



Machine Learning Approaches for Intrusion Detection in Computer Networks

Marisabel Belhocine

Department of Financial Management and Analytics, Kavir Polytechnic University of Technology, Iran
marisabel.belhocine@kput-ir.net

Peer Review Information	Abstract
<i>Submission: 04 Jan 2019</i> <i>Revision: 19 Feb 2019</i> <i>Acceptance: 07 March 2019</i> Keywords <i>intrusion detection system, machine learning, network security, anomaly detection, deep learning</i>	Intrusion detection is a central component of network defense because firewalls and signature rules alone cannot reliably identify previously unseen or rapidly evolving attacks. This methodology paper presents a systematic framework for applying machine-learning (ML) and deep-learning (DL) approaches to network intrusion detection using benchmark traffic data available in the literature up to 2019. The proposed procedure integrates dataset selection, cleaning, categorical encoding, feature scaling, imbalance treatment, feature selection, classifier training, hyperparameter tuning, cross-validation, and final testing. Classical models such as k-nearest neighbors, support vector machines and random forests are compared with multilayer neural networks, recurrent neural networks and autoencoder-based anomaly detectors. Performance is assessed with accuracy, precision, recall, F1-score, false-positive rate, receiver-operating characteristics and computational cost rather than accuracy alone. The methodology emphasizes leakage-free data partitioning and evaluation on modern datasets such as UNSW-NB15 and CICIDS2017 in addition to NSL-KDD. Literature through 2019 shows that deep models can learn complex traffic representations, while classical models remain valuable for efficiency and interpretability. The paper therefore recommends a comparative, dataset-aware evaluation strategy in which detection quality, false alarms, generalization and deployment cost are considered jointly.

Introduction

Computer networks carry business, academic, industrial and personal services, making the reliable detection of malicious traffic a continuing security requirement. Intrusion Detection Systems (IDSs) inspect network or host activity and generate alerts when behavior matches known attacks or deviates from expected patterns. Signature-based systems are effective for recognized threats, whereas anomaly-based systems can potentially detect novel behavior but often suffer from false alarms and evaluation difficulties (García-Teodoro et al., 2009). General anomaly-detection research similarly shows that the usefulness of a detector depends not only on the learning algorithm but

also on how normality, rarity and contextual deviation are defined (Chandola et al., 2009).

Machine learning provides a practical way to model these patterns from traffic features. Supervised algorithms learn boundaries between normal and attack classes, while unsupervised or semi-supervised methods attempt to identify unusual observations without requiring complete attack labels. Buczak and Guven (2016) reviewed a broad range of classification, clustering, ensemble and data-mining methods for cyber intrusion detection and highlighted that dataset quality and feature representation are as important as the chosen algorithm. At the same time, Sommer and Paxson

(2010) warned that strong results obtained in closed benchmark settings may not transfer directly to operational networks because real traffic is non-stationary, attacks are rare, and training labels are imperfect.

This paper therefore focuses on methodology rather than presenting unverified new performance claims. Its objective is to define a reproducible comparative process for evaluating ML and DL approaches to network intrusion detection using references published no later than 2019. The framework is designed to answer three questions: (i) how should network traffic be prepared so that model comparisons are fair; (ii) which model families should be included to represent classical, ensemble and deep learning; and (iii) which evaluation measures best reflect practical intrusion-detection performance. The resulting procedure can be implemented on NSL-KDD, UNSW-NB15, CICIDS2017, or a combination of benchmark datasets, with an emphasis on generalization and false-alarm control.

Literature Review

Tavallaee et al. (2009) examined weaknesses in the KDD Cup 1999 benchmark, including redundancy that can bias learning algorithms toward frequently repeated records. Their analysis led to the NSL-KDD dataset, which has been widely used because it reduces duplicate records and creates a more useful benchmark for comparing classifiers. The study is important methodologically because it demonstrates that apparently high classification accuracy can be caused by dataset structure rather than genuine attack generalization.

Sommer and Paxson (2010) argued that network intrusion detection differs from many standard ML problems because operational data do not follow a stable, closed-world distribution. Their work emphasized the semantic gap between statistical anomalies and security-relevant attacks, the high cost of false positives, and the danger of training and testing under unrealistically similar conditions. These observations motivate explicit controls against data leakage and the use of multiple evaluation metrics.

Liao et al. (2013) presented a comprehensive IDS taxonomy covering data sources, analysis strategies, architectures and detection approaches. They described the progression from knowledge-based detection toward behavior-based and learning-based methods. Their review supports a comparative methodology in which classifier selection is

linked to the type of traffic features, available labels and deployment environment rather than treating all IDS tasks as identical classification problems.

Moustafa and Slay (2015) introduced UNSW-NB15 to address limitations of older benchmarks. The dataset contains modern normal traffic and multiple contemporary attack categories generated using a controlled testbed. Its inclusion is important because models that perform well on KDD-derived data may not preserve the same behavior on more recent traffic distributions.

Ahmed et al. (2016) surveyed network anomaly-detection techniques and organized approaches into statistical, knowledge-based and machine-learning categories. The authors also discussed the importance of public datasets and the relationship between different forms of anomalies and network attacks. This work reinforces the need to evaluate the false-positive rate and class-specific detection performance, not only overall accuracy.

Yin et al. (2017) developed an RNN-based IDS and evaluated binary and multiclass classification on NSL-KDD. The study showed the value of recurrent architectures for learning dependencies in traffic-feature sequences and reported competitive performance against conventional classifiers. It also illustrates a broader transition from manually designed decision boundaries to models that learn richer internal representations.

Shone et al. (2018) proposed a deep network intrusion detection approach using a nonsymmetric deep autoencoder for unsupervised feature learning followed by classification. Their work demonstrated how learned representations can reduce dependence on fully hand-engineered features. In a related direction, Mirsky et al. (2018) presented Kitsune, an ensemble of autoencoders designed for efficient online anomaly detection, showing that unsupervised neural methods can be adapted to resource-constrained network environments.

Sharafaldin et al. (2018) produced CICIDS2017 to provide realistic benign traffic and multiple modern attack families. The dataset includes detailed flow features and is useful for testing whether models can discriminate attack categories under more representative conditions than legacy benchmarks. Vinayakumar et al. (2019) subsequently performed broad experiments with deep neural networks across several benchmark datasets, including KDDCup99, NSL-KDD, UNSW-NB15 and

CICIDS2017, demonstrating the importance of cross-dataset benchmarking and systematic hyperparameter selection.

Methodology

The proposed methodology is organized as a sequence of reproducible steps. The same train/validation/test partitions and preprocessing transformations are applied to all algorithms so that observed differences arise from the models rather than inconsistent data treatment.

Step 1 - Dataset selection and problem definition

Use NSL-KDD as a legacy benchmark and either UNSW-NB15 or CICIDS2017 as a modern benchmark. NSL-KDD permits comparison with a large body of earlier work (Tavallaei et al., 2009), while UNSW-NB15 and CICIDS2017 introduce newer protocols and attack behaviors (Moustafa and Slay, 2015; Sharafaldin et al., 2018). Define two tasks: binary classification (normal versus attack) and multiclass classification (normal plus attack categories). When multiple datasets are used, results must be reported separately before any pooled interpretation.

Step 2 - Data cleaning, encoding and partitioning

Remove exact duplicate records when present, inspect missing or infinite values, and eliminate identifiers that directly leak the label. Encode categorical variables such as protocol, service and connection state using one-hot or ordinal encoding as appropriate. Fit encoders and scalers only on the training set, then apply the fitted transformations to validation and test sets. Use a stratified split such as 70% training, 15% validation and 15% testing, or the official dataset split where it exists. This procedure follows the practical concern raised by Sommer and Paxson (2010) that evaluation must avoid unrealistic access to future or test information.

Step 3 - Class imbalance and feature preparation

Network datasets are usually imbalanced because high-volume attack families and normal flows dominate rare attacks. During training, compare class weighting with controlled resampling; never balance the independent test set. Standardize continuous variables for distance-based and neural models. For classical ML, perform feature selection using mutual information, recursive elimination or tree-based importance. For deep networks, retain a broader

feature set and allow the model to learn representations. These choices are consistent with survey findings that feature quality and anomaly representation strongly influence detection performance (Buczak and Guven, 2016; Ahmed et al., 2016).

Step 4 - Model construction

Evaluate at least five representative learners: k-nearest neighbors (KNN), support vector machine (SVM), random forest (RF), multilayer perceptron/deep neural network (DNN), and recurrent neural network (RNN). KNN provides a simple local baseline; SVM provides a margin-based classifier; RF represents an ensemble method with nonlinear decision capability and feature importance; DNN learns hierarchical nonlinear representations; and RNN represents sequential modeling. An optional autoencoder can be trained primarily on benign traffic and used as an anomaly detector by thresholding reconstruction error, reflecting the approaches used by Shone et al. (2018) and Mirsky et al. (2018).

Step 5 - Hyperparameter tuning and training control

Tune each model only on training and validation data. Candidate parameters include K for KNN; C, kernel and gamma for SVM; number of trees, tree depth and minimum samples per leaf for RF; number of hidden layers, units, dropout and learning rate for DNN; and hidden units, sequence representation and learning rate for RNN. Use stratified k-fold cross-validation for classical models and early stopping for neural models. Record training time, inference time and model size because a marginal increase in accuracy may not justify a large deployment cost. Vinayakumar et al. (2019) similarly emphasized systematic deep-network parameter selection across intrusion datasets.

Step 6 - Evaluation metrics

Calculate the confusion matrix for each model and report accuracy, precision, recall (detection rate), F1-score, false-positive rate (FPR) and area under the ROC curve were meaningful. For imbalanced multiclass data, report macro-averaged and weighted F1 in addition to overall accuracy. The core expressions are: Precision = $TP/(TP+FP)$, Recall = $TP/(TP+FN)$, $F1 = 2(Precision \times Recall)/(Precision+Recall)$, and $FPR = FP/(FP+TN)$. Class-specific recall should be examined for rare attacks because high overall accuracy can coexist with poor detection of low-frequency categories.

Step 7 - Robustness and deployment analysis

Repeat evaluation with different random seeds and report mean and variability. If resources permit, train on one dataset and test selected model behavior on a second dataset after compatible feature mapping; this is not expected

to yield identical performance, but it reveals distribution sensitivity. The final model should be chosen by a balanced operational criterion: strong recall and F1, an acceptably low false-positive rate, manageable inference latency, and a retraining procedure that accounts for traffic drift.



Figure 1. Proposed machine-learning intrusion detection methodology.

1. Comparative model design

Table 1. Role of representative machine-learning approaches in the proposed IDS comparison.

Approach	Learning form	Main strength for IDS	Main limitation / control
KNN	Supervised	Simple baseline; captures local similarity	Slow prediction on large data; scale features
SVM	Supervised	Strong margin-based classification in moderate dimensions	Training cost and kernel tuning on large datasets
Random Forest	Supervised ensemble	Handles nonlinear interactions; robust and interpretable importance	Can become large; tune depth and class weights
DNN / MLP	Supervised deep learning	Learns complex nonlinear feature combinations	Requires tuning, compute and regularization
RNN	Supervised deep learning	Can model sequential or ordered dependencies	Higher training cost; sensitive to representation
Autoencoder	Unsupervised / semi-supervised	Useful for anomaly or previously unseen behavior	Threshold selection may increase false alarms

Results

This paper defines a methodology and does not claim a newly executed benchmark; therefore, the results are presented as an evidence-based interpretation of what the proposed evaluation should reveal, grounded in studies published through 2019. First, no single classifier should be judged solely by overall accuracy. Older benchmarks can contain redundancies and class distributions that inflate apparent performance, as demonstrated by Tavallaee et al. (2009). Consequently, a trustworthy result set should include class-specific recall and false-positive rates and should be repeated on at least one modern dataset.

Second, classical ML models are expected to remain competitive baselines. Random forests are particularly useful where network features

contain nonlinear interactions and mixed importance, while SVM can perform strongly after careful scaling and parameter tuning. KNN offers an interpretable reference but its prediction cost rises with the number of stored training instances. The surveys by Buczak and Guven (2016) and Ahmed et al. (2016) show that such classical methods remain relevant because they can provide effective detection with lower training complexity than many deep architectures.

Third, deep models should be evaluated for representation learning rather than assumed to be universally superior. Yin et al. (2017) showed that RNN-based intrusion detection can achieve strong binary and multiclass performance on NSL-KDD, while Shone et al. (2018) demonstrated the value of autoencoder-based

feature learning. Kitsune further showed that an ensemble of small autoencoders can support online anomaly detection under constrained resources (Mirsky et al., 2018). Vinayakumar et al. (2019) extended the evidence across several datasets and found deep neural networks useful for learning complex attack representations. However, these advantages must be weighed against training time, parameter sensitivity and deployment cost.

Fourth, modern datasets are expected to reduce overly optimistic conclusions derived from legacy data. UNSW-NB15 was specifically created to include contemporary normal and malicious traffic (Moustafa and Slay, 2015), and CICIDS2017 was designed with realistic traffic profiles and multiple attack families (Sharafaldin et al., 2018). If a model shows excellent NSL-KDD performance but a large decline on one of these datasets, the correct interpretation is limited generalization rather than algorithmic failure in isolation. Such a result would support the closed-world concerns of Sommer and Paxson (2010).

The expected practical outcome of the methodology is therefore a multi-criteria comparison. A deployment candidate should combine high attack recall with a low false-positive rate, stable performance across repeated splits, and acceptable inference latency. For a resource-limited gateway, RF or a compact autoencoder may be preferable; for offline analysis with abundant training data, DNN or RNN models may justify their greater complexity. This interpretation follows the broader IDS taxonomy and deployment considerations summarized by Liao et al. (2013) and the anomaly-detection challenges discussed by García-Teodoro et al. (2009).

Conclusion and Discussion

This methodology paper proposes a structured approach for comparing machine-learning methods for intrusion detection in computer networks. The procedure begins with careful dataset selection, continues through leakage-free preprocessing and imbalance treatment, and evaluates classical ML, ensemble methods and deep architectures under common data partitions. It emphasizes that an intrusion detector is not useful merely because it achieves high accuracy; it must also detect minority attacks, control false alarms, generalize beyond the training distribution and operate within realistic computational limits. The literature up to 2019 shows a clear evolution from hand-designed anomaly and classification methods toward representation learning with deep neural

networks, RNNs and autoencoders. Nevertheless, the foundational warnings remain relevant: benchmark redundancy can distort results (Tavallaee et al., 2009), operational networks are not closed and stationary environments (Sommer and Paxson, 2010), and anomaly detectors can generate costly false positives (García-Teodoro et al., 2009). Modern datasets such as UNSW-NB15 and CICIDS2017 improve experimental realism but do not eliminate distribution drift or labeling uncertainty. Accordingly, future implementations of this methodology should report complete confusion-matrix metrics, computational cost and repeated-run variability, and should include cross-dataset or temporally separated testing when compatible data are available. Model interpretability and adversarial robustness should also be considered before operational deployment. The central conclusion is methodological: reliable intrusion-detection research requires disciplined evaluation of data, models and deployment assumptions together rather than selecting a classifier from benchmark accuracy alone.

References

- Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60, 19–31. DOI: 10.1016/j.jnca.2015.11.016.
- 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. DOI: 10.1109/COMST.2015.2494502.
- Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), Article 15. DOI: 10.1145/1541880.1541882.
- García-Teodoro, P., Díaz-Verdejo, J., Maciá-Fernández, G., & Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1–2), 18–28. DOI: 10.1016/j.cose.2008.08.003.
- Liao, H.-J., Lin, C.-H. R., Lin, Y.-C., & Tung, K.-Y. (2013). Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1), 16–24. DOI: 10.1016/j.jnca.2012.09.004.

Mirsky, Y., Doitshman, T., Elovici, Y., & Shabtai, A. (2018). Kitsune: An ensemble of autoencoders for online network intrusion detection. *Network and Distributed System Security Symposium (NDSS)*. DOI: 10.14722/ndss.2018.23204.

Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems. 2015 Military Communications and Information Systems Conference (MilCIS). DOI: 10.1109/MilCIS.2015.7348942.

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, 108–116. DOI: 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. DOI: 10.1109/TETCI.2017.2772792.

Sommer, R., & Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. 2010 IEEE Symposium on Security and Privacy, 305–316. DOI: 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. 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications. DOI: 10.1109/CISDA.2009.5356528.

Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. *IEEE Access*, 7, 41525–41550. DOI: 10.1109/ACCESS.2019.2895334.

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. DOI: 10.1109/ACCESS.2017.2762418.