Comparative Analysis of Machine Learning Models for Cybersecurity Intrusion Detection on UNSW-NB15 and Kaggle Datasets

Aditya Jadhav
New Mexico State University
Las Cruces, USA

Nikshith Kilaru
New Mexico State University
Las Cruces, USA

Dinesh Kumar Gowd Jadapalli
New Mexico State University
Las Cruces, USA

Siva Sai Kumar Boina
New Mexico State University
Las Cruces, USA

Md Omar Faruque
New Mexico State University
Las Cruces, USA

Abstract

Cybersecurity threats are increasingly sophisticated, with attackers using advanced methods to bypass conventional security measures. Traditional rule-based Intrusion Detection Systems (IDS) struggle to keep pace with evolving attack patterns, leading to false positives or missed threats. Machine learning (ML) offers a promising solution by automatically learning patterns from network behavior to detect anomalies indicative of cyber threats. This project develops and evaluates several ML-based IDS models, including Random Forest (RF), Artificial Neural Networks (ANN), and Fuzzy C-Means (FCM), using the UNSW-NB15 and a Kaggle cybersecurity dataset. We analyze the impact of data preprocessing techniques like outlier removal and feature scaling, evaluate model performance using metrics like F1-score and recall, assess deployment considerations like model size and inference time, and use SHAP for feature importance analysis. Results indicate that RF and ANN models achieve high detection accuracy, with performance varying significantly based on dataset characteristics and the handling of outliers. Key features driving detection are identified, providing insights for building more robust and efficient IDS.

Keywords

Intrusion Detection, Machine Learning, Random Forest, Neural Networks, UNSW-NB15, Feature Importance, SHAP

1 Introduction

As cyber threats continue to evolve in complexity and frequency, traditional Intrusion Detection Systems (IDS) face significant limitations. Conventional signature-based and rule-based IDS struggle against zero-day attacks, polymorphic malware, and advanced persistent threats (APTs), often generating high false positive rates or failing to detect novel attack strategies [9]. This necessitates the development of more intelligent, adaptive, and scalable IDS capable of identifying anomalies in real-time.

Machine learning (ML)-based IDS have emerged as a powerful alternative, leveraging behavioral analysis and pattern recognition to detect both known and unknown threats [4]. However, developing effective ML-based IDS presents challenges, including achieving high accuracy while minimizing false alerts (especially false negatives, which represent missed attacks), selecting relevant features from high-dimensional network data, ensuring real-time processing capabilities despite computational demands, and maintaining robustness against potential adversarial evasion techniques [1].

The goal of this project is to develop, evaluate, and compare several ML-based IDS approaches using two distinct benchmark datasets: UNSW-NB15 and a Kaggle cybersecurity dataset. We specifically investigate the performance of Random Forest (RF), Artificial Neural Networks (ANN) combined with SMOTE for imbalance handling, Principal Component Analysis (PCA) for dimensionality reduction coupled with RF, and Fuzzy C-Means (FCM) clustering. We analyze the impact of data preprocessing techniques (outlier removal, scaling), feature selection strategies (correlation-based), and dimensionality reduction (PCA) on model performance, interpretability (using SHAP), and deployment considerations (model size, inference time). Our findings highlight the effectiveness of RF and ANN models and underscore the critical influence of dataset characteristics and preprocessing choices on IDS performance.

2 Related Work

Intrusion Detection Systems (IDS) are crucial for cybersecurity. Early signature-based systems struggled with novel threats, leading to the development of ML-based approaches [9]. Supervised methods like Random Forest (RF) and Support Vector Machines (SVM) have shown promise [4], while unsupervised techniques like Isolation Forest target zero-day attacks. Deep Learning (DL), particularly LSTMs and CNNs, has further improved detection by capturing complex temporal and spatial patterns in network traffic [2, 3].

Effective feature engineering and selection are vital. Features like packet size, duration, and IP reputation are often discriminative [1]. Techniques like PCA and RFE help reduce dimensionality and improve interpretability [8]. Benchmark datasets like NSL-KDD, CICIDS2017, UNSW-NB15 [6], and the Kaggle dataset used here [5] are essential for evaluating IDS models.

Hybrid systems combining signature and anomaly detection [4], robustness against adversarial attacks [1, 2], and explainability using methods like SHAP and LIME [7] are active areas of research aiming to enhance the practicality and trustworthiness of ML-based IDS.

3 Datasets

Two datasets were used to evaluate the ML models:

3.1 UNSW-NB15

Created by the Australian Centre for Cyber Security (ACCS), this dataset (https://research.unsw.edu.au/projects/unsw-nb15-dataset) contains a mix of normal traffic and 9 types of synthetic attacks across 49 features [6]. The full dataset has over 250,000 instances; for this study, we used a sample of 20,000 instances. The target variable is 'label', where 0 indicates normal traffic and 1 indicates an attack. Our goal is to classify network traffic instances as either normal or attack. The dataset exhibits imbalance in our sample (approx. 36% normal, 64% attack).

Table 1: UNSW-NB15 Dataset Overview

Property	Value
Format	CSV
Total Instances	175,341 training, 82,332 testing
Features	49 (including the label)
Attack Types	9 different attack categories
Size	100MB (split into training and testing)

3.2 Cybersecurity IDS Dataset (Kaggle)

Available on Kaggle (https://www.kaggle.com/datasets/dnkumars/cybersecurity-intrusion-detection-dataset/), this dataset contains 9,537 instances with 10 features plus a binary target variable named 'attack_detected' [5]. It includes 7 numerical and 3 categorical features. The goal is to classify each instance based on whether an attack was detected (1) or not (0). The target variable is relatively balanced (approx. 55% normal/no attack, 45% attack). The 'encryption_used' feature contained missing values, which we handled during preprocessing.

Table 2: Cybersecurity IDS Dataset Overview

Property	Value
Format	CSV
Total Instances	9,537
Features	10 features + 1 target variable
Feature Types	3 categorical, 7 numerical
Label	Binary (attack_detected)
Size	Small (< 1MB)

4 Methodology

We implemented a comprehensive analysis pipeline in Python, leveraging standard data science libraries such as Pandas for data manipulation, Scikit-learn and TensorFlow for machine learning model implementation, and SHAP for model interpretability. Our methodology involved the following key steps:

1 **Data Loading & Initial Cleaning:** We began by loading the datasets. For UNSW-NB15, we combined the provided training and testing sets to create a unified dataset before sampling. For the Kaggle dataset, we addressed potential mixed data types often encountered in real-world data. We dropped ID columns as they provide no predictive value. To manage computational resources and ensure timely analysis, we sampled the larger UNSW-NB15 dataset down to 20,000 instances, aiming to maintain the original class distribution. 2 **Exploratory Data Analysis (EDA):** We conducted EDA to understand the data's characteristics. This included basic checks like summary statistics, identifying missing values, and examining the target variable distribution to understand class balance. We performed advanced EDA by calculating the correlation matrix for numerical features. We chose a high correlation threshold of 0.9 to identify potentially redundant features, as highly correlated features can sometimes negatively impact model performance and interpretability without adding significant information. 3 **Outlier Handling:** Recognizing that outliers can disproportionately affect some ML models, we investigated their impact. We performed analyses on both the original data and data where outliers were removed from numerical features. We chose the Interquartile Range (IQR) method with a standard factor of 1.5, a common technique for identifying statistical outliers, to systematically assess the sensitivity of our models to extreme values. 4 **Preprocessing:** We developed robust preprocessing pipelines using Scikit-learn's `ColumnTransformer` to ensure consistent data transformation. For numerical features, we imputed missing values using the median, which is less sensitive to outliers than the mean. We then applied `StandardScaler` (zero mean, unit variance) as many ML algorithms perform better with scaled data. For categorical features, we imputed missing values with a distinct 'Missing' category and then applied one-hot encoding to convert them into a numerical format suitable for ML models. To evaluate the impact of scaling, particularly for models like Random Forest that are less sensitive to it, we created separate pipelines producing both scaled and unscaled versions of the numerical data. 5 **Feature Selection/Reduction/Engineering:** To address potential issues with high dimensionality and redundancy, we evaluated two common dimensionality reduction techniques:

- *Correlation-based Selection:* Based on our EDA, we trained models using a subset of features, excluding those identified with a pairwise correlation greater than 0.9. This approach aims to remove redundant information while retaining most of the original feature interpretability.
- *PCA:* We applied Principal Component Analysis, a technique that transforms the data into a smaller set of uncorrelated components capturing maximum variance. We applied PCA only to the scaled data, as it is sensitive to feature scales, and evaluated the performance of RF using the resulting principal components to see if a more compact representation could maintain performance.

6 **Model Training & Evaluation:** We partitioned the data using a stratified 80/20 train-test split. Stratification ensures that the proportion of target classes is maintained in both sets, which is crucial for imbalanced datasets like UNSW-NB15. We trained and evaluated the following diverse set of models to compare different algorithmic approaches:

- *Random Forest (RF):* We chose RF due to its strong performance in many classification tasks, robustness to feature scaling, and inherent ability to handle non-linearities. We trained RF on both scaled and unscaled data. We tuned key hyperparameters (`n_estimators`, `max_depth`, `min_samples_split`, `min_samples_leaf`) using `GridSearchCV` with 3-fold cross-validation, optimizing for accuracy, to find the best configuration. We also included a baseline RF with default parameters for comparison.
- *Artificial Neural Network (ANN):* We selected ANNs for their ability to model complex patterns. We designed a sequential model with Dense layers and Dropout for regularization. We used the Adam optimizer and binary cross-entropy loss, standard choices for binary classification. To address class imbalance observed particularly in the UNSW-NB15 dataset, we applied SMOTE (Synthetic Minority Over-sampling Technique) only to the training data before feeding it to the ANN. We employed early stopping during training to prevent overfitting.
- *PCA + RF:* To assess the effectiveness of PCA as a preprocessing step, we trained a baseline RF model using the principal components derived from the scaled data as input features. We tuned the number of components to retain.
- *Fuzzy C-Means (FCM):* We included FCM, an unsupervised clustering algorithm, to explore if natural groupings in the data corresponded well to the intrusion labels. We applied it to scaled numerical data, tuned its parameters (number of clusters c, fuzzifier m), and evaluated its performance by assigning the majority class label to each cluster.

7 Evaluation Metrics: We evaluated model performance using standard classification metrics. Accuracy provides an overall measure of correctness. F1-Score (harmonic mean of precision and recall) is useful for imbalanced datasets. We paid particular attention to Recall (sensitivity), especially for the attack class, as minimizing false negatives (missed attacks) is critical in IDS. Additionally, we recorded model size (in MB) and prediction time on the test set (in seconds) as practical considerations for potential deployment scenarios. 8 **Feature Importance:** To understand which features were most influential in the predictions of our best models, we employed SHAP (SHapley Additive Explanations). We chose SHAP because it provides theoretically sound, consistent, and locally accurate feature attributions. We applied it to the best-performing tuned Random Forest models (using scaled data) to gain insights into the drivers of intrusion detection for each dataset.

5 Results

All experiments were conducted using Python on a system running Ubuntu 24.04. The specific library versions used are detailed in the project's Conda environment file (`tf_gpu_env.yml`). The hardware configuration included a 12th Gen Intel(R) Core(TM) i7-12700KF CPU, an NVIDIA RTX 3060 GPU (utilized by TensorFlow if enabled), and associated system RAM of 16 GB. The analysis script executed was `merged_ids_analysis.py`.

Trained models were saved for potential future use. Random Forest models were serialized using Python's 'pickle' library ('.pkl' files), storing the entire trained model object including its structure (trees, splits) and parameters. Fuzzy C-Means cluster centers were similarly saved as '.pkl' files. The Artificial Neural Network models were saved using the Keras native format ('.keras'), which stores the model architecture, learned weights and biases, and optimizer state. These saved files allow the models to be reloaded using Scikit-learn ('.pkl') or TensorFlow/Keras ('.keras') for inference on new data without retraining, facilitating deployment or further analysis.

The performance of the evaluated models varied significantly between the two datasets and was heavily influenced by preprocessing steps, particularly outlier removal. Key results from the original (non-outlier-removed) data are presented in Table 4 and Table 5.

Table 3 presents the optimal hyperparameters for the Tuned Random Forest models on the original data, which achieved the highest performance on both datasets. These models slightly outperformed the ANN+SMOTE variants across all metrics. The impact of outlier removal is summarized in Table 6.

Table 3: Best Random Forest Model Parameters

Dataset	Accuracy	Parameters
Kaggle	0.886	max_depth=10 min_samples_leaf=1 min_samples_split=5 n_estimators=150
UNSW	0.935	max_depth=None min_samples_leaf=1 min_samples_split=2 or 5 n_estimators=100

Scaling numerical features had minimal impact on RF performance for both datasets. Using PCA for dimensionality reduction resulted in lower accuracy compared to using the full feature set. Correlation-based feature selection on the UNSW dataset (where 14 features were dropped) slightly reduced RF accuracy (to 0.927) but significantly decreased model size.

Outlier removal had a modest negative impact on performance for the Kaggle dataset but a more pronounced negative effect on RF accuracy for the UNSW dataset (dropping from 0.935 to 0.910). Interestingly, ANN+SMOTE maintained high recall on the outlier-removed UNSW data, suggesting better robustness in that scenario.

Deployment metrics show that ANN models were significantly smaller in size compared to RF models, but had substantially longer prediction times on the test set. Correlation-based feature selection offered a good trade-off for RF on UNSW, reducing model size considerably with only a minor performance dip.

SHAP analysis identified distinct top features for each dataset (Table 7). For Kaggle, login-related features and IP reputation were most important. For UNSW-NB15, Time-To-Live (TTL) fields, load, and byte counts were dominant, reflecting the different nature of the features and potential attacks in each dataset.

6 Discussion

The results demonstrate the effectiveness of Random Forest and ANN models for intrusion detection on these datasets. On the original UNSW data, both the Base RF and the Tuned RF achieved high

Table 4: Model Performance & Deployment Metrics (Kaggle Dataset - Original Data)

Model Configuration	Accuracy	F1-Score	Recall	Size (MB)	Pred. Time (s)	Train Acc (CV)
Base RF (Scaled)	0.8826	0.8800	0.8800	-	0.02	-
Tuned RF (Scaled)	0.8857	0.8800	0.8900	2.327	0.01	0.8954
Tuned RF (Unscaled)	0.8857	0.8800	0.8900	2.321	0.01	0.8954
ANN+SMOTE (Scaled)	0.8836	0.8800	0.8800	0.066	0.89	-
PCA (n=15) + Base RF (Scaled)	0.8422	0.8400	0.8400	-	0.02	-
FCM (Tuned, n=4, m=2.0)	0.6801	0.6800	0.6800	0.000	0.02	-

Table 5: Model Performance & Deployment Metrics (UNSW-NB15 Dataset - Original Data)

Model Configuration	Accuracy	F1-Score	Recall	Size (MB)	Pred. Time (s)	Train Acc (CV)
Base RF (Scaled)	0.9347	0.9300	0.9300	-	0.03	-
Tuned RF (Scaled)	0.9347	0.9300	0.9300	16.313	0.03	0.9339
Tuned RF (Unscaled)	0.9350	0.9300	0.9400	11.763	0.03	0.9346
ANN+SMOTE (Scaled)	0.9070	0.9100	0.9100	0.192	0.52	-
PCA (n=20) + Base RF (Scaled)	0.9103	0.9100	0.9100	-	0.03	-
Tuned RF (Corr. Selected, Scaled)	0.9270	0.9300	0.9300	5.976	0.02	0.9247
FCM (Tuned, n=4, m=1.5)	0.7796	0.7600	0.7800	0.001	0.01	-

Table 6: Impact of Outlier Removal on Scaled Model Performance (Weighted Avg F1/Recall)

Model	Dataset	Accuracy		F1-Score (Weighted)		Recall (Weighted)	
		Original	No Outliers	Original	No Outliers	Original	No Outliers
Tuned RF	Kaggle	0.886	0.879	0.880	0.870	0.890	0.880
ANN+SMOTE (64>32>1, Dropout 0.3)	Kaggle	0.884	0.873	0.880	0.870	0.880	0.870
Tuned FCM (Best: n=4,m=2.0 / n=4,m=2.5)	Kaggle	0.680	0.646	0.680	0.650	0.680	0.650
Tuned RF	UNSW-NB15	0.935	0.910	0.930	0.910	0.940	0.910
ANN+SMOTE (64>32>1, Dropout 0.3)	UNSW-NB15	0.907	0.893	0.910	0.900	0.910	0.890
Tuned FCM (Best: n=4,m=1.5 / n=2,m=1.5)	UNSW-NB15	0.780	0.819	0.760	0.740	0.780	0.820

Table 7: Top 5 Features by Mean Absolute SHAP Value (Best Tuned RF, Original Data)

Kaggle Dataset	UNSW-NB15 Dataset
1. failed_logins	1. sttl
2. ip_reputation_score	2. ct_state_ttl
3. login_attempts	3. dttl
4. browser_type_Unknown	4. sload
5. session_duration	5. dbytes

accuracy and F1-scores (0.935 Acc, 0.93 F1), indicating that even default RF parameters perform well, although tuning provides potential for minor optimization (Table 5). The ANN+SMOTE model also performed strongly (0.91 Acc, 0.91 F1), slightly below the RF models on this dataset. RF's robustness is further highlighted by the minimal impact of feature scaling on its performance, confirming its suitability for datasets where scaling might not be straightforward or desirable.

The impact of outlier removal warrants significant discussion. While often considered a standard preprocessing step, its application here, especially on the UNSW-NB15 dataset where it removed 77% of the data, led to a decrease in performance for RF models (Table 6). This suggests that some data points identified as outliers by the IQR method might represent legitimate, albeit rare, attack patterns or network behaviors crucial for accurate classification. The ANN model's relatively stable recall on the outlier-removed UNSW data might indicate its ability to capture underlying patterns even with reduced data, potentially benefiting from the SMOTE oversampling.

The tuned Fuzzy C-Means clustering results provide additional insights. On the Kaggle dataset, the best FCM achieved 68.0% accuracy (n=4, m=2.0), while on UNSW-NB15 it reached approximately 78.0% accuracy (n=4, m=1.5) (Tables 4, 5). The performance difference between datasets suggests varying degrees of natural cluster separation corresponding to the labels. FCM's extremely small model size (around 0.001MB) makes it potentially attractive for highly resource-constrained environments, though its significantly lower accuracy compared to RF and ANN limits its standalone use in critical security applications.

This highlights a critical consideration for IDS development: aggressive outlier removal might inadvertently discard valuable information about sophisticated or low-frequency attacks. The high false negative rate (low recall) for the minority class (normal traffic) in the outlier-removed UNSW FCM results (Accuracy 0.819, but Recall for class 0 is 0.00, see summary file) further emphasizes the potential pitfalls of both outlier removal and relying solely on clustering for this task.

Analyzing potential overfitting is also important. Comparing the cross-validation training accuracy with the final test accuracy for the Tuned RF models (Tables 4, 5) reveals very small differences (e.g., Kaggle Scaled: 0.895 CV vs 0.886 Test; UNSW Scaled: 0.934 CV vs 0.935 Test). This close alignment suggests that the Tuned RF models did not significantly overfit the training data and generalized well to the unseen test set. For the ANN models, the use of dropout layers and early stopping during training are standard techniques employed specifically to mitigate overfitting. The relatively low performance of FCM compared to RF/ANN might indicate underfitting for this specific classification task.

Analyzing false negatives via the Recall metric (Tables 4, 5, 6) is crucial. While overall accuracy is high, ensuring high recall for the attack class (label=1) is paramount. For the original data, both RF and ANN achieve high recall (>0.93 on UNSW, >0.74 on Kaggle for the attack class based on classification reports in summaries), indicating a good ability to detect attacks. The slight drop in recall after outlier removal, particularly for RF on UNSW, reinforces the concern about potentially removing informative attack instances.

Feature importance analysis via SHAP provided valuable insights across all three approaches. For RF models, the top features were: - Kaggle: failed_logins (0.192), ip_reputation_score (0.097), login_attempts (0.090) - UNSW-NB15: sttl (0.145), ct_state_ttl (0.138), dttl (0.135)

The FCM clustering revealed similar feature importance patterns, though with less distinct separation between top features. The differing top features between Kaggle (login/reputation focused) and UNSW-NB15 (TTL/traffic volume focused) underscore the importance of dataset context and feature engineering tailored to specific network environments or expected threat models. This consistency across different analytical approaches strengthens confidence in these features' importance for intrusion detection.

Deployment considerations reveal a trade-off. ANNs offer significantly smaller model sizes, beneficial for resource-constrained environments. However, their inference latency was much higher than RF in our tests. RF models, while larger, provide very fast predictions. Feature selection (like correlation-based) can mitigate RF's size disadvantage with minimal performance loss, offering a practical compromise. PCA, while reducing dimensions, did not yield competitive performance in this study.

Limitations include the use of sampled data for UNSW-NB15, potentially affecting generalizability. The hyperparameter search space was not exhaustive. Evaluation was performed offline; real-time performance might differ.

7 Conclusion

This study compared several machine learning models for network intrusion detection using the UNSW-NB15 and Kaggle datasets. Tuned Random Forest and ANN+SMOTE models demonstrated strong performance, achieving high accuracy and F1-scores on both datasets. Our analysis revealed the critical impact of preprocessing choices, particularly outlier removal, which significantly affected performance on the UNSW-NB15 dataset, potentially by removing informative attack samples. Feature importance analysis using SHAP highlighted distinct key features for each dataset, emphasizing the context-dependent nature of IDS. While ANNs offer smaller model sizes, RF provides faster inference, with feature selection presenting a viable method to reduce RF model size. Future work could involve evaluating models on the full datasets, exploring more advanced feature engineering, investigating different outlier handling strategies, and assessing real-time deployment performance.

References

[1] Damilola Adesina, Chung-Chu Hsieh, Yalin E. Sagduyu, and Lijun Qian. 2023. Adversarial Machine Learning in Wireless Communications Using RF Data: A Review. *IEEE Communications Surveys & Tutorials* 25, 1 (2023), 77–100. doi:10.1109/COMST.2022.3205184

[2] Jadil Alsamiri and Khalid Alsubhi. 2023. Federated learning for intrusion detection systems in internet of vehicles: a general taxonomy, applications, and future directions. *Future Internet* 15, 12 (2023), 403.

[3] Asmaa Halbouni, Teddy Surya Gunawan, Mohamed Hadi Habaebi, Murad Halbouni, Mira Kartiwi, and Robiah Ahmad. 2022. CNN-LSTM: Hybrid Deep Neural Network for Network Intrusion Detection System. *IEEE Access* 10 (2022), 99837–99849. doi:10.1109/ACCESS.2022.3206425

[4] G Indra, E Nirmala, G Nirmala, and P Gururama Senthilvel. 2024. An ensemble learning approach for intrusion detection in IoT-based smart cities. *Peer-to-Peer Networking and Applications* 17, 6 (2024), 4230–4246.

[5] D. N. Kumar. 2023. CyberSecurity Intrusion Detection Dataset. https://www.kaggle.com/datasets/dnkumars/cybersecurity-intrusion-detection-dataset. Retrieved February 28, 2025.

[6] Wenhao Li, Duohe Ma, Zhaoxuan Li, Huaifeng Bao, Shuai Wang, Huamin Jin, and Xiao-Yu Zhang. 2024. Poster: Towards Real-Time Intrusion Detection with Explainable AI-Based Detector. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security* (Salt Lake City, UT, USA) *(CCS '24)*. Association for Computing Machinery, New York, NY, USA, 4934–4936. doi:10.1145/3658644.3691410

[7] Christoph Molnar. 2020. *Interpretable machine learning*. Lulu. com.

[8] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, and Ali A Ghorbani. 2012. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *computers & security* 31, 3 (2012), 357–374.

[9] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. 2009. A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications*. Ieee, 1–6.