ENHANCING WSN INTRUSION DETECTION: TWO-TIER FEATURE SELECTION AND OPTUNA-OPTIMIZED ENSEMBLE LEARNING

Dilip Dalgade, NileshPatil, ManujJoshi and Dilendra Hiran

Department of Computer Engineering, Pacific Academy of Higher Education and Research, Udaipur, Rajasthan, India.

ABSTRACT

Wireless Sensor Networks (WSNs) are key for ubiquitous computing. Despite advantages, they face security challenges due to decentralized nature and threats. Intrusion detection helps protect WSNs from security threats. This study proposes an Optuna-implemented stacking technique (OXCRF) the method combines SHapley Additive exPlanations, CatBoost, Mutual Information, and cross-validated Recursive Feature Elimination with Random Forest for feature selection, while SMOTE handles data imbalance. The stacking ensemble, XGBoost, CatBoost and Random Forest are used as the base learners, with hyperparameters being optimized using Optuna. Experiments on the NSL-KDD and UNSW-NB15 datasets show that OXCRF achieves higher accuracy (99.60% for binary and 99.53% for multiclass on NSL-KDD; 98.62% for binary and 83.67% for multiclass on UNSW-NB15) and lower misclassification rates (0.0040 and 0.0047 on NSL-KDD; 0.0138 and 0.1633 on UNSW-NB15) compared to baseline models. Running an ablation study showed that OXCRF components worked as expected for multiclass intrusion detection in WSNs with overlapping classes and imbalanced data. The framework is efficient through feature selection, balanced data distribution and improved ensemble learning.

KEYWORDS

Wireless Sensor Networks, Intrusion Detection, Stacking Ensemble Learning, Optuna, Feature Selection, XGB, CatBoost.

1. INTRODUCTION

Wireless Sensor Networks (WSNs) are becoming increasingly important in many sectors, such as e-health, military surveillance, and smartenvironments. As networks become more interconnected, the openness of wireless sensor network areas and wireless communication broadcasting makes these networks vulnerable to external threats. Open deployment and resource constraints such as limited energy and computational power render their networks vulnerable to a variety of potential attacks, such as unauthorized access, tampering, and denial of service (DoS) attacks [1]

Intrusion detection systems are engineered to systematically monitor malicious or undesirable activities conducted by the nodes within a specific network[2]. Traditional IDSs, particularly those based on heuristic rules or fixed signature-based IDSs, are frequently unable to react to new threats[3]. Overcoming these limitations, machine learning (ML)-based intrusion detection systems have become increasingly popular because they can identify complex patterns from extensive datasets and adapt to address new attack processes[4], designing an efficient IDS for WSNs is fraught with a significant challenge. These are coping with the extremely high dimensionality of network traffic data, class imbalance, and high accuracy without compromising

DOI:10.5121/ijwmn.2025.17301

the limited resources of sensor nodes. In addition, noisy and redundant features in the learning samples can reduce the efficiency of ML classifiers.

To address these issues, this study proposes an adaptive intrusion detection system with SMOTE-based data balancing, a two-stage feature selection mechanism (SHAP with CatBoost and Mutual Information in the first stage and Recursive Feature Elimination with cross-validation with Random Forest in the second stage), and stacking ensemble classifier with Optuna's Tree-structured Parzen Estimator (TPE) optimization. The proposed model can efficiently identify both binary and multiclass intrusions in WSN scenarios with minimal resource consumption and enhanced detection accuracy.

To address the aforementioned issues, this study proposes a robust and scalable intrusion detection system for WSNs. The contributions of this research are as follows:

- Hybrid Feature Selection: A two-stage feature selection process is suggested. The firststageemploys SHAP with CatBoost and Mutual Information for the first stage of feature filtering, and then the second stage employs Recursive Feature Elimination with cross-validationwith Random Forests to further narrow the feature space.
- Integrated Model Optimization: The ensemble model is hyperparameter optimized usingTPE algorithm of Optuna to provide efficient generalization as well as optimization interms of performance.
- Stacked Ensemble Learning: A stacked OXCRF is proposed, where XGBoost and CatBoostare used as base learners, and Random Forest is used as the meta-learner and is optimized withOptuna.
- Binary and Multiclass Intrusion Detection: The model is tested for binary and multiclass classification using the NSL-KDD and UNSW-NB15 dataset.

Section 2 presents the literature and defines the research gaps that exist in IDS for WSNs. Section 3 describes the research methodology, including data preprocessing, feature selection, model construction, and optimization. Section 4 presents the experimental setup and results, including the comparative performance. Section 5 summarizes the study and specifies future work.

2. RELATED RESEARCH

The increase in the number of assaults on wireless sensor networks (WSNs) highlights the need for effective intrusion detection. Numerous machine-learning (ML)-based intrusion detection models in the literature aim to overcome the constraints of traditional techniques in WSNs. While many studies have made notable contributions to addressing specific aspects of the intrusion detection problem, there were considerable differences in each with respect to the class imbalance, redundant and irrelevant features, multiclass detection capability, and model generalization. In [5] proposed DLS-IDS, a deep-learning spark intrusion detection system using the NSL-KDD dataset. The system addresses class imbalance through SMOTE and LSTM for intrusion identification. The Spark Cluster setup accelerated the preparation, allowing IDS applications with optimized hyperparameters. In [6] based on ADASYN oversampling for IDS. In WSNs, an anomaly-based IDS uses mutual information (MI) for feature selection, dataset balancing using SMOTE, and ML algorithms, such as SVM, SGD, and KNN. In [7] focused on NADSs and improved data imbalance in minority class classification by combining SMOTE with K-means clustering. This was followed by a Denoising Autoencoder and XGB algorithm for anomaly detection through dimensionality reduction. In [8] proposed a learning-based aid for feature selection in a random forest algorithm by combining three ML models (SVM, LR, and k-NN). This study enhanced network intrusion detection via ML, emphasizing the role of feature selection in improving the detection rate. In [9] An intrusion detection system was proposed using a Random Forest Classifier aimed at protecting the network from different forms of attacks. This study highlights feature selection through correlation analysis and PCA for enhanced security. In [10] This study examined cybersecurity challenges using random forest recursive feature elimination, demonstrating high accuracy on the NSL-KDD, UNSW-NB15, and CSE-CIC-IDS2018 datasets. In [11] proposed a network intrusion detection system using a DNN and RNN to determine normal and malignant network traffic activities. The integration of neural networks leverages detection by exploiting the strength of each approach. In [12] proposed a flow-based NIDS using a stacked unsupervised FL approach for increased generalization in a cross-silo setup. Improved IDS using " a stacked unsupervised federated learning approach" for dynamic networking conditions. In [13] proposed an IDS for Wireless Sensor Networks using Sequential Minimal Optimization (SMA), SVM, and decision trees for classification. SMA reduces the dataset dimensionality from 41 to 5. In [14] A machine-learning approach utilizing a Weighted Score Selector (WSS) for WSN attack detection. The results demonstrated the efficacy of WSS with Boosting, Bagging, and Stacking, all of which worked in DoS attack detection. In [15] addressed NIDS problems in terms of accuracy, trustworthiness, and big-data processing. This highlights the need for dimension reduction and feature selection to increase the efficiency and model performance. This study mitigates class imbalance using SMOTE to improve the detection performance.

3. METHODOLOGY

This section outlines the proposed OXCRF model, which includes data preprocessing, feature selection, and model training. The model adopts Optuna to optimize hyperparameters and involves a stacking-based ensemble learning approach that enhances the performance and generalization for improved intrusion detection in WSNs.

3.1. Dataset Description

NSL-KDD is a widely used dataset and is highly regarded in industry for evaluating intrusion detection methods[16]. Each entry includes 41 features: 38 digital attributes, three symbolic attributes, and a class label indicating the type of network traffic. The dataset was categorized into one normal and four attack types: DoS, Probing, U2R, and R2L. DoS attacks disrupt network-resource access by consuming bandwidth or overloading resources. Probing attacks involve network scanning to gather information before an assault. U2R attacks occur when users gain unauthorized root access. R2L attacks involve accessing local hosts by using specially crafted network packets. Table 1 summarizes the class labels that distinguish between the four types of attack data.

The UNSW-NB15 [17]dataset has 42 characteristics and has 257,673 rows with nine attack categories: Worms, Shellcodes, Reconnaissance, Generic applications, Fuzzers, Exploits, Denial of Service (DoS), Backdoor, and Analysis. Each entry in the dataset labels a binary classification alongside the attack type.

To elucidate the proposed model, the dataset description, data preprocessing, model engineering, and ML algorithms are detailed as in figure1, and the steps are as follows:

1. Imports NSL-KDD / UNSW-NB15 dataset.
2. The dataset was divided into two parts: “80% for training and 20% for testing”.
3. Data preprocessing was performed.
4. Apply SMOTE.
5. Rank feature importance using SHAP with CatBoost and Mutual Information (MI) for the

initial feature selection.

6. Conduct feature selection using cv-RFE with an RF classifier.
7. On the training dataset, the Optuna Stacking Ensemble was used to train the OXCRF method.
8. The OXCRF method was analysed using the base class XGB-CatBoost and metaclass RF classifier on the test dataset.

Table 1: NSL-KDD Dataset Attack Class to Attack Type Mapping[18]

Attack Class	**Attack Type**
DoS	Back, Land, Neptune, Pod, Smurf, Teardrop, Apache2, Udpstorm, Processtable, Worm
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint
R2L	Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named
U2R	Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps

3.2. Data Preprocessing

Preprocessing data in a Wireless Sensor Network (WSN) serves an interim but fundamental role in the improvement of intrusion-detection system performance. They usually form the transition of data into such a form or format that they can be easily used with machine learning algorithms. Table 2 presents an 80%-20% split for training and testing. Additionally, it outlines the distribution of normal and attack instances within these sets, highlighting the differences in distribution across various training-to-testing ratios for binary classification. Data preprocessing involves handling a dataset in various steps.

Table 2: Data Distribution in Binary Classification on NSL-KDD and UNSW-NB15 Datasets.

Data Set		Binary classification		Total
		Normal	Attack	
NSL-KDD	Train	61643	57170	118813
	Test	15411	14293	29704
UNSW- NB15	Train	74400	131738	206138
	Test	18600	32935	51535

Table 3and algorithm 1 illustrates the training class distribution prior to resampling. For instance, the NSL-KDD dataset has SMOTE applied so that there are 61,643 instances in each class to ensure a balanced representation for the multiclass data. In the case of the UNSW-NB15 dataset, majority classes were downsampled to 10,000 instances before using SMOTE in order to control for data imbalance and ensure reproducibility. SMOTE was then applied to the remaining minority classes, resulting in a balanced training set with 10,000 samples.

Table 3: Distribution of Multiclass Training Data by Class on the NSL-KDD and UNSW-NB15 Datasets

Data Set	Attack (Type: Volume)
NSL-KDD	dos:42708,normal:61643,probe:11261,r2l:2999,u2r:202
UNSW-NB15	Analysis:2142, Backdoor:1863, DoS:10000, Exploits:10000, Fuzzers:10000, Generic:10000, Normal:10000, Reconnaissance:10000, Shellcode:1209, Worms =139.

3.2.1. Standardization

Features in the dataset may vary in scale, which can bias distance-based learning or gradient-based optimization. The standardization technique adjusts the data for each feature so that 0 represents the mean and 1 represents the standard deviation. To ensure that each feature contributed equally to the model, we applied Z-score standardization:

$$Z\text{-score} = \frac{x_i-\mu}{\sigma}\ ,\quad \sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\ (x_i-\mu)^2} \tag{1}$$

Here, μ and σ represent the mean and standard deviation, respectively, of a given feature. Standardization enables the model to converge faster and with better accuracy.

Algorithm 1: ***UNSW-NB15 Larger Data Preprocessing***

Input: Preprocessed training dataset, label vector

Output: Balanced training dataset capped

1. for each class label L in X_train:

2. a. if number of samples in class L > 10,000:

3. i. randomly sample 10,000 instances from class L (using fixed seed = 42)
4. b. else:
5. i. retain all instances from class L
6. c. concatenates all class-wise subsets into a single dataset

---

3.2.2. Label Encoding

Machine learning models operate with numerical values; therefore, categorical variables must be transformed into numerical values. To transform categorical labels into a machine-readable numeric format, we used label encoding.

Binary classification: Normal = 1, Attack = 0
Multiclass classification (NSDL-KDD): normal = 1, DoS = 0, probe = 2, u2r = 3, r2l = 4
Multiclass classification (UNSW-NB15): Analysis = 0, Backdoor = 1, DoS = 2, Exploits = 3, Fuzzers = 4, Generic = 5, Normal = 6, Reconnaissance = 7, Shellcode = 8, Worms = 9.

This encoding is guaranteed to be a numerical input classifier compatible with class semantics. These two steps serve as the basis for formulating distinct machine learning algorithms for developing an effective intrusion detection system for WSNs.

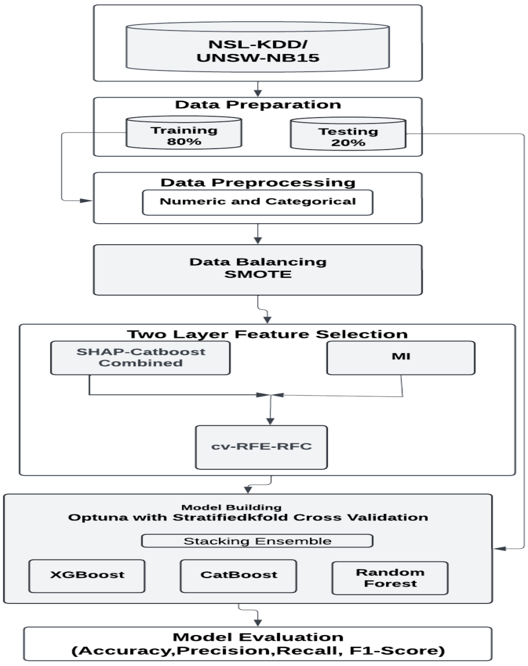


Fig.1. The proposed method for OXCRF model

3.3. Data Balancing

WSN data such as data set are inherently imbalanced with benign traffic, far outpacing instances of attacks. To address this, we applied SMOTE the Synthetic Minority Oversampling Technique (SMOTE) to the training dataset. SMOTE creates new instances of the minority class by interpolating between instances, thereby enhancing the generalization of the model and reducing the bias. Fig. 2, 3, 4 and 5 show the balanced class distribution in the training set before and after applying SMOTE for binary and multiclass classifications.

3.4. Two-Tier Feature Extraction

Following standardization, a two-tier feature selection strategy was implemented to extract relevant features to refinethe intrusion detection prediction accuracy and eliminate irrelevant and redundant features. The first layer uses an improved filtering method to remove redundant features, which is a feature selection method that minimizes the search space by selectively filtering out irrelevant or redundant features and enhancing the accuracy and speed of the second layer. This is a wrapper approach that assesses each feature's significance using a machine learning algorithm, eliminating less important features in an iterative fashion.

3.4.1. First Layer: Filter-Based Selection

The first feature extraction layer comprises the feature importance ranking and redundancy analysis. SHAP (SHapley Additive exPlanations), used in combination with CatBoost to compute feature importance scores in terms of contribution to predictions, and Mutual

Information (MI), used to compute dependency between each feature and the class label.The average SHAP value for each feature was computed using the mean formula (abs(shap_values)), where the shap_values were derived from the SHAP explainer applied to the CatBoost model. To represent these relationships, the following equations were established:

- SHAP Value:

$$SHAP\ (i) = \sum_{S \subseteq N\{i\}} \frac{|S|!(M-|S|-1)!}{M!} [f_x(S \cup \{i\}) - f_x(S)] \tag{2}$$

- Mutual Information:

$$I(X, Y) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right) \tag{3}$$

where variables X and Y are random, $P_{(X,\,Y)}$ is the joint probability distribution, and $P_{(X)}$ and $P_{(Y)}$ are the marginal probability distributions. Given these definitions, the relationships can be quantified using Equations 2 and 3:

Mean SHAP = mean(abs(shap_values))

MI = mi (x,y)

where x and y are the feature set and target variables, respectively, of the balanced training data. These metrics help in understanding the influence and dependency of the features within a dataset.

Algorithm 2 illustrate the selected features in the training set after SHAP was combined with CatBoost and MI for binary and multiclass classification.

Algorithm 2: ***SHAP and MI Based Feature Selection***

Input: X_train (Pre-processed and SMOTE-balanced training features)
y_train (Encoded target labels)

Step 1: SHAP-Based Feature Importance (Model-Driven)

Procedure calculate-shap-values (X-train, y-train):

i. train CatBoostClassifier on X-train, y-train

ii. compute shap-values

iii. compute shap-summary = mean absolute shap-values per feature

iv. Return shap-summary

Step 2: MI-Based Feature Ranking (Statistical)

Procedure calculate-mi (X-train, y-train):

i. compute MI scores for each feature

ii. Return mi-values

Step 3: Intersection of Top Features from SHAP and MI

i. Selected shap-features = shap-summary$_i$

ii. Selected mi-features = mi-values$_i$

iii. Selected-features = Intersection (shap-features, mi-features)

Step 4: Subset the data using selected features

i. X-train selected = X-train[selected-features]

ii. X-test selected = X-test-preprocessed[selected-features]

3.4.2. Second Layer: Wrapper-Based Selection (cv-RFE)

In this layer, we used Recursive Feature Elimination with Cross-Validation using a Random Forest classifier to recursively eliminate the most irrelevant features. The Random Forest model is iteratively trained using RFECV, which then assesses the significance of each feature, eliminates the least important feature, and retrains the model using a smaller set. This process employs a Stratified 5-fold cross-validation for robust feature selection. Algorithm 3 illustrates the detail about the feature section based on wrapper method. Table 4 and 5 summarizes the total number of features selected using a combination of SHAP and Mutual Information (SHAP+MI) and Recursive Feature Elimination with Cross Validation (RFECV) for the binary and multiclass classification tasks on the NSL-KDD and UNSW-NB15 dataset.

Algorithm 3: ***RFECV for Further Refinement (Wrapper Method)***

Procedure rfecv (X-train selected, y-train-encoded)

i. Initialize RandomForestClassifier as estimator

ii. Initialize RFECV with estimator, StratifiedKFold CV, and scoring metric

iii. Fit RFECV on X_train_selected, y_train_encoded

iv. Extract selected features via cvrfe

v. Return selected-features-rfe

Step 6: Final Feature Subset

i. X-train-final = Columns of X-train-selected corresponding to selected-features-rfe.

ii. X-test-final = Columns of X-test-selected corresponding to selected-features-rfe.

Table 4: Number of Selected Features on the NSL-KDD Dataset

Classification	Feature Selection	
	SHAP + MI	RFEcv
Binary	35	33
Multiclass	35	35

Table 5: Number of Selected Features on the UNSW-NB15 Dataset

Classification	Feature Selection	
	SHAP + MI	RFEcv
Binary	36	16
Multiclass	36	25

3.5. Machine Learning Model

Wolpert first introduced a stacking algorithm in 1992, and Breiman later published Stacked Regressions in 1996[19]. The stacking model architecture was composed of two distinct layers. To reach a final solution, an ensemble learning methodology uses the predictions of several base models. This ensemble method was built using three machine-learning algorithms: XGB and CatBoost were the base learners, and the Random Forest was the meta-learner. The meta model hyperparameters were optimized using the Optuna optimization method.

Extreme Gradient Boosting: The development of gradient boosting decision trees (GBDT) [20]was proposed and developed with an emphasis on the improvement of efficiency, adaptability, and portability. The improvement in GBDT performance comes from second-order derivatives and regularization techniques.

CatBoost: Is an improved version of GBDT. CatBoost reduces the gradient and prediction biases using Greedy Target-based Statistics, and prior distribution items reduce the effects of noisy data. Ordered boosting during the iteration can remove gradient bias in the gradient boosting process. CatBoost employs oblivious trees as base predictors to achieve unbiased gradient estimation and performs gradient descent, thereby reducing overfitting.

$$\widehat{x_k^l} = \frac{\sum_{j=1}^{p-1}\left(\frac{x_j - \overline{x}_{\sigma_{j},k}}{y_{\sigma_j} + a\rho}\right)}{\sum_{j=1}^{p-1}\left(\frac{x_j - \overline{x}_{\sigma_{j},k}}{1} + a\right)} \tag{4}$$

where the weight coefficient is denoted by p and the added prior term is denoted by a.

Random Forests: RF are derivative techniques that build multiple decision treesand aggregated their predictions to increase stability and accuracy. With reference to the gathering of decision trees, is akin to the forest [21]. It comprises of various tree predictors. For the prediction, each dependent variable in the tree was a random vector sampled independently, with each tree rendered from the same distribution across the entire forest.

Optuna TPEs: Optuna is an improved Bayesian method that can automatically optimize the fitting error calculation process for hyperparameters. The parameter search space can be dynamically constructed to implement pruning and search methods efficiently. In this study, TPE was used to optimize the sampling method[22].

TPE maximizes expected improvement (EI)

$$EI = \int_{-\infty}^{\infty} \max\{(x^* - x), 0\}\, P_M(x \mid y)\; dv \qquad (5)$$

where M is the hyperparameter of the model, x^* is the desired performance, the objective loss is represented by x, and $p_M(x \mid y)$ resembles the objective function and is the surrogate function. The surrogate function is modeled by TPE using the Bayes theorem.

$$p(x \mid y) = \begin{cases} \partial(x), & \text{if } x < y^* \\ g(x), & \text{if } x > y^* \end{cases} \qquad (6)$$

The density of observations formed for each observation is $\partial(x)$, such that the corresponding loss l(x) is less than y∗, and the density formed by the remaining observations is denoted by g(x). ith a high probability of $\partial(x)$ and a low probability of g(x) at point x, the tree-structured Parzen algorithm relies on y∗ being greater than the best observed y to maximize the improvement.

4. RESULTS ANALYSIS AND DISCUSSION

4.1. Experimental Setup

This study employed the Jupyter notebook in Python 3.11, utilizing libraries such as Pandas, NumPy, Matplotlib, Seaborn, and Scikit-learn.The environment ran on Windows 11 Professional with an Intel Core i7-8665U CPU at 1.90GHz (two cores and four logical processors), 500GB SSD, and 16GB RAM.

4.2. Optuna-Stacking Ensemble Learning Model

XGB and CatBoost have numerous hyperparameters, which results in a large search space for each algorithm. In the stacking approach, the performance improvement relies heavily on optimizing the meta model hyperparameters, as confirmed by Optuna in Tables 6 and 7. Algorithm 4 describes the process of using Optuna to conduct a hyperparameter search aimed at finding the best configuration for a stacking ensemble. This ensemble uses XGBoost and CatBoost as the base models and Random Forest as the meta-model. The search is directed by the average accuracy obtained from a 10-fold cross-validation.

Algorithm 4: Optimizing Hyperparameters for a Stacking Ensemble of XGBoost and CatBoost Using Optuna, with a Random Forest as the Meta Learner

Input: Pre-processed training features (X_train), Encoded labels (y_train)
Output: Best hyperparameter combination with value

Initialization:

- number of optimization trials: n_trials

- Optuna search space:

XGB parameters: n_estimators, max_depth, learning_rate

CatBoost parameters: iterations, depth, learning_rate

1. repeat for each trial in n_trials

2. sample hyperparameters for XGB from defined search space

3. sample hyperparameters for CatBoost from defined search space

4. initialize XGB and CatBoost models using sampled parameters

5. initialize Random Forest as meta-learner

6. construct a stacking ensemble model:

7. -base models: XGB and CatBoost

8. - meta model: Random Forest

9. - internal CV: StratifiedKFold (5 splits)

10. perform 10-fold cross-validation using the stacking model

11. compute mean accuracy from the 10-fold scores

12 report score to Optuna for evaluation

13 end repeat

14 return score

Table 6: Parameter settings for the Optuna algorithm used in the proposed method.

Parameters	Value
Sampler	TPES
Direction	Maximize
Iterations (n_trials)	50 (NSL-KDD), 25 (UNSW-

4.3. Evaluation Metrics

This study employed four key metrics–Accuracy, Recall, Precision, and F1-Score–to assess the model effectiveness. These metrics were calculated using the fundamental components of the confusion matrix. These measurements were established by considering four main features of the confusion matrix: *TP stands for true positive, FP for false positive, TN for true negative, and FN for false negative.*

The efficiency of the proposed approach in correctly identifying and classifying assaults was assessed using these methods.

$$\mathbf{Accuracy} = \frac{\mathbf{TP + TN}}{\mathbf{TP + TN + FP + FN}} \quad (7)$$

$$\mathbf{Precision} = \frac{\mathbf{TP}}{\mathbf{TP+FP}} \quad (8)$$

$$\mathbf{Recall} = \frac{\mathbf{TP}}{\mathbf{TP+FN}} \quad (9)$$

$$\mathbf{F1\ Score} = \mathbf{2} \times \frac{\mathbf{Precision \times Recall}}{\mathbf{Precision + Recall}} \quad (10)$$

Misclassification Rate = (Number of incorrect predictions) / (Total number of predictions) (11)

Receiver Operating Characteristic (ROC) curves were plotted based on the true-positive rate (TPR) versus false-positive rate (FPR).

$$\mathbf{AUC} = \int_{\mathbf{0}}^{\mathbf{1}} \frac{\mathbf{TP}}{\mathbf{TP+FN}}\, \boldsymbol{d}\left(\frac{\mathbf{FP}}{\mathbf{TN+FP}}\right) \quad (12)$$

Table 7: Hyperparameters of the suggested approach.

Model	Parameters	Value
XGboost	n_estimators	100-300
	max_depth	3-12
	learning rate	1e-4 – 1e-1
	eval_metric	Mlogloss
Catboost	Iterations	50-200
	Depth	3-12
	learning rate	1e-4 – 1e-1
Random Forest	n_estimators	100
	max_depth	None
	min_leaf_nodes	1
	max_leaf_nodes	None
	Random_state	42

4.4. Binary Classification

The performance evaluation outcomes for the various machine-learning models employed in intrusion detection are presented in Table 8 and 10. These models include ExtraTrees, LGBM, XGB, ensemble stacking approaches, OLCRF, and OXCRF. The hyperparameters (Tables 6 and 7) underwent 50 and 25 iterationsof optimization using the Optuna frameworkfor NSL-KDD and UNSW-NB15 dataset. The optimized hyperparameter historywhere trial 47 produced the best results: 99.58% accuracy with parameters **{'n_estimators': 271,'max_depth': 10, 'learning_rate': 0.0897, 'iterations': 122, 'depth': 11}. Figure 15** where trial 11 produced the best results: 98.80% accuracy with parameters **{'n_estimators': 170, 'max_depth': 12, 'learning_rate': 0.0843, 'iterations': 200, 'depth': 3}.**

Table 8: Evaluation Matrix of Binary Classification Models for Intrusion Detection on NSL-KDD.

Model	Precision	Recall	F1-Score	Misclassification Rate
ExtraTrees	93.60%	98.77%	96.12%	0.0414
LGBM	99.43%	99.65%	99.54%	0.0048
XGB	99.53%	99.65%	99.59%	0.0043
OLCRF	99.12%	98.90%	99.01%	0.0073
OXCRF	**99.60%**	**99.62%**	**99.61%**	0.0040

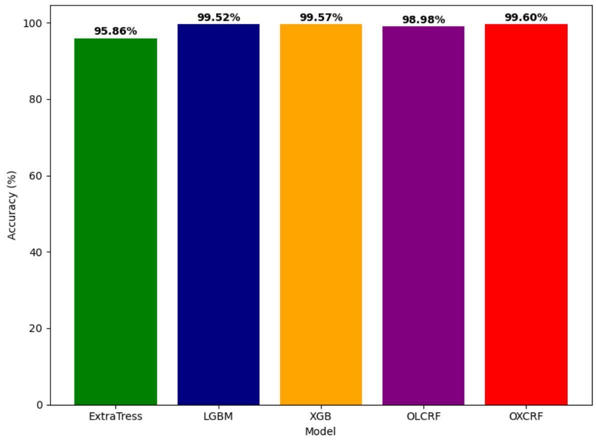


Fig.2. Performance Evaluation of Models Based on Accuracy (%) on NSL-KDD.

The OXCRF model achieved an accuracy of 99.60%, 98.62% on NSL-KDD and UNSW-NB15 respectively based on the binary classification presented in Figure 2 and 3. Table 9 and 11 presents how well various classification models performed on the NSL-KDD and UNSW-NB15 dataset, broken down by class. The accuracy for both the 'Normal' and 'Attack' classes is shown separately to highlight each model's performance in each specific category.Tables 8 and 10 summarize the overall performance of various classification models on both the dataset using key evaluation metrics, including precision, recall, F1-score, and misclassification rate.OXCRF exhibited a misclassification rate of 0.0040 and 0.0138 respectively on both the dataset. Figure 6 and 7 illustrates the confusion matrix and ROC curve of the OXCRF model for binary classification on NSL-KDD and UNSW-NB15. Along with fewer false positives and negatives, NSL-KDD provides a marginally better balance between precision and recall. Conversely, UNSW-NB15 exhibits a higher ROC-AUC, indicating greater robustness and superior class separability, despite having slightly lower accuracy.

Table 9: Accuracy for each model in Binary Classification on NSL-KDD.

Model	Normal	Attack
ExtraTrees	98.77%	92.72%
LGBM	99.65%	99.38%
XGB	99.65%	99.45%
OLCRF	99.35%	99.18%
OXCRF	**99.62%**	**99.57%**

Table 10: Evaluation Matrix of Binary Classification Models for Intrusion Detection on UNSW-NB15.

Model	Precision	Recall	F1-Score	Misclassification
ExtraTrees	93.21%	96.17%	94.67%	0.0693
LGBM	98.36%	97.51%	97.94%	0.0263
XGB	95.74%	98.25%	96.98%	0.0391
OLCRF	98.22%	97.61%	97.91%	0.0266
OXCRF	**99.11%**	**98.72%**	**98.92%**	0.0138

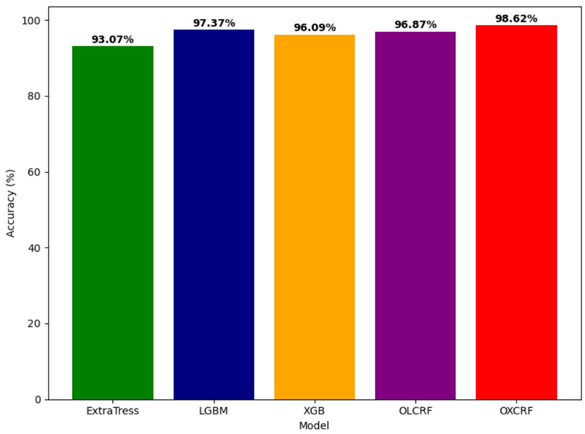


Fig. 3. Performance Evaluation of Models Based on Accuracy (%)on UNSW-NB15.

Table 11: Accuracy for each model in Binary Classification on UNSW-NB15.

Model	Normal	Attack
ExtraTrees	87.59%	96.17%
LGBM	97.12%	97.51%
XGB	92.25%	98.25%
OLCRF	96.87%	97.61%
OXCRF	**98.44%**	**98.72%**

The classification accuracies and the corresponding t-test p-values for each of the binary classification tasks on the NSL-KDD and UNWS-NB15 datasets are shown in Tables 12 and 13, respectively, utilising 10-fold cross-validation. A p-value of less than 0.05 indicates that the null hypothesis of statistical equivalence across classifiers can be rejected. For the NSL-KDD dataset (Table 12), OXCRF is significant and achieved the highest classification accuracy and outperformed classification accuracy compared to both OLCRF and ExtraTree. However, there was no significant improvement over the XGB classifier as indicated by its p-value of 0.0859. For the UNSW-NB15 dataset (Table 13), the OXCRF classifier made the highest improvement in performance over all the baseline classifiers, achieving an average accuracy of 98.320% ± 0.094 and the results indicate a statistically significant.

Table 12: Results of average accuracy (%) with standard deviations, Paired t-test p-value in terms of 10-fold cross validation for Binary Classification on NSL-KDD.

Method	Accuracy	Paired t-test p-value
OXCRF	99.581±0.068	--
OLCRF	99.212±0.094	0.0000
XGB	99.533±0.058	0.0859
LGB	99.506±0.046	0.0071
ExtraTree	95.745±0.201	0.0000

Table 13: Results of average accuracy (%) with standard deviations, Paired t-test p-value in terms of 10-fold cross validation for Binary Classification on UNSW-NB15.

Method	Accuracy	Paired t-test p-value
OXCRF	98.320±0.094	--
OLCRF	97.585±0.075	0.0000
XGB	95.425±0.106	0.0000
LGB	98.255±0.085	0.0241
ExtraTree	92.039±0.147	0.0000

4.5. Multiclass Classification

The accuracy of the ensemble-stacking OXCRF model for multiclass was 99.53% and 83.67% on NSL-KDD and UNSW-NB15 respectively. Over 50 and 25 iterations on NSL-KDD and UNSW-NB15, the Optuna framework was used to optimize the hyperparameters (Tables 6 and 7). The optimization history where trial 22 produced an accuracy of 99.83% with parameters **{{'n_estimators': 244,'max_depth': 8, 'learning_rate': 0.0989, 'iterations': 69, 'depth': 6} on NSL-KDD.**Best is trial 21 with value: 77.75 with parameters: **{'n_estimators': 48, 'max_depth': 6, 'learning_rate': 0.08908701476048053, 'iterations': 41, 'depth': 6}** on UNSW-NB15.

Tables 14 and 15 summarize the overall performance of various classification models on both the dataset using key evaluation metrics, including precision, recall, F1-score, and misclassification rate. OXCRF exhibited a misclassification rate of 0.0047 and 0.1633 respectively on both the dataset. Table 16 and 17 presents the evaluation outcomes for the machine learning models employed in intrusion detection, compares the class-specific accuracies on both the data set, highlighting the performance of the models. Figure 4and 5compares the accuracies highlighting the performance of eachmodel on NSL-KDD and UNSW-NB15.Figure 8,9 and 10 illustrates the classification report, confusion matrix and ROC curve. Even though all the models perform well on the NSL-KDD dataset when it comes to accuracy, just looking at accuracy isn't enough for multi-class intrusion detection. This is especially true when the classes are very uneven. So, we evaluated each model using macro-F1, micro-F1, and weighted-F1 scores, along with the macro-average ROC-AUC. Our proposed OXCRF model did great, getting a macro-F1 score of 0.9658 and a macro-average AUC of 0.9969. surpassing baseline models in both minority and majority class performance. Also, per-class confusion matrices for each class validatethe OXCRF model is really good at correctly identifying rare attack types like u2r and r2l, where conventional models such as ExtraTrees and LGBM experience significant performance decline.

Table 14: Results for Intrusion Detection in Multiclass Classification on NSL-KDD.

Model	Precision	Recall	F1-Score	Misclassification Rate
ExtraTrees	99.44%	99.44%	99.44%	0.0056
LGBM	99.48%	99.47%	99.47%	0.0053
XGB	99.50%	99.49%	99.49%	0.0051
OLCRF	99.53%	99.52%	99.52%	0.0048
OXCRF	**99.54%**	**99.53%**	**99.53%**	0.0047

Table 15: Results for Intrusion Detection in Multiclass Classification on UNSW-NB15.

Model	Precision	Recall	F1-Score	Misclassification Rate
ExtraTrees	84.46%	80.41%	81.70%	0.1959
LGBM	87.59%	81.59%	82.94%	0.1841
XGB	88.33%	88.06%	84.18%	0.1694
OLCRF	87.16%	81.88%	82.85%	0.1812
OXCRF	**87.94%**	**83.67%**	**84.73%**	0.1633

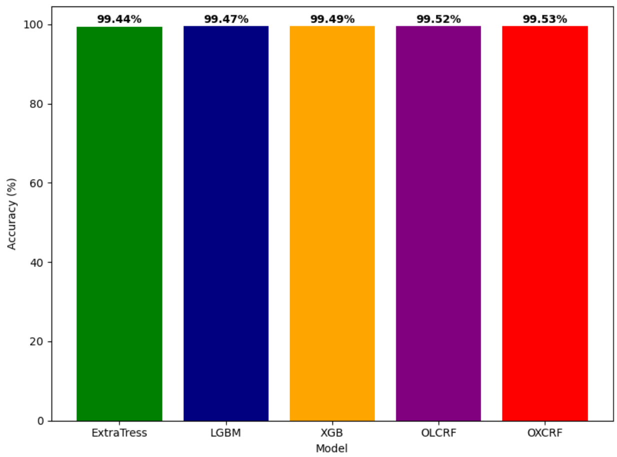


Fig. 4. Performance Evaluation of Models Based on Accuracy (%)on NSL-KDD.

Table 16: Accuracy for each class in multiclass classification on NSL-KDD.

Model	normal	Dos	probe	r2l	u2r
ExtraTrees	99.43%	99.91%	99.61%	93.33%	84.00%
LGBM	99.33%	**99.93%**	99.36%	**96.67%**	92.00%
XGB	99.44%	99.91%	99.36%	95.73%	92.00%
OLCRF	99.48%	99.90%	99.50%	95.47%	**94.00%**
OXCRF	**99.49%**	99.91%	**99.47%**	95.73%	**94.00%**

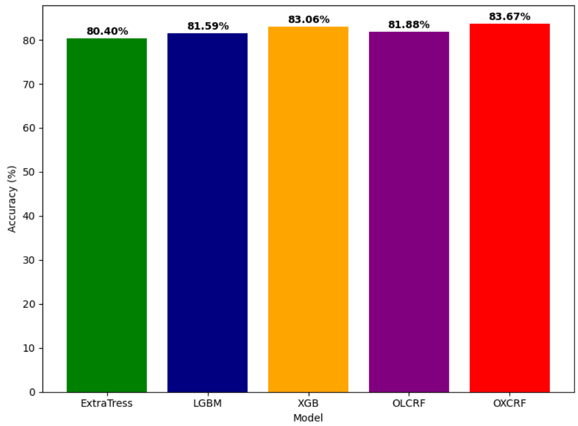


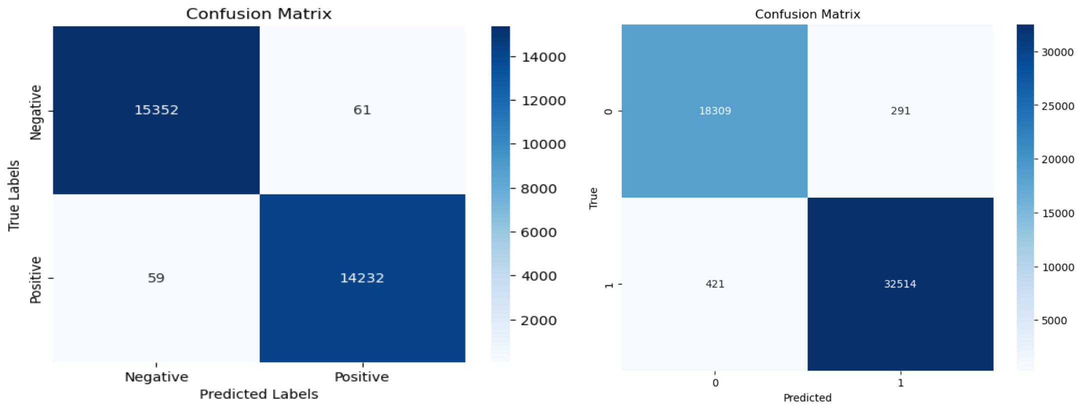
Fig. 5. Performance Evaluation of Models Based on Accuracy (%)on UNSW-NB-15.

Figure 11,12 and 13 illustrates illustrate the classification report, confusion matrix and ROC model curve. The best overall performance was obtained by the OXCRF ensemble with best macro-F1 (0.6307), mictheF1 (0.8367), weighted-F1 (0.8473) and accuracy (0.84), indicating good performance for attack class balance. As the macro-F1) outperformed LGB UNSW-NB15 is imbalanced, OXCRF (0.6307 for (0.5805) and ExtraTree (0.5746), the three showed better minority attack on Normal, Generic, and detection. Although all models produced good results Exploits classes, they showed variations on rare attacks. Poor F1-scoring classes (except in OXCRF) in Analysis, Backdoor, Worms in Backdoor and Analysis and LGB. The comparisons in Backdoor, as well as Shellcode, Worms in ExtraTree between the OLCRF and OXCRF models indicated that the ensemble-based models learn were more robust than any individual all scores, er. OXCRF outperformed in Normal and Generic was and ExtraTrees rejected with poorest macro-AUC. The universally well classified in confusion matrix, but minority classes suffered of misclassifications in non-ensemble models. The overall class discrimination OXCRF was better.

Table 17: Accuracy for each class in multiclass classification on UNSW-NB15.

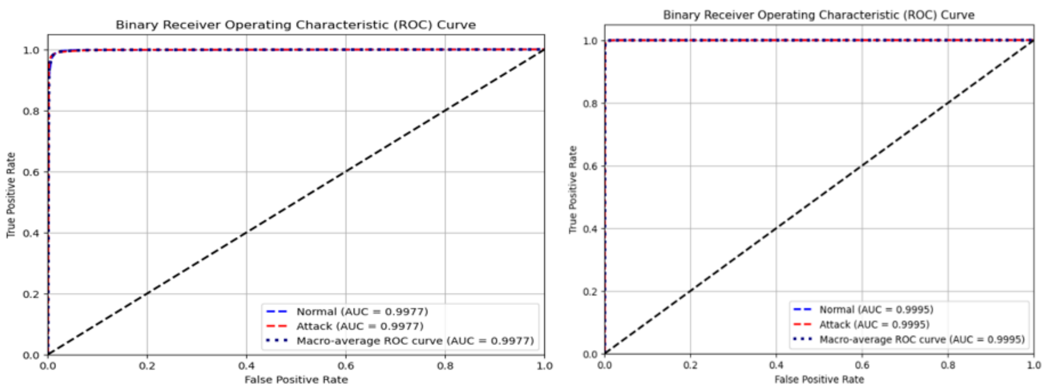
Model	0	1	2	3	4	5	6	7	8	9
ExtraTrees	**27.10%**	18.67%	48.67%	**58.32%**	**83.93%**	97.60%	87.54%	62.45%	81.13%	68.57%
LGBM	25.98%	**32.62%**	77.19%	48.30%	82.08%	97.36%	90.74%	**82.52%**	**91.06%**	**80.00%**
XGB	26.73%	29.40%	**79.67%**	50.48%	83.42%	97.66%	92.60%	82.42%	88.74%	**80.00%**
OLCRF	36.45%	11.16%	77.80%	45.40%	82.88%	97.61%	92.87%	82.38%	83.77%	57.14%
OXCRF	24.04%	26.61%	74.75%	53.85%	82.88%	**97.97%**	**93.91%**	82.42%	84.44%	68.57%

Analysis = 0, Backdoor = 1, DoS = 2, Exploits = 3, Fuzzers = 4, Generic = 5, Normal = 6, Reconnaissance = 7, Shellcode = 8, Worms = 9.



(a) NSL-KDD (b) UNSW-NB-15

Fig.6. Confusion Matrix for Binary Classification(a) NSL-KDD and (b) UNSW-NB15



(a) NSL-KDD (b) UNSW-NB15

Fig. 7: ROC Curve for Binary Classification (a) NSL-KDD and (b) UNSW-NB15

Classification Reportof XGB-NSL-KDD:

	precision	recall	f1-score	support
dos	1.00	1.00	1.00	10677
normal	1.00	0.99	1.00	15411
probe	0.99	0.99	0.99	2816
r2l	0.93	0.96	0.94	750
u2r	0.81	0.92	0.86	50
accuracy			0.99	29704
macro avg	0.95	0.97	0.96	29704
weighted avg	1.00	0.99	0.99	29704

Overall F1 Metrics:
Macro-F1 : 0.9582
Micro-F1 : 0.9949
Weighted-F1 : 0.9950

(a) XGB

Classification Reportof LGB-NSL-KDD:

	precision	recall	f1-score	support
dos	1.00	1.00	1.00	10677
normal	1.00	0.99	1.00	15411
probe	0.99	0.99	0.99	2816
r2l	0.91	0.97	0.94	750
u2r	0.82	0.92	0.87	50
accuracy			0.99	29704
macro avg	0.94	0.97	0.96	29704
weighted avg	0.99	0.99	0.99	29704

Overall F1 Metrics:
Macro-F1 : 0.9588
Micro-F1 : 0.9947
Weighted-F1 : 0.9947

(b) LGBM

Classification Reportof ExtraTree-NSL-KDD:

	precision	recall	f1-score	support
dos	1.00	1.00	1.00	10677
normal	1.00	0.99	0.99	15411
probe	0.99	1.00	1.00	2816
r2l	0.92	0.93	0.93	750
u2r	0.88	0.84	0.86	50
accuracy			0.99	29704
macro avg	0.96	0.95	0.95	29704
weighted avg	0.99	0.99	0.99	29704

Overall F1 Metrics:
Macro-F1 Score : 0.9546
Micro-F1 Score : 0.9944
Weighted-F1 Score: 0.9944

(c)ExtraTree

Classification Reportof OLCRF-NSL-KDD:

	precision	recall	f1-score	support
dos	1.00	1.00	1.00	10677
normal	1.00	0.99	1.00	15411
probe	0.99	1.00	0.99	2816
r2l	0.93	0.95	0.94	750
u2r	0.85	0.94	0.90	50
accuracy			1.00	29704
macro avg	0.96	0.98	0.97	29704
weighted avg	1.00	1.00	1.00	29704

Overall F1 Metrics:
Macro-F1 : 0.9656
Micro-F1 : 0.9952
Weighted-F1 : 0.9952

(d) OLCRF

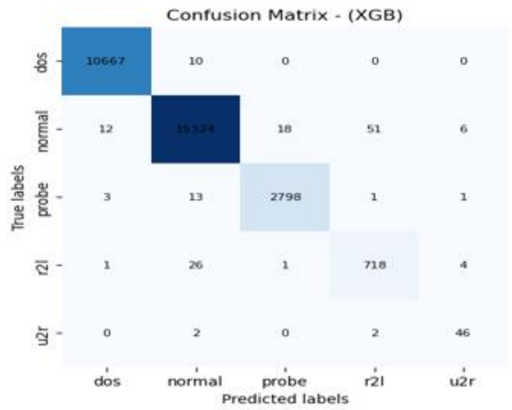
Classification Reportof OXCRF-NSL-KDD:

	precision	recall	f1-score	support
dos	1.00	1.00	1.00	10677
normal	1.00	0.99	1.00	15411
probe	0.99	0.99	0.99	2816
r2l	0.93	0.96	0.94	750
u2r	0.85	0.94	0.90	50
accuracy			1.00	29704
macro avg	0.96	0.98	0.97	29704
weighted avg	1.00	1.00	1.00	29704

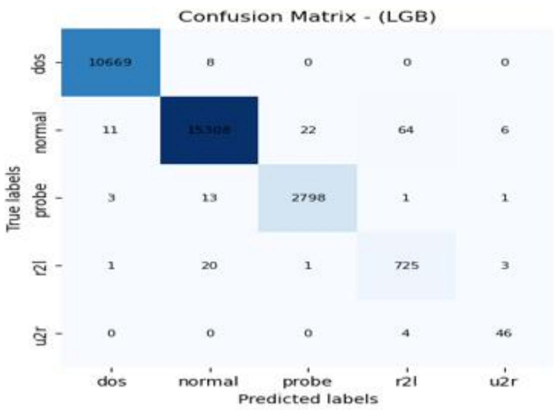
Overall F1 Metrics:
Macro-F1 : 0.9658
Micro-F1 : 0.9953
Weighted-F1 : 0.9953

(e) OXCRF

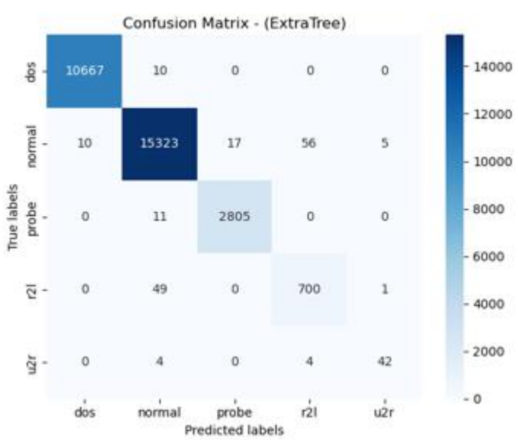
Fig.8. Classification Report for Multiclass Classication on NSL-KDD a) XGB b)LGB c)ExtraTree d)OLCRF e) OXCRF



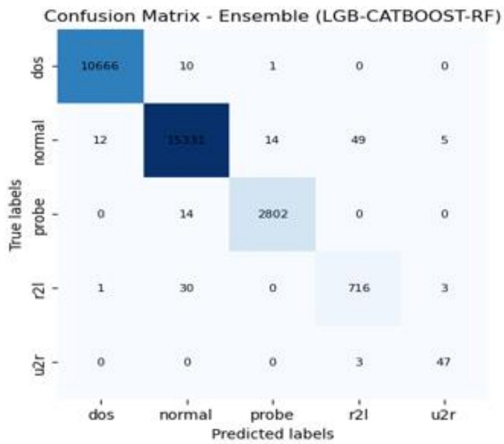
(a) XGB



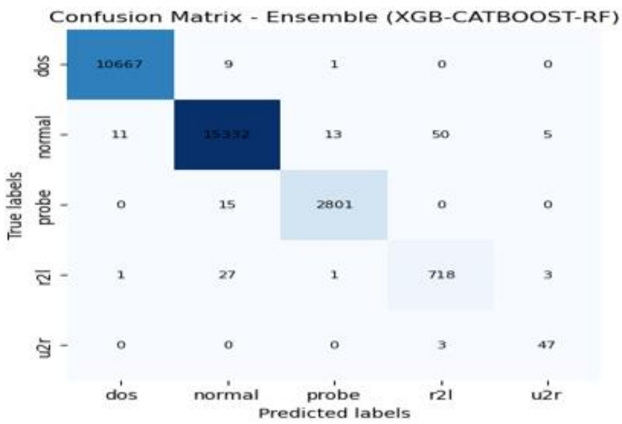
(b) LGBM



(c) ExtraTree

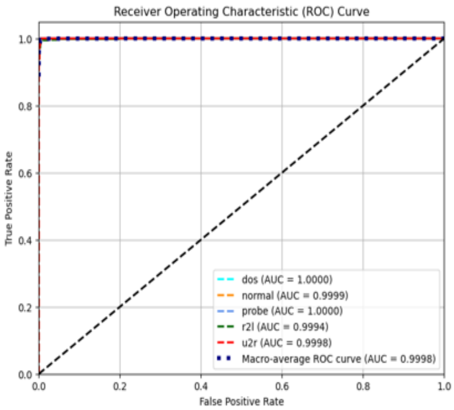


(d) OLCRF

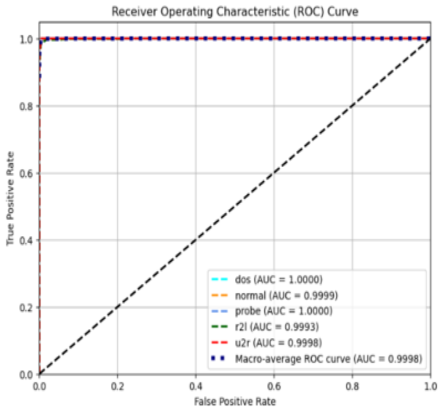


(e) OXCRF

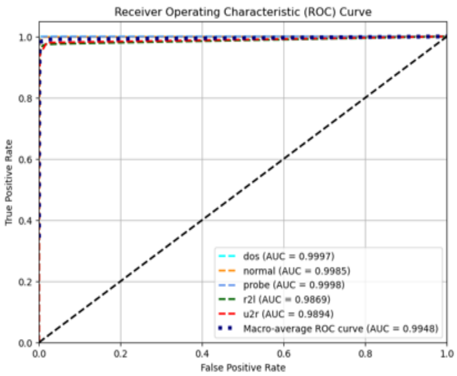
Fig.9. Confusion Matrix for Multiclass Classication on NSL-KDD a) XGB b)LGB c)ExtraTree d)OLCRF e) OXCRF



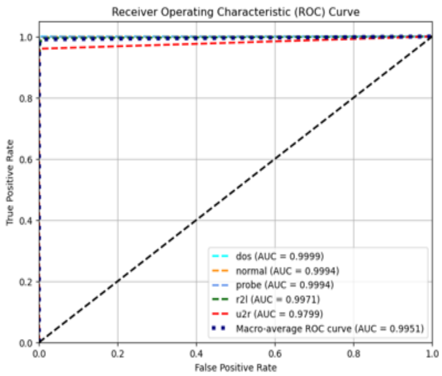
(a) XGB



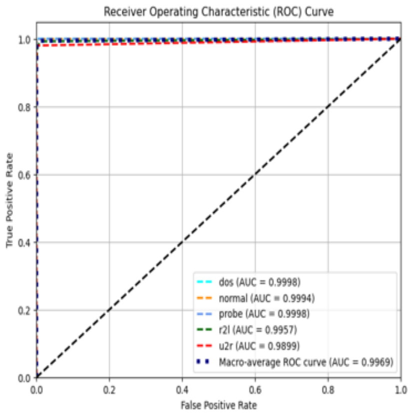
(b) LGBM



(c) Extra Tree



(d) OLCRF



(e) OXCRF

Fig.10. ROC for Multiclass Classication on NSL-KDD a) XGB b)LGB c)ExtraTree d)OLCRF e) OXCRF

Classification Reportof XGB-UNSW-NB:

	precision	recall	f1-score	support
Analysis	0.27	0.27	0.27	535
Backdoor	0.17	0.29	0.22	466
DoS	0.36	0.80	0.49	3271
Exploits	0.88	0.50	0.64	8905
Fuzzers	0.76	0.83	0.79	4849
Generic	1.00	0.98	0.99	11774
Normal	0.99	0.93	0.96	18600
Reconnaissance	0.83	0.82	0.83	2798
Shellcode	0.45	0.89	0.60	302
Worms	0.22	0.80	0.35	35
accuracy			0.83	51535
macro avg	0.59	0.71	0.61	51535
weighted avg	0.88	0.83	0.84	51535

Overall F1 Metrics:
Macro-F1 : 0.6135
Micro-F1 : 0.8306
Weighted-F1 : 0.8418

(a) XGB

Classification Reportof LGB-UNSW-NB:

	precision	recall	f1-score	support
Analysis	0.26	0.26	0.26	535
Backdoor	0.16	0.33	0.21	466
DoS	0.35	0.77	0.48	3271
Exploits	0.86	0.48	0.62	8905
Fuzzers	0.71	0.82	0.76	4849
Generic	1.00	0.97	0.99	11774
Normal	0.99	0.91	0.95	18600
Reconnaissance	0.82	0.83	0.83	2798
Shellcode	0.34	0.91	0.50	302
Worms	0.12	0.80	0.21	35
accuracy			0.82	51535
macro avg	0.56	0.71	0.58	51535
weighted avg	0.88	0.82	0.83	51535

Overall F1 Metrics:
Macro-F1 : 0.5805
Micro-F1 : 0.8159
Weighted-F1 : 0.8294

(b) LGBM

Classification Reportof ExtraTree-UNSW-NB:

	precision	recall	f1-score	support
Analysis	0.13	0.27	0.18	535
Backdoor	0.11	0.19	0.14	466
DoS	0.35	0.49	0.41	3271
Exploits	0.79	0.58	0.67	8905
Fuzzers	0.61	0.84	0.71	4849
Generic	1.00	0.98	0.99	11774
Normal	0.99	0.88	0.93	18600
Reconnaissance	0.70	0.82	0.76	2798
Shellcode	0.43	0.81	0.57	302
Worms	0.29	0.69	0.40	35
accuracy			0.80	51535
macro avg	0.54	0.65	0.57	51535
weighted avg	0.84	0.80	0.82	51535

Overall F1 Metrics:
Macro-F1 : 0.5746
Micro-F1 : 0.8041
Weighted-F1 : 0.8170

(c) ExtraTree

Classification Reportof OLCRF-UNSW-NB:

	precision	recall	f1-score	support
Analysis	0.21	0.36	0.27	535
Backdoor	0.24	0.11	0.15	466
DoS	0.33	0.78	0.46	3271
Exploits	0.83	0.45	0.59	8905
Fuzzers	0.73	0.83	0.78	4849
Generic	1.00	0.98	0.99	11774
Normal	0.99	0.93	0.96	18600
Reconnaissance	0.83	0.82	0.83	2798
Shellcode	0.45	0.84	0.59	302
Worms	0.24	0.57	0.33	35
accuracy			0.82	51535
macro avg	0.59	0.67	0.59	51535
weighted avg	0.87	0.82	0.83	51535

Overall F1 Metrics:
Macro-F1 : 0.5946
Micro-F1 : 0.8188
Weighted-F1 : 0.8285

(d) OLCRF

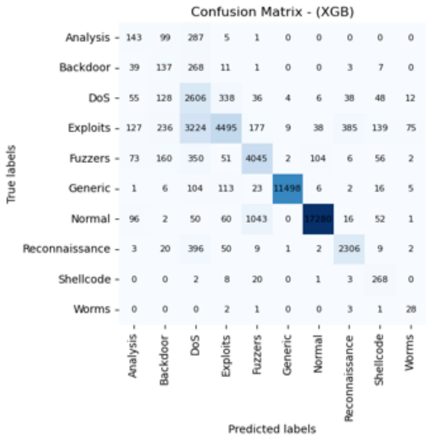
Classification Reportof OXCRF-UNSW-NB:

	precision	recall	f1-score	support
Analysis	0.24	0.28	0.26	535
Backdoor	0.18	0.27	0.21	466
DoS	0.36	0.75	0.48	3271
Exploits	0.84	0.54	0.66	8905
Fuzzers	0.78	0.83	0.81	4849
Generic	1.00	0.98	0.99	11774
Normal	0.99	0.94	0.96	18600
Reconnaissance	0.83	0.82	0.83	2798
Shellcode	0.51	0.84	0.64	302
Worms	0.36	0.69	0.48	35
accuracy			0.84	51535
macro avg	0.61	0.69	0.63	51535
weighted avg	0.88	0.84	0.85	51535

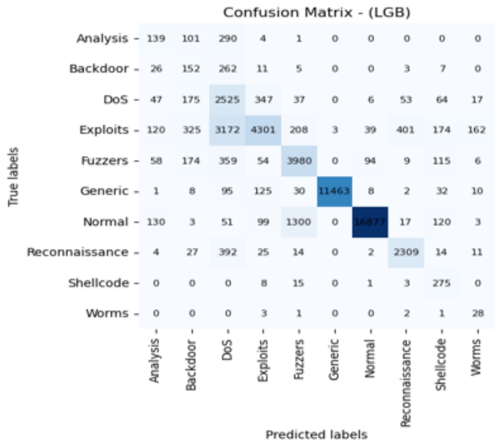
Overall F1 Metrics:
Macro-F1 : 0.6307
Micro-F1 : 0.8367
Weighted-F1 : 0.8473

(e) OXCRF

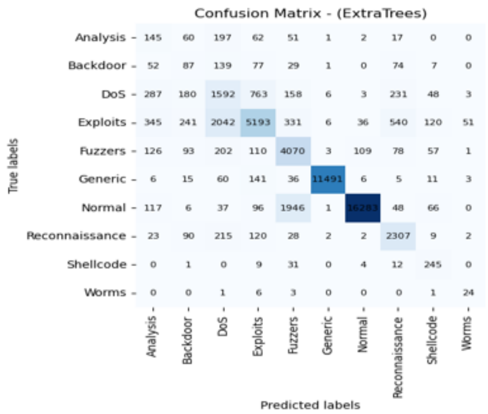
Fig.11. Classification Report for Multiclass Classication on UNSW-NB15 a) XGB b)LGB c)ExtraTree d)OLCRF e) OXCRF



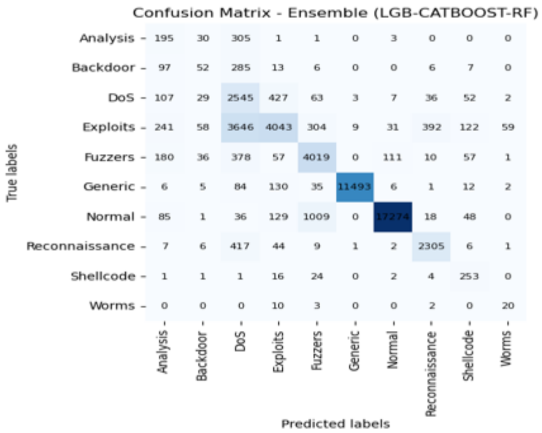
(a) XGB



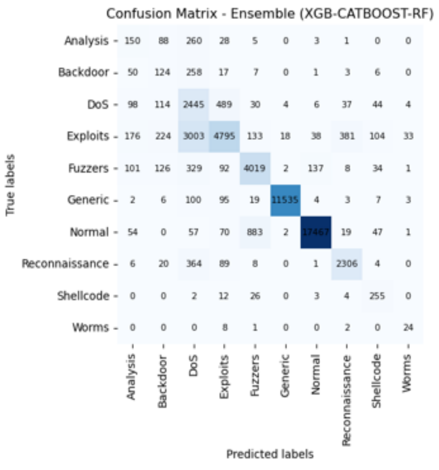
(b) LGBM



(c) ExtraTree



(d) OLCRF



(e) OXCRF

Fig.12. Confusion Matrix for Multiclass Classication on UNSW-NB15 a) XGB b)LGB c)ExtraTree d)OLCRF e) OXCRF

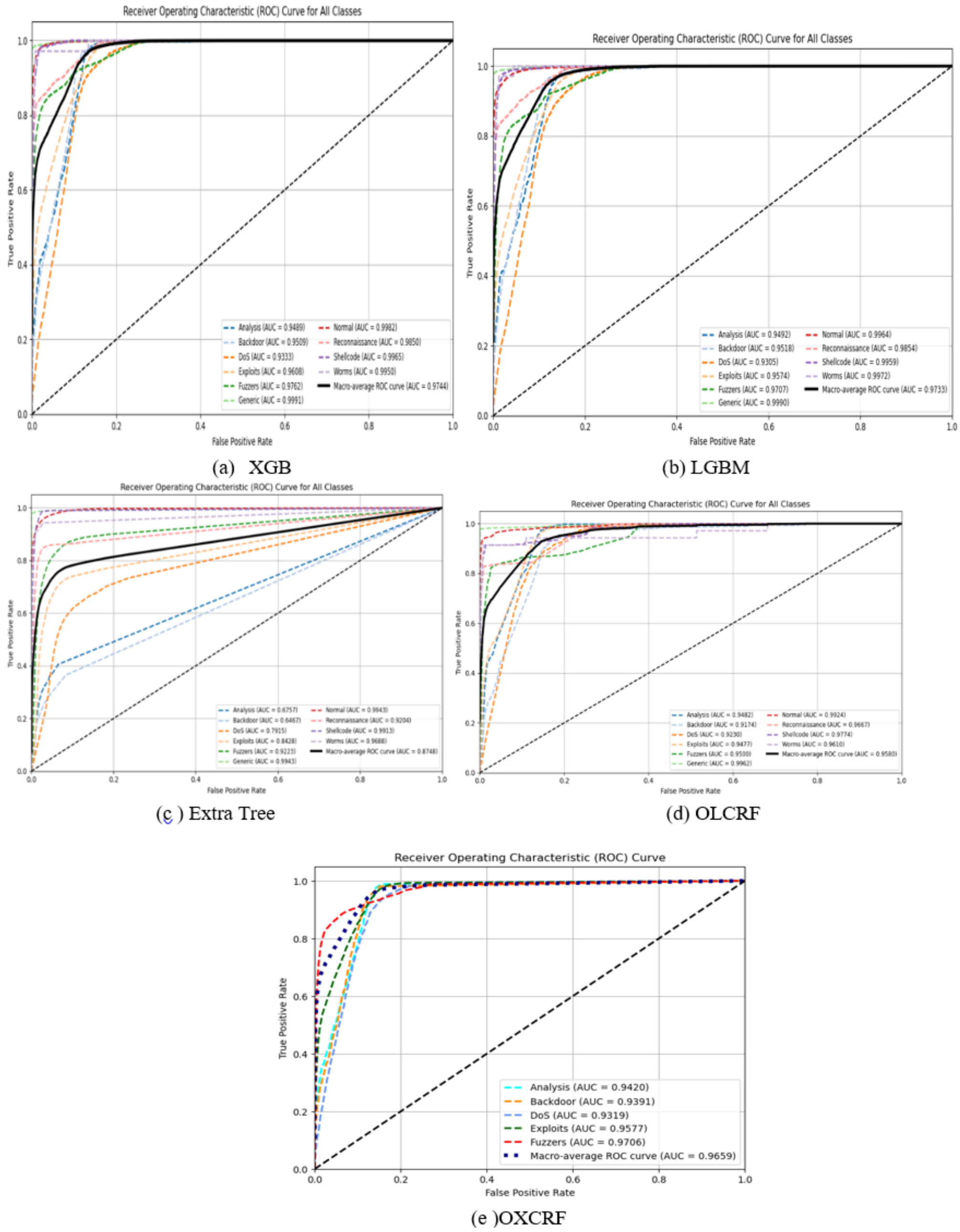
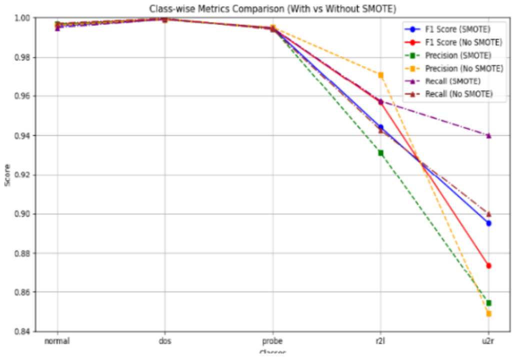


Fig.13. ROC for Multiclass Classication on NSL-KDD a) XGB b)LGB c)ExtraTree d)OLCRF e) OXCRF

4.6. Ablation Study

An in-depth ablation study was conducted to evaluate the effects of the components within the OXCRF pipeline. Elements such as SMOTE for balancing classes, two-stage feature selection process, and Optuna-driven hyperparameter optimization were removed individually while keeping the other components intact. Figure14illustrates the performance evaluation for multiclass precision, recall, and F1 measures with and without SMOTE. Without SMOTE, minority classes such as R2L and U2R had a very low recall because the data imbalance favoredthe majority classes. With SMOTE, the minority classes were better balanced, and the scores were higher in all categories.



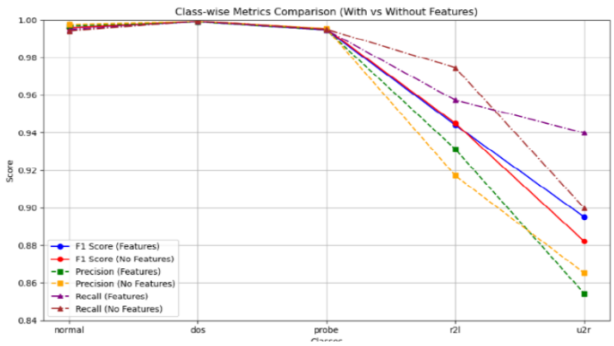


Fig. 14. Comparison for with and without SMOTE.Fig.15.Comparison for with and without feature selection

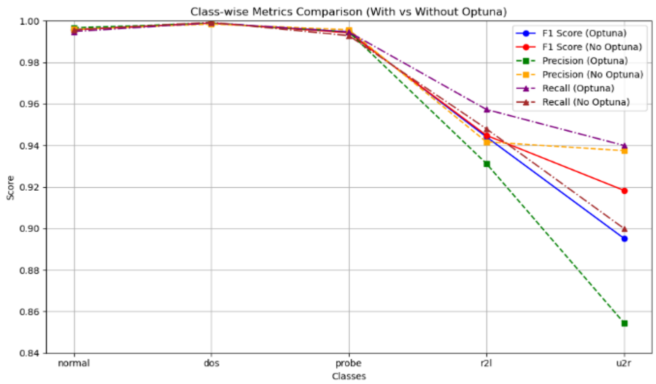


Fig. 16. Comparison for with and without Optuna.

Deployment Challenges of WSNs Deploying this model in WSN environment faces challenges on deployment: using pruned decision trees or thin boosted trees; model distillation for reducing ensemble size; hierarchical IDS architecture, using light-weight pre-detector on the motes and ensemble on cluster heads. Limitations of NSL-KDD Dataset While NSL-KDD full be seen as an improvement to KDD'99, it is not without its limitations. The can as adversarial above dataset does not account for changing attack vectors such hop count and trust evasion and routing misbehavior in WSNs. It lacks energy, include types of level which are the essential issues of WSN. It does not attacks such as Wormhole and Sybil, which restricts generalization. Because is unable to model spatial correlations or support NSL-KDD is flow-based, it node mobility.

4.7. Comparison with Previous Study

The significance of the proposed approach is reviewed based on existing literature. The role played by feature selection methods is crucial for discovering the right and advanced features. Many studies have discussed traditional selection methods and obtained satisfactory results. OXCRF exhibited a misclassification rate of 0.0047, compared to 0.0056, 0.0053, 0.0051, and 0.0048 for ExtraTrees, LGBM, XGB, and stacking ensemble OLCRF, respectively. Table 6 presents the evaluation outcomes for the machine learning models employed in intrusion detection. Table 7 compares the class-specific accuracies (normal, dose, probe, r2l, and u2r). For example, in[8], the correlation-based method, as presented above, achieved 90.38% and 91.33% accuracy through the SVM and KNN classifiers, respectively. The models presented better results than the present study: all such binary and multiclass classifications for the IDS-RF model in the research [23] were 98.67% and 98.54%, respectively, and utilized the SMOTE and RF approaches. Table 18 and Figures20show a comparison of the significance of the current study with that of earlier published studies.

Table 18: Evaluation of OXCRF in relation to other binary and multiclass classification studies.

Binary		**Multiclass**	
Study	Accuracy	Study	Accuracy
IDS-RF[23]	98.67%	IDS-RF[23]	98.54%
SMA[13]	99.39%	adaptive SVM [2]	84.00%
SKM-XGB[7]	99.37%	MARL[1]	97.44%
CNN-LSM-SA [24]	89.36%	CNN-LSM-SA[24]	93.72%
OXCRF(NSL-KDD)	99.60%	OXCRF(NSL-KDD)	99.53%
OXCRF(UNSW-NB15)	98.62%	OXCRF(UNSW-NB15)	83.67%

5. CONCLUSIONS AND FUTURE WORK

In this work, we proposed a stacking ensemble learning technique with the help of Optuna to WSN a as OXCRF. The model showed based intrusion detection and referred to it significant performance in binary and multiclass classification, tested on NSL-KDD and UNSW-NB15 datasets.

Key components of the OXCRF model included:

1. CatBoost, Mutual An ensemble method for feature selection with SHAP, Information Correlation, and cross-validated Recursive Feature Elimination with Random Forest.
2. Class Application of SMOTE to overcome the Imbalances.
3. stacking ensemble of Optuna with XGBoost and CatBoost The tuning included a as base learners and Random Forest as the meta-learner.

The accuracy of the model for the binary and multiclass classification NSL-KDD data set got to 99.60% and 99.53% respectively which is better using than existing method and single classifier. High-dimensional data problem was successfully handled by two stage feature selection method to reduce for WSNs. The ablation study dimensionality and improve model efficiency verified that each component in the OXCRF framework plays an important role, especially for the class overlap and data imbalance issue in intrusion detection. multiclass

The ablation study validated the importance of every aspect of the OXCRF framework, particularly The class overlap and multiclass intrusion detection data imbalance management. Future research should focus on the effectiveness of the model on various datasets as well as integrating deep learning-based models to address new attack problems in real-time applications.

AUTHOR CONTRIBUTIONS

All the authors contributed equally to, read, and approved the final manuscript.

FUNDING

No funding.

DATA ACCESS STATEMENT

Data is available based on the request.

CONFLICT OF INTEREST

None the authors have any conflict of interests to disclose. of Yes, we confirm the proposed study work on Research Square preprint is [25]

REFERENCES

[1] F. Louati, F. B. Ktata, and I. Amous, "Big-IDS: a decentralized multi agent reinforcement learning approach for distributed intrusion detection in big data networks," Cluster Comput, vol. 27, no. 5, pp. 6823–6841, Aug. 2024, doi: 10.1007/s10586-024-04306-9.

[2] G. M. Borkar, L. H. Patil, D. Dalgade, and A. Hutke, "A novel clustering approach and adaptive SVM classifier for intrusion detection in WSN: A data mining concept," Sustainable Computing: Informatics and Systems, vol. 23, pp. 120–135, Sep. 2019, doi: 10.1016/j.suscom.2019.06.002.

[3] W. F. Urmi et al., "A stacked ensemble approach to detect cyber attacks based on feature selection techniques," International Journal of Cognitive Computing in Engineering, vol. 5, pp. 316–331, Jan. 2024, doi: 10.1016/j.ijcce.2024.07.005.

[4] H. Lin, Q. Xue, J. Feng, and D. Bai, "Internet of things intrusion detection model and algorithm based on cloud computing and multi-feature extraction extreme learning machine," Digital Communications and Networks, vol. 9, no. 1, pp. 111–124, Feb. 2023, doi: 10.1016/J.DCAN.2022.09.021.

[5] M. Haggag, M. M. Tantawy, and M. M. S. El-Soudani, "Implementing a deep learning model for intrusion detection on apache spark platform," IEEE Access, vol. 8, pp. 163660–163672, 2020, doi: 10.1109/ACCESS.2020.3019931.

[6] B. Al-Fuhaidi, Z. Farae, F. Al-Fahaidy, G. Nagi, A. Ghallab, and A. Alameri, "Anomaly-Based Intrusion Detection System in Wireless Sensor Networks Using Machine Learning Algorithms," Applied Computational Intelligence and Soft Computing, vol. 2024, no. 1, Jan. 2024, doi: 10.1155/2024/2625922.

[7] M. K. Hooshmand, M. D. Huchaiah, A. R. Alzighaibi, H. Hashim, E. S. Atlam, and I. Gad, "Robust network anomaly detection using ensemble learning approach and explainable artificial intelligence (XAI)," Alexandria Engineering Journal, vol. 94, pp. 120–130, May 2024, doi: 10.1016/j.aej.2024.03.041.

[8] A. G. Mari, D. Zinca, and V. Dobrota, "Development of a Machine-Learning Intrusion Detection System and Testing of Its Performance Using a Generative Adversarial Network," Sensors, vol. 23, no. 3, Feb. 2023, doi: 10.3390/s23031315.

[9] M. A. Hossain and M. S. Islam, "Ensuring network security with a robust intrusion detection system using ensemble-based machine learning," Array, vol. 19, Sep. 2023, doi: 10.1016/j.array.2023.100306.

[10] Q. Abbas, S. Hina, H. Sajjad, K. S. Zaidi, and R. Akbar, "Optimization of predictive performance of intrusion detection system using hybrid ensemble model for secure systems," PeerJComput Sci, vol. 9, 2023, doi: 10.7717/peerj-cs.1552.

[11] B. Mohammed and E. Gbashi, "Intrusion Detection System for NSL-KDD Dataset Based on Deep Learning and Recursive Feature Elimination," Engineering and Technology Journal, vol. 39, no. 7, pp. 1069–1079, Jul. 2021, doi: 10.30684/etj.v39i7.1695.

[12] G. de C. Bertoli, L. A. P. Junior, A. L. dos Santos, and O. Saotome, “Generalizing intrusion detection for heterogeneous networks: A stacked-unsupervised federated learning approach,” Sep. 2022, doi: 10.1016/j.cose.2023.103106.

[13] M. H. Alwan, Y. I. Hammadi, O. A. Mahmood, A. Muthanna, and A. Koucheryavy, “High Density Sensor Networks Intrusion Detection System for Anomaly Intruders Using the Slime Mould Algorithm,” Electronics (Switzerland), vol. 11, no. 20, Oct. 2022, doi: 10.3390/electronics11203332.

[14] S. Ismail, Z. El Mrabet, and H. Reza, “An Ensemble-Based Machine Learning Approach for Cyber-Attacks Detection in Wireless Sensor Networks,” Applied Sciences (Switzerland), vol. 13, no. 1, Jan. 2023, doi: 10.3390/app13010030.

[15] M. A. Talukder et al., “A dependable hybrid machine learning model for network intrusion detection,” Journal of Information Security and Applications, vol. 72, Feb. 2023, doi: 10.1016/j.jisa.2022.103405.

[16] M. Ghurab, G. Gaphari, F. Alshami, R. Alshamy, and S. Othman, “A Detailed Analysis of Benchmark Datasets for Network Intrusion Detection System,” Asian Journal of Research in Computer Science, pp. 14–33, Apr. 2021, doi: 10.9734/ajrcos/2021/v7i430185.

[17] B. A. Tama, M. Comuzzi, and K. H. Rhee, “TSE-IDS: A Two-Stage Classifier Ensemble for Intelligent Anomaly-Based Intrusion Detection System,” IEEE Access, vol. 7, pp. 94497–94507, 2019, doi: 10.1109/ACCESS.2019.2928048.

[18] G. Vilas Rasane and S. P. Rathod Student, “Engineering and Technology (A High Impact Factor,” International Journal of Innovative Research in Science, vol. 9, 2020, doi: 10.15680/IJIRSET.2020.0903009.

[19] K. M. Ting and I. H. Witten, “Issues in stacked generalization,” Journal of Artificial Intelligence Research, vol. 10, pp. 271–289, 1999, doi: 10.1613/jair.594.

[20] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

[21] X. Tan et al., “Wireless sensor networks intrusion detection based on SMOTE and the random forest algorithm,” Sensors (Switzerland), vol. 19, no. 1, Jan. 2019, doi: 10.3390/s19010203.

[22] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, “Optuna: A Next-generation Hyperparameter Optimization Framework,” in Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, Jul. 2019, pp. 2623–2631. doi: 10.1145/3292500.3330701.

[23] R. Alshamy and M. A. Akcayol, “INTRUSION DETECTION MODEL USING MACHINE LEARNING ALGORITHMS ON NSL-KDD DATASET,” International Journal of Computer Networks and Communications, vol. 16, no. 6, pp. 75–88, Nov. 2024, doi: 10.5121/ijcnc.2024.16605.

[24] B. Hui and K. L. Chiew, “An Improved Network Intrusion Detection Method Based On CNN-LSTM-SA,” Journal of Advanced Research in Applied Sciences and Engineering Technology, vol. 44, no. 1, pp. 225–238, Feb. 2025, doi: 10.37934/araset.44.1.225238.

[25] D. Dalgade, N. Patil, M. Joshi, and D. Hiran, “Enhancing WSN Intrusion Detection: Two-Tier Feature Selection and Optuna- Optimized Ensemble Learning,” May 26, 2025. doi: 10.21203/rs.3.rs-6726618/v1.