



Machine Learning-Based Intrusion Detection System for Secure Communication Networks

Marisabel Yamashiro

Department of Electrical and Computer Engineering, Padma Institute of Business and Management, Bangladesh

marisabel.yamashiro@pibm-bd.org

Peer Review Information

Submission: 28 Dec 2020

Revision: 15 Jan 2021

Acceptance: 07 Feb 2021

Keywords

Intrusion Detection System, Machine Learning, Network Security, Anomaly Detection, Secure Communication.

Abstract

Secure communication networks must distinguish legitimate traffic from rapidly changing malicious activity without introducing excessive false alarms or processing delay. This methodology paper presents a machine-learning-based network intrusion detection system (NIDS) designed around flow-level traffic analysis, reproducible preprocessing, feature selection, supervised classification, and security-oriented performance evaluation. The framework uses modern benchmark traffic, with UNSW-NB15 as the primary dataset and CICIDS2017 as an external robustness reference, because both contain diverse benign and attack behaviors that are more representative than legacy KDD-only evaluations [1], [2], [7]. The proposed study compares a linear baseline, support vector machine, random forest, and multilayer perceptron under identical train/validation/test rules. Model selection is based on macro-F1, recall, false-alarm rate, ROC-AUC, and inference latency rather than accuracy alone. A literature-based validation of the methodology shows that learning-based IDSs can achieve strong detection performance, but results are highly sensitive to dataset age, class imbalance, feature engineering, and evaluation protocol [5], [6], [9]–[12]. The paper therefore emphasizes leakage-free preprocessing, held-out testing, class-aware metrics, and deployability for secure communication networks.

Introduction

Communication networks carry authentication data, financial transactions, industrial telemetry, cloud traffic, and private user information. Their increasing scale and heterogeneity have enlarged the attack surface and made manual inspection impractical. A network intrusion detection system monitors traffic and identifies behavior that may violate confidentiality, integrity, or availability. Traditional signature-based systems are effective for known attacks, but they depend on previously defined patterns and can miss new or obfuscated threats. Anomaly-oriented machine learning provides a complementary mechanism because the classifier learns statistical differences between normal and

malicious traffic and can generalize beyond exact signatures [9], [12].

The quality of an ML-based IDS is determined by more than classifier accuracy. The dataset must represent realistic traffic, the preprocessing pipeline must prevent information leakage, minority attack classes must not be ignored, and the resulting model must respond quickly enough for operational deployment. Earlier evaluation practices relied heavily on KDD Cup 99. Tavallaee et al. showed that duplicated and redundant records in KDD Cup 99 can bias learning algorithms and motivated the cleaner NSL-KDD benchmark [1]. Later datasets were designed to reflect more modern network behavior. UNSW-NB15 combines normal traffic with nine attack

families and supplies dedicated training and testing partitions [2]. CICIDS2017 further introduced contemporary benign behavior and attacks such as brute-force, botnet, denial-of-service, infiltration, port scanning, and web attacks [7].

Machine learning approaches have therefore evolved from shallow classifiers and manually engineered features toward neural and hybrid architectures. Singh et al. combined traffic profiling, feature selection, and online sequential extreme learning to reduce complexity while maintaining strong detection performance [3]. Saied et al. demonstrated artificial neural networks for detecting known and unknown DDoS behavior [4]. Yin et al. applied recurrent neural networks to intrusion classification [5], while Shone et al. used a nonsymmetric deep autoencoder with a random-forest classifier to learn compact representations [6]. Large comparative studies have since shown that no single model should be judged without consistent datasets, metrics, and preprocessing [8]–[12].

The objective of this paper is to define a reproducible methodology for a machine-learning-based IDS suitable for secure communication networks. The study is organized around five questions: (1) which preprocessing steps are required for mixed numerical and categorical network features, (2) which feature-selection strategy can reduce computational burden without removing attack-discriminating information, (3) how do classical and neural classifiers compare under identical conditions, (4) which metrics best represent operational security, and (5) how should the selected model be integrated into a low-latency network monitoring workflow. Because this is a methodology-focused paper, the Results section reports validated benchmark evidence from the cited literature and the expected decision criteria for the proposed experimental protocol rather than presenting fabricated measurements from an unexecuted experiment.

Literature Review

Benchmark design has been a persistent issue in intrusion-detection research. Tavallaee et al. identified structural limitations in KDD Cup 99 and proposed NSL-KDD as a less redundant evaluation set [1]. Moustafa and Slay later introduced UNSW-NB15 to represent modern normal and malicious traffic using a hybrid traffic-generation environment; the dataset contains nine attack categories and a broad flow-feature set [2]. Sharafaldin et al. developed CICIDS2017 after reviewing older datasets and emphasizing traffic diversity, current attack scenarios, complete labeling, and flow-level

characteristics [7]. These studies collectively support the use of a modern dataset as the primary benchmark and an independent dataset for robustness testing.

Classical machine learning remains relevant because tree ensembles and margin-based classifiers are comparatively efficient and interpretable. Singh et al. reduced the feature space and training data before using an online sequential extreme learning machine; their results demonstrated that profiling and feature reduction can improve computational efficiency without abandoning detection quality [3]. In a DDoS context, Saied et al. trained an artificial neural network on discriminative traffic characteristics and showed that learned patterns can separate legitimate and attack traffic, including previously unseen attack variations [4]. These findings motivate explicit feature analysis and a strong classical baseline in any modern IDS experiment.

Deep models reduce some dependence on manual feature engineering. Yin et al. proposed RNN-IDS and evaluated binary and multiclass intrusion recognition on NSL-KDD, reporting that recurrent modeling can improve classification relative to several traditional algorithms [5]. Shone et al. proposed a stacked nonsymmetric deep autoencoder with random forest and reported strong accuracy while reducing training time relative to deeper symmetric alternatives [6]. Vinayakumar et al. compared deep neural models across multiple public cybersecurity datasets and argued for systematic benchmarking because algorithm rankings can change by dataset and attack distribution [8].

Survey evidence reinforces this need for disciplined evaluation. Khraisat et al. reviewed IDS techniques, datasets, and challenges, highlighting zero-day detection, false alarms, and benchmark limitations [9]. Ferrag et al. compared seven deep-learning families across modern datasets and emphasized accuracy, detection rate, and false-alarm rate as core measures [10]. Gamage and Samarabandu performed a controlled comparison of feed-forward networks, autoencoders, deep belief networks, and LSTM models across legacy and modern datasets; their benchmark found that feed-forward neural networks offered a favorable balance of accuracy, F1 score, and computational cost [11]. Ahmad et al. similarly concluded that ML/DL-based NIDS design must be assessed jointly in terms of methodology, datasets, and evaluation metrics [12]. Kilincer et al. extended comparative analysis of ML methods for cybersecurity intrusion detection in 2021, reinforcing the value of consistent experimental conditions when comparing classifiers [13].

Methodology

1. Research Design and Dataset Strategy

The proposed work follows a controlled comparative experimental design. UNSW-NB15 is selected as the primary dataset because it provides an established train/test split, mixed feature types, normal traffic, and multiple contemporary attack families [2]. The main experiment is conducted as binary classification (normal versus attack) because this maps directly to real-time screening. A secondary multiclass experiment predicts the attack category for traffic already identified as malicious. CICIDS2017 is used as an external robustness reference rather than being merged with UNSW-NB15, because merging datasets generated under different environments can introduce domain artifacts and misleading performance gains [7], [11].

For UNSW-NB15, the identifier field is removed and the label is retained only as the target. The attack-category field is excluded from the binary feature matrix because it directly reveals malicious class semantics and would constitute target leakage. Categorical protocol/service/state variables are encoded, numerical variables are checked for missing and infinite values, and duplicate rows are removed only from training data when necessary. The official test partition remains untouched until final evaluation. This separation is essential because resampling or feature selection performed before the split can leak information from test data into the learned model.

2. Preprocessing and Feature Selection

Numerical features are standardized using parameters estimated only from the training set. Categorical variables are one-hot encoded with an unknown-category handler so that previously unseen service or state values do not terminate inference. Because attack data are imbalanced, the study uses class-weighted learning for models that support it. Synthetic oversampling may be tested inside cross-validation folds, but it is never applied to validation or test data. The main selection strategy ranks features using mutual information and then checks stability using random-forest importance. Candidate subsets of 15, 25, and all available post-encoding

features are compared. The smallest subset whose macro-F1 is within one percentage point of the best validation score is preferred, because it reduces latency and makes deployment easier.



Figure 1. Proposed machine-learning intrusion detection workflow for secure communication networks.

3. Model Development and Selection

Four classifiers are trained under the same data partitions. Logistic regression provides a transparent linear baseline. An RBF support vector machine represents a nonlinear margin-based classifier; for large training sets, its hyperparameter search is conducted on a stratified subset and the selected configuration is then retrained on the available training data if computationally feasible. Random forest represents a robust nonlinear ensemble with limited preprocessing sensitivity. A multilayer perceptron (MLP) represents a neural model capable of learning higher-order feature interactions. Hyperparameters are tuned only using stratified five-fold cross-validation on the training partition. The selected model is the one with the highest mean macro-F1 subject to a practical false-alarm-rate constraint; if two models are statistically similar, the model with lower inference latency and fewer features is preferred.

Table 1. Proposed experimental configuration

Component	Configuration	Purpose
Primary data	UNSW-NB15 official training/testing partitions	Modern multi-attack benchmark with fixed hold-out test
External reference	CICIDS2017	Check whether conclusions are dataset-specific

Component	Configuration	Purpose
Feature handling	One-hot encoding; standardization; leakage-free selection	Prepare mixed flow features and reduce dimensionality
Models	Logistic regression, RBF-SVM, Random Forest, MLP	Compare linear, margin, ensemble, and neural learning
Validation	Stratified 5-fold CV on training data	Tune without exposing the held-out test set
Primary metrics	Macro-F1, recall, FAR, ROC-AUC, latency	Prioritize security detection and operational cost

4. Evaluation Metrics and Security Interpretation

A confusion matrix provides true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Accuracy is computed as $(TP + TN)/(TP + TN + FP + FN)$. Precision is $TP/(TP + FP)$, recall or detection rate is $TP/(TP + FN)$, F1 is the harmonic mean of precision and recall, and false-alarm rate is $FP/(FP + TN)$. ROC-AUC is included to evaluate ranking quality across thresholds. For multiclass detection, macro-averaged precision, recall, and F1 are reported so that rare attack classes receive equal weight. Weighted F1 is also reported for completeness but is not used alone for model selection because majority classes can dominate it.

Operational performance is evaluated using per-flow inference latency and throughput. The final threshold is selected on the validation data using a cost-sensitive rule that penalizes missed attacks more strongly than false alerts while imposing a maximum acceptable false-alarm rate. This is important because a model that raises excessive alerts can overwhelm analysts even if its raw

accuracy is high. Confidence scores are logged so that uncertain cases can be escalated rather than automatically blocked. In deployment, the model runs passively on mirrored network traffic, and confirmed incidents are periodically added to a controlled retraining dataset. Model versions, preprocessing parameters, feature lists, and evaluation results are retained to support reproducibility and rollback.

Results

Because the present contribution is a methodology paper, this section summarizes published benchmark evidence that validates the proposed design choices and defines the results that should be produced when the protocol is executed. It does not claim that a new training run was performed for this manuscript. The literature demonstrates three consistent patterns: feature reduction can lower computational cost, neural or hybrid models can improve nonlinear attack recognition, and dataset/evaluation differences can substantially change reported performance.

Table 2. Published benchmark evidence supporting the proposed methodology

Study	Dataset	Method	Published finding
Singh et al. [3]	NSL-KDD / Kyoto	Traffic profiling + OS-ELM	Reported 98.66% accuracy and 1.74% false-positive rate in the main evaluation; profiling reduced processing complexity.
Yin et al. [5]	NSL-KDD	RNN-IDS	Binary KDDTest+ accuracy reached 83.28% for the stated configuration; training-set accuracy was much higher, illustrating the importance of held-out testing.
Shone et al. [6]	KDD Cup 99 / NSL-KDD	Stacked NDAE + RF	Reported 97.85% average accuracy on KDD Cup 99 and 85.42% total accuracy on NSL-KDD, showing strong dataset dependence.
Gamage & Samarabandu [11]	KDD99, NSL-KDD, CICIDS2017, CICIDS2018	FFNN, AE, DBN, LSTM	Controlled comparison found feed-forward neural networks favorable across accuracy, F1, and computational measures.
Kilincer et al. [13]	Multiple IDS datasets	Comparative ML study	Reinforced that classifier comparison should use common datasets, preprocessing, and metrics rather than isolated accuracy claims.

These findings directly motivate the proposed result-reporting format. First, training accuracy should never be used as the principal indicator: the RNN-IDS results in [5] show a clear gap

between training and held-out performance. Second, the contrast between Shone et al. results on KDD Cup 99 and NSL-KDD demonstrates that high values on an easier or older benchmark do

not automatically translate to modern traffic [6]. Third, the favorable performance of a comparatively straightforward feed-forward architecture in the controlled benchmark of Gamage and Samarabandu shows that model complexity alone is not a sufficient justification for deployment [11].

When the proposed protocol is executed, the primary results table should therefore report, for each model, held-out accuracy, precision, recall, macro-F1, ROC-AUC, false-alarm rate, number of retained features, and median inference latency. A second table should give per-attack recall for multiclass evaluation. The expected decision pattern is not “choose the highest accuracy”; rather, a deployable model must sustain strong attack recall and macro-F1 while keeping false alarms and latency within the network operator’s limits. Random forest is expected to provide a strong classical baseline because it captures nonlinear feature interactions and is robust to mixed traffic characteristics, while the MLP is expected to become competitive when training data are sufficiently large. SVM may remain accurate but can be computationally expensive at scale, and logistic regression provides a useful lower-complexity reference.

A secure deployment should also include an error analysis. False negatives are grouped by attack family to determine whether the detector systematically misses low-frequency or stealthy attacks. False positives are examined by protocol, service, and traffic intensity to identify normal behaviors that resemble attacks. If performance collapses on the external dataset, the result should be interpreted as evidence of domain shift rather than hidden by retraining on the test data. Such analysis is consistent with survey findings that class imbalance, outdated data, and inconsistent evaluation remain major obstacles in learning-based IDS research [9], [10], [12].

Conclusion and Discussion

This paper presented a structured methodology for developing a machine-learning-based intrusion detection system for secure communication networks. The framework combines a modern benchmark dataset, leakage-free preprocessing, class-aware feature selection, multiple classifier families, stratified cross-validation, held-out testing, and deployment-oriented metrics. The design is intentionally comparative: the purpose is not to assume that a deep model is automatically superior, but to determine which model offers the best security-operational trade-off under identical data and evaluation conditions. This position is supported by comparative IDS literature showing that apparent performance can change substantially

with dataset generation, attack composition, and model-evaluation protocol [8], [10]–[13]. The methodology also addresses common weaknesses in published IDS work. Accuracy alone can be misleading when normal traffic or a small number of attack classes dominate the dataset. For this reason, macro-F1, per-class recall, false-alarm rate, and ROC-AUC are included alongside accuracy. The official test data are isolated from feature selection and resampling, preventing optimistic leakage. Modern datasets are prioritized because legacy KDD-style data can exaggerate performance due to redundancy and outdated traffic characteristics [1]. UNSW-NB15 and CICIDS2017 provide stronger coverage of contemporary network behavior, although neither can perfectly represent a live enterprise network [2], [7]. The proposed system is suitable for passive NIDS deployment in which flow records are collected from a mirrored switch port or network sensor, transformed by the same preprocessing pipeline used during training, and scored in near real time. A confidence-aware alert layer allows uncertain cases to be escalated rather than immediately blocked, reducing the operational risk of false positives. Periodic retraining can incorporate analyst-confirmed incidents, but retraining should be version-controlled and evaluated against a fixed regression test to avoid catastrophic degradation. Future work should extend the protocol to concept-drift detection, adversarial manipulation of network features, encrypted-traffic representation, federated or privacy-preserving training, and calibrated uncertainty for zero-day attacks. The central conclusion is that an effective ML-based IDS is not defined by the most complex classifier; it is defined by reproducible evaluation, realistic data, strong minority-attack recall, controlled false alarms, and inference efficiency.

References

- M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” in *Proc. IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6. doi: 10.1109/CISDA.2009.5356528.
- N. Moustafa and J. Slay, “UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set),” in *2015 Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6. doi: 10.1109/MilCIS.2015.7348942.
- R. Singh, H. Kumar, and R. K. Singla, “An intrusion detection system using network traffic profiling

and online sequential extreme learning machine,” *Expert Systems with Applications*, vol. 42, no. 22, pp. 8609–8624, 2015. doi: 10.1016/j.eswa.2015.07.015.

Saied, R. E. Overill, and T. Radzik, “Detection of known and unknown DDoS attacks using Artificial Neural Networks,” *Neurocomputing*, vol. 172, pp. 385–393, 2016. doi: 10.1016/j.neucom.2015.04.101.

Yin, Y. Zhu, J. Fei, and X. He, “A deep learning approach for intrusion detection using recurrent neural networks,” *IEEE Access*, vol. 5, pp. 21954–21961, 2017. doi: 10.1109/ACCESS.2017.2762418.

N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018. doi: 10.1109/TETCI.2017.2772792.

Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in *Proc. 4th International Conference on Information Systems Security and Privacy (ICISSP)*, 2018, pp. 108–116. doi: 10.5220/0006639801080116.

R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, “Deep learning approach for intelligent intrusion detection system,” *IEEE*

Access, vol. 7, pp. 41525–41550, 2019. doi: 10.1109/ACCESS.2019.2895334.

Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, “Survey of intrusion detection systems: techniques, datasets and challenges,” *Cybersecurity*, vol. 2, art. 20, 2019. doi: 10.1186/s42400-019-0038-7.

M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, “Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study,” *Journal of Information Security and Applications*, vol. 50, art. 102419, 2020. doi: 10.1016/j.jisa.2019.102419.

S. Gamage and J. Samarabandu, “Deep learning methods in network intrusion detection: A survey and an objective comparison,” *Journal of Network and Computer Applications*, vol. 169, art. 102767, 2020. doi: 10.1016/j.jnca.2020.102767.

Z. Ahmad, A. S. Khan, C. W. Shiang, J. Abdullah, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, e4150, 2021. doi: 10.1002/ett.4150.

I. F. Kilincer, F. Ertam, and A. Sengur, “Machine learning methods for cyber security intrusion detection: Datasets and comparative study,” *Computer Networks*, vol. 188, art. 107840, 2021. doi: 10.1016/j.comnet.2021.107840.