FOUNTAIN JOURNAL OF NATURAL & APPLIED SCIENCES

A Publication of the College of Natural & Applied Sciences
Fountain University, Osogbo, Nigeria

Comparative analysis of machine learning models for network intrusion detection: a focus on execution speed and performance

Ibraheem, I. O.*[1], Jidda, M. T.[3], Abdulrasaq, A. A.[3], Segbefia, S. K.[4]

[1]*Department of Mathematical and Computer Science, Fountain University, Osogbo, Nigeria*
[2]*Ahmadu Bello University, Zaria, Nigeria*
[3]*Department of Computer Science, Al-Hikmah University, Ilorin-Kwara, Nigeria*
[4]*University of South Africa*

**Correspondence:* ibraheem.idris@fuo.edu.ng

ABSTRACT

As cyber threats grow more complex, the need for Network Intrusion Detection Systems (NIDS) that are effective and capable of real-time processing becomes inevitable. Traditional rule-based systems are effective against known threats; however, they tend to struggle with novel and more sophisticated attacks. This study evaluates three widely used machine learning models, Random Forest (RF), XGBoost, and Support Vector Machines (SVM), to identify an optimal balance between detection accuracy and computational efficiency. Using the UNSW-NB15 dataset and feature selection based on XGBoost's importance ranking, this study compares these models across accuracy, precision, recall, F1-score, and execution time. The results show that an optimised XGBoost model achieved the highest accuracy (87.18%), the lowest training time (35.2s), and a prediction time of 0.16s, making it an effective practical choice for real-time NIDS deployments. Random Forest achieved comparable accuracy (86.95%) but at a higher computational cost, whereas SVM's high computational cost makes it unsuitable for large-scale use. Feature Importance Analysis shows packet duration, source/destination bytes, and TCP flags as key indicators of network intrusion. These findings provide critical insights for cybersecurity practitioners focusing on efficient, adaptive threat detection. Hyperparameter optimisation for XGBoost was performed via grid search; the absence of k-fold cross-validation across all models is acknowledged as a scope limitation. Future work will explore deep learning architectures, hybrid models, and adversarial defence mechanisms to further advance NIDS performance.

ARTICLE INFO

Article history:
Received June 2026
Revised July 2026
Accepted August 2026

Keywords:
Network Intrusion Detection System (NIDS), Machine Learning, XGBoost, Random Forest, Support Vector Machines, Cybersecurity

Introduction

Organisations are facing the rapid spread of cyber threats that risk data breaches, operational disruptions, and financial damage in today's digital landscape. Traditional intrusion detection systems (IDS), such as signature- and rule-based approaches, are effective against known threats. However, they struggle when confronted with novel or evolving attack patterns. As cybercriminal methods grow more destructive, intelligent and adaptive security solutions are essential [1].

Machine learning (ML) has emerged as a promising approach for intrusion detection because it can analyse large volumes of network traffic to detect anomalous patterns often overlooked by conventional systems. However, real-time deployment of ML models in cybersecurity environments also requires careful consideration of a critical trade-off between detection accuracy and processing speed [2]. Highly accurate models that require long processing delays may fail to detect immediate threats, while fast models may compromise accuracy by allowing attacks to go undetected. Consequently, selecting a model that balances these factors is important for implementing effective NIDS.

DOI: https://doi.org/10.53704/fujnas.v15i1.1146

Some popular ML algorithms for NIDS, including Random Forest (RF), XGBoost, and Support Vector Machines (SVM), have shown strong detection capabilities. RF is an ensemble of decision trees known for efficiency and effectiveness; XGBoost is a gradient boosting framework optimised for speed and accuracy; and SVM is more effective for high-dimensional, nonlinear data classification [3]. Each algorithm has distinct advantages and limitations.

This study compares these models using the UNSW-NB15 dataset, a modern benchmark dataset reflecting diverse, realistic cyberattack types. The study evaluates detection performance using these metrics: accuracy, precision, recall, F1-score, and computational efficiency (training and prediction time). Using extensive feature selection and hyperparameter optimisation, the study aims to guide practitioners in selecting the most suitable model for real-time cybersecurity environments.

The UNSW-NB15 dataset was selected for its realistic portrayal of modern network conditions, diverse attack categories, and its established status as a benchmark in the NIDS literature, making it more representative than older datasets such as KDD99 or NSL-KDD. Most prior comparative studies in this space optimise solely for detection accuracy, leaving the equally critical dimension of execution speed underexplored. This study directly addresses that gap by evaluating all three models under consistent conditions with execution time as a primary evaluation criterion alongside standard classification metrics.

This study contributes empirical evidence on the accuracy–execution speed trade-offs in network intrusion detection, supporting informed model selection for NIDS design and deployment.

Literature review

Signature-based and anomaly detection

Intrusion detection has undergone a major shift over the past three decades. The earliest approaches were signature-based detection systems (SIDS), which rely on predefined rules and known attack patterns to identify malicious traffic. Snort and Suricata, among other open-source tools, remain widely used representatives of signature-based detection; they are very effective at detecting known threats with low false-positive rates. However, these systems also have some critical limitations, including the need for constant manual updating, failure to detect zero-day attacks or polymorphic variants, and poor adaptation to rapidly evolving threat landscapes [1].

Anomaly-based intrusion detection systems can overcome shortcomings peculiar to signature-based systems. By modelling normal network behaviour, they can detect deviations and categorise them as potential attacks. Clustering techniques, Principal Component Analysis (PCA), and statistical profiling are the main backbone of anomaly detection. Although anomaly-based IDS improve adaptability and can detect novel attacks, they are prone to high false-positive rates. Encrypted traffic often produces misleading deviations, which can lead to analyst fatigue and limited operational deployment [4]. These challenges paved the way for integrating machine learning techniques, which have been shown to detect anomalies more accurately.

Machine learning paradigms in IDS

Machine learning has significantly transformed intrusion detection systems by enabling data-driven modelling of intricate network traffic patterns. These approaches fall into three paradigms: supervised, unsupervised, and semi-supervised learning. Supervised learning dominates intrusion detection research because labelled datasets allow models to learn direct mappings between traffic features and attack categories. As a result, supervised classifiers achieve higher detection accuracy most of the time [5]. Notably, supervised learning depends heavily on the quality and completeness of labelled datasets, which are costly and time-consuming to produce.

Clustering, autoencoders, and density-based models are unsupervised learning methods and are advantageous for detecting unknown or zero-day attacks. They do not require labelled data, but their accuracy is low, and false positives remain an unaddressed issue. Finally, semi-supervised learning combines the strengths of supervised and unsupervised learning, relying on small amounts of labelled data and a large volume of unlabeled traffic. Semi-supervised learning is becoming a prominent option in IDS research for environments where attack labels are scarce [6].

The role of datasets in developing ML-based IDS cannot be overemphasised. Earlier datasets, such as KDD'99 and NSL-KDD, were foundational to early IDS research but are now criticised for their inability to represent modern network conditions and evolving attack patterns. CICIDS2017 and UNSW-NB15 are more recent and include simulations of modern network conditions and diverse attacks, making them

DOI: https://doi.org/10.53704/fujnas.v15i1.1146

the benchmark for most IDS studies in recent years [7]. Despite this progress, no dataset fully captures the scale, diversity, and encryption levels common in modern traffic, creating a gap between experimental research and real-world deployment.

Machine learning algorithms for IDS

Among the ML algorithms are Random Forest (RF), XGBoost, and Support Vector Machines (SVM), which have been the most frequently applied to intrusion detection.

Random forest (RF): This ensemble method constructs multiple decision trees and aggregates their outputs. It is valued for its effectiveness, interpretability, and its resistance to overfitting. RF performs well on high-dimensional data and has proven to detect different types of network intrusions effectively [2]. However, it can become computationally demanding with large datasets, limiting its utility in high-throughput, real-time systems.

XGBoost: Extreme Gradient Boosting (XGBoost) is an advanced boosting algorithm; it has become popular among ML algorithms for its scalability, speed, and high predictive accuracy. Its parallel processing and ability to handle imbalanced datasets make XGBoost a strong option for developing NIDS applications. Studies show that XGBoost consistently outperforms other tree-based classifiers in accuracy and execution speed [8, 7]. However, its sensitivity to noisy features and overreliance on hyperparameter tuning remain key limitations.

Support vector machines (SVM): SVMs perform well in high-dimensional spaces and can handle non-linear decision boundaries using kernel functions, making them another viable option. They are mostly effective in data-scarce scenarios. However, like RF, it suffers from high computational complexity on large-scale datasets, leading to poor scalability in real-time environments [1].

Comparative evaluations

Direct comparisons show that XGBoost has consistently outperformed both RF and SVM in accuracy and speed [9]. While RF remains attractive for its interpretability and resilience to noisy data, SVM has largely fallen out of favour for large-scale applications because of its computational inefficiency, although it remains effective in data-scarce scenarios. Recent work explores hybrid models combining these classical algorithms by leveraging the robustness of RF, the efficiency of XGBoost, and the precision of SVM [10].

Deep learning and hybrid approaches

While classical algorithms such as XGBoost and Random Forest remain highly effective and relevant in IDS development, modern intrusion detection implementations increasingly embrace deep learning (DL) because it can automatically learn complex, hierarchy-based patterns in network traffic. Recent studies show that Convolutional Neural Networks (CNNs) are effective at capturing spatial dependencies, while Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) models perform well in modelling temporal sequences. Combining models such as CNNs and LSTMs can achieve superior detection accuracy, as seen in intrusion datasets like UNSW-NB15 and CICIDS2017 [11]. More recent architectures include attention techniques with CNNs and LSTMs, such as an Attention-CNN-LSTM model that selectively focuses on relevant spatiotemporal patterns and delivers strong performance on the NSL-KDD and Bot-IoT datasets [12]. A systematic review of DL techniques further reiterates the importance of handling spatiotemporal feature extraction and class imbalance and suggests Generative Adversarial Networks (GANs) as a promising augmentation strategy [13]. Hybrid frameworks that combine ML and DL methods have also been explored. XGBoost and CNN have been used for feature selection, and LSTM for classification across multiple datasets, including UNSW-NB15 and CICIDS2017, indicating effective detection rates and lower false positive rates [14]. The application of DL and hybrid approaches in IDS development indicates a shift toward more expressive models to improve detection capabilities, although challenges with explainability and resource demands remain.

Challenges and research gaps

Several obstacles continue to limit the practical application of both ML- and DL-based IDS, despite all the advancements recorded in classifying ML, DL, and Hybrid approaches. Public datasets such as UNSW-NB15, NSL-KDD, and CICIDS2017, while widely used, do not fully capture the diversity, encryption, and traffic volatility that are encountered in modern networks [1]. Moreover, adversarial attacks such as data poisoning and evasion threaten these models' integrity. Researchers have highlighted that vulnerabilities can be exploited to bypass detection.

DOI: https://doi.org/10.53704/fujnas.v15i1.1146

Interpretability remains a challenge, as many DL models act as black boxes, hindering trust and adoption by security professionals. The challenge of scalability is not fully addressed because high computational resource demands render these methods less viable in high-throughput or resource-constrained environments [4]. Finally, most IDS studies lack operational integration with SIEM systems, which leaves practical deployment questions underexplored.

Emerging directions

To address these challenges, ongoing research is working toward more resilient, scalable, and interpretable IDS frameworks. Researchers are applying Explainable AI (XAI) technologies such as SHAP and LIME to IDS to provide transparency in decision-making [1]. Federated Learning (FL) also offers privacy-preserving, distributed model training with optimal results in intrusion detection. Different FL-based IDS frameworks have indicated enhanced data privacy and effectiveness in distributed or IoT contexts, enabling collaborative model updates without centralising sensitive data [15, 16, 17]. An emerging transformer-based federated model, FetFIDS, using feature embeddings and attention mechanisms, also demonstrates strong potential for edge-centric deployment and high accuracy in decentralised environments [18].

Methodology

Dataset and data preprocessing

Dataset selection

The UNSW-NB15 dataset, curated by the Australian Centre for Cyber Security, forms the basis of our experiments. It supersedes older datasets (e.g., KDD99) by encompassing realistic modern attacks including Denial of Service, Exploits, Reconnaissance, Backdoor, and Worms. The dataset features 49 attributes spanning flow-based, time-based, content-based, and host- based properties, providing comprehensive network traffic characterisation suitable for ML modelling.

Preprocessing

To prepare data for modelling:

i. Missing values were imputed using means for numerical and modes for categorical features.
ii. Categorical data, such as protocols (TCP, UDP) and services (HTTP, FTP), were one-hot encoded to maintain numerical compatibility.
iii. Numeric features were normalised with Min-Max scaling as:

$$X' = \frac{\mathrm{X\text{-}X_{min}}}{\mathrm{X_{max}\text{-}X_{min}}}$$

where X is the original feature value, X_{min} and X_{max} are the minimum and maximum values of the feature across the training set, and X_norm is the normalised value bounded in [0, 1].
iv. The dataset was partitioned into training (70%) and testing (30%) subsets.

Feature selection

Feature dimensionality reduction was conducted leveraging XGBoost's feature importance ranking. The top 20 features selected among them, packet duration, source/destination bytes, packet rate, TCP flags, and inter-packet arrival time, were retained to optimise detection accuracy and computational efficiency.

Mathematically, feature selection can be expressed as minimising the loss function L over a selected subset of features F_s, where:

$$L(F_s) = \sum_{i=1}^{N} (yi - f(xi \mid F_s))^2$$

where:

i. y_i is the actual class label,
ii. x_i is the feature vector,
iii. $f(xi|Fs)$ is the classifier function with selected features.

Importantly, feature importance scoring was computed exclusively on the training partition (70% split) prior to any contact with the held-out test set. The top-ranked features were then applied consistently to both training and test data, ensuring no information leakage from the test set into the feature selection process.

Machine learning models

Random forest (RF)

RF constructs multiple decision trees $ht(x)$ and predicts classes by majority vote:

$$\hat{y} = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$$

Key hyperparameters tuned include: 100 trees (n_estimators), max depth 10, minimum samples per split 5.

XGBoost

XGBoost performs gradient boosting by incrementally adding learners fk to minimise the objective:

DOI: https://doi.org/10.53704/fujnas.v15i1.1146

$$L(\theta) = \sum_{i=1}^{N} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

where ℓ is the log loss, and Ω regularises model complexity. Tuned hyperparameters: learning rate 0.05, 300 estimators, max depth 8, subsample ratio 0.8.

Support vector machines (SVM)

SVM finds the hyperplane maximising the margin via:

$$\min_{w,b} \frac{1}{2} \| w \|^2 + Ci \sum \xi i$$

subject to:

$$yi(w \cdot xi + b) \geq 1 - \xi i, \xi i \geq 0$$

where C=1.0, kernel linear.

Hyperparameter optimisation

Hyperparameters for Random Forest and XGBoost were selected using grid search with 3-fold cross-validation applied exclusively to the training partition. For Random Forest, the search space covered n_estimators ∈ {50, 100, 200}, max_depth ∈ {5, 10, 15}, and min_samples_split ∈ {2, 5, 10}. For XGBoost, the search space covered learning_rate ∈ {0.01, 0.05, 0.1}, n_estimators ∈ {100, 200, 300}, max_depth ∈ {4, 6, 8}, and subsample ∈ {0.6, 0.8, 1.0}. We selected the configuration with the highest mean validation accuracy. For SVM, we used a linear kernel with C = 1.0 without grid search, justified by the high-dimensional, linearly separable nature of the feature space and the prohibitive training cost of kernel-based grid search at this dataset scale.

Evaluation metrics

We evaluated models on accuracy, precision, recall, F1-score, and computational time (training and prediction). Definitions follow standard measures:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$Pr\,e\,cision = \frac{TP}{TP + FP}$$

$$Re\,c\,all = \frac{TP}{TP + FN}$$

$$F_1\text{-Score} = \quad F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

where TP = True Positives, TN = True Negatives, FP = False Positives, FN = False Negatives.

Training time and prediction time are reported in seconds (s).

Experimentation and results

Model performance comparison

Table 1 summarises the performance of all four model variants evaluated in this study. Optimised XGBoost achieved the highest accuracy (87.18%) alongside the fastest training time (35.2 s) and prediction time (0.16 s), making it the most suitable candidate for real-time NIDS deployment. Random Forest achieved comparable accuracy (86.95%) but required a longer training time (45.3 s). Baseline XGBoost performed similarly to Random Forest in accuracy (86.84%) but with faster execution. SVM showed lower efficiency and accuracy than the other evaluated models, mainly because its training complexity is quadratic on large-scale datasets; it took 231.5 s to train and achieved 75.49% accuracy.

Table 1: Performance comparison of ML models

Evaluation Criteria	Random Forest	XGBoost	Optimized XGBoost	SVM
Accuracy	86.95%	86.84%	87.18%	75.49%
False Positives	9,825	9,764	9,764	18,130
False Negatives	917	792	792	1,814
Training Time (s)	45.3	38.7	35.2	231.5
Prediction Time (s)	0.23	0.18	0.16	5.12

Classification reports and confusion matrices

To further analyse the models' performance, the classification reports and confusion matrices are as follows. Figure 1 describes the Random Forest confusion matrix, which displays the classification performance of Random Forest, showing a high number of true positives (44,415) along with 9,825 false positives and 917 false negatives.

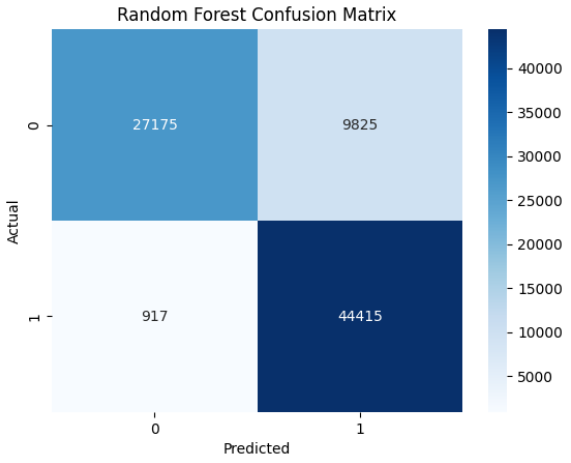


Figure 1. Random forest confusion matrix

DOI: https://doi.org/10.53704/fujnas.v15i1.1146

Figure 2 describes the SVM Confusion Matrix, which details the error profile for SVM and illustrates its relatively weak performance, with 18,130 false positives and 1,814 false negatives.

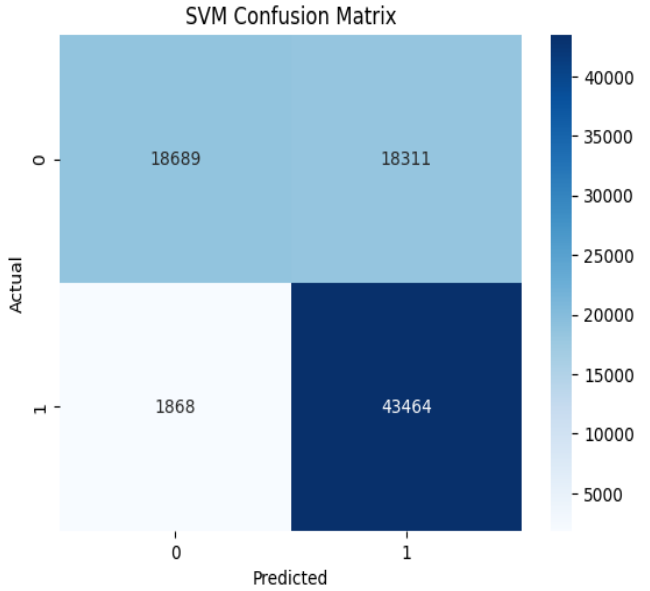


Figure 2. SVM confusion matrix

Figure 3 describes the XGBoost confusion matrix, which highlights standard XGBoost performance, showing a slight reduction in false negatives to 918 while maintaining false positives at 9,917.

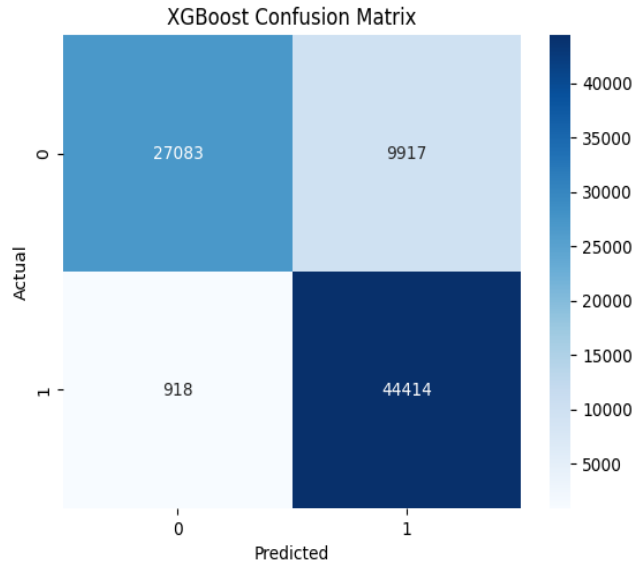


Figure 3. XGBoost confusion matrix

Figure 4 describes the Optimised XGBoost Confusion Matrix, which demonstrates the best overall classification balance, achieving the lowest false positive count of 9,764 and manageable false negatives of 792.

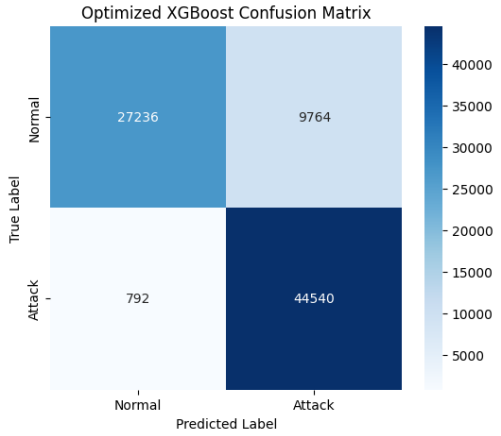
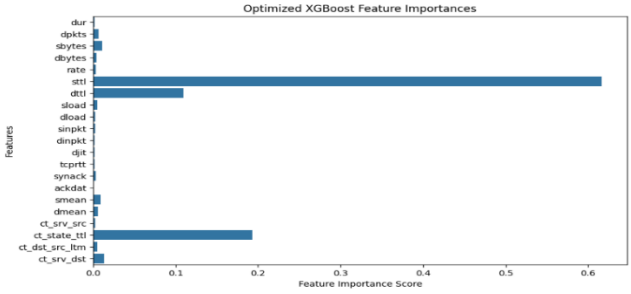


Figure 4. Optimised XGBoost confusion matrix

The confusion matrices reveal that SVM performs the poorest due to high misclassifications, while Random Forest and baseline XGBoost catch most attacks at the cost of frequent false alarms. Optimised XGBoost achieves the ideal balance, minimising missed threats while keeping false alarms low for reliable deployment.

Feature importance analysis

As shown in Figure 5, the feature importance plot for the Optimised XGBoost model indicates the key network traffic attributes that most influence intrusion detection decisions. Packet duration stands out as the most significant feature, highlighting how attackers frequently manipulate packet lengths to evade detection mechanisms. Source and destination bytes also rank highly, as unusual data transfer sizes often indicate malicious activity. TCP flags are also important because several attack types exploit the TCP handshake process to initiate or hide intrusions. Additionally, inter-packet arrival time played a major role, as sudden bursts or irregular packet timing can signal denial-of-service (DoS) attacks or other network anomalies. Combining flow-based, packet-level, and temporal features enabled the Optimised XGBoost model to effectively distinguish between benign and malicious network traffic, achieving strong detection accuracy while maintaining computational efficiency.



DOI: https://doi.org/10.53704/fujnas.v15i1.1146

Figure 5. Feature importance plot for xgboost

ROC and precision-recall curves

Figure 6 shows the Receiver Operating Characteristic (ROC) curve for the combined models, with the Optimised XGBoost model showing its strong ability to distinguish between benign and malicious network traffic. The ROC curve plots the true positive rate against the false positive rate across all classification thresholds. As illustrated in Figure 6, Optimised XGBoost achieves an Area Under the Curve (AUC) of 0.94, followed by Random Forest (AUC = 0.93) and SVM (AUC = 0.81), confirming the superior discriminative ability of the ensemble-based approaches under the UNSW-NB15 experimental conditions. This reflects the model's effectiveness in correctly detecting intrusions while minimising false alarms across different threshold settings.

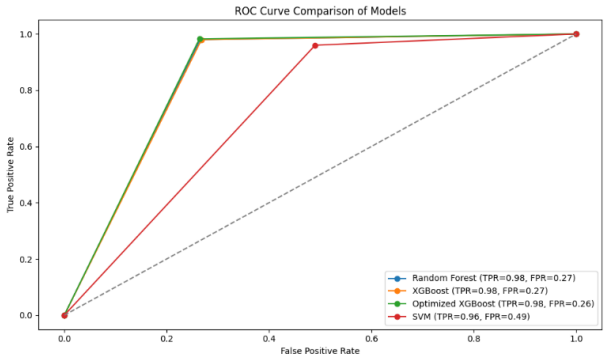


Figure 6. ROC curve for the combined models

Figure 7 presents the Precision-Recall curve for the combined model; it offers more insight into the model's performance on imbalanced intrusion-detection data.

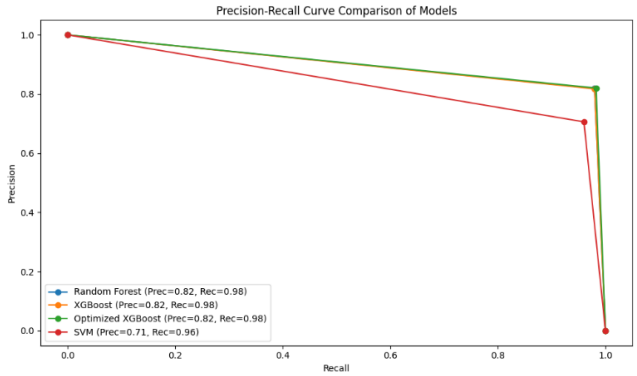


Figure 7. Precision-recall curve for combined models

The curve indicates that the Optimised XGBoost maintains high precision across recall levels, meaning it identifies a large share of actual intrusions with high recall without excessive false positives. This balance matters in cybersecurity, where missing real attacks can be consequential and analysts can be overwhelmed by false alarms. Together, these curves show that the Optimised XGBoost model effectively manages the trade-off between detection sensitivity and specificity, making it reliable for real-time network intrusion detection applications.

Execution time comparison

Because real-time intrusion detection requires fast processing, execution time is a crucial factor in model selection.

Figure 8 shows the execution time for training and prediction for the machine learning models used in network intrusion detection. Among the models evaluated, Optimised XGBoost performed the fastest, with a training time of 35.2 seconds and a prediction time of 0.16 seconds, making it highly suitable for real-time applications. Random Forest trailed behind, with slightly longer training times, but it remained efficient within acceptable limits. In contrast, SVM is the least efficient, requiring a substantial 231.5 seconds for training, which makes it less suitable for real-time network security deployment. While prediction times are minimal across models, SVM returned a notably higher prediction time, as shown in red on the chart. Overall, this comparison indicates that XGBoost offers a better balance of speed and performance for intrusion detection tasks.

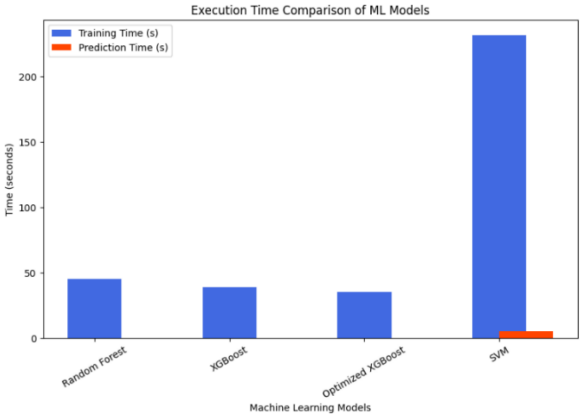


Figure 8. Execution time bar chart

Summary of results

Key performance metrics summarised alongside computational costs show Optimised XGBoost as the most balanced for practical NIDS in Table 1.

Key findings

The key takeaway is that Optimised XGBoost delivered the highest accuracy (87.18%) with the fastest execution, making it the best choice for real-time deployment, as shown in Table 1. Random Forest showed comparable accuracy to Optimised XGBoost

DOI: https://doi.org/10.53704/fujnas.v15i1.1146

but was slower; SVM's high computational cost made it impractical despite reasonable recall. Feature selection further improved efficiency without reducing accuracy, confirming XGBoost as the most balanced and effective model.

Discussion

This discussion section interprets the experimental findings and links them to the aim of intrusion detection research. Each subsection addresses specific aspects of practicability and performance of the models tested.

Accuracy versus execution speed

The experiments confirmed that Optimised XGBoost delivered the best balance between accuracy and computational efficiency. With an achievement of 87.18%, the highest accuracy, while maintaining the fastest training and prediction times of 35.2s and 0.16s, respectively. Random Forest achieved an accuracy of 86.95% but required a slightly longer training time of 45.3s. SVM showed significantly lower accuracy (75.49%) and a very high computational cost, with training time lasting over 230 seconds. In real-world NIDS, where both speed and accuracy are essential, Optimised XGBoost indicates clear superiority.

False positives and negatives

Intrusion detection must minimise false positives and false negatives. Optimised XGBoost achieved the lowest false negatives (792), making it less likely to miss real attacks. Random Forest showed slightly higher false negatives (917) and false positives (9,825). SVM's poor performance was clear, with a higher false positive rate of 18,130 and false negatives of 1,814, which severely affects its reliability. From an operational standpoint, reducing false negatives is more important because an undetected intrusion poses a huge threat.

Feature impact

Feature importance analysis showed that packet duration, source and destination bytes, TCP flags, and inter-packet arrival time are important indicators of malicious activity. These findings align with known dataset (UNSW-NB15) limit the generalisability of the findings to other network environments and traffic distributions.

Deployment Recommendations

attack behaviours, such as packet flooding in denial-of-service attacks or abnormal data transfer in exfiltration attempts. Focusing on discriminative features not only improves detection accuracy but also increases efficiency by reducing unnecessary dimensionality.

Precision-recall and real-world use

The precision-recall trade-off matters in cybersecurity: excess false alarms can overwhelm analysts, and missed attacks can be destructive. Optimising XGBoost to maintain high recall while preserving precision provides a good balance for operational use. Random Forest performed well, but it produced more false positives. SVM performed the worst, with low precision at high recall levels. These results show that Optimised XGBoost is best suited for deployment in live environments where both detection and manageable or lower alert volumes are required.

Computational efficiency

Beyond predictive performance, computational efficiency is also important in real-time NIDS. Optimised XGBoost consistently outperforms other models by delivering the fastest training and inference times. RF was slower but was still within acceptable limits for many enterprise networks. However, the computational demands made SVM unsuitable for real-time use, especially in high-throughput networks.

Limitations

While Optimised XGBoost was deemed the most effective, the study has its limitations. First, results were obtained using just the UNSW-NB15 dataset; this may not fully capture the diversity of real-world traffic. Additionally, the study did not evaluate adversarial robustness; attackers can manipulate inputs to evade detection. Lastly, the study measured performance in a controlled environment, not under live traffic. These limitations point to areas for future research. Additionally, the study does not evaluate adversarial robustness; an adversarial actor may manipulate input features to evade detection, and the models' vulnerability to such attacks has not been assessed. Furthermore, the absence of cross-validation across all models and the use of a single

For real-time NIDS deployment, Optimised XGBoost is recommended for its better balance of accuracy, speed, and false-negative reduction. Random Forest can also perform in settings where interpretability is as important, due to its simpler

DOI: https://doi.org/10.53704/fujnas.v15i1.1146

decision-tree structure. SVM should not be used in a large-scale environment because of scalability issues. Additionally, integrating feature selection into deployment pipelines can further improve efficiency, and models should be embedded within broader security frameworks such as SIEM systems to maximise operational utility.

Future research

Future studies should validate these findings using additional datasets, such as CICIDS2017 or real-world enterprise traffic, to enhance generalizability. The architecture based on deep learning, such as CNNs, LSTMs, and Transformers, which have better potential of capturing temporal dependencies in traffic patterns, should be explored. Hybrid ensemble approaches, such as combining RF and XGBoost, could further improve effectiveness. Addressing adversarial attacks through robust training techniques should also be a priority. Finally, real-world testing on live traffic and integration with cloud or edge computing platforms would provide stronger evidence of operational readiness.

Conclusion

This study compared Random Forest, XGBoost, and SVM for network intrusion detection using the UNSW-NB15 dataset, with equal emphasis on detection accuracy and computational efficiency. Optimised XGBoost achieved the highest accuracy (87.18%) and the fastest execution (training: 35.2 s; prediction: 0.16 s), making it the most suitable model for real-time NIDS deployment. Random Forest offered comparable accuracy (86.95%) but at a higher computational cost, while SVM was impractical due to excessive training time (231.5 s) and reduced accuracy (75.49%). Feature importance analysis identified packet duration, source/destination bytes, TCP flags, and inter-packet arrival time as the most discriminative indicators of malicious activity, confirming the value of principled feature selection. Optimised XGBoost is recommended for operational NIDS; future work should validate findings on additional datasets, investigate deep learning and hybrid ensemble architectures, and evaluate adversarial robustness under realistic threat conditions.

Data Availability

No additional data or materials are available beyond those included in the manuscript.

Funding

No funding was received

Contributions

Idris Ibraheem conceptualised the study and designed the methodology. Muhammad Tijjani Jidda and Idris Ibraheem supervised the research process. Idris, Ahmed and Abdulrasaq developed the codes and conducted technical validations. Samuel Kwabla Segbefia and Muhammad Tijjani Jidda provided critical insights into the analysis of the results. All the authors reviewed and approved the final manuscript.

Conflict of interest

There is no conflict of interest

References

1. Jacob SL, Habibullah PS. A systematic analysis and review on intrusion detection systems using machine learning and deep learning algorithms. *J Comput Cogn Eng*. 2024. doi:10.47852/bonviewJCCE42023249.
2. Alzahrani AO, Alenazi MJF. Designing a network intrusion detection system based on machine learning for software defined networks. *Future Internet*. 2021. doi:10.3390/fi13050111.
3. Golande SV, Sanket V, Aniket P, Vivekanand K, Vedant P. An efficient network intrusion detection and classification system using machine learning. *Int J Adv Res Sci Commun Technol*. 2024. doi:10.48175/IJARSCT-22045.
4. Leon M, Tijana M, Punnekkat S. Comparative evaluation of machine learning algorithms for network intrusion detection and attack classification. *IEEE Int Joint Conf Neural Netw*. 2022. doi:10.1109/IJCNN55064.2022.9892293.
5. Hossain A, Islam S. Ensuring network security with a robust intrusion detection system using ensemble-based machine learning. *Array*. 2023. doi:10.1016/j.array.2023.100306.
6. Himthani P, Dubey GP. Application of machine learning techniques in intrusion detection systems: a systematic review. In: *Proceedings of the Third International Conference on Sustainable Computing (SUSCOM 2021)*. Singapore Springer Nature; 2022. p. 97–105. doi:10.1007/978-981-16-4538-9_10.
7. Talukder A, Islam M, Uddin A, Hasan KF, Sharmin S. Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction. *J Big Data*. 2024. doi:10.1186/s40537-024-00886-w.
8. Zhang C, Jia D, Wang L, Wang W, Liu F. Comparative research on network intrusion detection methods based on machine learning. *Comput Secur*. 2022. doi:10.1016/j.cose.2022.102861.
9. Maseer ZK, Yusof R, Bahaman N, Mostafa SA, Mohd Foozy CF. Benchmarking of machine learning for anomaly-based intrusion detection systems in the CICIDS2017 dataset. *IEEE Access*. 2021. doi:10.1109/ACCESS.2021.3056614.
10. Rege PR, Aarti K, Anishkumar D, Rahul S, Rupali SK. Exploring machine learning's role in intrusion detection systems for network security. In: *2024 International Conference on Emerging Smart Computing and Informatics (ESCI)*. 2024. doi:10.1109/ESCI59607.2024.10497357.

DOI: https://doi.org/10.53704/fujnas.v15i1.1146

11. Sun Y, Liu Z, Li G, Wang H. A hybrid CNN–LSTM model for network intrusion detection. *IEEE Access*. 2020;8:137361–137371. doi:10.1109/ACCESS.2020.3009843.
12. Alashjaee AM. Deep learning for network security: an Attention-CNN-LSTM model for accurate intrusion detection. *Sci Rep*. 2025;15:21856. doi:10.1038/s41598-025-07706-y.
13. Zhang Y, Muniyandi RC, Qamar F. A review of deep learning applications in intrusion detection systems: overcoming challenges in spatiotemporal feature extraction and data imbalance. *Appl Sci*. 2025;15(3):1552. doi:10.3390/app15031552.
14. Sajid M, Malik KR, Almogren A, Malik TS, Khan AH, Tanveer J, Rehman AU. Enhancing intrusion detection: a hybrid machine and deep learning approach. *J Cloud Comput*. 2024;13(1):123. doi:10.1186/s13677-024-00685-x.
15. Buyuktanir B, Altinkaya Ş, Karatas Baydogmus G, et al. Federated learning in intrusion detection: advancements, applications, and future directions. *Clust Comput*. 2025;28:473. doi:10.1007/s10586-025-05325-w.
16. Hernandez-Ramos JL, Karopoulos G, Chatzoglou E, Kouliaridis V, Marmol E, Gonzalez-Vidal A, Kambourakis G. Intrusion detection based on federated learning: a systematic review. *ACM Comput Surv*. 2025;57(12):1–65. doi:10.1145/3731596.
17. Agrawal S, Sarkar S, Aouedi O, Yenduri G, Piamrat K, Alazab M, et al. Federated learning for intrusion detection system: concepts, challenges and future directions. *Comput Commun*. 2022;195:346–361. doi:10.48550/arXiv.2106.09527.
18. Ghosh S, Jameel ASMM, Gamal AE. FetFIDS: a feature embedding attention-based federated network intrusion detection algorithm. *arXiv Preprint*. 2025. arXiv:2508.09056. doi:10.48550/arXiv.2508.0

DOI: https://doi.org/10.53704/fujnas.v15i1.1146