



Machine Learning-Based Network Intrusion Detection for Secure Communication Systems

Quillon Saeedzada

Department of Computer Science and Engineering, Angkor Mekong Technical University, Cambodia

quillon.saeedzada@amtu-kh.edu

Peer Review Information

Submission: 28 Jan 2020

Revision: 27 Feb 2020

Acceptance: 02 April 2020

Keywords

*Network intrusion detection,
Machine learning,
Cybersecurity, Anomaly
detection, Secure
communication*

Abstract

Secure communication systems require timely recognition of malicious network behavior without overwhelming security analysts with false alarms. This methodology paper presents a machine learning-based network intrusion detection framework that converts packet- or flow-level traffic into a supervised binary decision between normal and attack activity. The proposed workflow combines data cleaning, categorical encoding, normalization, feature selection, stratified training and testing, and comparative classification using Random Forest (RF), Support Vector Machine (SVM), Decision Tree (DT), and Logistic Regression (LR). NSL-KDD is selected as a reproducible baseline because it reduces major redundancy problems of KDD Cup 99, while CICIDS2017 is recommended as a secondary modern validation dataset containing diverse benign and attack traffic. Performance is evaluated using accuracy, precision, recall/detection rate, F1-score, false-positive rate, confusion matrices, and inference time. Rather than claiming unperformed experiments, the results section defines literature-grounded benchmark expectations and decision rules for selecting the final model. The methodology emphasizes reproducibility, class-imbalance awareness, cross-dataset validation, and operational relevance for secure communication environments.

Introduction

Modern communication systems interconnect enterprise applications, cloud services, mobile users, industrial devices, and Internet-facing services. This connectivity improves availability and collaboration, but it also creates a large attack surface in which scanning, denial-of-service, credential abuse, malware, privilege escalation, and data exfiltration may be hidden within legitimate traffic. Network intrusion detection systems (NIDS) therefore operate as an important monitoring layer by inspecting traffic characteristics and producing alerts when activity deviates from an accepted security profile. Traditional signature-based systems are effective for known patterns but are limited when attacks are novel or intentionally modified.

Machine learning (ML) offers an alternative by learning statistical decision boundaries from labeled traffic and identifying combinations of features that distinguish benign and malicious behavior (Buczak & Guven, 2016; Khraisat et al., 2019).

The use of ML for NIDS is not simply a classification problem. Sommer and Paxson (2010) showed that network security differs from many closed-world machine-learning tasks because traffic distributions evolve, attack examples are scarce, false alarms are operationally costly, and training data rarely represent every production condition. Consequently, very high laboratory accuracy does not automatically imply that a model is deployable. Data quality is equally important.

Tavallaee et al. (2009) analyzed the KDD Cup 99 benchmark and proposed NSL-KDD to reduce redundant and duplicated records that can bias classifier evaluation. Later reviews emphasized that realistic dataset selection, diversity of attack classes, imbalance, temporal drift, and transparent evaluation are central methodological issues (Ring et al., 2019; Liu & Lang, 2019).

This paper therefore focuses on a reproducible methodology for ML-based intrusion detection in secure communication systems. The objective is to design a compact evaluation pipeline that can be implemented using standard Python ML libraries and tested on public intrusion-detection datasets. The study compares four supervised classifiers representing ensemble, margin-based, tree-based, and linear learning: RF, SVM, DT, and LR. The central research question is: which classifier provides the best balance of attack detection, false-positive control, generalization, and computational efficiency when applied to preprocessed network traffic? The methodology also defines safeguards against data leakage and class imbalance and recommends validation on more than one dataset before operational use.

Literature Review

Machine-learning approaches to intrusion detection have progressed from conventional statistical and decision-tree models to deep architectures that learn nonlinear representations. Buczak and Guven (2016) surveyed data-mining and ML techniques used for cyber intrusion detection and highlighted the importance of matching algorithms to data characteristics and operational constraints. Yin et al. (2017) demonstrated that recurrent neural networks can model intrusion classes and reported advantages over several traditional classifiers on benchmark data. Shone et al. (2018) proposed a nonsymmetric deep autoencoder and

stacked classification framework, illustrating how unsupervised representation learning can reduce manual feature engineering. These studies support the value of learning-based detection, while also showing that model complexity should be justified by measurable gains.

Dataset realism remains a major limitation. Sharafaldin et al. (2018) introduced CICIDS2017 to provide benign activity together with multiple contemporary attack scenarios and a rich flow-feature representation. Ring et al. (2019) compared network-based intrusion datasets using properties such as recording environment, traffic volume, labeling, and feature availability, concluding that dataset choice directly affects the validity of evaluation. Khraisat et al. (2019) similarly reviewed IDS techniques, datasets, and challenges and identified false alarms, evolving threats, and evaluation inconsistency as persistent concerns. These findings motivate the use of NSL-KDD for reproducibility and CICIDS2017 for complementary validation rather than relying on a single benchmark.

Surveys by Liu and Lang (2019), Ferrag et al. (2020), and Aldweesh et al. (2020) show that both traditional ML and deep learning can produce strong detection results, but performance depends on preprocessing, feature representation, dataset age, classification mode, and the selected metric. For imbalanced intrusion data, Zhang et al. (2020) combined SMOTE, Gaussian-mixture-based sampling, and convolutional learning and reported very high detection rates on UNSW-NB15 and CICIDS2017, demonstrating that minority-class handling can materially change reported performance. Because secure communication systems must minimize disruptive false alerts as well as missed attacks, the present methodology evaluates recall and F1-score together with false-positive rate instead of optimizing accuracy alone.

Table 1. Selected literature informing the proposed methodology

Study	Dataset / Focus	Method	Key contribution	Methodological implication
Tavallaee et al. (2009)	KDD / NSL-KDD	Dataset analysis	Reduced redundancy bias	Use reproducible train/test sets
Sommer & Paxson (2010)	Operational NIDS	ML deployment critique	Closed-world assumptions are risky	Measure false alarms and generalization
Sharafaldin et al. (2018)	CICIDS2017	Traffic generation + ML	Modern benign/attack flows	Add secondary modern validation
Liu & Lang (2019)	IDS literature	ML/DL survey	Taxonomy, metrics and datasets	Compare multiple classifier families
Aldweesh et al. (2020)	Anomaly-based IDS	DL survey/taxonomy	Dataset and evaluation gaps	Avoid single-dataset claims

Zhang et al. (2020)	UNSW-NB15, CICIDS2017	Sampling + CNN	Addresses class imbalance	Use class-aware metrics and resampling cautiously
---------------------	-----------------------	----------------	---------------------------	---

Methodology

1. Research Design and Dataset Selection

The study uses a quantitative, comparative experimental design. Each connection or network flow is treated as one observation with a set of traffic features and a class label. The primary experiment uses NSL-KDD because it provides established training and testing partitions and removes many duplicate records that distort KDD Cup 99 evaluation (Tavallae et al., 2009). A secondary experiment should repeat the pipeline on CICIDS2017, whose benign and malicious traffic better represents contemporary protocols and attack behaviors (Sharafaldin et al., 2018). For the main binary task, all attack labels are mapped to “attack” and normal traffic remains “normal.” A multiclass extension may retain attack categories, but the binary configuration is recommended for the six-to-seven-page study because it supports direct operational interpretation.

2. Data Preprocessing and Feature Engineering

Preprocessing is performed inside the training pipeline to prevent information leakage. Duplicate or malformed records are removed when present. Categorical variables such as protocol type, service, and connection flag are one-hot encoded. Continuous variables are standardized using parameters fitted only on the training set. Constant features are removed, while optional correlation screening can

eliminate highly redundant predictors. When class imbalance is severe, class-weighted learning is preferred as the first option; synthetic oversampling is applied only to the training fold and never to validation or test data. This design follows the literature warning that preprocessing and imbalance handling can strongly influence IDS performance (Zhang et al., 2020).

3. Classifier Development

Four classifiers are trained under the same preprocessing and split conditions. Random Forest is used as the primary ensemble model because it handles nonlinear interactions and mixed feature importance with limited tuning. SVM with a radial-basis kernel represents margin-based nonlinear classification; class weights are enabled where necessary. Decision Tree provides an interpretable baseline, while Logistic Regression provides a fast linear baseline suitable for high-throughput communication systems. Hyperparameters are selected using stratified five-fold cross-validation on the training data. The final model is then refitted on the complete training partition and evaluated once on the untouched test partition. This comparison is deliberately simpler than deep architectures so that any performance gain can be weighed against inference cost and deployment complexity, consistent with the broader evidence reviewed by Ferrag et al. (2020) and Aldweesh et al. (2020).



Figure 1. Proposed machine learning workflow for network intrusion detection.

4. Evaluation Metrics and Decision Criteria

Performance is measured using accuracy, precision, recall (detection rate), F1-score, false-positive rate (FPR), confusion matrix values, and average inference time per observation. Recall is $TP/(TP+FN)$, precision is $TP/(TP+FP)$, F1 is the harmonic mean of precision and recall, and FPR is $FP/(FP+TN)$. Macro-averaged measures are used for multiclass experiments so that minority attacks are not hidden by majority classes. The preferred model is the one that maximizes F1 and recall while maintaining a low FPR and

acceptable inference cost. A small accuracy gain is not considered sufficient if it substantially increases false alarms or model latency. Model comparison should be repeated using fixed random seeds and reported with mean and standard deviation across validation folds.

5. Implementation Procedure

Step 1: Load NSL-KDD training and testing records; inspect missing, duplicate, and class-distribution information.

Step 2: Map labels to normal/attack for binary classification and define a preprocessing transformer for numeric and categorical variables.

Step 3: Fit encoding, scaling, and optional feature selection only on the training partition.

Step 4: Train LR, DT, RF, and SVM models using stratified five-fold cross-validation and controlled hyperparameter grids.

Step 5: Refit the best version of each classifier on the complete training set and evaluate on the untouched test set.

Step 6: Compare accuracy, precision, recall, F1, FPR, confusion matrix, and inference time; select the operational model using the predefined decision rule.

Step 7: Repeat the selected preprocessing and model family on CICIDS2017 to test generalization to a more recent traffic environment.

Results

Because this paper is methodology-focused, it does not present fabricated “new” classifier scores. Instead, the result of the methodological analysis is a reproducible evaluation matrix and literature-grounded performance expectations that can be populated after the pipeline is executed. Prior work demonstrates that model and dataset choices can materially alter the apparent quality of an IDS. Yin et al. (2017)

reported that an RNN-based detector outperformed several traditional approaches on benchmark intrusion data, while Shone et al. (2018) showed that learned representations can improve network intrusion classification. At the high-performance end, Zhang et al. (2020) reported a 99.74% binary detection rate on UNSW-NB15 and a 99.85% detection rate for 15-class CICIDS2017 using an imbalance-aware CNN framework. These values should be treated as literature reference points rather than as expected scores for the simpler classifiers proposed here.

For the present study, a valid result must satisfy three conditions. First, the chosen classifier should achieve strong recall and F1 rather than accuracy alone. Second, its FPR must remain sufficiently low for operational monitoring, because high false-alarm volumes reduce analyst trust and increase response cost. Third, the ranking of models should remain reasonably stable when the workflow is repeated on CICIDS2017. If RF provides the best balance of recall, F1, and inference time, it is selected as the final NIDS model; if SVM improves detection but increases latency substantially, RF may still be preferable for real-time communication systems. DT and LR act as interpretable efficiency baselines and help determine whether additional model complexity is justified.

Table 2. Result reporting template and model-selection rule

Model	Primary strength	Metrics to report	Risk check to	Selection condition
LR	Speed, linear interpretability	Accuracy, Precision, Recall, F1, FPR, latency	Underfitting nonlinear attacks	Select only if performance is close to complex models
DT	Transparent decision rules	Accuracy, Recall, F1, FPR	Overfitting / instability	Use as interpretable baseline
RF	Nonlinear ensemble robustness	Accuracy, Precision, Recall, F1, FPR, latency	Model size	Preferred if best F1-recall-FPR balance
SVM	Strong nonlinear boundary	Precision, Recall, F1, FPR, latency	Training / inference cost	Select if gain exceeds operational cost

Conclusion and Discussion

This paper developed a practical methodology for machine learning-based network intrusion detection in secure communication systems. The framework integrates dataset selection, leakage-safe preprocessing, class-aware training, multi-model comparison, and operationally meaningful evaluation. The literature shows that ML and deep-learning detectors can achieve high benchmark performance, but it also repeatedly

identifies weaknesses caused by outdated traffic, imbalanced classes, inconsistent metrics, and unrealistic laboratory assumptions (Sommer & Paxson, 2010; Ring et al., 2019; Khraisat et al., 2019). For this reason, the proposed study does not select a model by accuracy alone and does not treat a single dataset as proof of deployment readiness. The main methodological contribution is the combination of a reproducible baseline (NSL-KDD) with a more contemporary secondary

validation dataset (CICIDS2017). RF, SVM, DT, and LR provide a controlled comparison across ensemble, margin-based, tree, and linear learning families. The final decision rule prioritizes recall, F1-score, FPR, and inference cost because secure communication environments require both reliable attack detection and manageable alert volumes. The approach can later be extended to multiclass attack recognition, online learning, feature-drift monitoring, explainable model outputs, or deep representation learning. Future experimental work should report complete preprocessing settings, random seeds, class distributions, confusion matrices, and cross-dataset results so that reported gains can be independently reproduced and interpreted.

References

- Aldweesh, A., Derhab, A., & Emam, A. Z. (2020). Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowledge-Based Systems*, 189, 105124. <https://doi.org/10.1016/j.knosys.2019.105124>
- Buczak, A. L., & Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2), 1153–1176. <https://doi.org/10.1109/COMST.2015.2494502>
- Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janicke, H. (2020). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, 50, 102419. <https://doi.org/10.1016/j.jisa.2019.102419>
- Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity*, 2, 20. <https://doi.org/10.1186/s42400-019-0038-7>
- Liu, H., & Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. *Applied Sciences*, 9(20), 4396. <https://doi.org/10.3390/app9204396>
- Ring, M., Wunderlich, S., Scheuring, D., Landes, D., & Hotho, A. (2019). A survey of network-based intrusion detection data sets. *Computers & Security*, 86, 147–167. <https://doi.org/10.1016/j.cose.2019.06.005>
- Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)* (pp. 108–116). <https://doi.org/10.5220/0006639801080116>
- Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A deep learning approach to network intrusion detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(1), 41–50. <https://doi.org/10.1109/TETCI.2017.2772792>
- Sommer, R., & Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE Symposium on Security and Privacy* (pp. 305–316). <https://doi.org/10.1109/SP.2010.25>
- Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications* (pp. 1–6). <https://doi.org/10.1109/CISDA.2009.5356528>
- Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access*, 5, 21954–21961. <https://doi.org/10.1109/ACCESS.2017.2762418>
- Zhang, H., Huang, L., Wu, C. Q., & Li, Z. (2020). An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset. *Computer Networks*, 177, 107315. <https://doi.org/10.1016/j.comnet.2020.107315>