± 0.00011	0.9794 ± 0.0012	0.9624 ± 0.0012	0.99648 ± 0.00021	0.164 ± 0.007
	ADASYN Only	0.9018 ± 0.0031	0.9942 ± 0.0008	0.9457 ± 0.0015	0.00612 ± 0.00022	0.9940 ± 0.0003	0.9438 ± 0.0015	0.99444 ± 0.00026	0.162 ± 0.006
	ENN Only	0.9100 ± 0.0032	0.9933 ± 0.0007	0.9498 ± 0.0018	0.00556 ± 0.00021	0.9939 ± 0.0004	0.9479 ± 0.0019	0.99510 ± 0.00038	0.157 ± 0.006
	HyRes-Boost	0.8893 ± 0.0117	0.9974 ± 0.0027	0.9402 ± 0.0054	0.00704 ± 0.00085	0.9952 ± 0.0010	0.9384 ± 0.0053	0.99167 ± 0.00263	0.160 ± 0.008
CIC-IDS2018	No Balancing	0.9993 ± 0.0002	0.9997 ± 0.0001	0.9995 ± 0.0001	0.00025 ± 0.00006	0.9997 ± 0.0000	0.9994 ± 0.0001	0.99999 ± 0.00000	0.453 ± 0.025
	ADASYN Only	0.9981 ± 0.0003	1.0000 ± 0.0000	0.9990 ± 0.0002	0.00072 ± 0.00011	0.9996 ± 0.0001	0.9987 ± 0.0002	0.99994 ± 0.00001	0.453 ± 0.010
	ENN Only	0.9992 ± 0.0002	0.9998 ± 0.0001	0.9995 ± 0.0001	0.00030 ± 0.00006	0.9998 ± 0.0000	0.9993 ± 0.0001	0.99999 ± 0.00000	0.419 ± 0.016
	HyRes-Boost	0.9981 ± 0.0003	1.0000 ± 0.0000	0.9990 ± 0.0002	0.00073 ± 0.00012	0.9996 ± 0.0001	0.9986 ± 0.0002	0.99994 ± 0.00002	0.446 ± 0.007
CIC-IoT2023	No Balancing	0.9967 ± 0.0001	0.9957 ± 0.0001	0.9962 ± 0.0001	0.13717 ± 0.00482	0.9293 ± 0.0024	0.8417 ± 0.0034	0.99995 ± 0.00000	0.283 ± 0.019
	ADASYN Only	0.9994 ± 0.0001	0.9880 ± 0.0002	0.9936 ± 0.0001	0.02610 ± 0.00217	0.9809 ± 0.0011	0.7973 ± 0.0019	0.99993 ± 0.00000	0.276 ± 0.032
	ENN Only	0.9990 ± 0.0001	0.9912 ± 0.0002	0.9951 ± 0.0001	0.04069 ± 0.00240	0.9752 ± 0.0012	0.8287 ± 0.0025	0.99995 ± 0.00000	0.227 ± 0.047
	HyRes-Boost	0.9995 ± 0.0002	0.9875 ± 0.0010	0.9935 ± 0.0004	0.02234 ± 0.00916	0.9826 ± 0.0041	0.7944 ± 0.0059	0.99994 ± 0.00000	0.231 ± 0.045

Note: Latency excludes preprocessing and is the per-sample prediction time after 3 warm-up passes and 10 measurement repetitions.

3.2 Generate-then-Clean Mechanism Validation

The ablation results in Table 3 indicate that the impact of the Generate-then-Clean sequence varies with class distribution and classification complexity. Instead of consistently improving performance across datasets, this sequence shifts the operating point differently based on which class is underrepresented and the degree of class separation.

In UNSW-NB15, where attack traffic is the minority class, HyRes-Boost increases recall by about 3.7 percentage points compared to the no-balancing baseline. However, this improvement comes with a higher FAR. This outcome reflects a sensitivity-oriented trade-off: greater minority coverage enhances attack detection but also increases the likelihood of misclassifying benign traffic as malicious. Post-generation editing does not resolve this trade-off and maintains a higher sensitivity to attacks.

CICIDS2018 demonstrates a different pattern. Baseline performance is nearly saturated without resampling, so further generation and editing yield minimal changes. In this scenario, the slight increase in recall does not offset the rise in FAR. This suggests that class imbalance alone does not justify resampling when the original decision boundary already separates the classes effectively.

For CICIoT2023, where benign traffic is the minority class, the trade-off shifts direction. HyRes-Boost reduces FAR by about 84% compared to the no-balancing baseline, while attack recall drops by less than one percentage point. Resampling improves recognition of rare benign traffic and significantly reduces false alarms, though at the cost of a slight

decrease in attack sensitivity. In this context, the method's primary value is in controlling false alarms rather than maximizing attack recall.

Overall, these results indicate that HyRes-Boost primarily shifts the balance between sensitivity and false alarms, rather than uniformly improving predictive performance. Attack-minority data favor higher sensitivity, benign-minority data benefit from stronger false-alarm control, and highly separable data see little advantage from additional resampling. The practical value of this approach depends on whether the resulting trade-off matches the error priorities of the target environment.

3.3 Classifier Robustness

Table 4 assesses whether the operating behavior seen with HyRes-Boost is unique to XGBoost or remains when the same resampled data are applied to other classifiers. Random Forest and CNN offer contrasting learning architectures, which helps separate the effects of the modified training distribution from those specific to each classifier.

Table 4. Classifier Robustness under the HyRes-Boost Resampling Condition

Dataset	Configuration	Recall	F1-Score	FAR	MCC	Latency (µs)
UNSW-NB15	XGBoost	0.9974	0.9402	0.00704	0.9384	0.160
	Random Forest	0.9919	0.9499	0.00549	0.9481	0.949
	CNN	0.9998	0.9338	0.00802	0.9320	3.490
CIC-IDS2018	XGBoost	1.0000	0.9990	0.00073	0.9986	0.446
	Random Forest	0.9997	0.9987	0.00086	0.9982	1.855
	CNN	0.9994	0.9774	0.01723	0.9691	3.825
CIC-IoT2023	XGBoost	0.9875	0.9935	0.02234	0.7944	0.231
	Random Forest	0.9901	0.9945	0.04359	0.8147	1.486
	CNN	0.9822	0.9909	0.00832	0.7470	3.937

For UNSW-NB15, all three classifiers demonstrate high sensitivity to attack traffic after HyRes-Boost, though the extent of the recall–FAR trade-off varies. The consistent shift indicates that the resampled distribution significantly increases sensitivity, while classifier architecture influences the degree of false alarms.

For CICIoT2023, FAR decreases across all three classifiers, with only minor differences in attack recall. This consistent trend suggests that better representation of the benign minority class reduces false alarms. However, each classifier achieves a different recall–FAR balance, confirming that resampling sets the general direction of change but does not solely determine the final trade-off.

For CICIDS2018, differences are limited because baseline performance is nearly saturated. XGBoost and Random Forest remain stable, while CNN is more influenced by the modified training distribution.

Overall, the direction of the HyRes-Boost effect is consistent across the evaluated algorithms, though final performance depends on the classifier. XGBoost remains the primary classifier due to its balanced sensitivity–FAR profile and the lowest measured decision-engine latency in this environment.

3.4 Performance Benchmarking

Figure 3 shows that HyRes-Boost affects the recall–FAR operating point differently across the three datasets. In UNSW-NB15, the method increases both attack sensitivity and the false-alarm rate. In CICIDS2018, recall remains high, with the main change being an increase in FAR. In contrast, CICIoT2023 experiences a significant reduction in FAR with only a minor decrease in recall. These results highlight that the same resampling strategy can lead to different operational outcomes depending on minority-class conditions.

Table 5 compares these patterns across resampling strategies. On UNSW-NB15, HyRes-Boost increases recall by about 3.7 percentage points over no balancing, but this comes with higher FAR and lower F1-score and MCC. Compared to ROS and RUS, HyRes-Boost achieves similar recall with a slightly lower FAR. The benchmark does not identify a single best configuration; instead, it shows that configurations prioritizing attack sensitivity vary in the false-alarm cost they incur.

For CICIDS2018, recall gains are minimal across resampling methods because baseline detection is nearly saturated. HyRes-Boost and ADASYN increase FAR compared to no balancing, without a meaningful improvement in recall. In this scenario, the trade-off is unfavorable, as additional resampling alters the error profile without significant detection benefit.

The greatest false-alarm reduction is observed in CICIoT2023. HyRes-Boost lowers FAR by about 84% compared to no balancing, with recall decreasing by less than one percentage point. ROS, RUS, and SMOTE-ENN achieve even lower FAR, but with greater recall loss. HyRes-Boost therefore offers a balance between maintaining attack sensitivity and reducing false alarms.

AUC-PR provides further insight, remaining nearly unchanged despite large differences in FAR, especially in CICIoT2023. This suggests resampling mainly affects the final operating point, not the classifier's overall ranking ability. As a result, two configurations may have similar AUC-PR values but lead to significantly different operational outcomes at a fixed classification threshold.

3.5 Statistics, Operational Costs, and Complexity Analysis

Table 6 shows that benchmark differences are reproducible across cross-validation folds. The

strongest effects occur in UNSW-NB15 and CICIoT2023, where Kendall's *W* approaches 1 for the main endpoints. CICIDS2018 exhibits smaller effects, consistent with its near-saturation behavior.

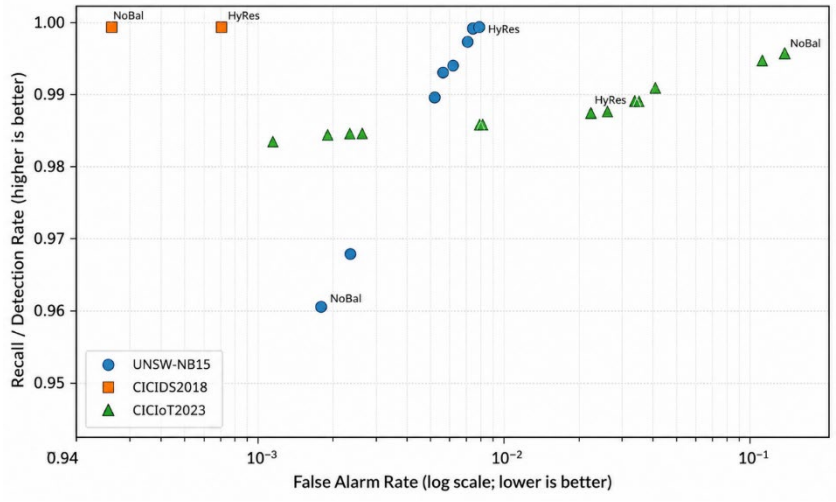


Figure 3. Recall–FAR operating-point shift from no balancing to HyRes-Boost across the three datasets. Other resampling configurations are shown as contextual reference, while NoBal and HyRes denote the primary comparison.

Table 5. Representative XGBoost Resampling Benchmark (Mean Over 10-folds)

Dataset	Method	Accuracy	Recall	F1-Score	FAR	MCC	AUC-PR
UNSW-NB15	No balancing	0.9961	0.9605	0.9643	0.0017	0.9623	0.9964
	ROS	0.9928	0.9997	0.9371	0.0075	0.9354	0.9964
	RUS	0.9925	0.9998	0.9346	0.0079	0.9329	0.9949
	ENN	0.9943	0.9932	0.9498	0.0055	0.9478	0.9951
	ADASYN	0.9938	0.9941	0.9457	0.0061	0.9437	0.9944
	SMOTE-ENN	0.9945	0.9897	0.9511	0.0051	0.9490	0.9956
	HyRes-Boost	**0.9932**	**0.9973**	**0.9402**	**0.0070**	**0.9383**	**0.9916**
CIC-IDS2018	No balancing	0.9997	0.9997	0.9995	0.0002	0.9993	0.9999
	ROS	0.9997	0.9998	0.9995	0.0002	0.9993	0.9999
	RUS	0.9997	0.9998	0.9995	0.0002	0.9993	0.9999
	ENN	0.9997	0.9998	0.9995	0.0002	0.9993	0.9999
	ADASYN	0.9994	0.9999	0.9990	0.0007	0.9986	0.9999
	SMOTE-ENN	0.9997	0.9997	0.9995	0.0002	0.9993	0.9999
	HyRes-Boost	**0.9994**	**0.9999**	**0.9990**	**0.0007**	**0.9986**	**0.9999**
CIC-IoT2023	No balancing	0.9925	0.9957	0.9962	0.1371	0.8417	0.9999
	ROS	0.9851	0.9849	0.9923	0.0026	0.7767	0.9999
	RUS	0.9840	0.9837	0.9917	0.0011	0.7656	0.9999
	ENN	0.9904	0.9911	0.9950	0.0406	0.8287	0.9999
	ADASYN	0.9876	0.9879	0.9936	0.0260	0.7972	0.9999
	SMOTE-ENN	0.9862	0.9861	0.9929	0.0078	0.7868	0.9999
	HyRes-Boost	**0.9873**	**0.9875**	**0.9934**	**0.0223**	**0.7944**	**0.9999**

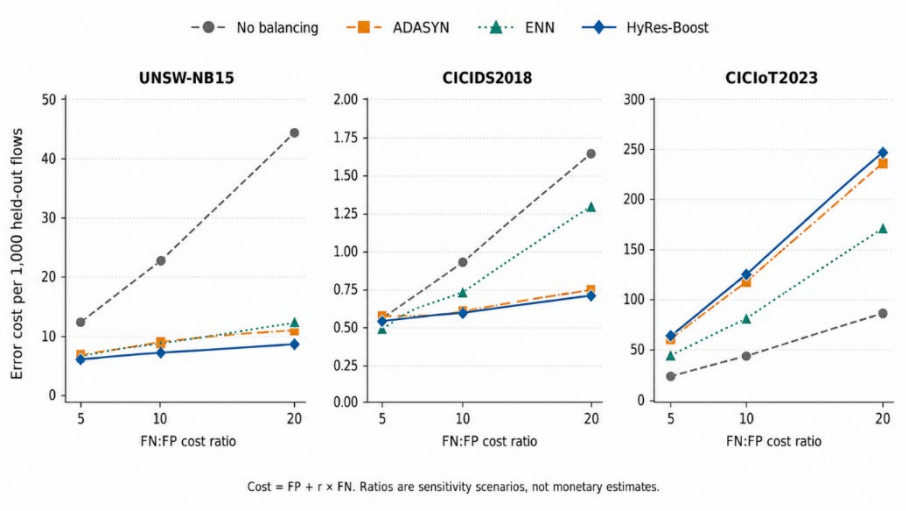


Figure 4. Sensitivity of operational error cost to increasing false-negative penalties across the evaluated resampling configurations. Costs are computed per 1,000 held-out flows for FN:FP ratios of 5, 10, and 20 on UNSW-NB15, CICIDS2018, and CICIoT2023.

The post-hoc results align with the performance patterns described in Section 3.4. Statistical evidence confirms differences among the evaluated operating points; however, statistical significance does not imply that HyRes-Boost is always preferable. Its value depends on the trade-off direction: higher sensitivity in UNSW-NB15 results in increased FAR, CICIDS2018 shows minimal benefit from additional resampling, and CICIoT2023 achieves better false-alarm control at the expense of some attack sensitivity.

Table 6. Statistical Validation across HyRes-Boost

Dataset	Metric	Friedman x^2	p-value	Kendall's W
UNSW-NB15	F1-Score	124.915	$2.14e^{-20}$	0.961
	FAR	126.752	$9.25e^{-21}$	0.975
CIC-IDS2018	F1-Score	78.927	$1.76e^{-11}$	0.607
	FAR	101.393	$8.91e^{-16}$	0.780
CIC-IoT2023	F1-Score	126.923	$8.56e^{-21}$	0.976
	FAR	125.988	$1.31e^{-20}$	0.969

Figure 4 further analyzes how trade-offs shift as false negatives receive higher penalties. In UNSW-NB15, the HyRes-Boost cost curve increases more gradually than the no-balancing baseline as the FN:FP ratio rises. This higher attack sensitivity is especially valuable when missed intrusions have significant operational impact.

CICIDS2018 again shows minimal differences among configurations, highlighting the limited benefit of resampling in a highly separable problem. In contrast, CICIoT2023 displays the opposite pattern: HyRes-Boost achieves lower FAR when the false-negative penalty is modest, but its cost increases more quickly as missed attacks become more costly. Therefore, a small reduction in recall that seems acceptable for false alarms can become significant when false negatives are weighted more heavily.

Tables 5 and 6, along with Figure 4, offer complementary views of the same trade-off. Table 5 shows how predictive metrics vary across resampling strategies, while Table 6 confirms these differences are consistent across folds. Figure 4 illustrates how operational value shifts as FP/FN priorities change. Together, these results clarify why the same HyRes-Boost operating point may be valuable in sensitivity-focused settings, unnecessary when baseline performance is saturated, or less suitable when even minor increases in missed attacks are costly. From a computational standpoint, ADASYN and ENN modify the training distribution but do not add processing stages to the XGBoost prediction path. The similar XGBoost latency values in Table 3 reflect this. These measurements capture only decision-engine latency and exclude preprocessing, flow construction, communication overhead, and model training. As a result, they should not be considered as end-to-end real-time NIDS performance.

4. Conclusions

This study shows that the impact of Generate-then-Clean resampling in network intrusion detection depends on class imbalance, minority-class identity, baseline separability, and the operational error to be minimized. HyRes-Boost produces distinct sensitivity and false-alarm trade-offs across datasets, rather than a uniform performance gain. When attack traffic is the minority, as in UNSW-NB15, the framework increases attack sensitivity but also raises false alarms. When benign traffic is the minority, as in CICIoT2023, it improves false-alarm control with a slight reduction in attack recall. In CICIDS2018, where baseline classification is nearly saturated, further resampling offers little benefit. Ablation analysis shows that ADASYN drives most of the operating-point shift, while ENN provides a smaller adjustment that depends on the remaining data structure. The consistent shift across XGBoost, Random Forest, and CNN suggests that changes in the training distribution largely explain the observed effects, though the final outcome depends on the classifier. Statistical and cost-sensitive analyses confirm that a significant performance difference does not always translate to operational improvement; the value of a configuration depends on the relative importance of false positives and missed attacks. HyRes-Boost's main contribution is not universal superiority, but a controlled framework for separating and interpreting the effects of generation and post-generation editing across diverse traffic regimes. These results support choosing imbalance-handling strategies based on the desired sensitivity and false-alarm balance, rather than a single aggregate metric. Future work should extend this analysis to temporal and streaming traffic, independent datasets, adaptive resampling, probability calibration, and end-to-end deployment.

Declarations

AI Usage Statement

The authors used ChatGPT (OpenAI) solely to assist with language editing and improving the readability of the manuscript during the revision process. All scientific content, experimental design, data analysis, interpretation of the results, and conclusions were reviewed and remain the responsibility of the authors.

Author Contributions

A.A.: conceptualization, methodology, validation, supervision, project administration, writing – original draft, and writing – review and editing; E.S.: methodology, validation, supervision, and writing – review and editing; E.: methodology, validation, supervision, and writing – review and editing; S.R.: investigation, data curation, validation, and writing – review and editing; P.H.M.: software, validation, visualization, and writing – review and editing; S.: resources, validation, supervision, and writing – review and editing; F.S.M.: conceptualization, methodology, investigation, formal analysis, software, data curation, visualization, validation, writing – original draft, and writing – review and editing. All authors reviewed and approved the final manuscript.

Funding Statement

This work was supported by the Directorate of Research and Public Services, Institut Teknologi Sepuluh Nopember (ITS), under Department Research Grant Batch 2 2025, Grant No. 2781/PKS/ITS/2025.

Conflict of Interest Statement

The authors declare no conflict of interest.

Ethical Clearance

Ethical approval was not required for this study because the research was conducted exclusively using publicly available benchmark network traffic datasets and did not involve direct participation, intervention, or experimentation involving human participants or animals.

Data and Code Availability

The datasets used in this study, namely UNSW-NB15, CICIDS2018, and CICIoT2023, are publicly available from their original sources as cited in References [19]–[21]. The experimental code, configuration files, and environment information supporting the findings of this study are available from the corresponding author upon reasonable request.

References

[1] S. Tariq, M. Baruwal Chhetri, S. Nepal, and C. Paris, "Alert Fatigue in Security Operations Centres: Research Challenges and Opportunities," ACM Comput. Surv., vol. 57, no. 9, pp. 1–38, Sep. 2025, doi: https://doi.org/10.1145/3723158 .

[2] F. Sisilia Mukti, E. Setijadi, A. Affandi, A. Basuki, and M. Ali Akbar, "In-Depth Network Traffic Analysis using Machine Learning Perspective: Characterization and Classification," in 2023 6th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), IEEE, Dec. 2023, pp. 415–421. doi: https://doi.org/10.1109/ISRITI60336.2023.10467384 .

[3] F. S. Mukti, A. Basuki, B. A. Pratomo, A. Affandi, H. Studiawan, and E. Setijadi, "SPARCOF: A Two-stage Decision-Support Framework for Efficient Feature Sets in Network Intrusion Detection Datasets," Int. J. Intell. Eng. Syst., vol. 19, no. 10, 2026, doi: https://doi.org/10.22266/ijies2026.1031.17 .

[4] N. Gupta, V. Jindal, and P. Bedi, "CSE-IDS: Using cost-sensitive deep learning and ensemble algorithms to handle class imbalance in network-based intrusion detection systems," Comput. Secur., vol. 112, p. 102499, Jan. 2022, doi: https://doi.org/10.1016/j.cose.2021.102499 .

[5] S. Gopikrishnan, P. Jonnalagadda, M. Driss, and W. Boulila, "EFS-IDS: An Enhanced Feature-Selective Intrusion Detection System for Imbalanced IoT Traffic Data," IEEE Open J. Commun. Soc., vol. 6, pp. 9673–9695, 2025, doi: https://doi.org/10.1109/OJCOMS.2025.3630471 .

[6] A. Abdelkhalek and M. Mashaly, "Addressing the class imbalance problem in network intrusion detection systems using data resampling and deep learning," J. Supercomput., vol. 79, no. 10, pp. 10611–10644, Jul. 2023, doi: https://doi.org/10.1007/s11227-023-05073-x .

[7] S. Bagui, D. Mink, S. Bagui, S. Subramaniam, and D. Wallace, "Resampling Imbalanced Network Intrusion Datasets to Identify Rare Attacks," Futur. Internet, vol. 15, no. 4, p. 130, Mar. 2023, doi: https://doi.org/10.3390/fi15040130 .

[8] T.-T.-H. Le, Y. Shin, M. Kim, and H. Kim, "Towards unbalanced multiclass intrusion detection with hybrid sampling methods and ensemble classification," Appl. Soft Comput., vol. 157, p. 111517, May 2024, doi: https://doi.org/10.1016/j.asoc.2024.111517 .

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," J. Artif. Intell. Res., vol. 16, pp. 321–357, Jun. 2002, doi: https://doi.org/10.1613/jair.953 .

[10] M. Zakariah, S. A. AlQahtani, and M. S. Al-Rakhami, "Machine Learning-Based Adaptive Synthetic Sampling Technique for Intrusion Detection," Appl. Sci., vol. 13, no. 11, p. 6504, May 2023, doi: https://doi.org/10.3390/app13116504 .

[11] J. Liu, Y. Gao, and F. Hu, "A fast network intrusion detection system using adaptive synthetic oversampling and LightGBM," Comput. Secur., vol. 106, p. 102289, Jul. 2021, doi: https://doi.org/10.1016/j.cose.2021.102289 .

[12] F. Omer Albasheer, R. Ramesh Haibatti, M. Agarwal, and S. Yeob Nam, "A Novel IDS Based on Jaya Optimizer and Smote-ENN for Cyberattacks Detection," IEEE Access, vol. 12, pp. 101506–101527, 2024, doi: https://doi.org/10.1109/ACCESS.2024.3431534 .

[13] T. Sameer, J. Rashmi, B. Khushi, M. Khushi, R. Rahul, and J. Prachi, "Enhanced intrusion detection framework: Integrating SMOTE-ENN with random forest and XGBoost," i-manager's J. Inf. Technol., vol. 13, no. 3, p. 19, 2024, doi: https://doi.org/10.26634/jit.13.3.21454 .

[14] L. Xue and T. Zhu, "Hybrid resampling and weighted majority voting for multi-class anomaly detection on imbalanced malware and network traffic data," Eng. Appl. Artif. Intell., vol. 128, p. 107568, Feb. 2024, doi: https://doi.org/10.1016/j.engappai.2023.107568 .

[15] M. A. Umar, Z. Chen, K. Shuaib, and Y. Liu, "Effects of feature selection and normalization on network intrusion detection," Data Sci. Manag., vol. 8, no. 1, pp. 23–39, Mar. 2025, doi: https://doi.org/10.1016/j.dsm.2024.08.001 .

[16] A. Salehpour, M. Norouzi, M. A. Balafar, and K. SamadZamini, "A cloud-based hybrid intrusion detection framework using XGBoost and ADASYN-Augmented random forest for IoMT," IET Commun., vol. 18, no. 19, pp. 1371–1390, Dec. 2024, doi: https://doi.org/10.1049/cmu2.12833 .

[17] 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, p. 69, Dec. 2024, doi: https://doi.org/10.3390/electronics14010069 .

[18] D. Kasina, V. S. Reddy, S. Rajamanickam, J. Jose, and B. Palani, "A novel intrusion detection system for class imbalance datasets using hybrid sampling with deep learning techniques," Inf. Sci. (Ny)., vol. 741, p. 123208, Jun. 2026, doi: https://doi.org/10.1016/j.ins.2026.123208 .

[19] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in 2015 Military Communications and Information Systems Conference (MilCIS), IEEE, Nov. 2015, pp. 1–6. doi: https://doi.org/10.1109/MilCIS.2015.7348942 .

[20] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in Proceedings of the 4th International Conference on Information Systems Security and Privacy, SCITEPRESS - Science and Technology Publications, 2018, pp. 108–116. doi: https://www.scitepress.org/Link.aspx?doi=10.5220/0006639801080116 .

[21] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment," Sensors, vol. 23, no. 13, p. 5941, Jun. 2023, doi: https://doi.org/10.3390/s23135941 .

[22] S. T. Hamidou and A. Mehdi, "Enhancing IDS performance through a comparative analysis of Random Forest, XGBoost, and Deep Neural Networks," Mach. Learn. with Appl., vol. 22, p. 100738, Dec. 2025, doi: https://doi.org/10.1016/j.mlwa.2025.100738 .

[23] D. Chicco, N. Tötsch, and G. Jurman, "The Matthews correlation coefficient (MCC) is more reliable than balanced accuracy, bookmaker informedness, and markedness in two-class confusion matrix evaluation," BioData Min., vol. 14, no. 1, p. 13, Feb. 2021, doi: https://doi.org/10.1186/s13040-021-00244-z .

[24] T. Saito and M. Rehmsmeier, "The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets," PLoS One, vol. 10, no. 3, p. e0118432, Mar. 2015, doi: https://doi.org/10.1371/journal.pone.0118432 .

[25] O. Rainio, J. Teuho, and R. Klén, "Evaluation metrics and statistical tests for machine learning," Sci. Rep., vol. 14, no. 1, p. 6086, Mar. 2024, doi: https://doi.org/10.1038/s41598-024-56706-x .

[26] C. Padró-Ferragut, F. Martínez-Plumed, and M. J. Ramírez-Quintana, "Robustness under noise: assessing the impact of perturbed key attributes on machine learning models," Int. J. Data Sci. Anal., vol. 22, no. 1, p. 71, Dec. 2026, doi: https://doi.org/10.1007/s41060-025-01006-4 .