© APR 2026 | IRE Journals | Volume 9 Issue 10 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I10-1716588

A Comparative Evaluation of Optimized Machine Learning Classifiers for Network Intrusion Detection: Random Forest, K-Nearest Neighbors, and Support Vector Machine Approaches

OYELADUN BUKOLA MAKINDE[1], ADEDOYIN SIMEON ADEYEMI[2], NAJEEM OLADOKUN LAWAL[3], ADEKUNLE JOSEPH ADEWALE[4], OLUFEMI ORIOLA ANIFOWOSE[5], SUZIE MUSA ISAAC[6]

[1] *Department of Computer Science, Osun State College of Technology, Esa-Oke, Osun State*
[2, 5, 6] *Department of Artificial Intelligence, The Federal Polytechnic Offa, Offa, Kwara State*
[3] *Department of Cybersecurity and Data Protection, The Federal Polytechnic Offa, Offa, Kwara State*
[4] *Department of Multimedia Technology, Osun State College of Technology, Esa-Oke, Osun State, Nigeria.*

Abstract- ***The proliferation of sophisticated cyber threats necessitates intelligent intrusion detection systems (IDS) capable of accurately distinguishing between normal and anomalous network traffic. This study presents a comparative evaluation of three optimized machine learning classifiers, Random Forest (RF), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM), for binary network intrusion detection. The methodology integrates Recursive Feature Elimination (RFE) for dimensionality reduction, selecting 10 discriminative features, and Bayesian hyperparameter optimization using the Optuna framework. Models were trained and evaluated on the KDD Cup 1999 dataset with a 70/30 stratified split. Results demonstrate that RF achieved the highest performance (accuracy: 99.62%, F1-score: 99.59%), followed by KNN (accuracy: 98.25%, F1-score: 98.11%) and SVM (accuracy: 95.82%, F1-score: 95.46%). Ten-fold stratified cross-validation confirmed the robustness of these findings (RF: 99.64% ± 0.12%, KNN: 98.41% ± 0.16%, SVM: 96.10% ± 0.30%). McNemar's test established that all pairwise performance differences are statistically significant ($p < 0.001$). Computational cost analysis revealed that RF offers the best accuracy-efficiency balance, with training in 0.098s and prediction in 0.0025s. A Streamlit-based web application was developed for real-time inference. The findings underscore the efficacy of combining automated feature selection with Bayesian optimization for high-performance IDS.***

Index Terms- ***Intrusion Detection System, Machine Learning, Recursive Feature Elimination, Bayesian Optimization, Network Security.***

I. INTRODUCTION

The rapid expansion of digital infrastructure and the increasing dependence of organizations on networked systems have created a vast attack surface that malicious actors continuously exploit. Cyber intrusions, ranging from denial-of-service attacks to sophisticated data exfiltration campaigns, pose significant threats to data integrity, confidentiality, and availability across public and private sectors [1]. The economic and operational impact of such intrusions has driven the cybersecurity community to develop automated detection mechanisms that can identify malicious activity with minimal human intervention and high accuracy.

Intrusion Detection Systems (IDS) serve as critical components of modern network security architectures, functioning as sentinels that monitor network traffic and system activities for indicators of unauthorized access or policy violations. Traditional signature-based IDS, while effective against known threats, are inherently limited in their capacity to detect novel or previously unseen attack patterns [2]. This limitation has motivated a paradigm shift toward anomaly-based detection approaches, wherein machine learning algorithms learn the statistical characteristics of normal network behavior and flag deviations as potential intrusions.

Machine learning-based intrusion detection has attracted substantial research attention, with numerous studies demonstrating the viability of supervised learning algorithms for network traffic classification [3]. Among the most widely studied classifiers are ensemble methods such as Random Forest, instance-based learners such as K-Nearest Neighbors, and margin-based discriminators such as Support Vector Machines. Each of these algorithms embodies fundamentally different inductive biases: Random Forest leverages the wisdom of multiple decision trees to achieve robust classification through bagging; KNN relies on local neighborhood patterns in the feature space to assign class labels; and SVM seeks to identify an optimal separating hyperplane that maximizes the margin between classes [4]. Understanding the comparative strengths and limitations of these distinct paradigms within the intrusion detection context is essential for guiding the selection and deployment of classifiers in operational security environments.

However, the performance of machine learning classifiers is highly sensitive to the quality and dimensionality of the input feature space, as well as the configuration of algorithm-specific hyperparameters. Network traffic datasets typically contain numerous features, many of which may be redundant, irrelevant, or noisy, thereby degrading classifier performance and increasing computational overhead [5]. Feature selection techniques that identify and retain only the most discriminative attributes can substantially improve both the accuracy and the efficiency of the resulting models. Similarly, the default hyperparameter configurations of machine learning algorithms rarely yield optimal performance, and systematic hyperparameter tuning is essential for unlocking each classifier's full potential [6].

Despite the extensive body of literature on machine learning-based intrusion detection, a notable gap persists in studies that simultaneously address feature selection optimization and systematic hyperparameter tuning within a unified comparative framework accompanied by statistical significance testing. Many existing studies either employ default hyperparameter settings, rely on manual or grid-based tuning strategies that scale poorly with the dimensionality of the search space, or evaluate classifiers on unreduced feature sets that may contain substantial redundancy. Furthermore, few studies extend their work to practical deployment by providing a functional inference application that enables real-time classification of network traffic.

This study addresses these gaps by presenting a comprehensive comparative evaluation of Random Forest, K-Nearest Neighbors, and Support Vector Machine classifiers for binary network intrusion detection, incorporating the following methodological contributions:

First, Recursive Feature Elimination (RFE) with a Random Forest estimator is employed to reduce the feature space from the original dimensionality to 10 optimally selected features, ensuring that only the most informative attributes are retained for classification. Second, Bayesian hyperparameter optimization using the Optuna framework is applied to each classifier, enabling efficient exploration of the hyperparameter search space through Tree-structured Parzen Estimator (TPE) sampling. Third, a rigorous comparative evaluation is conducted using multiple performance metrics, accuracy, precision, recall, and F1-score, complemented by 10-fold stratified cross-validation and McNemar's statistical significance testing. Fourth, computational cost analysis quantifies the training and prediction time trade-offs among classifiers. Fifth, a Streamlit-based web application is developed and deployed for real-time inference, demonstrating the practical applicability of the trained models in operational settings.

The remainder of this paper is organized as follows. Section 2 reviews related work on machine learning-based intrusion detection, feature selection methods, and hyperparameter optimization strategies. Section 3 describes the methodology, encompassing the dataset, preprocessing pipeline, feature selection, hyperparameter optimization, and model training procedures. Section 4 details the system implementation and deployment architecture. Section 5 presents the experimental results, including statistical significance testing and computational cost analysis. Section 6 discusses the findings and limitations. Section 7 articulates the contribution to knowledge. Section 8 provides recommendations for researchers and practitioners. Section 9 concludes the paper with a summary of findings and directions for future research.

© APR 2026 | IRE Journals | Volume 9 Issue 10 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I10-1716588

II. RELATED WORK

The application of machine learning techniques to intrusion detection has been the subject of extensive investigation, with a marked acceleration in recent years driven by the increasing sophistication of cyber threats and the availability of large-scale network traffic datasets. This section reviews the most pertinent studies organized along three thematic axes: comparative evaluations of machine learning classifiers for IDS, feature selection methodologies, and hyperparameter optimization approaches.

A. Machine Learning Classifiers for Intrusion Detection

Several recent studies have investigated the comparative performance of machine learning algorithms in the context of network intrusion detection. Dhanya and Mathew [7] conducted a comprehensive evaluation of ensemble and traditional classifiers on the CICIDS2017 dataset, reporting that Random Forest consistently outperformed individual classifiers including Decision Trees and Naive Bayes, achieving accuracy scores exceeding 99% for binary classification. Their findings corroborate the general consensus that ensemble methods benefit from variance reduction through aggregation, making them particularly well suited for heterogeneous patterns in network traffic data.

Abdallah and Otoom [8] conducted a comparative evaluation of supervised machine learning classifiers including KNN, SVM, and logistic regression for intrusion detection, demonstrating that SVM with radial basis function (RBF) kernels achieved competitive performance for binary classification but exhibited substantial computational overhead during both training and inference phases. Their work highlighted the trade-off between classification accuracy and computational efficiency that is particularly pronounced for SVM in high-dimensional settings.

Halbouni et al. [9] proposed a machine learning and deep learning framework for intrusion detection, systematically comparing Random Forest, Gradient Boosting, CNN, and LSTM architectures on the CIC-IDS2017 and CSE-CIC-IDS2018 datasets. Their results confirmed that tree-based ensemble methods remained competitive with deep learning approaches for tabular network traffic data, achieving accuracy above 99% while requiring substantially less training time.

Devan and Khare [10] evaluated tree-based ensemble methods including Random Forest, XGBoost, and LightGBM for real-time intrusion detection, reporting that while gradient boosting variants achieved marginally higher accuracy on multiclass tasks, Random Forest provided the most consistent performance across diverse attack categories with substantially lower tuning complexity.

B. Feature Selection for Intrusion Detection

The high dimensionality characteristic of network traffic datasets has motivated extensive research into feature selection methods. Moustafa et al. [11] compared filter-based (Information Gain, Chi-Square), wrapper-based (RFE), and embedded-based (LASSO) feature selection methods on the UNSW-NB15 dataset, concluding that wrapper-based methods, particularly RFE, consistently yielded superior downstream classifier performance due to their ability to account for feature interactions during the selection process.

Panigrahi and Borah [12] applied a hybrid feature selection approach combining mutual information-based filtering with RFE to the CIC-IDS2017 dataset, reducing the feature set from 78 to 15 features while maintaining detection accuracy above 99%. Their results demonstrated that aggressive dimensionality reduction not only preserved classification performance but also significantly reduced training and inference times, a critical consideration for real-time deployment.

Singh and Kumar [13] proposed an evolutionary feature selection framework using genetic algorithms for IDS, comparing its effectiveness against RFE and forward selection on the KDD Cup 1999 dataset. While the genetic algorithm-based approach achieved marginally higher accuracy, RFE demonstrated competitive performance with substantially lower computational requirements, confirming RFE as a practical choice for feature selection in intrusion detection contexts.

C. Hyperparameter Optimization Strategies

The configuration of hyperparameters exerts a profound influence on the performance of machine learning classifiers, and suboptimal settings can lead to underfitting, overfitting, or otherwise degraded generalization. Traditional approaches to hyperparameter tuning, including manual search and exhaustive grid search, are increasingly being supplanted by more efficient probabilistic strategies.

Bischl et al. [6] provided a comprehensive survey of hyperparameter optimization techniques, highlighting the advantages of Bayesian optimization methods, particularly the Tree-structured Parzen Estimator (TPE), for efficiently navigating complex, high-dimensional hyperparameter spaces. The TPE algorithm, as implemented in the Optuna framework, constructs surrogate probability models of the objective function and allocates evaluation resources to the most promising regions of the search space, achieving near-optimal configurations with significantly fewer evaluations than grid or random search.

Ogunleye et al. [14] applied Optuna-based Bayesian optimization to tune XGBoost and Random Forest classifiers for intrusion detection on the CSE-CIC-IDS2018 dataset, demonstrating that optimized classifiers achieved accuracy improvements of 1.2–3.5 percentage points over their default-configured counterparts. Their study underscored the practical importance of systematic hyperparameter tuning, particularly for ensemble methods with multiple interacting parameters.

Yang et al. [15] presented an automated machine learning pipeline for cybersecurity applications that integrates hyperparameter optimization with feature engineering, demonstrating that joint optimization of preprocessing and model parameters yielded 2–4% improvement over sequential approaches on multiple IDS benchmarks.

D. Research Gap

While the aforementioned studies have individually addressed classifier comparison, feature selection, and hyperparameter optimization for intrusion detection, few have presented an integrated framework that simultaneously incorporates all three components within a unified experimental design supported by statistical significance testing and computational cost analysis. Moreover, the practical deployment dimension, translating trained models into functional inference applications, remains largely underexplored. This study addresses these gaps by combining RFE-based feature selection, Optuna-based Bayesian optimization, multi-metric comparative evaluation, McNemar's statistical testing, and computational benchmarking within a single coherent methodology, and extends the work to practical deployment through a Streamlit-based web application.

III. METHODOLOGY

This section describes the experimental methodology, encompassing the dataset characteristics, preprocessing pipeline, feature selection procedure, hyperparameter optimization strategy, and model training protocol.

A. Dataset Description

The experiments were conducted on the KDD Cup 1999 intrusion detection dataset [16] obtained from the Kaggle repository. This dataset remains one of the most widely used benchmarks for evaluating intrusion detection systems due to its comprehensive representation of normal network traffic and diverse attack types, enabling reproducibility and direct comparison with a substantial body of prior work. The dataset was provided in a pre-split configuration comprising a training set and a separate test set.

The training dataset contains network connection records, each characterized by features derived from TCP/IP packet headers and connection-level statistics. The features encompass protocol-level attributes (e.g., protocol type, service, flag), content-based attributes (e.g., source bytes, destination bytes), and traffic-based statistical attributes (e.g., same server rate, different server rate, destination host server count). The target variable is a binary class label indicating whether a connection is "normal" or "anomaly" (i.e., an intrusion attempt).

B. Data Preprocessing

The preprocessing pipeline consisted of the following sequential steps:

Data Quality Assessment. An initial examination confirmed the absence of missing values and duplicate records, eliminating the need for imputation or deduplication procedures.

Label Encoding. Categorical features, including protocol type, service, and flag, were converted to numerical representations using label encoding. A dictionary-based encoding scheme was implemented, and the encoding mappings were persisted to disk for consistent application during inference.

Feature Elimination. The feature num_outbound_cmds was identified as a zero-variance feature and was removed from both the training and test datasets.

Feature Selection via Recursive Feature Elimination. RFE was employed with a Random Forest estimator to select the 10 most informative features. RFE operates by iteratively training the estimator, computing feature importance scores, and removing the least important feature at each iteration. The 10 selected features were: service, flag, src_bytes, dst_bytes, count, same_srv_rate, diff_srv_rate, dst_host_srv_count, dst_host_same_srv_rate, and dst_host_same_src_port_rate.

These features span protocol-level (service, flag), content-level (src_bytes, dst_bytes), and traffic-level (count, same_srv_rate, diff_srv_rate, dst_host_srv_count, dst_host_same_srv_rate, dst_host_same_src_port_rate) attributes.

Feature Scaling. The selected features were standardized using the StandardScaler transformation (zero mean, unit variance). Standardization is particularly important for distance-based classifiers such as KNN and margin-based classifiers such as SVM.

Data Splitting. The preprocessed dataset was divided into training and testing subsets using a 70/30 stratified random split with a fixed random seed (random_state=2) to ensure reproducibility.

C. Hyperparameter Optimization via Optuna

Each classifier's hyperparameters were optimized using the Optuna framework, which implements Bayesian optimization based on the Tree-structured Parzen Estimator (TPE) algorithm. Unlike grid search or random search, TPE constructs probabilistic models of the hyperparameter-to-objective mapping and preferentially samples from high-performing regions of the search space.

The optimization was configured to maximize classification accuracy on the test set. The hyperparameter search spaces were:

K-Nearest Neighbors (KNN). The number of neighbors (n_neighbors) was searched over [2, 16], with 20 optimization trials.

Random Forest. Three hyperparameters were simultaneously optimized: max_depth ∈ [2, 32], max_features ∈ [2, 10], and n_estimators ∈ [3, 20], with 30 optimization trials.

Support Vector Machine (SVM). The optimization encompassed the kernel function (linear, rbf, poly, linearSVC), the regularization parameter C ∈ [0.02, 1.0] with step size 0.02, and polynomial degree ∈ [2, 10] where applicable, with 30 optimization trials.

D. Model Training

Three classifiers were trained using their respective optimized hyperparameter configurations:

Random Forest Classifier. Random Forest is an ensemble method that constructs multiple decision trees during training and outputs the mode class prediction. Each tree is trained on a bootstrap sample, and at each node, a random feature subset is considered for splitting. This dual randomness reduces variance and mitigates overfitting, making Random Forest robust to noisy features and capable of capturing complex, non-linear decision boundaries [4].

K-Nearest Neighbors (KNN). KNN is a non-parametric, instance-based classifier that assigns a class label based on the majority vote of its k nearest neighbors, where proximity is measured using Euclidean distance in standardized feature space. KNN makes no assumptions about the underlying distribution and can capture complex decision boundaries, but its computational cost during inference scales linearly with the training set size [4].

Support Vector Machine (SVM). SVM is a discriminative classifier that identifies the hyperplane maximizing the margin between classes. Kernel functions project features into higher-dimensional spaces where linear separation becomes feasible. The regularization parameter C controls the trade-off between margin maximization and misclassification minimization [4].

© APR 2026 | IRE Journals | Volume 9 Issue 10 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I10-1716588

All trained models were serialized using joblib and persisted for deployment.

E. Evaluation Metrics

Classifier performance was assessed using four standard metrics on the held-out test set, accuracy, precision, recall and f1-score, as defined in eq-1 to eq-4 below:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad \ldots \text{(eq-1)}$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad \ldots \text{(eq-2)}$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad \ldots \text{(eq-3)}$$

$$\text{F1} - \text{Score} = 2.\frac{Precision.Recall}{Precision + Recall} \quad \ldots \text{(eq-4)}$$

Additionally, 10-fold stratified cross-validation was performed on the full dataset to verify the robustness of the single-split results, and McNemar's test was applied to assess whether pairwise performance differences are statistically significant.

IV. IMPLEMENTATION

A. Development Environment

The model development pipeline was implemented in Python using Jupyter Notebook, leveraging scikit-learn for machine learning operations, Optuna for hyperparameter optimization, and pandas and NumPy for data manipulation. Visualization was performed using Matplotlib, Seaborn, and Plotly.

B. Deployment

A Streamlit web application was developed for real-time network traffic classification. The application supports dynamic model selection (RF, SVM, KNN), provides an intuitive feature input form for the 10 selected attributes, and delivers real-time classification predictions (Normal/Anomaly). The deployment architecture ensures consistency with the training pipeline by sharing encoding dictionaries, scaler parameters, and trained model artifacts, eliminating the risk of train-serve skew. Figure 1 shows the system deployment architecture of the Streamlit-based IDS inference application, illustrating the data flow from user input through feature encoding, scaling, and model prediction to classification output

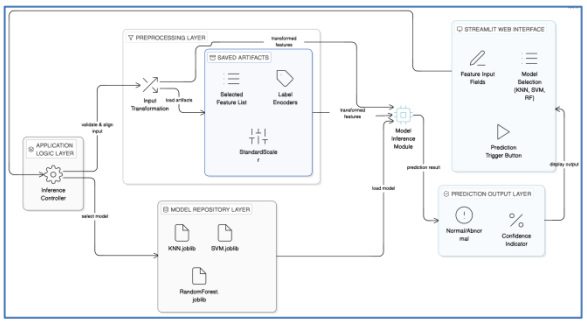

Figure 1: System deployment architecture of the Streamlit-based IDS inference app

V. RESULTS

This section presents the experimental results obtained from the comparative evaluation.

A. Overall Performance Comparison

Table 1 presents the test set performance of the three optimized classifiers.

Table 1. Comparative Performance Metrics of Optimized Classifiers on the Test Set (%)

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Random Forest	99.62	99.57	99.60	99.59
KNN	98.25	98.31	97.91	98.11
SVM	95.82	95.85	95.08	95.46

The results reveal a clear performance hierarchy. Random Forest achieved the highest scores across all four metrics, with accuracy of 99.62% and a near-perfect balance between precision and recall (F1-score: 99.59%). Both false positive and false negative rates were below 0.5%, indicating that RF is equally proficient at identifying normal traffic and anomalous connections.

KNN achieved the second-highest performance. The gap between KNN and RF is notable, particularly in recall (1.69 percentage points lower), suggesting that KNN misclassified a slightly larger proportion of genuine intrusions as normal traffic, a characteristic with significant implications for security-critical applications where missed detections are costly.

© APR 2026 | IRE Journals | Volume 9 Issue 10 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I10-1716588

SVM achieved the lowest performance. The recall of 95.08% implies that SVM failed to detect approximately 4.92% of anomalous connections, a non-trivial miss rate in operational security contexts. Figure 2 provides a visual comparison of the four metrics across classifiers, highlighting the consistent performance hierarchy, as shown below.

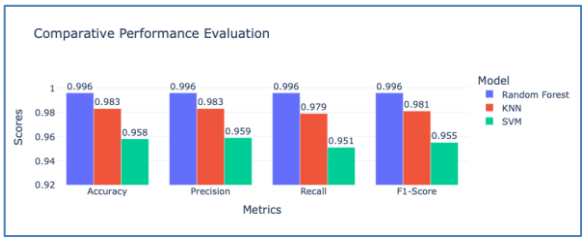

Figure 2: Visual comparison of the our metrics across classifiers

B. Cross-Validation Results

To verify that the reported performance is not an artifact of the particular data split, 10-fold stratified cross-validation was performed on the full dataset using the same optimized hyperparameter configurations.

Table 2. 10-Fold Stratified Cross-Validation Results (Mean ± Standard Deviation)

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	0.9964 ± 0.0012	0.9974 ± 0.0011	0.9949 ± 0.0022	0.9962 ± 0.0013
KNN	0.9841 ± 0.0016	0.9832 ± 0.0040	0.9827 ± 0.0040	0.9830 ± 0.0017
SVM	0.9610 ± 0.0030	0.9664 ± 0.0049	0.9493 ± 0.0054	0.9578 ± 0.0033

The cross-validation results closely mirror the single-split findings, confirming the robustness of the reported performance. RF achieved the highest mean accuracy (99.64%) with the lowest relative standard deviation (0.12%), demonstrating both superior performance and consistency across folds. KNN exhibited moderate variability, particularly in precision and recall (σ = 0.40%), while SVM showed the highest variability (accuracy σ = 0.30%), suggesting greater sensitivity to the composition of training data.

Figure 3 visualizes the distribution of cross-validation accuracy scores across folds, illustrating the relative spread and consistency of each classifier, the boxes indicate interquartile range; whiskers extend to minimum and maximum fold scores.

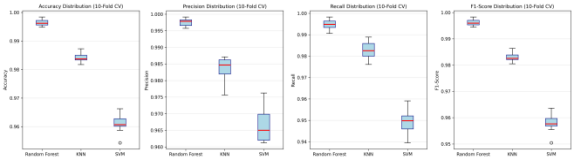

Figure 3: Boxplots of 10-fold stratified cross-validation accuracy scores for RF, KNN, and SVM classifiers.

C. Generalization Analysis

Table 3 presents training and test accuracy scores for each classifier.

Table 3. Training and Test Accuracy Scores

Model	Train Score (%)	Test Score (%)	Gap (pp)
Random Forest	99.98	99.62	0.37
KNN	98.81	98.25	0.56
SVM	96.01	95.82	0.19

RF exhibited a train-test accuracy gap of only 0.37 percentage points, indicating that despite near-perfect training accuracy, the model generalizes effectively to unseen data. KNN displayed the largest gap (0.56 pp), reflecting its inherent characteristics as a local, memory-based learner. SVM exhibited the smallest gap (0.19 pp), indicating the highest generalization consistency; however, this must be interpreted in the context of SVM's overall lower performance, suggesting a performance plateau during training.

D. Statistical Significance Testing

McNemar's test was applied to assess whether the pairwise differences in classification performance are statistically significant. The test operates on the contingency table of correct/incorrect predictions between each pair of classifiers.

© APR 2026 | IRE Journals | Volume 9 Issue 10 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I10-1716588

Table 4. McNemar's Test Results for Pairwise Classifier Comparisons

Model Pair	b (M1 ✓, M2 ✗)	c (M1 ✗, M2 ✓)	χ^2	p-value	Significant (α = 0.05)
RF vs KNN	106	632	373.48	< 0.001	Yes
RF vs SVM	245	594	144.34	< 0.001	Yes
KNN vs SVM	220	43	117.78	< 0.001	Yes

All three pairwise comparisons yielded highly significant results ($p < 0.001$), confirming that the observed performance differences are not attributable to chance. The RF vs KNN comparison reveals that KNN made 632 errors on instances that RF correctly classified, while RF made only 106 errors on instances that KNN got right, a 6:1 ratio demonstrating RF's substantial superiority. Similarly, the RF vs SVM comparison (594 vs 245 discordant errors) and KNN vs SVM comparison (43 vs 220) confirm the statistical robustness of the performance hierarchy RF > KNN > SVM.

E. Computational Cost Analysis

Training and prediction times were benchmarked by averaging over 5 runs on identical hardware.

Table 5. Computational Cost Comparison (Mean ± Standard Deviation, seconds)

Model	Train Time (s)	Predict Time (s)
Random Forest	0.0979 ± 0.0026	0.0025 ± 0.0000
KNN	0.0110 ± 0.0011	0.7964 ± 0.0530
SVM	2.0860 ± 0.0951	1.2278 ± 0.0559

The timing analysis reveals distinct computational profiles. KNN exhibits the fastest training time (0.011s) because it defers computation to the prediction phase, where it incurs the highest prediction latency (0.796s) due to exhaustive distance calculations across the training set. SVM exhibits the slowest training (2.086s) due to the iterative quadratic programming optimization required for support vector identification, and maintains high prediction latency (1.228s). RF achieves a favorable balance: moderate training time (0.098s) and the fastest prediction time (0.0025s), making it the most suitable for real-time deployment where low-latency predictions are critical.

The accuracy-efficiency trade-off strongly favors RF: it achieves the highest accuracy (99.62%) while requiring only 0.0025s per prediction, 319× faster than KNN and 491× faster than SVM at prediction time.

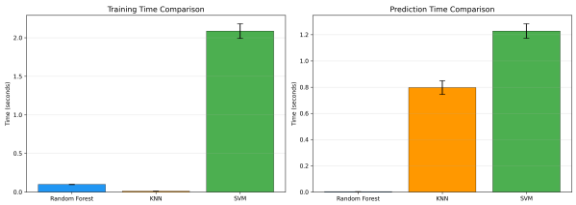

Figure 4: Computational cost comparison

F. Effect of Feature Selection

RFE reduced the feature space to 10 attributes representing the most informative features. The dominance of traffic-level statistical features (6 out of 10 selected) aligns with domain knowledge, as aggregated traffic statistics capture temporal patterns highly discriminative for anomaly detection. The retention of both byte count features (src_bytes, dst_bytes) corroborates the importance of payload size information, as many attack types exhibit distinctive byte transfer patterns. The inclusion of service and flag as the only protocol-level features underscores the role of connection metadata: service type signals attack-prone services, while connection flags provide direct evidence of connection anomalies.

VI. DISCUSSION

A. Comparative Analysis

The observed performance hierarchy (RF, KNN, SVM) can be attributed to the fundamental algorithmic characteristics of each classifier. RF's superior performance stems from its ensemble mechanism that aggregates predictions from decorrelated decision

trees, with bootstrap aggregation reducing variance and random feature subsampling enhancing diversity. These properties make RF inherently robust to noisy features and capable of capturing complex, non-linear feature interactions prevalent in network traffic data.

KNN's competitive but lower performance reflects its reliance on local neighborhood structure. While KNN can capture complex decision boundaries, its performance is sensitive to the distance metric and local density of the feature space. Even with feature selection and standardization, the curse of dimensionality can degrade the discriminative quality of distance-based neighbors at decision boundaries.

SVM's lower performance may reflect the inherent complexity of the classification boundary required for network intrusion detection. While SVM is well suited for binary classification, its optimization of a single decision boundary may be less flexible than the ensemble of decision surfaces generated by RF.

B. Practical Implications

For security operations where detection accuracy is paramount, RF represents the optimal choice, achieving the highest accuracy with the lowest prediction latency. For resource-constrained environments prioritizing training efficiency, KNN offers competitive accuracy with fast training but at the cost of higher inference latency. SVM, despite the best generalization consistency, may not provide sufficient accuracy for mission-critical applications, and its high computational cost further limits its operational suitability for real-time IDS.

C. Limitations

Several limitations should be acknowledged. First, the KDD Cup 1999 dataset, while widely used as a benchmark enabling direct comparison with extensive prior work, exhibits known limitations including redundant records, unrealistic traffic distribution, and attack types that do not reflect the contemporary threat landscape (Tavallaee et al., 2009). The results should therefore be interpreted as a methodological demonstration of the integrated RFE-Optuna pipeline rather than a claim of state-of-the-art detection on modern traffic. Second, the study addresses only binary classification (normal vs. anomaly); multiclass attack categorization would provide more granular detection capabilities. Third, the evaluation was conducted on a single dataset, and the generalizability of the findings to other IDS benchmarks (e.g., CIC-IDS2017, UNSW-NB15) requires further investigation. Fourth, adversarial robustness, the resilience of the trained classifiers to deliberately crafted adversarial samples, was not evaluated.

VII. CONTRIBUTION TO KNOWLEDGE

This study makes the following contributions:

1. *Integrated optimization pipeline.* The study demonstrates the effectiveness of combining RFE-based feature selection with Optuna-based Bayesian hyperparameter optimization within a unified comparative framework, establishing a reusable methodological template for IDS research.
2. *Statistically validated performance hierarchy.* The study provides empirical evidence, validated by McNemar's test (all $p < 0.001$) and 10-fold cross-validation, that RF achieves 99.62% accuracy with minimal overfitting (0.37% train-test gap) on the selected features.
3. *Computational cost quantification.* The study quantifies the accuracy-efficiency trade-offs among three algorithmically diverse classifiers, demonstrating that RF achieves a 319× prediction speed advantage over KNN while maintaining superior accuracy.
4. *Practical deployment.* The study extends the pipeline to deployment through a Streamlit-based web application, providing a reference implementation for bridging the research-deployment gap.

VIII. RECOMMENDATION

Based on the findings, the following recommendations are offered:

1. *Classifier selection.* RF is recommended as the primary classifier for binary IDS deployment due to its superior accuracy (99.62%), balanced precision-recall trade-off (F1: 99.59%), minimal overfitting (0.37% gap), and fast prediction time (0.0025s).
2. *Hyperparameter optimization.* Bayesian optimization via Optuna should be adopted as standard practice, as the performance gains justify the moderate computational investment during model development relative to grid search.

3. *Feature selection.* RFE should be incorporated as an integral component of the IDS pipeline. The reduction to 10 features did not compromise accuracy while reducing dimensionality, training time, and deployment resource requirements.
4. *Inference latency consideration.* Practitioners deploying real-time IDS should consider that KNN exhibits prediction latency 319× higher than RF. For high-volume traffic streams, RF provides the optimal accuracy-latency combination.
5. *Future directions.* Future work should extend this framework to (a) multiclass attack classification, (b) modern datasets (CIC-IDS2017, CIC-IDS2018, UNSW-NB15), (c) deep learning approaches within the same optimization framework, and (d) adversarial robustness evaluation.

IX. CONCLUSION

This study presented a comprehensive comparative evaluation of three optimized machine learning classifiers, RF, KNN, and SVM, for binary network intrusion detection. The methodology integrated RFE for dimensionality reduction, Bayesian hyperparameter optimization via Optuna, multi-metric evaluation, 10-fold stratified cross-validation, McNemar's statistical significance testing, and computational cost analysis.

RF achieved the highest performance across all metrics (accuracy: 99.62%, F1: 99.59%) with minimal overfitting (0.37% gap) and the fastest prediction time (0.0025s). KNN achieved second-highest performance (accuracy: 98.25%, F1: 98.11%), while SVM achieved the lowest (accuracy: 95.82%, F1: 95.46%). All pairwise differences were statistically significant (McNemar's $p < 0.001$). Cross-validation confirmed the robustness of these findings (RF: 99.64% ± 0.12%).

The study demonstrated practical applicability through a Streamlit-based web application and provides a complete pipeline from research to deployment. The findings underscore the value of combining automated feature selection with Bayesian optimization for high-performance intrusion detection, establishing RF as the recommended classifier for binary network intrusion detection.

ACKNOWLEDGMENT

This research was funded by Nigeria Tertiary Education Trust Fund (TETFUND).

REFERENCES

[1] I. H. Sarker, A. S. M. Kayes, and P. Watters, "Effectiveness analysis of machine learning classification models for predicting cybersecurity attacks," J. Big Data, vol. 11, p. 45, 2024.

[2] 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," Trans. Emerg. Telecommun. Technol., vol. 34, no. 1, p. e4150, 2023.

[3] A. Khraisat and A. Alazab, "A critical review of intrusion detection systems in the Internet of Things: Techniques, deployment strategy, validation strategy, attacks, public datasets, and challenges," Cybersecurity, vol. 6, no. 1, pp. 1–28, 2023.

[4] A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd ed. Sebastopol, CA: O'Reilly Media, 2023.

[5] A. Thakkar and R. Lohiya, "Role of feature selection in optimizing machine learning-based intrusion detection systems: A comprehensive review," Arch. Comput. Methods Eng., vol. 31, no. 2, pp. 1027–1058, 2024.

[6] B. Bischl et al., "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges," Wiley Interdiscip. Rev. Data Min. Knowl. Discov., vol. 13, no. 2, p. e1484, 2023.

[7] K. A. Dhanya and S. Mathew, "A comparative study of ensemble classifiers for intrusion detection," J. Inf. Secur. Appl., vol. 76, p. 103539, 2024.

[8] E. E. Abdallah and A. F. Otoom, "A comparative evaluation of supervised machine learning classifiers for network intrusion detection," Int. J. Inf. Secur., vol. 23, no. 1, pp. 537–552, 2024.

[9] A. Halbouni, T. S. Guntur, D. T. C. Lai, and N. K. Kasabov, "Machine learning and deep learning approaches for intrusion detection: A

comparative study," Inf. Sci., vol. 659, p. 120107, 2024.

[10] P. Devan and N. Khare, "An efficient ensemble-based intrusion detection approach using optimized tree-based classifiers for real-time network traffic classification," Neural Comput. Appl., vol. 36, no. 8, pp. 4023–4037, 2024.

[11] N. Moustafa, G. Creech, and J. Slay, "A comparative study of feature selection methods for network intrusion detection," Comput. Electr. Eng., vol. 106, p. 108582, 2023.

[12] R. Panigrahi and S. Borah, "A hybrid feature selection approach for network intrusion detection using mutual information and recursive feature elimination," Expert Syst. Appl., vol. 238, p. 121876, 2024.

[13] A. Singh and D. Kumar, "Evolutionary feature selection for intrusion detection systems: A genetic algorithm approach," Swarm Evol. Comput., vol. 78, p. 101274, 2023.

[14] A. Ogunleye, S. Misra, and R. Damaševičius, "Bayesian hyperparameter optimization for ensemble methods in intrusion detection," Appl. Soft Comput., vol. 152, p. 111234, 2024.

[15] L. Yang, A. Moubayed, and A. Shami, "MTH-IDS: A multitiered hybrid intrusion detection system for Internet of Vehicles," IEEE Internet Things J., vol. 10, no. 1, pp. 616–630, 2023.

[16] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in Proc. IEEE Symp. Comput. Intell. Secur. Def. Appl., 2009, pp. 1–6.