Scientific Journal of Computer Science
futuristech PT. Teknologi Futuristik Indonesia
Date of publication March 3, 2026, date of current version March 3, 2026.
Digital Object Identifier 10.64539/sjcs.v2i1.2026.378

Article

Breaking Class Imbalance Barriers in Intrusion Detection Systems: A Clustering-Based Hybrid Framework

Moshood Abiola Hambali[1,*], Nahum Zhema Bako[2], Mu'awuya Dalhatu[2], Ashraf Ishaq[2]

[1] Department of Software Engineering, Faculty of Computing and Information Systems, Federal University Wukari, PMB 1020, Wukari, Nigeria.; e-mail: hambali@fuwukari.edu.ng
[2] Department of Computer Science, Faculty of Computing and Information Systems, Federal University Wukari, PMB 1020, Wukari, Nigeria.; e-mail: nahumbako002@gmail.com; dalhatu@fuwukari.edu.ng; ishaqashraf@fuwukari.edu.ng
* Correspondence

The authors received no financial support for the research, authorship, and/or publication of this article.

Abstract: Intrusion Detection Systems (IDS) deal with issues concerning the ever-escalating level of sophistication observed within cyber threats. Nonetheless, IDS performance is deteriorated by class imbalance and excessively high-dimensional features, which cause biased classifier training towards major traffic patterns. Thus, this research introduces an innovative hybrid clustering IDS approach that utilizes MiniBatchKMeans clustering and ensemble machine learning strategies to mitigate these challenges. The suggested IDS approach utilizes the Synthetic Minority Over-sampling Technique for addressing class imbalance problems, Fast Correlation-Based Filter for reducing high-dimensional features, and Hyperopt Tree-structured Parzen Estimator for optimizing clustering and machine classifiers' parameters. Four supervised machine classifiers — Decision Tree classifier, Random Forest classifier, Extra Trees classifier, and XGBoost classifier— were trained and validated on the NSL-KDD IDS dataset. Additionally, experimental analysis indicated a superior detection accuracy for all classifiers, for which the best-optimized XGBoost classifier and best-optimized Random Forest classifier provided 99.57% and 99.51% accuracy, respectively. The proposed clustering-optimized machine IDS approach provided substantial improvements for identifying minority class attacks, along with sustainability and high generalization capabilities. The obtained outcomes support the research premise concerning the efficacy of cluster-aware sampling and ensemble optimizations for designing more balanced, accurate, and adaptive IDS systems for effectively protecting against ever-escalating real-life threats within the cyberworld.

Keywords: IDS, MiniBatchKmeans, Data Imbalance, Cyber-attack, Hyperopt, SMOTE.

1. Introduction

Cyber-attacks come off as major threats to the security of both individuals and organizations, given the increasing level of interconnectedness associated with the online world. An Intrusion Detection Systems (IDS) play the important role of a cornerstone defense system, detecting unusual patterns associated with activities observed on networks. However, the high level of complex patterns associated with high network data, specifically its high dimensionality and dynamic patterns associated with persistent threats, creates important challenges for typical IDS models to effectively design new approaches for the detection of unusual patterns [1]-[3].

Techniques utilizing machine learning algorithms have shown enormous benefits in the enhancement of IDS performance through the ability to identify prognostic patterns from labeled databases [4], [5]. However, these approaches can be associated with high imbalance in their classes, in which attack instances are masked by a large number of normal traffic instances, such that biased classifiers are produced, as well as a reduced capacity to detect rare attacks [6], [7]. Moreover, high-dimensional spaces are less efficient with high chances of overfitting problems in which feature selection/dimensionality reduction techniques are required to enhance performance [8]-[10]. Various combination approaches like bagging,

boosting, random forest algorithms, gradient boosting algorithms, stacked deep networks simultaneously employ many base classifiers that combine their strengths in making a more accurate prediction as well as addressing the problem brought about by high imbalance in classes through a combination of Misclassification Errors Weighted, bootstrap sampling approaches [11], [12]. However, despite their enormous benefits, these approaches still may end up completely neglecting the systemic structure pattern in the attack data.

One of the most essential questions in IDS study is how best to integrate clustering approaches with expert classifiers. This integration largely focuses on identifying highly imbalanced minority attack classes in the study, as shown by recent research.

Yin et al. [2] proposed the use of Birch clustering with MLP, considering the cluster label information as supplement features, and achieved an accuracy rate of 99.73% on the CICIDS-2017 dataset, considering the class label information of the clusters as supplement features. Farooqi et al. [1] proposed an ensemble voting classifier that integrates Decision Tree, Random Forest, and XGBoost (Extreme Gradient Boosting) classifiers, coupled with SMOTE, and demonstrated accuracy above 99.9% on NSL-KDD, UNSW-NB15, and CIC-IDS databases. Nassreddine et al. [4] combined the concept of correlation-based and embedded FS, coupled with classifier stacking of RF, Decision Tree, and XGBoost, and closely approached the perfect accuracy level of ~99.9%. Zhu and Liu [13] presented the design of an efficient framework that applies the concepts of subspace clustering and ensemble learning to the Internet of Things (IoT) intrusion detection, demonstrating excellent gains over the benchmark classifiers. In addition, the use of ADASYN oversampling, an SMOTE variant, with RF on the CICIDS databases by Chen et al. [14] resulted in a remarkable enhancement of dataset recall and F1-score values, particularly for the less frequent events. These works indicate that the integration of ML-ensemble, or other specific classifiers, with some proper oversampling and FS, outperforms the other ML-approaches to some level, but none of them explored the concept of biased classifier designs on the individual clusters, as done in the proposed model presented.

There is a limited body of IDS research that employs both oversampling and ensemble methods to handle class imbalance. Le et al. [15] observe that while "various methods" (resampling methods, ensemble methods, etc.) have been employed to deal with imbalanced IDS data, "there exists a notable gap in the literature for effective hybrid methodologies." Some recent studies report very high accuracy on the NSL-KDD data set. For example, Farooqi et al. (2024) [1] report 99.88% accuracy using a DRX Voting Ensemble (Decision Tree, Random Forest, XGBoost). Similarly, Nassreddine et al. [16] report ~99.99% accuracy using the SMOTE Tomek Resampling method. However, none of these studies that report very high accuracy use a clustering step in the process. A small number of studies have also, utilized clustering before a classification model is applied. For example, [17] introduced a new approach called KCLSTM, which is a hybrid approach that makes use of a CNN-LSTM model for detection on the NSL KDD dataset. A similar approach is utilized by [18], who employed a K-means model before a classification model using the XGBoost algorithm on the NSL KDD dataset. However, this approach has been utilized to increase overall accuracy but has yet to be utilized with sampling methods for each individual cluster. Most importantly, the minority classes of attacks (R2L and U2R on the NSL KDD dataset, classes 3 and 4) have yet to be addressed. Joshi et al. (2025) noted that even though advanced models have been utilized on the dataset, they have yet to be successful in detecting the minority classes. "CNN-BiLSTM often misclassified Classes 3 and 4," they noted. A similar result was found with the XGBoost model. Overall, while various models have been utilized on the dataset, none have employed a combination of models to increase detection accuracy on the minority classes.

In this study, we present a novel hybrid model for Intrusion Detection Systems that uses MiniBatchKMeans clustering and a separate ensemble model (e.g., XGBoost, Random Forest) per cluster. Although K-means has been explored in Intrusion Detection Systems (e.g., KCLSTM), our contribution is combining clustering and ensemble learning in a single model. The ensemble model learns from a more homogeneous subset of data within each cluster and uses a specialized decision boundary. Fast Correlation-Based Filter (FCBF) feature selection approach within each cluster. The FCBF filter starts with a full feature set and uses a symmetrical uncertainty criterion (information-theoretic correlation measure) to filter out irrelevant and redundant features. The process continues iteratively until a reduced subset of relevant features is obtained. The reduced feature subset contains a small number of relevant features that are useful for each cluster. This improves each ensemble model by removing noise and speeding up model training. After feature filtering, we balance the class distribution for each cluster by using SMOTE on the minority classes in the cluster. This is like creating new instances of infrequent intrusions in a cluster. This is important to ensure that infrequent attacks have enough training instances in a cluster. SMOTE has been shown to be highly successful; for example, SMOTE Tomek Re-Sampling with the NSL KDD dataset has an accuracy rate of almost 99.99%. Our approach is more accurate because we use SMOTE for each cluster. Also, we use the Hyperopt library's "Tree-

structured Parzen Estimator" (TPE) to optimize the hyperparameters for each cluster's model. This is important to ensure that the model is tuned to the best possible parameters for the distribution in the cluster. This is important to ensure that we can detect infrequent attacks in a cluster. By tuning the hyperparameters for each model in a cluster, we ensure that we can detect infrequent attacks in a cluster.

This paper contributes to the body of knowledge: 1) Improving rare class detection: Our first goal is to significantly improve IDS detection rates for the least represented classes in the NSL-KDD data set (such as Class 4: U2R attacks). Regular global models perform poorly in this regard (Classes 3/4 are "often misclassified" according to Joshi et al.), so we specifically address this problem. Our goal is to significantly improve recall/F1 for R2L/U2R attacks. 2) Cluster-aware training: Our model benefits from the homogeneous nature of the data in the clusters created by MiniBatchKMeans. Each model will be trained on a more specific set of data, allowing it to specialize in the detection of anomalies in that set. For example, if one of the clusters is comprised mainly of one type of attack, the model will be able to specialize in that type of attack. 3) Balancing and reduction: In each cluster, SMOTE will create additional examples of rare classes, and FCBF will remove irrelevant features. The goal of these operations is to enhance the detection of rare intrusion classes by providing sufficient positive examples of each rare class and eliminating irrelevant features to help each ensemble model better understand the nature of rare intrusion classes.

The rest of this paper is organized as follows: Section 2 presents an overview of existing work on clustering-based IDS systems, feature selection, and imbalanced datasets. Section 3 presents an overview of our methodology for clustering, biased classifier design, feature development, and optimization. The rest of this paper is organized as follows: Section 4 presents an overview of our dataset, experiment setup, and evaluation metrics, and presents an overview of our research process and comparison to existing approaches. Finally, we provide an overview of conclusions and future work in Section 5 and Section 6.

2. Literature Review

2.1. Clustering-Based Intrusion Detection Using K-means

Clustering algorithms have emerged as significant components in anomaly-related intrusion detection systems, and it has been identified that among the widely used algorithms is the K-means algorithm due to its simplicity and efficiency [19]. Many researchers have introduced modifications of the algorithm in an attempt to increase its precision and mitigate the problem of producing false alarms.

A new K-means algorithm for anomaly detection with adaptive determination of optimal clusters based on the Calinski-Harabasz index was presented by Kherbache et al. [20]. This showed enhanced efficiency for detection as well as processing speed when compared to traditional K-means and Support Vector Machine classifiers on NSL-KDD and CICIDS2017 datasets. A collaborative approach using PCA with K-means for dimensionality reduction with enhanced clustering accuracy was presented by Chapagain et al. [19]. Their approach showed enhanced detection capabilities, a reduction in the rate of FP, as well as improvement in processing speed.

Later improvements in clustering algorithm approaches were suggested in the work of Hu et al. [21], who focused on developing an approach called multiple kernel clustering (MKC) to meet the challenges of having missing attributes in the Internet of Things (IoT) domain. The approach focused on similarity calculations and not rough imputation methods, thereby providing better accuracy in the detection of anomalies in incomplete datasets. On the other hand, Samunnisa et al. in [22] designed a hybrid approach to combine clustering and classification techniques, also utilizing the RF classification algorithm. The anomaly detection approach based on threshold values was very accurate, with up to 99.85% on the NSL KDD and KDD Cup 99 datasets.

Aamir et al. [23] worked in the area of semi-supervised clustering, where the unlabeled network traffic data was classified using PCA-based K-means and Agglomerative algorithms. Finally, the labeled groups of the data were classified using classification algorithms like k-Nearest Neighbors (k-NN), Support Vector Machine (SVM), and Random Forest classification. This end-to-end solution showed remarkable accuracy of up to 96.66% and proved robustness against various forms of attacks. Talking about the benchmarking analysis, Maseer et al. [24] carried out a comparative study of ten supervised and unsupervised algorithms like K-means, Self-Organizing Map (SOM), and Expectation Maximization (EM) on the CICIDS2017 dataset. They critically analyzed the shortfalls of past studies, where the research contained shallow analysis and outdated data issues. They concluded that K-means remained a competitor in the unsupervised scheme of things but marked the need for parameter adjustments and in-depth fusion.

Younisse & Al-Haija [25] concentrated on the use of real-time K-means, specifically an online version for intrusion detection on the web. The results showed that the online clustering algorithm attained the same level of clustering purity as the offline algorithm when used for normalized IoT traffic, which attained values above 93% and F1 values of 94%.

Chormale et al. [26] focused on data streams that evoke high velocity and volume, testing the efficiency of

clustering-based Intrusion Detection Systems (IDSs) in a difficult corporate environment. Their preliminary work emphasized the flexibility and scalability of clustering approaches in a setting that involves the surveillance of packets in real-time.

The imbalance problem of datasets was overcome by Wang et al. [27], who developed a multicriteria GANs-assisted clustering-based Intrusion Detection System. The generated synthetic data was used for speeding up the learning of clusters by employing fuzzy C-means and CNN-LSTM deep CNN architectures. The obtained results that were proven to be correct by employing the dataset UNSW-NB15 had good generalizability and applicability to any type of attack scenario.

In general, all these studies exhibit the scenario of intrusion detection via K-means clustering. In particular, since traditional K-means clustering is affected by several drawbacks, such as sensitivity to initialization, as well as the requirement that the number of clusters must be predefined, all these drawbacks have been alleviated by hybridization (PCA, classifiers), by adaptive clustering (Multiple Kernel Clustering, Weighted Fuzzy C-Means), as well as by online machine learning. Benchmark datasets like NSL-KDD and CICIDS2017 remain mostly used, but new approaches tend towards more difficult tasks like unbalanced datasets, real-time systems, and missing attributes.

2.2 Biased classifier ensemble for network intrusion detection

It has been observed that recent works have demonstrated that there is potential to improve the performance capabilities of intrusion detection systems (IDS) by ensemble-based strategies, especially with regard to the existing imbalance class issue in network traffic datasets. The contribution by Thakkar and Lohiya [28] in this regard was through the development of a bagging-based deep neural network (DNN) model that assigns more weight to the minority classes during training, thus providing better accuracy and F1-value performance capabilities to four prominent benchmark datasets. Related to this, but with equal importance, was the contribution by Ren et al. [29] with the Dynamic Under-sampling Ensemble Network (DUEN) model, which uses boost-based strategies that utilize dynamic under-sampling, giving preference to boundary regions, thus resolving imbalances and noisy issues, thereby enabling efficient identification or detection of rare classes associated with attack traffic.

After this, with respect to strategies related to IoT, Musthafa et al. [5] have made contributions through balancing classes, along with feature selection represented by ANOVA, followed by ensemble strategies involving Support Vector Machines and LSTM networks, thus enabling high detection accuracy with low false alarms.

Apart from those approaches that specifically deal with tackling imbalance, other researchers have explored ensemble architectures that specifically aim to maximize overall efficiency. These works include the comparative study by Hossain and Islam [30] involving different ensemble models like Random Forest, Gradient Boosting Machine, AdaBoost, and XGBoost, whose results consistently showed that Random Forest delivered the best and most accurate results. The study by Thockchom et al. [31] adopted a light-weight base learner method composed of Naïve Bayes, Logistic Regression, and Decision Tree classifiers pertaining to the classification task, with an additional Stochastic Gradient Descent meta-classifier in favor of more efficacious generalization. In terms of large-scale intrusion detection in big data, Jemili et al. [32] proposed Random Forest integrated with XGBoost, which achieved an optimum accuracy rate of 97%. On the subject of optimization method-based approaches, Sarkar et al. [33] proposed the use of cascaded meta-specialist classifiers with the integration of data generation methods specifically pertaining to less common categories R2L & U2R, which performed more accurately in their detection. Last but not least, the utilization of evolutionary computation was explored, as presented in the study by Akhtar et al. [34], who established an efficient genetic ensemble method that was compared with common ensemble methods such as Voting, Bagging, Gradient Boosting Machine (GBM), and Bagger based on Random Forest, achieving impressive reduction in mean squared error & mean absolute error.

Taken collectively, these papers serve as a pointer to how ensemble methods can be very useful in improving the efficacy of an IDS, particularly in dealing with imbalances and heterogeneities in networks, without emphasizing significantly how clustering can be utilized as a preprocessing step in improving further the quality of data prior to classification.

3. Proposed Method

This paper proposes a hybrid solution for intrusion detection by integrating clustering-based sampling with supervised machine learning classifier models. The overall process includes data preprocessing, balancing classes, feature selection, clustering, training, and testing. The overall process is depicted in Figure 1.

3.1. Data Acquisition and Preprocessing

The NSL-KDD intrusion detection benchmark dataset, which removes redundancy issues related to the original KDD'99 data set, was utilized in conducting experiments [35]. The data was read from the CSV file, and inconsistencies in it were checked. The categorical variables were encoded as numerical using Label Encoding to make them compatible with machine learning models.

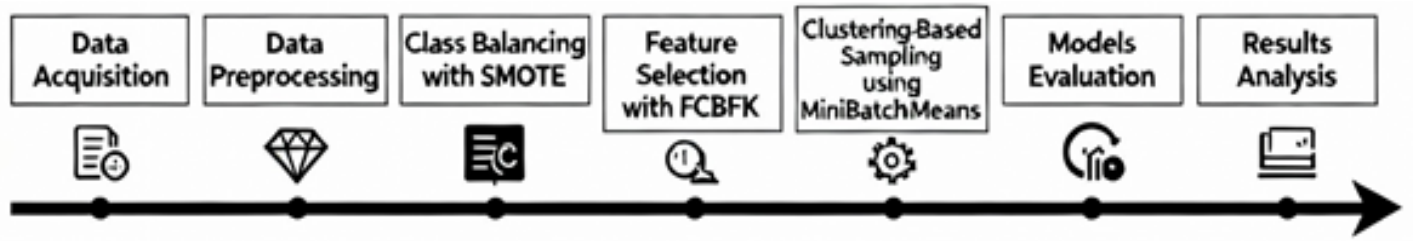

Figure 1. Workflow of the Proposed Hybrid Intrusion Detection System Based on Clustering and Supervised Learning.

Each record in the NSL-KDD dataset represents a single TCP/IP connection with 41 attributes (TCP features, content features, traffic features based on time and host information) and one label. Among the attributes, there are three nominal attributes (protocol, service, flag), some binary attributes (logged_in, is_guest_login), while the rest of the attributes are numeric. There are five classes in the label set: one class for normal instances and four classes for different kinds of attack instances: Denial of Service (DoS), Probe, Remote-to-Local (R2L), and User-to-Root (U2R). There are about 148,500 instances in the NSL-KDD dataset in total: 125,972 instances in the training set and 22,543 instances in the testing set. There are 39 kinds of attack instances in total: 22 kinds in the training set and 37 kinds in the testing set. Some examples of the most common attack instances include smurf and neptune for the DoS class, port sweep and satan for the Probe class, guess_passwd for the R2L class, while the U2R class includes instances of buffer_overflow. The class distribution is imbalanced in the NSL-KDD dataset. For example, the percentage of the Denial of Service class is much higher than the other classes in the attack instances.

3.2. Class Balancing

Owing to the imbalance present in intrusion detection datasets, the Synthetic Minority Oversampling Technique (SMOTE) was used on the training dataset to artificially create samples from minority classes. This allowed the models to see an equal distribution of attack and normal traffic samples.

3.3. Feature Selection

In this research, the Fast Correlation-Based Filter (FCBFK) algorithm is used as a dimensional reduction algorithm in order to improve the performance of the proposed IDS. This is undertaken in a bid to improve computational complexity by removing any redundant attributes.

The method relies on the Symmetrical Uncertainty (SU) measure, which evaluates the correlation between features and the target class. It is defined as:

$$SU(X,Y) = 2 \times \frac{IG(X \mid Y)}{H(X) + H(Y)} \quad (1)$$

where:

- $IG(X \mid Y) = H(X) - H(X \mid Y)$ is the information gain
- $H(X)$ Is the entropy of attribute (H)
- $IG(X \mid Y)$ is the conditional entropy of X given X

The SU values range between 0 and 1, with higher values indicating stronger associations with the class label.

Procedure:

- SU Computation: For each feature in the dataset, the SU score with respect to the class label was computed.
- Relevance Filtering: Features whose SU values exceeded the predefined threshold δ were selected as relevant.
- Ranking: Based on SU values, the chosen features were arranged in descending order.
- Redundancy Elimination: for each feature F_i, the SU score with other features F_j was calculated. If

$$SU(F_i, F_j) \geq SU(F_j, C) \quad (2)$$

then F_j was considered redundant and discarded.

- Final Subject: The process continued until only the most relevant and non-redundant features were retained.

The NSL-KDD database was then employed to reduce the first 41 features by using the Fast Correlation-Based Filter with k-selection, named as FCBFK. The process was done by ranking the features through symmetric uncertainty, where the top k(attributes) with high relevance to intrusion classes were chosen.

The final set of selected subsets contains variables like duration, protocol_type, src_bytes, dst_bytes, flag, count, srv_count, serror_rate, rerror_rate, logged_in, hot, num_access_files, and srv_rerror_rate. These variables encompass behavior from network traffic volumes and error rates to user or system-level activities like user logins and access to system files. For example, the two features, src_bytes and dst_bytes, are significant in identifying Denial-of-Service (DoS) attacks from normal sessions, while logged_in and num_access_files are significant in identifying Remote-to-Local (R2L) and User-to-

Algorithm 1. FCBFK–MiniBatchKMeans-Based Feature Selection.

Input:

NSL-KDD dataset D = $\{(\boldsymbol{X_i}, \boldsymbol{y_i})\}_{i=1}^{n}$
Feature set F = {F_1,F_2,...,F_{41}}
Class labels C∈{DoS, Probe, R2L, U2R, Normal}
SU threshold δ
Number of selected features k
Number of clusters K_c
Sampling ratio α
Train-test split ratio r = 0.8

Output:

Reduced feature subset F^*
Balanced training dataset S
Test dataset D_{test}

Steps:

1. Compute SU(F_i, *C) for all 41 features*
2. Select relevant features Fr where SU ≥ δ
3. Rank Fr in descending order of SU
4. Initialize F = {}*
5. For each Fi in Fr:
  *Add Fi to F**
  For each Fj after Fi:
    If SU(Fi, Fj) ≥ *SU(Fj, C):*
      Remove Fj
6. Select Top-k features → *F**
*7. Reduce dataset D using F**
8. Split D into D_train (80%) and D_test (20%)

9. Apply MiniBatchKMeans on D_train → *clusters C1...CKc*
10. For each cluster Cj:
  Sj = α × |*Cj*|
  Randomly sample Sj instances
11. Combine all Sj → *S*

Return F, S, D_test*
End

Root (U2R) attacks. Likewise, the two features, count and srv_count, are significant in identifying connection rate totals characteristic of Probe attacks. Reducing the features to the most significant ones resulted in a reduction of the data dimensions with consequent improvements in computational efficiency.

3.4. Clustering-Based Sampling

To counter class imbalances and to avoid domination by majority classes such as DoS during model training, there was a clustering-based sampling strategy. This was achieved by using MiniBatchKMeans to split the data into k homogenous clusters:

$$X = \{x_1, x_2, \dots, x_n\} \xrightarrow{\text{MiniBatchKMeans}} C = \{C_1, C_2, \dots, C_k\} \quad (3)$$

where X denotes the training samples and C_j represents the j^{th} cluster.

From each cluster, a proportional number of samples was selected:

$$S_j = \alpha \cdot |C_j| \quad (4)$$

where α is the sampling ratio and C_j is the cluster size.

This ensures that each cluster contributes a balanced subset of instances to the final training set:

$$S = \bigcup_{j=1}^{k} S_j \quad (5)$$

Because the sampling performed here is proportionally done from all the clusters, the diversity has been preserved, and there is a reduction in the over-representation of frequent patterns. It gives a view of relatively balanced sample distributions across attack categories: DoS, Probe, U2R, R2L, and Normal, which enhances the generalization capability of the classifiers that are downstream. The algorithm for feature selection is summarized in Algorithm 1.

Next, the data was divided into training and test subsets at a ratio of 80:20 using the train_test_split function provided by scikit-learn.

3.5. Model Training

Four different supervised learning algorithms were utilized in order to evaluate the detection capability of the developed hybrid model for intrusion detection. These algorithms are the Decision Tree (DT) Classifier, Random Forest (RF) Classifier, which also uses a hyperparameter-optimized model for improved detection capability, achieved through Hyperopt hyperparameter optimization with a Tree-structured Parzen Estimator (TPE), Extra Trees (ET) Classifier, and a hyperparameter-optimized Extreme Gradient Boosting model, also achieved through hyperparameter optimization, namely, XGBoost Classifier. The respective models were developed on the preprocessed training data sets, and their performances were also evaluated on a separate, unknown test data set.

3.6. Performance Evaluation

Various performance metrics were also used to evaluate the performance of the models. These include Accuracy, Precision, Recall, as well as the F1 score. Confusion matrices are also used to display errors in the classification process. Also, in a classification problem like this, calculations for ROC AUC score are done. Every performance metric was measured for comparison purposes. The criterion for each of the measures is determined based on four basic criteria: True Positive (TP),

Table 1. Hyperparameters of the Classifiers for Baseline.

Algorithm	Key Parameters
Decision Tree (DT)	criterion = "gini", max_depth = None, random_state = 42
Random Forest (RF)	n_estimators = 100, criterion = "gini", max_depth = None, random_state = 42
Extra Trees (ET)	n_estimators = 100, criterion = "gini", max_depth = None, random_state = 42
XGBoost (XGB)	n_estimators = 100, learning_rate = 0.1, max_depth = 6, objective = "multi:softmax", eval_metric = "mlogloss", random_state = 42

True Negative (TN), False Positive (FP), and False Negative (FN).

1) True Positive (TP): Indicates whether the model correctly identified an atack when there was indeed an attacked.
2) False Negative (FN): Occurs when the model incorrectly labels an instance record as not an attack, despite it actually being an attack.
3) False Positive (FP): Indicates instances where the model incorrectly identifies an attack when there is actually no attack present.
4) True Negative (TN): Indicates when the model correctly identifies that there is no attack and there is indeed no attack.

Precision: is a quantitative measure that specifically evaluates the accuracy of identifying positive instances, by determining the proportion of anticipated positive attack that are actually true. The question it addresses is: "What is the proportion of true positive predictions among all the predicted positive instances?" Equation 6 represents the mathematical formula for the precision metrics.

$$precision = \frac{TP}{TP + FP} \tag{6}$$

Recall: Recall, sometimes referred to as sensitivity or true positive rate, measures the accuracy of predicting actual attack records. It provides the answer to the question: "Out of all the true positive cases, how many were accurately identified as positive?" The formula is displayed in equation 7:

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

F-measure (or F-score): The F1 score is calculated as the harmonic mean of precision and recall. The evaluation of a classification model is conducted in a balanced manner, considering both false positives and false negatives. It is especially advantageous when working with datasets that have an unequal distribution of data. Equation 8 presents the mathematical expression for the F-score.

$$F - Measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{8}$$

Accuracy: Accuracy is a widely used measure to assess the overall effectiveness of a classification model. Equation 9 demonstrates how the accuracy is calculated by determining the ratio of correctly classified cases (including true positives and true negatives) to the total number of instances:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

4. Experimental Results

The proposed approach was implemented in Python 3.9 using the scikit-learn, imbalanced-learn, and XGBoost libraries, and experiments were conducted on a workstation with an Intel® Core™ i7 processor, 16 GB RAM, and Windows 10 OS. Using the NSL-KDD dataset, split into 80% training and 20% testing sets, SMOTE oversampling, FCBF feature selection, and clustering-based sampling were applied before classifier training. Four models — XGBoost, Random Forest (RF), Decision Tree (DT), and Extra Trees (ET),— were evaluated on 5,360 samples spanning seven target classes, with performance measured using accuracy, precision, recall, and F1-score. Hyperparameter tuning was carried out with Hyperopt, and the baseline configurations for each classifier are summarized in Table 1.

4.1. XGBoost Classifier

The XGBoost model was evaluated for intrusion detection using both the baseline configuration and a hyperparameter-optimized configuration obtained through Hyperopt. The optimization process selected a learning rate of 0.7340, 70 estimators, and a maximum depth of 14 as the best-performing parameters.

Table 2 presents the comparative performance of the baseline and optimized models. The baseline XGBoost-model achieved an accuracy of 99.18%, precision of 0.9918, recall of 0.9918, and an F1-score of 0.9918. Class-wise evaluation revealed strong performance across most attack and normal traffic categories, although Class 4 (a minority class with only 7 instances) exhibited a reduced recall of 0.71, indicating occasional misclassifications.

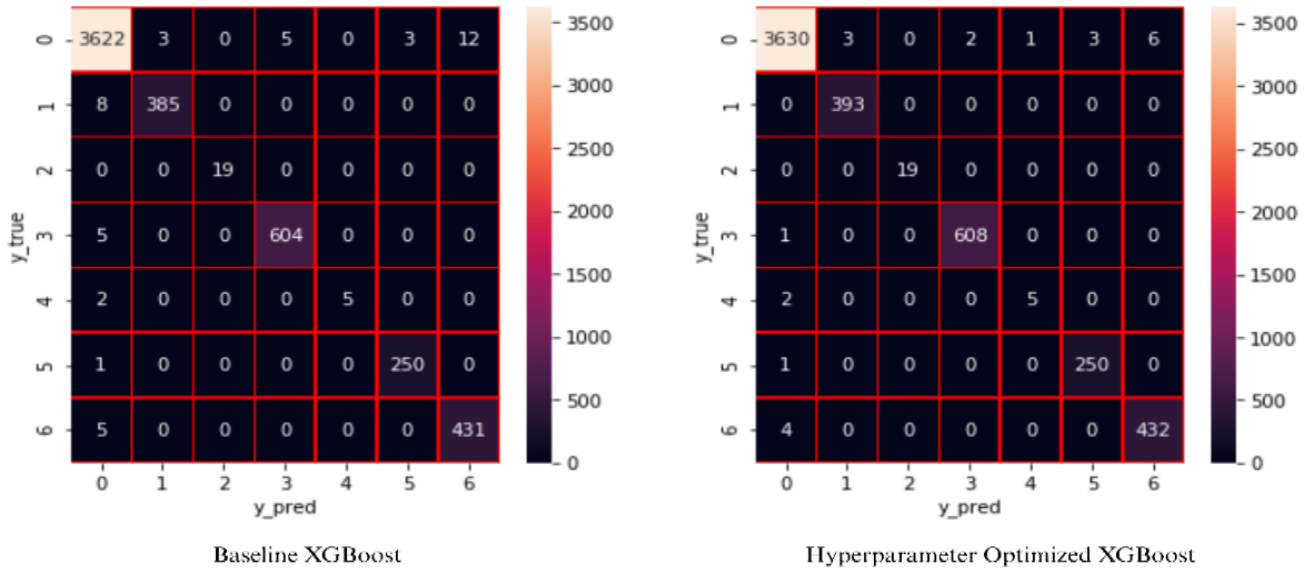

Figure 2. Confusion Matrices for Baseline (left) and Hyperparameter-Optimized (right) XGBoost Models in Intrusion Detection.

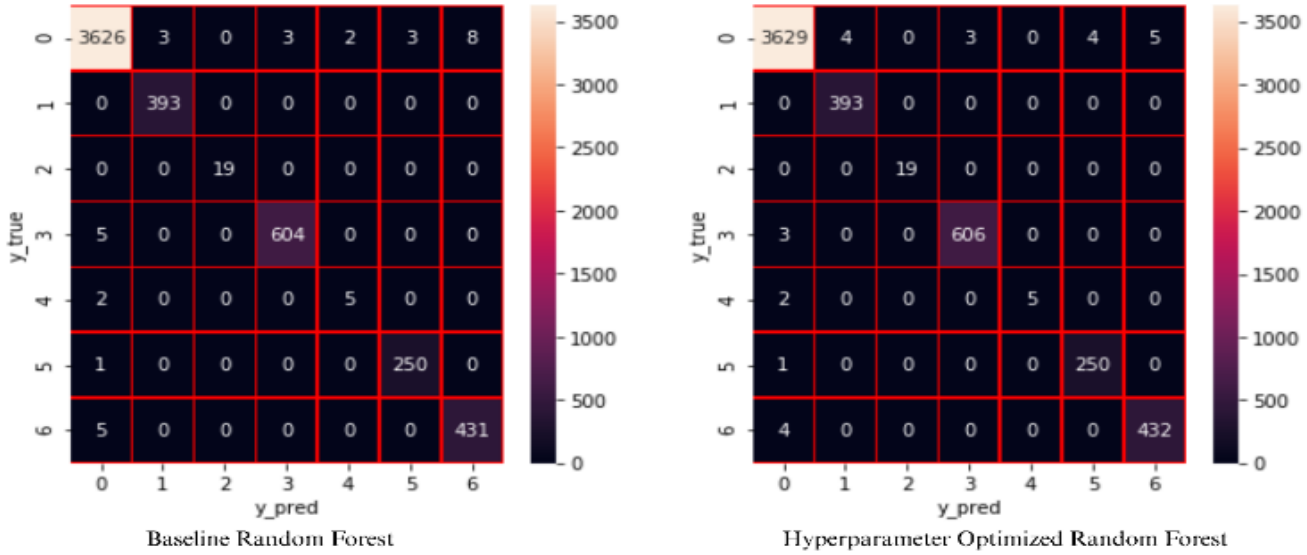

Figure 3. Confusion Matrices for Baseline (left) and Hyperparameter-Optimized (right) Random Forest Models in Intrusion Detection.

Table 2. Comparative Performance of Baseline and Hyperparameter-Optimized XGBoost for Intrusion Detection.

Configuration	Accuracy	Precision	Recall	F1-score
Baseline	0.9918	0.9918	0.9918	0.9918
Hyperparameter Optimized	0.9957	0.9957	0.9957	0.9957

Table 3. Comparative Performance of Baseline and Hyperparameter-Optimized Random Forest for Intrusion Detection.

Configuration	Accuracy	Precision	Recall	F1-score
Baseline	0.994	0.994	0.994	0.994
Hyperparameter Optimized	0.9951	0.9952	0.9951	0.9951

Following hyperparameter optimization, the XGBoost model achieved improved performance, with an accuracy of 99.57%, precision of 0.9957, recall of 0.9957, and an F1-score of 0.9957. While the recall for Class 4 remained at 0.71, all other classes either maintained or slightly improved in detection performance. Notably, minority classes such as Class 2 (19 samples) and Class 5 (251 samples) achieved perfect or near-perfect detection rates.

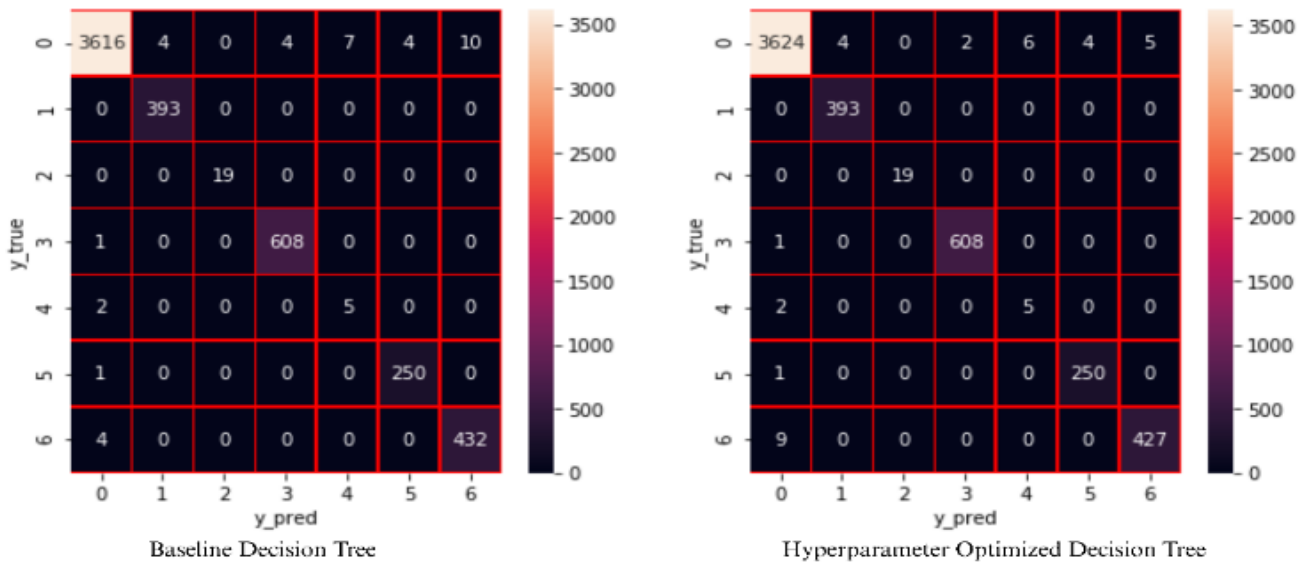

Figure 4. Confusion Matrices for Baseline (left) and Hyperparameter-Optimized (right) Decision Tree Models in Intrusion Detection.

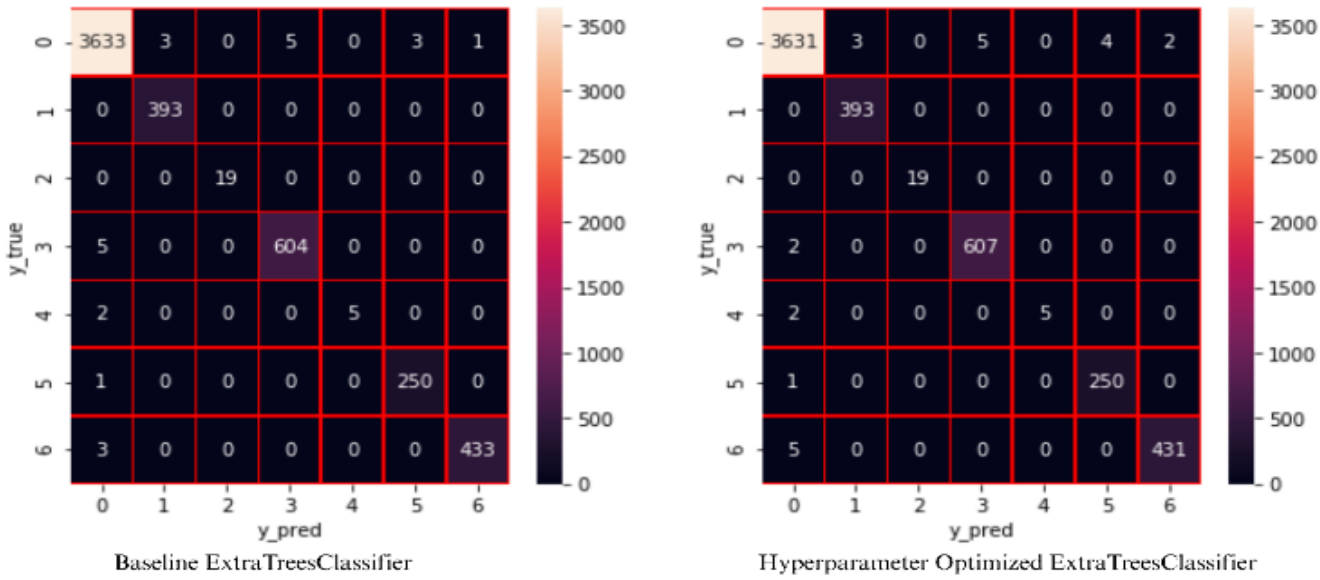

Figure 5. Confusion Matrices for Baseline (left) and Hyperparameter-Optimized (right) ExtraTreesClassifier Models in Intrusion Detection.

Table 4. Comparative Performance of Baseline and Hyperparameter-Optimized Decision Tree for Intrusion Detection.

Configuration	Accuracy	Precision	Recall	F1-score
Baseline	0.9931	0.9937	0.9931	0.9933
Hyperparameter Optimized	0.9937	0.9941	0.9937	0.9938

Table 5. Comparative Performance of Baseline and Hyperparameter-Optimized ExtraTreesClassifier for Intrusion Detection.

Configuration	Accuracy	Precision	Recall	F1-score
Baseline	0.9957	0.9957	0.9957	0.9957
Hyperparameter Optimized	0.9955	0.9955	0.9955	0.9955

The confusion matrices (Figure 2) illustrate the distribution of correct and incorrect predictions for the baseline and optimized models, respectively. In both cases, the majority of errors occurred in minority classes, underscoring the inherent difficulty of detecting rare intrusion types. Nevertheless, the optimized model reduced false negatives in several classes, contributing to the observed performance gain.

These results confirm that hyperparameter tuning offers measurable benefits for XGBoost in intrusion de-

tection, particularly in reducing misclassifications for less frequent attack types. However, extreme class imbalance continues to challenge the complete elimination of false negatives in rare categories.

4.2. Random Forest Classifier Performance

The Random Forest (RF) model was also evaluated for intrusion detection using both the baseline configuration and a hyperparameter-optimized configuration obtained through Hyperopt. The optimization process selected n_estimators = 71, min_samples_leaf = 1, max_depth = 46, min_samples_split = 9, max_features = 20, and criterion = 1 (entropy) as the best-performing parameters.

Table 3 presents the comparative performance of the baseline and optimized models. The baseline Random Forest model achieved an accuracy of 99.40%, precision of 0.9940, recall of 0.9940, and an F1-score of 0.9940. Class-wise evaluation revealed strong performance across most attack and normal traffic categories, although Class 4 (a minority class with only 7 instances) exhibited a reduced recall of 0.71, indicating occasional misclassifications.

Following hyperparameter optimization, the Random Forest model achieved improved performance, with an accuracy of 99.51%, precision of 0.9952, recall of 0.9951, and an F1-score of 0.9951. While the recall for Class 4 remained at 0.71, all other classes either maintained or slightly improved in detection performance. Minority classes such as Class 2 (19 samples) and Class 5 (251 samples) maintained perfect or near-perfect detection rates.

The confusion matrices (Figure 3) illustrate the distribution of correct and incorrect predictions for the baseline and optimized models, respectively. In both cases, the majority of errors occurred in minority classes, underscoring the inherent difficulty of detecting rare intrusion types. Nevertheless, the optimized model reduced false negatives in several classes, contributing to the observed performance gain.

4.3. Decision Tree Classifier Performance

The Decision Tree (DT) model was evaluated for intrusion detection using both a baseline configuration and a hyperparameter-optimized configuration obtained through Hyperopt. The optimization process selected a minimum samples per leaf of 2, a maximum depth of 47, minimum samples split of 3, a maximum feature count of 19, and the "gini" criterion (criterion = 0) as the best-performing parameters.

Table 4 presents the comparative performance of the baseline and optimized models. The baseline DT model achieved an accuracy of 99.31%, precision of 0.9937, recall of 0.9931, and an F1-score of 0.9933. Class-wise evaluation revealed strong detection rates across most categories, though Class 4 (only 7 instances) exhibited reduced precision (0.42) and recall (0.71), indicating that rare intrusion types remain challenging to classify accurately.

Following hyperparameter optimization, the DT model achieved a slight improvement, with an accuracy of 99.37%, precision of 0.9941, recall of 0.9937, and an F1-score of 0.9938. Class 4's recall remained at 0.71, though its precision improved modestly to 0.45. Other minority classes, such as Class 2 (19 samples) and Class 5 (251 samples), maintained perfect or near-perfect classification rates.

The confusion matrices (Figure 4) depict the distribution of correct and incorrect predictions for the baseline and optimized models, respectively. As with other classifiers, the majority of misclassifications occurred within minority classes, reflecting the impact of severe class imbalance. While the optimization yielded only incremental overall performance gains, it did enhance precision in certain underrepresented classes.

4.4. Extra Trees Classifier Performance

The Extra Trees Classifier was assessed for intrusion detection using both a baseline configuration and a hyperparameter-optimized configuration identified via Hyperopt. The optimization process selected 53 estimators, a maximum depth of 31, a minimum of 1 sample per leaf, a minimum split size of 5 samples, 20 maximum features, and the "entropy" criterion as the optimal parameters.

In the baseline configuration, the model achieved an accuracy of 99.57%, precision of 0.9957, recall of 0.9957, and an F1-score of 0.9957. Class-wise evaluation demonstrated near-perfect performance for most categories, including minority classes such as Class 2 (19 samples) and Class 4 (7 samples). However, Class 4 recorded a recall of 0.71, indicating that rare intrusion types remained challenging to detect perfectly.

Following hyperparameter optimization, the model attained an accuracy of 99.55%, precision of 0.9955, recall of 0.9955, and an F1-score of 0.9955 (Table 5). While this configuration slightly reduced the overall accuracy compared to the baseline, performance across most classes remained nearly identical. The recall for Class 4 persisted at 0.71, and all other classes maintained high detection rates.

The confusion matrices (Figure 5) visualize the correct and incorrect predictions for the baseline and optimized models. In both cases, errors were concentrated in minority classes, underscoring the challenge of extreme class imbalance. Nonetheless, the ExtraTreesClassifier consistently delivered strong results, confirming its robustness for intrusion detection.

4.5. Discussion

The experimental results demonstrate that the evaluated classifiers achieved high overall detection perfor-

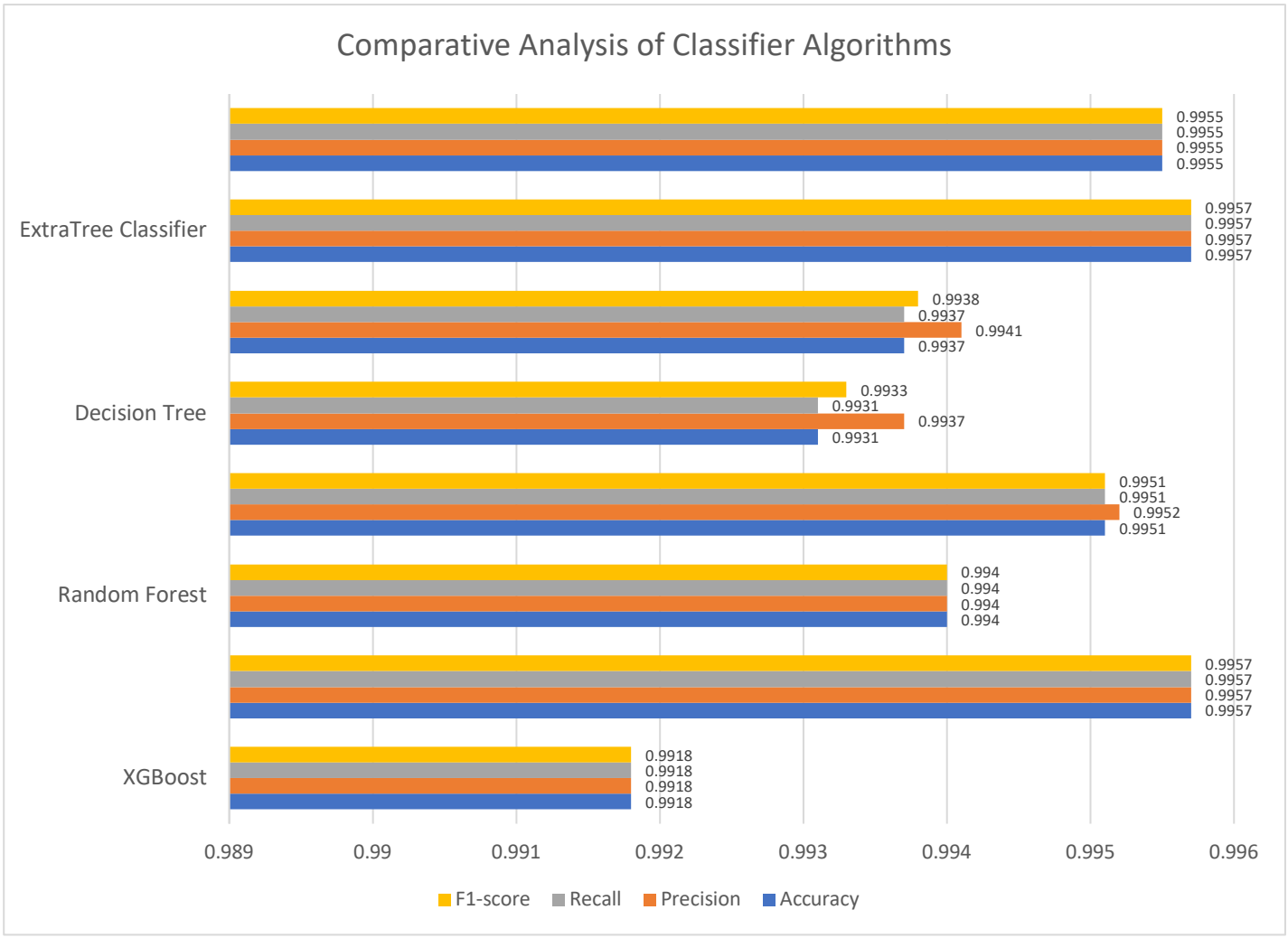

Figure 6. Comparative of Algorithms' Results.

mance for intrusion detection, with accuracies consistently exceeding 99% in both baseline and optimized configurations. The Decision Tree classifier provided strong baseline results but was slightly outperformed by the ensemble-based methods, namely XGBoost, Random Forest, and Extra Trees. These ensemble models benefited from aggregating multiple decision paths, which contributed to improved generalization and slightly higher precision and recall values.

Across all classifiers, misclassifications were concentrated in minority attack categories, particularly Class 4, which contained only seven instances in the test set. Despite near-perfect detection for the majority of classes, such as normal traffic (Class 0) and larger attack categories (Classes 1 and 3), recall for Class 4 remained at 0.71 for most models, indicating the persistent difficulty of detecting rare intrusion types. Minority classes with slightly more samples, such as Class 2 (19 samples) and Class 5 (251 samples), achieved perfect or near-perfect detection in optimized configurations.

Hyperparameter optimization via Hyperopt led to measurable improvements in most classifiers, with XGBoost and Random Forest showing the largest gains. In particular, optimized configurations reduced false negatives in several minority classes, contributing to higher weighted F1-scores. However, the results also highlight that extreme class imbalance continues to be a limiting factor, as even the best-performing models did not fully eliminate errors in rare categories. Overall, the findings confirm that ensemble methods, when properly tuned, provide robust and reliable performance for the majority of intrusion categories in the dataset.

4.6. Comparative Analysis

The comparative evaluation of the performance of XGBoost, Random Forest, Decision Tree, and ExtraTrees shows (see Figure 6) that the classification performance of all these models was very good, with accuracy, precision, recall, and F1-score values greater than 99%. The similarity of values for these four metrics shows that none of the models has bias towards false positives or false negatives. This is particularly important for intrusion detection systems (IDS), as false negatives and false positives may affect the reliability of the IDS.

4.6.1. Effect of Ensemble Learning

The results clearly demonstrate that ensemble-based approaches outperform the results of the single Decision

Tree model. At the baseline configuration, the highest results were obtained by the ExtraTrees model (0.9957), followed by the Random Forest model (0.9940), whereas the results of the Decision Tree model were significantly lower (0.9931). This is in accordance with theoretical expectations that ensemble methods reduce variance and increase generalization ability due to the combination of multiple weak classifiers. The results of the Random Forest and ExtraTrees models demonstrate the robustness of bagging-based approaches in handling high-dimensional intrusion detection problems.

4.6.2. Impact of Hyperparameter Optimization

Hyperparameter optimization had varying results for the different models. XGBoost recorded the best results, with the greatest improvement from 0.9918 to 0.9957, a difference of +0.0039. This suggests that boosting-based models are highly sensitive to model configuration.

Random Forest recorded a moderate improvement, +0.0011, which implies that, although the model can be improved with configuration, it is already quite stable. Decision Tree recorded the least improvement, +0.0006, which supports the limited capabilities of individual trees. A notable observation was the negative improvement recorded for the ExtraTrees model, -0.0002, which implies that the model is already quite stable in its default configuration and may be sensitive to overfitting.

4.6.3. Model Ranking After Optimization

Finally, after hyperparameter tuning, the optimized model with the best performance is XGBoost (0.9957), which is almost equal to the maximum baseline performance achieved by the ExtraTrees model. Random Forest is close but with a slightly lower performance (0.9951), while the worst model is the Decision Tree model.

From the results obtained, it can be concluded that models based on boosting can perform almost equally or even better than bagging models. However, the difference in performance is small, which means that the choice of model may also depend on efficiency.

5. Conclusion

This study proposed a hybrid intrusion detection framework that combines MiniBatchKMeans clustering, biased Random Forest classifiers, Fast Correlation-Based Filter (FCBF) feature selection, SMOTE oversampling, and hyperparameter optimization using Hyperopt TPE. On the NSL-KDD dataset, the technique reported accuracy levels higher than 99% consistently, where ensemble techniques like optimized XGBoost and Random Forest surpassed individual classifiers. Complementing the technique with cluster-based sampling assisted in maintaining data diversity and enhanced detection of some of the minority classes, whereas feature selection and reduction in dimension assisted in higher computational efficiency without losses in performance. In light of these encouraging outcomes, the recognition of highly rare intrusion classes, especially Class 4, proved to be difficult because of extreme class imbalance. This indicates the importance of even higher-level imbalance-processing mechanisms and dedicated learning for the underrepresented classes. Beyond this work, upcoming research will consider adaptive online clustering, GAN-based oversampling, and testing on current large-scale network datasets for the purpose of increased real-time applicability and resilience to dynamically emerging cyber-attacks.

6. Limitations and Future Work

Despite the excellent empirical results achieved by the proposed clustering-based hybrid IDS framework, it is necessary to point out the following limitations:

1) Dependence on the NSL-KDD Dataset: The experimental validation of the proposed framework was performed using the NSL-KDD dataset. Although the NSL-KDD dataset has addressed the redundancy problems inherent in the KDD'99 dataset, it does not capture the network traffic characteristics of the present day. It does not contain encrypted network traffic, zero-day exploits, advanced persistent threats, or realistic network traffic distributions. Hence, the applicability of the model in real-world scenarios, especially in enterprise or cloud environments, might be restricted.
2) Extreme Minority Class Scarcity: In spite of the integration of the SMOTE technique along with the clustering-based sampling technique, extremely scarce minority classes, such as Class 4, which had only seven instances in the test dataset, achieved low recall values, i.e., 0.71 for the majority of the classifiers. Although the oversampling technique using SMOTE is employed, the complete absence of genuine diversity in the attack scenarios might affect the quality of the oversampled data.
3) Static Clustering Strategy: The MiniBatchKMeans clustering approach was used in an offline manner with the number of clusters fixed in advance. However, in real-world network environments, the situation is dynamic and constantly changing. New types of cyber-attacks are constantly being developed. The static clustering approach may not perform well in such dynamic environments.
4) Computational Complexity: The proposed framework includes different components such as clustering, SMOTE oversampling, FCBF feature selection, and hyperparameter tuning using

the Hyperopt approach. Although the inclusion of these different components is beneficial for the improvement of the overall system's performance, the complexity may increase the computational time required for the training process.

To address the aforementioned limitations and further improve the proposed framework, the following research directions are recommended:

1) Evaluation on Modern and Large-Scale Datasets: Future research should extend the evaluation of the proposed framework on recent and large-scale datasets for intrusion detection, including CICIDS2017, CICIDS2018, UNSW-NB15, and real-time traffic data. This would improve the generalizability of the proposed framework.
2) Advanced Minority Class Learning: Due to the difficulty in dealing with very rare class instances (e.g., U2R), the following research direction can be considered in the future such as GAN-based synthetic data generation, Cost-sensitive learning, Optimizing Focal Loss, etc. These methods might be more effective in modeling rare attack behaviors than the traditional SMOTE algorithm.
3) Adaptive and Online Clustering: In order to make the model more applicable in real-time scenarios, the following can be incorporated in future works such as Incremental clustering algorithms, Online MiniBatchKMeans, or Mechanisms for concept drift detection. This would allow the IDS to adapt the clustering structures according to the changing network conditions.

Explainability and Interpretability: In order to make the model more understandable and increase the chances of it being used, the model can incorporate various Explainability and Interpretability techniques such as SHAP or LIME. This would allow the security analysts to better understand the features and the model.

7. Declarations

7.1. Author Contributions

Moshood Abiola Hambali: Writing - Review & Editing, Visualization, Supervision, Project administration, Funding acquisition; **Nahum Zhema Bako:** Conceptualization, Methodology, Writing - Original Draft; **Mu'awuya Dalhatu:** Formal analysis, Investigation, Resources, Data Curation; **Ashraf Ishaq:** Software, Validation, Formal analysis, Investigation, Resources.

7.2. Institutional Review Board Statement

Not applicable.

7.3. Informed Consent Statement

Not applicable.

7.4. Data Availability Statement

The data presented in this study are available publicly online uploaded by A. A. Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, "NSL-KDD dataset," 2009.

7.5. Acknowledgment

Not applicable.

7.6. Conflicts of Interest

The authors declare no conflicts of interest.

8. References

[1] A. H. Farooqi, S. Akhtar, H. Rahman, T. Sadiq, and W. Abbass, "Enhancing network intrusion detection using an ensemble voting classifier for internet of things," *Sensors*, vol. 24, no. 1, p. 127, 2023. https://doi.org/10.3390/s24010127.

[2] Y. Yin, J. Jang-Jaccard, F. Sabrina, and J. Kwak, "Improving multilayer-perceptron (MLP)-based network anomaly detection with birch clustering on CICIDS-2017 dataset," in *2023 26th international conference on computer supported cooperative work in design (CSCWD)*, 2023, pp. 423–431. https://doi.org/10.1109/CSCWD57460.2023.10152640.

[3] M. A. Hambali and O. C. Peter, "An Android Malware Detection System Based on Hybrid

Artificial Neural Network and Decision Tree Algorithm," *SLU Journal Of Science And Technology*, 2024. https://slujst.com.ng/wp-content/uploads/2024/09/SLUJST482_PP_45-68.pdf.

[4] G. Nassreddine, M. Nassereddine, and O. Al-Khatib, "Ensemble learning for network intrusion detection based on correlation and embedded feature selection techniques," *Computers*, vol. 14, no. 3, p. 82, 2025. https://doi.org/10.3390/computers14030082.

[5] M. B. Musthafa *et al.*, "Optimizing IoT intrusion detection using balanced class distribution, feature selection, and ensemble machine learning techniques," *Sensors*, vol. 24, no. 13, p. 4293, 2024. https://doi.org/10.3390/s24134293.

[6] P. Bedi, N. Gupta, and V. Jindal, "I-SiamIDS: an improved Siam-IDS for handling class imbalance in network-based intrusion detection systems," *Appl. Intell.*, vol. 51, no. 2, pp. 1133–1151, 2021. https://doi.org/10.1007/s10489-020-01886-y.

[7] L. B. Asaju, P. B. Shola, N. Franklin, and H. M. Abiola, "Intrusion Detection System on a Computer Network Using an Ensemble of Randomizable Filtered Classifier, K-Nearest Neighbor Algorithm," *FUW Trends in Science & Technology Journal*, vol. 2, no. 1, pp. 550–553, 2017. https://www.semanticscholar.org/paper/INTRUSION-DETECTION-SYSTEM-ON-A-COMPUTER-NETWORK-AN-Asaju-Bolaji/91fc2ea30b385363e0644afcff4893864bed8372

[8] M. A. Hambali, T. O. Oladele, and K. S. Adewole, "Microarray cancer feature selection: Review, challenges and research directions," *Int. J. Cogn. Comput. Eng.*, vol. 1, no. October, pp. 78–97, 2020, https://doi.org/10.1016/j.ijcce.2020.11.001.

[9] I. Ramos-Pérez, J. A. Barbero-Aparicio, A. Canepa-Oneto, Á. Arnaiz-González, and J. Maudes-Raedo, "An extensive performance comparison between feature reduction and feature selection preprocessing algorithms on imbalanced wide data," *Information*, vol. 15, no. 4, p. 223, 2024. https://doi.org/10.3390/info15040223.

[10] M. A. Hambali, T. O. Oladele, K. S. Adewole, A. K. Sangaiah, and W. Gao, "Feature selection and computational optimization in high-dimensional microarray cancer datasets via InfoGain-modified bat algorithm," *Multimed. Tools Appl.*, vol. 1213, pp. 1–45, 2022, https://doi.org/10.1007/s11042-022-13532-5.

[11] O. Peter, M. Hambali, S. Tosin, A. Wreford, and C. Ifeoma, "Android Malware Detection System: a Review and Research Directions," *Int. Rev. Comput. Softw.*, vol. 19, no. 1, pp. 1–13, 2024. https://doi.org/10.15866/irecos.v19i1.23734.

[12] M. Altalhan, A. Algarni, and M. T.-H. Alouane, "Imbalanced data problem in machine learning: A review," *IEEE Access*, vol. 13, pp. 13686 – 13699, 2025. https://doi.org/10.1109/ACCESS.2025.3531662.

[13] J. Zhu and X. Liu, "An integrated intrusion detection framework based on subspace clustering and ensemble learning," *Comput. Electr. Eng.*, vol. 115, p. 109113, 2024. https://doi.org/10.1016/j.compeleceng.2024.109113.

[14] Z. Chen, L. Zhou, and W. Yu, "ADASYN– Random Forest based intrusion detection model," in *Proceedings of the 2021 4th International Conference on Signal Processing and Machine Learning*, 2021, pp. 152–159. https://doi.org/10.1145/3483207.3483232.

[15] H. Le, T.-T.-H., Shin, Y., Kim, M., & Kim, "Towards unbalanced multiclass intrusion detection with hybrid sampling methods and ensemble classification," *Appl. Soft Comput.*, vol. 15, p. 111517, 2024, https://doi.org/10.1016/j.asoc.2024.111517.

[16] O. Nassreddine, G., Nassereddine, M., & Al-Khatib, "Ensemble learning for network intrusion detection based on correlation and embedded feature selection techniques," *Computers*, vol. 14, no. 3, p. 82, 2025, https://doi.org/10.3390/computers14030082.

[17] Y. Lv, H., & Ding, "A hybrid intrusion detection system with K-means and CNN+LSTM," *EAI Endorsed Trans. Scalable Inf. Syst.*, vol. 11, no. 6, p. Article 6, 2024, https://doi.org/10.4108/eetsis.5667.

[18] J. Henriques, F. Caldeira, T. Cruz, and P. Simões, "Combining k-means and xgboost models for anomaly detection using log datasets," *Electronics*, vol. 9, no. 7, p. 1164, 2020. https://doi.org/10.3390/electronics9071164.

[19] P. Chapagain, A. Timalsina, M. Bhandari, and R. Chitrakar, "Intrusion detection based on PCA with improved K-means," in *Innovations in Electrical and Electronic Engineering*, 2022, pp. 13–27. https://doi.org/10.1007/978-981-19-1677-9_2.

[20] M. Kherbache, D. Espes, and K. Amroun, "An Enhanced approach of the K-means clustering for Anomaly-based intrusion detection systems," in *2021 International Conference on Computing, Computational Modelling and Applications (ICCMA)*, 2021, pp. 78–83. https://doi.org/10.1109/ICCMA53594.2021.00021.

[21] N. Hu, Z. Tian, H. Lu, X. Du, and M. Guizani, "A multiple-kernel clustering based intrusion detection scheme for 5G and IoT networks," *Int. J. Mach. Learn. Cybern.*, vol. 12, no. 11, pp.

3129–3144, 2021. https://doi.org/10.1007/s13042-020-01253-w.

[22] K. Samunnisa, G. S. V. Kumar, and K. Madhavi, "Intrusion detection system in distributed cloud computing: Hybrid clustering and classification methods," *Meas. Sensors*, vol. 25, p. 100612, 2023. https://doi.org/10.1016/j.measen.2022.100612.

[23] M. Aamir and S. M. A. Zaidi, "Clustering based semi-supervised machine learning for DDoS attack classification," *J. King Saud Univ. Inf. Sci.*, vol. 33, no. 4, pp. 436–446, 2021. https://doi.org/10.1016/j.jksuci.2019.02.003.

[24] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset," *IEEE access*, vol. 9, pp. 22351–22370, 2021. https://doi.org/10.1109/ACCESS.2021.3056614.

[25] R. Younisse and Q. A. Al-Haija, "An empirical study on utilizing online k-means clustering for intrusion detection purposes," in *2023 International Conference on Smart Applications, Communications and Networking (SmartNets)*, 2023, pp. 1–5. https://doi.org/10.1109/SmartNets58706.2023.10215737.

[26] A. A. Chormale, P. Ukhalkar, and U. A. Deshmukh, "Clustering-Based Intrusion Detection System for High Volume and High Velocity Packet Streams," in *2024 8th International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, 2024, pp. 1–5. https://doi.org/10.1109/ICCUBEA61740.2024.10775170.

[27] H. Wang, F. Kandah, T. Mendis, and L. Medury, "Clustering-based intrusion detection system meets multi-critics generative adversarial networks," *IEEE Internet Things J.*, 2025. https://doi.org/10.1109/JIOT.2025.3533918.

[28] A. Thakkar and R. Lohiya, "Attack classification of imbalanced intrusion data for IoT network using ensemble-learning-based deep neural network," *IEEE Internet Things J.*, vol. 10, no. 13, pp. 11888–11895, 2023. https://doi.org/10.1109/JIOT.2023.3244810.

[29] H. Ren, Y. Tang, W. Dong, S. Ren, and L. Jiang, "DUEN: Dynamic ensemble handling class imbalance in network intrusion detection," *Expert Syst. Appl.*, vol. 229, p. 120420, 2023. https://doi.org/10.1016/j.eswa.2023.120420.

[30] M. A. Hossain and M. S. Islam, "Ensuring network security with a robust intrusion detection system using ensemble-based machine learning," *Array*, vol. 19, p. 100306, 2023. https://doi.org/10.1016/j.array.2023.100306.

[31] N. Thockchom, M. M. Singh, and U. Nandi, "A novel ensemble learning-based model for network intrusion detection," *Complex Intell. Syst.*, vol. 9, no. 5, pp. 5693–5714, 2023. https://doi.org/10.1007/s40747-023-01013-7.

[32] F. Jemili, R. Meddeb, and O. Korbaa, "Intrusion detection based on ensemble learning for big data classification," *Cluster Comput.*, vol. 27, no. 3, pp. 3771–3798, 2024. https://doi.org/10.1007/s10586-023-04168-7.

[33] A. Sarkar, H. S. Sharma, and M. M. Singh, "A supervised machine learning-based solution for efficient network intrusion detection using ensemble learning based on hyperparameter optimization," *Int. J. Inf. Technol.*, vol. 15, no. 1, pp. 423–434, 2023. https://doi.org/10.1007/s41870-022-01115-4.

[34] M. A. Akhtar, S. M. O. Qadri, M. A. Siddiqui, S. M. N. Mustafa, S. Javaid, and S. A. Ali, "Robust genetic machine learning ensemble model for intrusion detection in network traffic," *Sci. Rep.*, vol. 13, no. 1, p. 17227, 2023. https://doi.org/10.1038/s41598-023-43816-1.

[35] A. A. Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, "NSL-KDD dataset," 2009.