computers

MDPI

Article

Ensemble Learning for Network Intrusion Detection Based on Correlation and Embedded Feature Selection Techniques

Ghalia Nassreddine [1], Mohamad Nassereddine [2,*] and Obada Al-Khatib [2]

[1] Computer and Information Systems Department, Rafik Hariri University, Damour-Chouf 2010, Lebanon; nassreddinega@rhu.edu.lb
[2] School of Engineering, University of Wollongong in Dubai, Knowledge Village, Dubai P.O. Box 20183, United Arab Emirates; obadaalkhatib@uowdubai.ac.ae
* Correspondence: mohamadnassereddine@uowdubai.ac.ae

Abstract: Recent advancements across various sectors have resulted in a significant increase in the utilization of smart gadgets. This augmentation has resulted in an expansion of the network and the devices linked to it. Nevertheless, the development of the network has concurrently resulted in a rise in policy infractions impacting information security. Finding intruders immediately is a critical component of maintaining network security. The intrusion detection system is useful for network security because it can quickly identify threats and give alarms. In this paper, a new approach for network intrusion detection was proposed. Combining the results of machine learning models like the random forest, decision tree, k-nearest neighbors, and XGBoost with logistic regression as a meta-model is what this method is based on. For the feature selection technique, the proposed approach creates an advanced method that combines the correlation-based feature selection with an embedded technique based on XGBoost. For handling the challenge of an imbalanced dataset, a SMOTE-TOMEK technique is used. The suggested algorithm is tested on the NSL-KDD and CIC-IDS datasets. It shows a high performance with an accuracy of 99.99% for both datasets. These results prove the effectiveness of the proposed approach.

Keywords: network intrusion detection; machine learning; feature selection; imbalanced dataset; stacking technique; embedded feature selection technique

Academic Editors: Helge Janicke and Leandros Maglaras

Received: 4 February 2025
Revised: 20 February 2025
Accepted: 20 February 2025
Published: 25 February 2025

Citation: Nassreddine, G.; Nassereddine, M.; Al-Khatib, O. Ensemble Learning for Network Intrusion Detection Based on Correlation and Embedded Feature Selection Techniques. *Computers* **2025**, *14*, 82. https://doi.org/10.3390/computers14030082

1. Introduction

Recently, the huge development in many sectors has led to a surge in the usage of smart devices. This increase has led to a growth in the size of the underlying network connecting these devices. However, along with the expansion of the network, there have also been more policy violations affecting information security. For example, T-Mobile revealed that an unauthorized user accessed its API and stole private account information from 37 million users [1]. Latitude Financial, based in Melbourne, experienced a data breach in March 2023 that revealed more than 14 million records. This included over 8 million driver's licenses and 53,000 passport numbers [2]. Thus, network-based intrusion detection systems have become more essential due to the significant increase in the volume and complexity of network traffic. These systems play a vital role in detecting and mitigating cyber threats when traditional security measures often fail to face these network attacks. Effective intrusion detection systems (IDSs) are crucial for protecting required infrastructure, securing sensitive data, and ensuring the overall security and resilience of computer networks [3].

One important part of guaranteeing network security is to quickly find intruders, spot threats, and send alerts. Here, there are two main categories of IDS:

- Host-based Intrusion Detection Systems (HIDSs): The hosts, such as servers or workstations, monitor activities on individual devices by analyzing system logs, file integrity, and processes to detect doubtful behavior [4].
- Network-based Intrusion Detection Systems (NIDSs): These systems observe and examine network traffic in real-time to detect possible threats like unauthorized access, malware, or anomalies, across the entire network [5].

Thus, HIDS checks single devices, but NIDS looks at the whole network. This paper focuses on NIDS.

Machine learning (ML) and deep learning (DL) have been utilized extensively for NIDS in recent years to detect various network threats, assisting administrators in taking the necessary precautions to stop network invasions. The majority of NIDSs are based on conventional machine learning approaches, such as k-nearest neighbor (KNN) [6], support vector machine (SVM) [7], and other learning techniques [8]. The NIDS ML-based systems face two main challenges. The first one is the huge number of features in the networks. Thus, the adoption of the feature selection step in the intrusion detection problem is crucial because it minimizes the number of variables and retains the most significant features by eliminating irrelevant features [9]. The second challenge is the type of available datasets. It has been noted that ML/DL algorithms for intrusion detection systems perform worse when deployed on imbalanced datasets, which is one of the most prevalent IDS problems. A dataset is said to be imbalanced when a certain class is underrepresented in it. For example, the NSL-KDD dataset [10] has a higher percentage of normal samples than attack samples.

The authors of [11] suggested a hybrid technique to address the imbalance issue to increase the detection rate for minority classes while maintaining efficiency. This hybrid method used the TOMEK link to minimize noise by combining undersampling and Synthetic Minority Over-Sampling (SMOTE). To improve the intrusion detection system, the authors employed two deep learning models: the Long Short-Term Memory (LSTM) and the Convolutional Neural Network (CNN). In [12], a hybrid IDS strategy that combines LSTM and LeNet-5 was proposed. Unlike other existing NIDSs, the authors focused on applying the Imbalanced Generative Adversarial Network (IGAN) technique to address the problem of class imbalance in the dataset. The problems of complex models' lengthy training and detection times were also addressed using the hybrid LSTM ensemble model technique.

This study proposed a new method for NIDS that concentrates on developing sophisticated feature selection methods and employs a hybrid sampling technique to address the issue of the imbalanced dataset. Therefore, the primary contributions of this research can be summarized as follows:

- Creates an advanced feature selection algorithm to increase model efficiency by reducing computation time and increasing detection accuracy.
- Handles the problem of imbalanced datasets by using a hybrid technique that includes data augmentation techniques, undersampling, and oversampling.
- Enhances the classification performance of the proposed approach by adopting an ensemble approach that merges the decisions of different classifiers into one using a stacking technique.
- Performs a comparison of the suggested strategy with existing techniques using NSL-KDD and CIC-IDS2017 datasets based on various performance metrics like accuracy, F1 score, precision, and recall.

This paper is organized into five main sections. A summary of related work on NIDS is illustrated in Section 2. In addition, the gaps in existing studies with the main motivation

of this research are explored in this section. Section 3 shows the proposed approach. The results of applying this approach to the NSL-KDD and CIC-IDS datasets are illustrated and compared with other studies in Section 4. This paper is concluded in Section 5.

2. Related Work

By addressing intrusion attacks, network security engineers strive to always maintain service availability. An intrusion detection system (IDS) is one of the available tools for identifying and categorizing any unusual activity. Therefore, to maintain service availability, confidentiality, and integrity, the IDS needs to be constantly updated with the most recent intruder attack signatures. Learning new assaults and the IDS's speed are also critical issues. The Knowledge Discovery and Data Mining (or Knowledge Discovery in Databases) KDD dataset is useful for testing and assessing various machine learning techniques, as demonstrated in [13]. To create a respectable and equitable experimental dataset, it primarily concentrated on the KDD preprocessing step. For this investigation, the J48, Multi-Layer Perceptron (MLP), and Bayes Network classifiers were selected. It has been demonstrated that the J48 classifier has the highest accuracy rate for identifying and categorizing all intrusions of the DOS, R2L, U2R, and PROBE types in the KDD dataset. The nonsymmetric deep autoencoder (NDAE) for unsupervised feature learning is a unique deep learning method for intrusion detection that was introduced by the authors in [14]. Additionally, a unique deep learning classification model built with a stacked non-symmetric deep auto-encoder was developed. TensorFlow, which is GPU-enabled, was used to implement the suggested classifier, and the benchmark KDD Cup'99 and NSL-KDD datasets were used for evaluation. Thus far, the suggested model has produced encouraging results that show advantages over current methods and a high potential for application in contemporary NIDSs. The authors of [15] presented a new generation of network intrusion detection techniques that combines a deep-feed forward neural network approach with reinforcement learning based on Q-learning. With the use of an automated trial-error method, the suggested method gave a network environment the capacity to continually learn and improve its detection capabilities for various kinds of network intrusions. They offer information on how to adjust the various hyperparameters in the suggested model for better self-learning. They verified that the lower discount factor, which is set at 0.001 under 250 training episodes, produced the best performance outcomes based on experimental results using the NSL-KDD dataset. The authors of [16] suggested a network intrusion detection technique based on federated learning in conjunction with the features of network traffic. If local data was kept, this technique enabled cooperative deep-learning training between several ISPs or other organizations. In addition to increasing the model's detection accuracy, it safeguards network traffic privacy. Workers who participated in federated learning had higher detection accuracy, according to trials conducted on the CICIDS2017 network intrusion detection dataset. Federated learning nearly equals centralized deep learning models in terms of accuracy and other performance metrics. A deep learning-based method for predicting network intrusion warnings was provided in [17]. It was demonstrated that a deep learning model based on Gated Recurrent Units (GRUs) could learn relationships in security warning sequences and generate likely future alerts based on a history of alerts from an attacking source. The Warden alert sharing platform's intrusion alert sequences were used to test the model's performance. In [18], the authors presented a new ML-based network intrusion detection model that was specially made for large and unbalanced datasets. It used Principal Component Analysis (PCA) for dimension reduction, Random Oversampling to address data imbalance, and Stacking Feature Embedding based on clustering results. UNSW-NB15, CIC-IDS-2017, and CIC-IDS-2018 are three state-of-the-art benchmark datasets that were used to thoroughly assess this model's performance.

In [19], the authors suggested a machine learning-based automated intrusion detection system. The CSE-CIC-IDS 2018 and UNSW-NB15 datasets were used to collect the data, which was then preprocessed using Min–Max normalization and null value management. The Advanced Synthetic Minority Oversampling Technique (ASmoT) was used to lessen the issue of class imbalance. While the Opposition-based Northern Goshawk Optimization technique identified attacks using a Mud Ring-assisted multilayer support vector machine (M-MultiSVM), the Modified Singular Value Decomposition was utilized for feature extraction. The authors of [20] suggested an IDS defense mechanism that uses anomaly detection and machine learning to strengthen IoT network security against DoS assaults. Network traffic was continuously observed for departures from typical characteristics using anomaly detection. They employed four different kinds of supervised classifier algorithms: SVM, RF, k-nearest neighbor (KNN), and decision tree (DT). They used two different feature selection algorithms—the Genetic Algorithm and the correlation-based feature selection algorithm—and evaluated how well they performed. To train the model, they used the IoTID20 dataset, which is among the most recent for identifying unusual activities in IoT networks. When DT and RF classifiers were trained using features chosen by GA, the best results were achieved. Maintaining security and privacy in the ever-changing information technology ecosystem is a growing problem for cyber workers. ML has shown promise in identifying the new malware that is emerging. Based on in-depth testing on the BODMAS dataset, the authors of [21] offered low-complexity techniques for malware detection on PCs, IoT devices, and servers. These techniques enable real-time virus detection without requiring a large amount of processing power. They offered a thorough examination of feature reduction and lightweight algorithms to make the suggested approach functional on a variety of platforms, including PCs, servers, and Internet of Things gadgets.

For a data-driven approach to network intrusion detection, the distribution of intrusion data in a network is essential. It encounters obstacles such as decision boundary discontinuity and hierarchical dependency omission. In [22], the authors suggested a novel detection framework for Hierarchical Dependency and Class Imbalance (HIDIM) to overcome these issues. Initially, they included the protocol hierarchy of attributes into a paragraph embedding model and regarded semantic attributes as words. Second, they created a synthetic oversampling technique that uses the mutual nearest neighbor method to identify each disjunct's bounds. It then crossed or mutated features according to their significance to create high-quality samples inside certain boundary areas. To create IDS for IoT devices, the work carried out in [23] suggested using both supervised and unsupervised deep learning models trained via Federal learning. Using the N-BaIoT dataset of nine IoT devices, the performance of models trained using Federal learning was contrasted with models taught using non-Federal learning. A randomized search for hyperparameter optimization was carried out to raise the accuracy of DL models. The prediction findings were assessed using a range of performance indicators. Based on testing both Federal learning and non-Federal learning-trained models on all nine IoT devices, the results showed that the unsupervised AutoEncoder model taught via Federal learning was the best overall in terms of all metrics [23].

According to the above literature review, most of the existing studies are based on oversampling techniques such as SMOTE and ASmot. Thus, class imbalance remains a challenge for NIDS. In addition, many studies used deep learning-based models such as autoencoders that need high computational resources. Other systems were based on federated learning (FL); however, they did not take into consideration privacy issues. Furthermore, the communication in FL was not rigorously evaluated, which may affect scalability. For feature selection issues, many studies mainly used filter techniques such

as PCA. However, many powerful techniques, such as wrappers or embedded, were not investigated.

Based on these gaps, the objectives for this work can be summarized as follows:

- Propose advanced feature selection techniques that combine filter and embedded methods with a concentration on feature importance.
- Use a hybrid technique that combines data augmentation techniques, undersampling, and oversampling.
- Use a stacking ensemble learning technique based on logistic regression as a meta-model. This technique can combine the strength of basic models and provide higher performance.
- Test the proposed approaches on two benchmark datasets, NSL-KDD and CIC-IDS2017.

3. Methodology

In this section, the proposed approach, including data preprocessing, feature scaling, data encoding, feature selection, data resampling, and performance evaluation, will be explained. Algorithm 1 illustrates the block diagram of the proposed approach. As shown in Algorithm 1, this approach is composed of seven main steps.

Algorithm 1. Proposed approach for network intrusion detection based on ensemble learning technique

Start

1. Import Python libraries
2. Load the dataset
3. Perform data preprocessing:
   - Handle missing and duplicate values
   - Convert data types
   - Apply feature scaling
   - Apply Feature Encoding
     - Label Encoding for the target variable.
     - One-Hot Encoding for feature variables like “protocol_type”, “Services”, and “flag”.
4. Feature Selection: Select the most significant features by applying first a Correlation-based Feature Selection technique. XGBoost Embedded Feature Selection techniques are used later to optimize the select features set.
5. Handle Imbalanced Data using SMOTE-TOMEK to balance class distribution
6. Apply the proposed ensemble learning model:
   a. Perform K-Fold Cross-Validation
   b. Train multiple classifiers (k-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), XGBoost)
   c. Combines the training model based on the staking technique: Logistic Regression is used as the meta-learner
7. Evaluate the model performance using Confusion Matrix, Precision, Recall, F1 Score, and Accuracy)

End

1. Data preprocessing: In this step, the data will be prepared to train the ML model. This is accomplished by handling the missing values, removing duplicated rows, clearing space from column names, converting data types from 64 to 32 bits to reduce the dataset size, and finally merging the classes with low numbers with similar types. Furthermore, this step ensures that each feature contributes equally to the model

and avoids biases brought on by the difference in the range of dataset features. In this study, the StandardScaler is used. It assists in scaling numerical features to have a mean of 0 and a standard deviation of 1. This procedure of centering the data improves its suitability for a variety of algorithms that presume a conventional normal distribution. The standard scaler can be done using:

$$X_{scaled} = \frac{X_i - X_{mean}}{\sigma_X} \quad (1)$$

where X_i is the i value of the feature X, X_{mean} is the mean of the feature X, and σ_X is the standard deviation of X.

2. Feature Encoding: The target variable, called outcome, can include categorical variables that represent the attack types. In addition, some features, such as "protocol_type", "Services", and "flag" contain non-numeric values. To resolve this issue, a transformation should be made to convert all categorical variables to numeric values. This critical step can be performed with a label encoder and one hot encoder. The label encoder is a technique that sets a distinctive numerical value to categorical variables, starting with 0 and incrementing by one for each unique label. Thus, it guarantees that categorical variables are compatible with the ML algorithm that necessitates numerical inputs. The label encoder is used in this study to transform the target variable into numeric values. One-hot encoding is another technique for converting categorical variables into a binary representation [24]. This technique transforms each category in the non-numeric variable into a new binary column (0 s and 1 s). A value of 1 indicates the presence of a category; however, a value of 0 stands for its absence. This technique is applied to the feature columns like "protocol_type", "Services", and "flag". Indeed, these variables are nominal categorical features that have a non-ordinal relationship between the categories. One-hot encoding ensures that no artificial ordinal relationship is created between these categories. It permits the model to treat each category independently and detect the true nature of network protocols, services, and flags.
3. The feature selection (FS) techniques focus on identifying a subset of features from the overall set that suitably represents the data. All the attributes in the selected subset must be highly relevant to the target variable. FS methods can primarily be grouped into wrapper, filter, and embedding approaches [25]. Filter approaches assess the significance of dataset characteristics using statistical measures. The wrapper approaches use classification performance as a criterion for the evaluation and identification of feature subsets. However, embedding approaches merge FS with the learning process [26]. In this research, a hybrid FS technique is proposed that merges the filter and embedded approaches. This technique aims first to examine the relevancy and redundancy of the features in the selected subset to choose the best ones. This technique is illustrated in Figure 1.

As shown in Figure 1, the proposed FS approach is composed of two techniques: correlation feature selection and the embedded algorithm.

- Correlation-based feature selection (CFS): This is an approach for identifying feature subsets that reveal a high correlation with the target variable while providing a low correlation among the features in the identified subsets. The idea behind CFS is to select a subset of features that expand information concerning the target variable while minimizing redundancy included in the features. At the beginning, the CFS algorithm calculates the correlation between each feature and the target variable. After that, it assesses the correlation between each feature pair. Following this, it selects the

subset of features exhibiting the strongest correlation with the target variable while maintaining the lowest correlation among themselves [27].

- Embedded methods integrate the FS process with the learning step. The embedded methods are quicker and more accurate than other filter and wrapper methods. Within the embedded approaches, learning techniques, such as regularization or tree-based methods like random forest and XGBoost, are used. Tree-based techniques offer feature importance for selecting the most suitable subset of features.

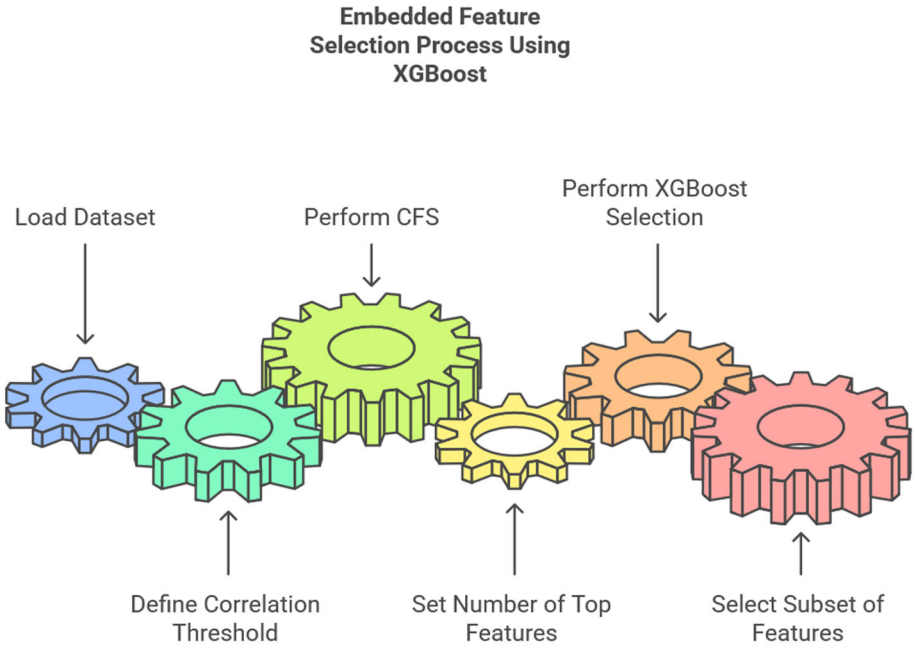


Figure 1. FS approach based on CFS and embedded algorithm.

This study combines the CFS and embedded methods to reduce the number of selected features and consequently increase the ML model performance. First, CFS is used to select a subset of features that have a high correlation with the target variable and a low correlation between themselves. Pearson correlation is employed in this step for evaluating this correlation. The embedded algorithm then takes the selected subset and performs optimization-based feature selection on it. XGBoost is used as a learning technique in this step. This two-stage approach effectually merges the simplicity of CFS with the adaptive optimization abilities of embedded techniques, producing a final reduced dataset suitable for an ML model.

4. Handling imbalanced dataset issues: An imbalanced dataset is a critical challenge for the ML models. It can lead to poor performance in detecting minority classes and biased model predictions. Several techniques are used to handle this issue, including anomaly detection and resampling (oversampling and undersampling). In this study, this issue is handled using a hybrid model that integrates the oversampling and undersampling techniques. This hybrid technique, based on SMOTE-TOMEK [28], combines both oversampling (based on SMOTE for the minority class) and undersampling (based on Tomek links to remove Tomek pairs). SMOTE specifically addresses imbalanced datasets by generating artificial instances of the minority class. TOMEK Link is a technique that eliminates noise from a dataset by removing pairs of instances from a dataset. These pairs include the points from minority and majority classes that are each other's nearest neighbors and belong to different classes. By eliminating these pairs, the decision boundary becomes clearer, class overlap will be minimized, and the overall model performance will be enhanced. Using TOMEK is important

because, after using SMOTE, the class clusters may intrude on one another's territory. The classifier model will show overfitting. Since the TOMEK linkages are based on the nearest neighbors of opposite-class paired samples, most class observations from these links are eliminated to enhance class separation around the decision borders. TOMEK linkages are used on the oversampled minority class instances that SMOTE creates to improve class clusters. Consequently, rather than eliminating data solely from the majority class, we often delete observations from both classes in the context of TOMEK linkages. Thus, SMOTE-TOMEK merges SMOTE and TOMEK links to eliminate the class overlap and noise. Thus, this technique guides clearer decision boundaries and may enhance classifier performance and minimize overfitting risks compared to employing SMOTE alone.

5. Handle Overfitting: Overfitting takes place when a machine learning algorithm extensively identifies noise or random fluctuations as significant patterns. This may lead to inferior performance, especially when the model is used on new and unfamiliar data due to inadequate generalization. Several techniques, such as regularization and cross-validation, can handle overfitting. This paper will employ K-fold cross-validation to address the overfitting challenge. K-fold cross-validation is a technique for assessing the prediction efficacy and generalization ability of the ML model [29]. The basic concept is to divide the dataset into "K" subsets (called folds) of roughly equivalent size. The model undergoes K training iterations, utilizing K-1 folds for training and reserving one-fold for validation at each iteration. This procedure is repeated K times, using a distinct fold as the validation set in each iteration. The performance metrics are averaged across multiple iterations of this process, each featuring a unique split. K-fold cross-validation improves the reliability of a model's performance estimate by mitigating the influence of a particular data split on the assessment. It is especially beneficial when the dataset is constrained or when there are concerns regarding the randomness of data splitting.
6. Ensemble Learning Approach: In this study, four ML models are used and compared:
    - The k-nearest neighbors (KNNs): This algorithm is a non-parametric, supervised learning method usually used for classification and regression tasks. It predicts the class of a new input by identifying the principal class among its k-nearest neighbors in the feature space. Performance enhancement can be achieved by hyperparameter adjustment [30].
    - Decision Tree (DT) [31]: This model is a supervised learning method widely employed in ML to model and predict outcomes based on input data. It has a hierarchical structure in which each internal node assesses an attribute, each branch correlates to an attribute value, and each leaf node indicates the ultimate decision or prediction. DT algorithms are applicable for handling both regression and classification issues. They are adaptable in modeling complex decision-making problems due to their interpretability and flexibility. Their hierarchical nature enables the illustrations of decision-making scenarios that consider several causes and effects.
    - Random Forest (RF): In [32], Breiman introduced RF, a decision tree methodology that creates several decision trees. It employs hundreds of input variables without excluding any and groups them according to their importance. RF is distinguished as a collection of classification trees, each of which produces a singular vote for the most prevalent class according to the input data. In contrast to other machine learning techniques, RF needs fewer parameters to be

defined before execution. A collection of individual tree-structured classifiers can be defined in RF as:

$$\{h(x, \theta_k),\ k = 1, 2,\ \ldots.\}$$

where h stands for the RF classifier, θ_k represents the random vectors distributed independently identical, and each tree has a vote for the most important class at input variable x. The achievement of RF depends on the construction of each decision tree that forms the forest. A bootstrapped subset of the training dataset is created to train each tree inside the forest. Hence, each tree employs around two-thirds of the training dataset as an average. The remaining elements are referred to as Out Of Bag (OOB) samples that are employed for internal cross-validation to examine the classification accuracy of RF [33].

- eXtreme Gradient Boosting (XGBoost): XGBoost was primarily created for capability and efficacy with gradient-boosted decision trees. It stands for a method for machine boosting, or, in other terms, the application of boosting to machines, initially developed by Tianqi Chen [34] and then used by many other developers. Boosting is a machine learning approach that can reduce bias and volatility within the dataset. Boosting smoothed the transformation of weak learners into strong learners. A weak learner classifier reveals a low correlation with the correct classification, while strong learners have a higher correlation. Most boosting algorithms continuously develop weak classifiers and merge them into a robust classifier. The incorporated data are weighted so that accurately classified data lose weight while misclassified data gain significance. XGBoost improves the employment of memory and hardware resources in tree-boosting algorithms. It offers benefits in algorithm improvement and model adjustment and can be used in computational systems.

The outputs of the different ML techniques will be combined to produce accurate and robust outcomes. The three main ensemble learning techniques are based on boosting, bagging, or stacking approaches [35]. By combining weak models to form a single strong model, boosting enhances the accuracy of ML outcomes. This technique minimizes variance and removes overfitting. The boosting approach sequentially trains models, each one learning from the errors of its predecessor. The second method is bagging, which is also known as bootstrap aggregating. This method involves training models separately on different parts of the data. Then, the results of all of them are added by either taking the average or voting on which results are the best. The last approach is stacking, which permits a simple ML algorithm to combine the outcomes of other ML models. This approach categorizes the ML models into either meta-models or sub-models. The meta-model uses the outputs of the sub-models as inputs and produces more accurate predictions. In this study, the stacking technique is used. The stacking process is illustrated in Figure 2. The logistic regression (LR) model is used as a meta-model. Indeed, LR is an ideal meta-model for stacking in NIDS due to its simplicity and capability to optimally integrate predictions from base models like DT, KNN, RF, and XGBoost. LR can be used in both binary and multiclass problems, making it appropriate for the various types of attacks in NIDSs. Its probabilistic outputs permit an effortless adjustment of the threshold and enable simple tuning between false positives and false negatives. The base models are DT, KNN, RF, and XGBoost.

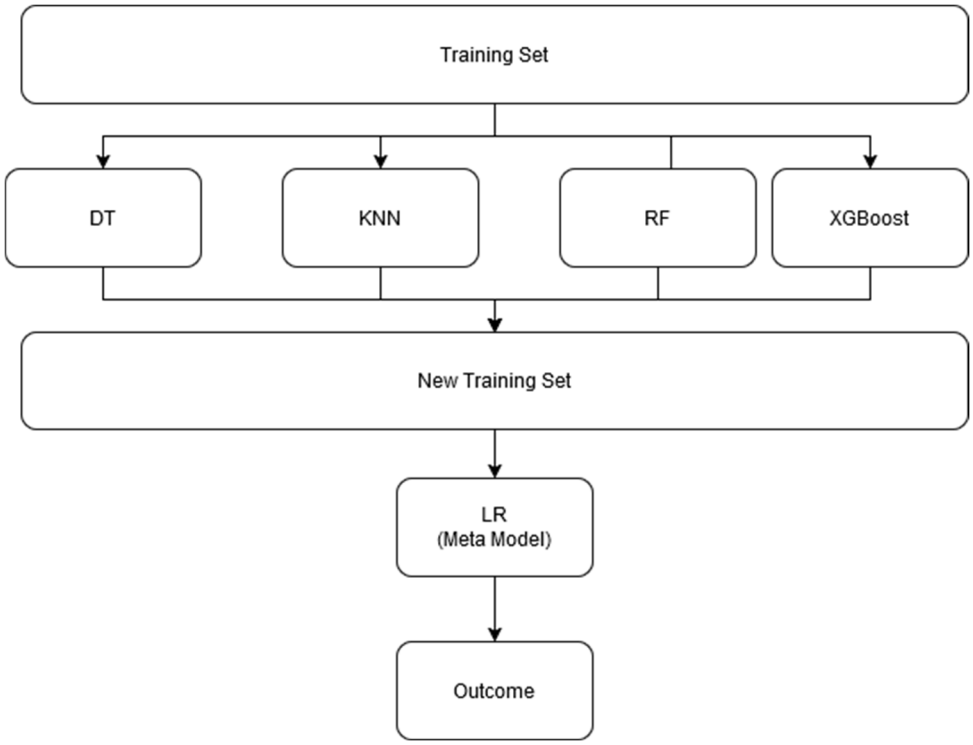


Figure 2. Proposed Stacking Approach.

7. Performance Metrics: the proposed approach is evaluated using a confusion matrix (CM), accuracy, precision, recall, and F1 score. The CM is illustrated in Table 1.

Table 1. Confusion Matrix for multi-class classification.

Actual/Predicted	Class 1	Class 2	Class 3	Class 4	Class 5
Class 1	TP1	A12	A13	A14	A15
Class 2	A21	TP2	A23	A24	A25
Class 3	A31	A33	TP3	A34	A35
Class 4	A41	A42	A43	TP4	A45
Class 5	A51	A52	A53	A54	TP5

TP_i represents the true positive for Class i; A_{ij} shows the situation where, in Actual Class i and Predicted Class j, FN_i is the false negative-associated TP class I, and FP_i stands for the false positive of class i. The performance metrics are illustrated in Table 2.

Table 2. Performance metrics.

Metrics	Description	Equation
Accuracy	Represents the ratio of correct predictions to total predictions made by a classifier, showing its effectiveness in predicting future outcomes.	$Accuracy = \frac{TP_1+TP_2+TP_3+TP_4}{\sum_i TP_i+\sum_{i,j} A_{ij}}$
Precision	Illustrates the part of accurately predicted instances of a specific class to all instances anticipated as positive for that class.	$Precision = \frac{TP_i}{TP_i+FP_i}$
Recall	Represents a value that is the precision of estimating the quantity of real observations for a specific category.	$Recall = \frac{TP_i}{TP_i+FN_i}$
F1 Score	Is a metric that mixes the precision and recall of a model by computing their harmonic mean.	$F1-score = 2 \times Precision \times \frac{Recall}{Precision+Recall}$

4. Evaluations and Results

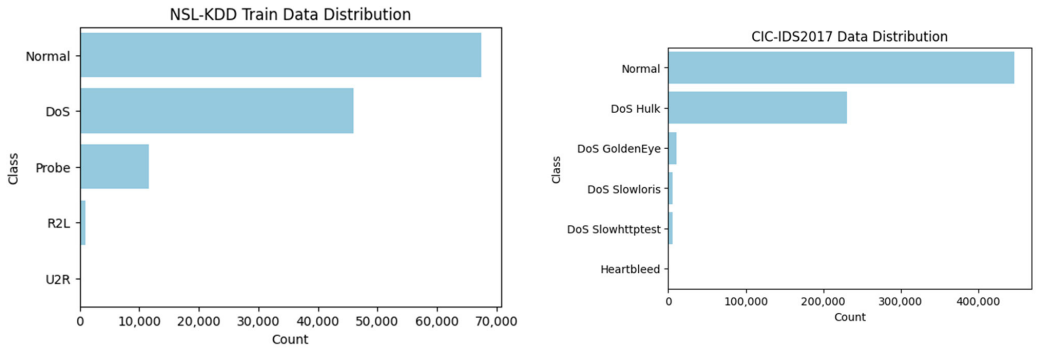
The goal of this paper is to design a NIDS that produces high accuracy and few false alarms. An advanced feature selection method that combines the CFS and XGBoost is used to pick the best subset of the original features in the proposed method. Thus, by removing all the irrelevant features, the accuracy of the model will be expanded. For the classification step, an ensemble learning model that combines the outcomes of KNN, RF, XGBoost, and DT will be employed. This combination is executed by employing a stacking technique. The experimentation is achieved on Colab and a desktop PC equipped with an Intel Core i7-5500U and 16 GB of RAM.

4.1. Benchmarks Datasets

The principal challenge when evaluating a NIDS approach is to find a suitable dataset. The collection of an unaltered real-world dataset that precisely shows network traffic remains a persistent challenge for the cybersecurity research community [36]. Whereas data are authorized for public release or sharing, they will be extensively anonymized or significantly modified. Therefore, there is a risk of losing many crucial data components. For this reason, many researchers try to use simulated datasets such as KDDCupp'99 and its current version, the dataset NSL-KDD. In addition, the Canadian Institute for Cybersecurity (CIC) published an intrusion detection dataset called CIC-IDS2017 that is very similar to real-world data packet captures (PCAPs) [37]. Thus, in this paper, the evaluation of the proposed approach will be conducted using the NSL-KDD and CIC-IDS2017 datasets.

4.1.1. NSL-KDD Dataset

The NSL-KDD dataset is an updated version of the original KDDCup'99 dataset that deals with several issues of the original dataset by eliminating redundant entries, optimizing the number of instances, and preserving the diversity of selected samples. The KDDTrain+, KDDTest+, and KDDTest-21 subsets of the NSL-KDD dataset will be used in this study. Figure 3a shows the distribution of the normal and attack intrusions in these subsets. Four types of attacks will be considered: DoS, PRB, R2L, and U2R. Cross-validation and validation tests are performed on the KDDTrain+ dataset to improve this benchmark. In addition, the KDDTest+ and KDDTest-21 datasets are employed to build an intrusion detection system using a basic hold-out (train–test) method.



(**a**) (**b**)

Figure 3. Attacks and normal class distribution in the NSL-KDD dataset (**a**) and CIC-IDS dataset (**b**).

4.1.2. CIC-IDS2017 Dataset

The CIC-IDS2017 dataset includes the outcomes of network traffic analysis conducted with CICFlowMeter, featuring labeled flows categorized by timestamp, source and destination IP addresses, source and destination ports, protocols, and attack types (CSV files). This dataset is considered one of the most recent intrusion detection collections, enclosing

essential criteria and featuring current attacks, including DDoS, Brute Force, XSS, SQL Injection, Infiltration, Port Scan, and botnet. The dataset comprises 2,830,743 records, arranged into eight files, each one includes 78 unique attributes and their corresponding labels. In this study, Wednesday working hour's collection has been chosen for applying the cross-validation method. Figure 3b illustrates the normal and attack intrusion in this dataset.

4.2. Data Preprocessing

Data preparation is the most work-intensive and time-consuming step in the ML model. Data are generally collected from diverse sources and may contain noise, redundancy, missing values, and inconsistency [38]. Therefore, it is critical to transform raw data into a format suitable for analysis and information extraction. In this study, the preprocessing step entails data filtration, data transformation, and data scaling.

- Data filtration: The dataset may result from various sources and platforms. Thus, the dataset can include anomalous and redundant occurrences, which may affect detection accuracy. Thus, these records must be removed from the dataset before applying NIDS. For instance, the attribute 'Fwd Header Length' is included two times in the CIC-IDS2017 dataset, and 'Flow Packets/s' contains abnormal values such as 'Infinity' and 'NaN'. Additionally, missing values are replaced with zeroes, and features with constant values are removed, as they do not aid in the classification process.
- Data transforming and scaling: The two datasets contain symbolic, continuous, and binary values. For instance, the 'protocol type' feature in the NSL-KDD datasets enclosed symbolic values like 'tcp', 'udp', and 'icmp'. Given that ML models consider only numerical inputs, the transforming process becomes essential and influences the performance of NIDS. This paper applied a one-hot encoder transformation to all features except for the target variable, which underwent a label encoder transformation. For the CIC-IDS2017 dataset, the variable 'Flow Duration' contains big numeric values comparable to other features. This difference in range may give more weight to the 'Flow Duration' feature and affect the performance of the classifier's model. For this reason, a scaling algorithm should be applied. In this study, a standard scaler is used.

In addition, the target variable in the NSL-KDD dataset is regrouped into five main classes, as illustrated in Figure 4. However, the attack types in the CIC-IDS2017 dataset are mapped into six labels: DoS Hulk, DoS GoldenEye, DoS slowloris, DoS Slowhttptest, Heartbleed 11, and normal.

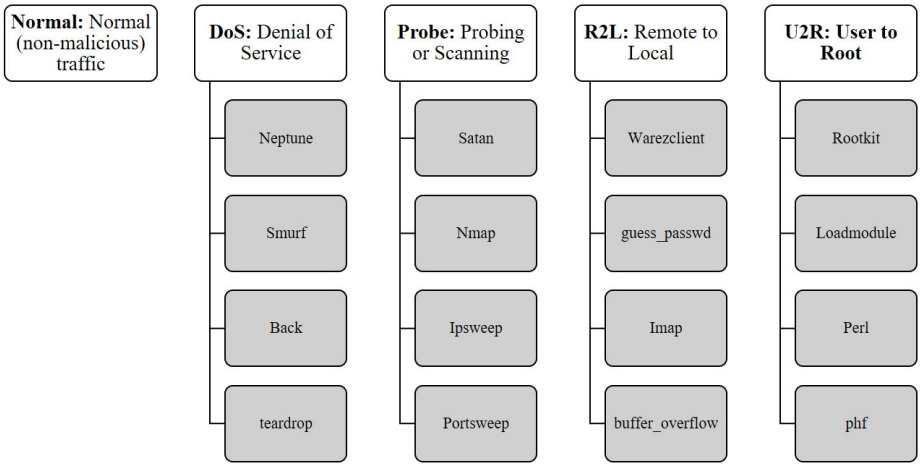


Figure 4. Target classes in the NSL-KDD dataset (main categories in the white rectangles and subcategories in the grey ones).

4.3. Results and Discussion

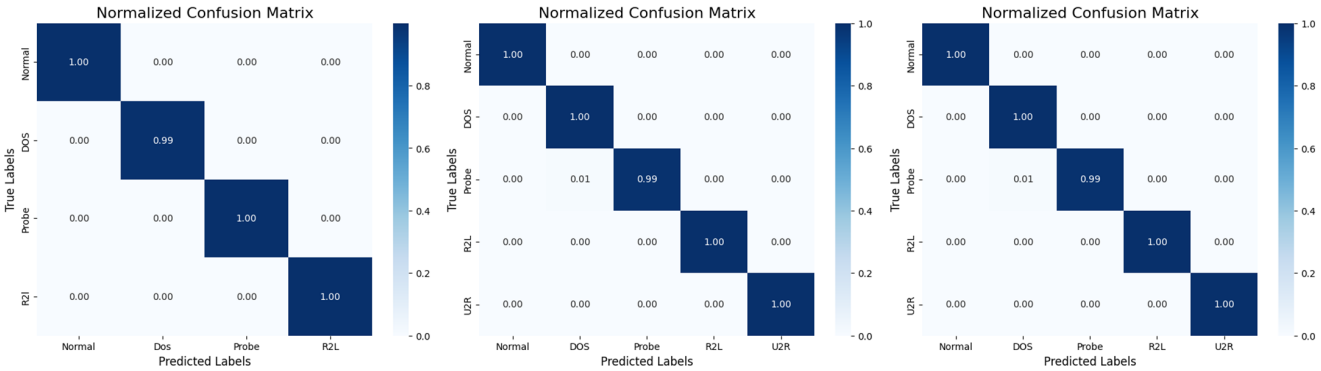
The efficacy of an intrusion detection system (IDS) is assessed by its ability to accurately classify network traffic. To mitigate the impact of data sampling for evaluating the IDS, we employed a repeated k-fold cross-validation procedure, with k set to 10. This study reports performance findings as the average output from 10 iterations of the 10-fold validation approach, with each experiment performed using different seeds to prevent biased outcomes. Specifically, for each dataset, we present the confusion matrix obtained from the testing of the proposed ensemble technique and compare it against several state-of-the-art methods using various detection metrics, including accuracy, precision, recall, and F1 score. We also present the results highlighting the importance of the feature selection step in the proposed technique. Initially, critical features are discerned by employing the suggested CFS with the integrated XGBoost method to assess the validity of the diminished feature subset during the feature selection phase. Subsequently, candidate characteristics are chosen from the original set for the subsequent phase. After applying the FS technique, only 10 features will be selected in the NSL_KDD and the CIC-IDS2017 datasets.

4.3.1. NSL-KDD Dataset Results

Figure 5 illustrates a normalized confusion matrix for NSL-KDD (KDDTrain+, KDDTest+, and KDDTest-21datasets). As illustrated in this figure, the proposed NIDS approach succeeds in detecting and classifying the attack types. Table 2 shows a comparison between the performance of the ML models and the proposed stacking learning techniques with the suggested FS techniques, correlation-based FS, embedded-based FS, and without any feature selection technique. Four cases are considered: (a) with the proposed mixed CFS and RF-based embedded FS technique, (b) without FS techniques, (c) with only CFS, and (d) with only the XGBoost-based embedded FS technique. First, without using any feature selection techniques, individual ML models and the proposed stacking model take all 48 features as inputs. As illustrated in part (b) of the table, KNN and DT show the lowest performance with accuracies of 0.7009 and 0.701. These models show limitations in terms of precision and recall (F1 score values are 0.70 and 0.71). Thus, these models have a lower capability to provide reliable predictions. However, RF and XGBoost outperform DT and KNN and provide accurate values of 0.89 and 0.93, with F1 scores equal to 0.884 and 0.92. XGBoost provides the highest precision, with a value equal to 0.93, showing that boosting techniques can handle more complex problems better. The proposed stacking learning approach, which combines the outputs of the individual ML models, achieved the highest accuracy with a value equal to 0.958. It also achieves good precision (0.969) and recall (0.964). Thus, ensemble learning, like stacking, can exceed individual learning methods and offer robust and accurate predictions.

In the suggested approach, all the individual ML models take as input the output of CFS and embedded FS techniques. As shown in Table 3, part (a), KNN and DT models have accuracy values of 0.87 and 0.82, which demonstrate that they have difficulty in identifying the positive cases. In addition, DT shows the lowest accuracy of all the individual models. RF has the highest performance with accuracy, precision, recall, and an F1 score of 0.992. This value reveals its robustness. XGBoost outperformed the RF with an accuracy of 0.992 and a notable F1 score of 0.991. These values demonstrate the capability of boosting techniques in handling complex scenarios. The proposed stacking approach exceeds all the individual models and achieves an accuracy of 0.9992. This result illustrates the effectiveness of combining these ML models to reproduce their strengths. In addition, it shows the efficacy of combining CFS and embedded XGBoost-based techniques to select the most suitable subsets. Part (b) shows the results without any FS technique. It illustrates that the stacking approach outperforms the basic ML models with a higher accuracy of

0.958 and an F1 score value of 0.967. Between all single models, XGBoost produces the highest performance with an accuracy of 0.93 and a precision of 0.93. The RF model works well, with a strong accuracy of 0.89.



(**a**) (**b**) (**c**)

Figure 5. Normalized confusion matrix for NSL-KDD parts: (**a**) train+, (**b**) Test+, and (**c**) Test-21.

Table 3. Comparison between performance metrics of ML models and the proposed stacking approach for the NSL-KDD dataset.

(a) Result with the suggested FS technique				
ML model	Accuracy	Precision	Recall	F1 score
KNN	0.87	0.85	0.78	0.81
DT	0.82	0.85	0.79	0.8
RF	0.991	0.994	0.99	0.992
XGBoost	0.992	0.992	0.99	0.991
Stacking approach	0.9992	0.9982	0.9984	0.9983
(b) Results without any FS technique				
ML model	Accuracy	Precision	Recall	F1 score
KNN	0.7009	0.69	0.70	0.70
DT	0.701	0.721	0.79	0.71
RF	0.89	0.870	0.891	0.884
XGBoost	0.93	0.93	0.89	0.92
Stacking approach	0.958	0.969	0.964	0.967
(c) Result with CFS				
ML model	Accuracy	Precision	Recall	F1 score
KNN	0.7701	0.7421	0.7752	0.7591
DT	0.79	0.804	0.79	0.799
RF	0.91	0.894	0.90	0.897
XGBoost	0.92	0.93	0.89	0.92
Stacking approach	0.978	0.978	0.981	0.98
(d) Result with embedded FS technique				
ML model	Accuracy	Precision	Recall	F1 score
KNN	0.831	0.84	0.8241	0.834
DT	0.809	0.811	0.809	0.81
RF	0.967	0.974	0.961	0.967
XGBoost	0.981	0.983	0.989	0.987
Stacking approach	0.991	0.994	0.991	0.993

Part (c) of Table 3 illustrates the performance of the correlation-based feature selection technique. Indeed, these results demonstrate a significant positive impact of the CFS on the performance metrics of all the models. The accuracy of the KNN model increased to

achieve a value of 0.7701. Also, DT with CFS achieved an accuracy of 0.79, with an increase in all other metrics, such as precision, recall, and F1 score. This modification indicates an improvement in prediction consistency. RF and XGBoost denote substantial gains with the use of CFS. They achieved accurate values of 0.91 for RF and 0.92 for XGBoost. The accuracy of the suggested approach achieves a value of 0.978 with a precision of 0.78 and a recall of 0.981. These results indicate the power of stacking multiple ML models.

In part (d), Table 3 illustrates the performance of the individual ML models and the suggested approach by using an embedded FS technique based on the XGBoost model. The proposed FS technique enhances the performance of the prediction of all the ML models. The KNN model indicates a notable improvement and produces an accuracy of 0.831. In addition, the performance of the DT model is enhanced and achieves an accuracy of 0.809. The RF and XGBoost also demonstrate significant enhancement and reach an accuracy of 0.967 and 0.987. This enhancement shows the importance of selecting the most suitable feature subsets. The proposed stacking learning approach outperforms all individual ML models and produces the best accuracy with a value of 0.991. Thus, the stacking method profits significantly from embedded FS and mixes the strengths of the individual models.

Figure 6 illustrates a comparison of the Receiver Operating Characteristic (ROC) for KKN, DT, RF, XGBoost, and the proposed stacking method. The ROC curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR), illustrating the trade-off between sensitivity and specificity. It shows that the stacking approach obtains a perfect area under the curve (AUC) of 1.00, demonstrating optimal classification. However, RF and XGBoost achieved high AUC values of 0.99, representing an excellent performance. The KNN and DT achieved similar performances with an AUC of 0.85. The results demonstrate that ensemble learning models such as RF, XGBoost, and stacking outperform basic classifiers. This result emphasizes the idea that the effectiveness of combining multiple models enhances classification accuracy.

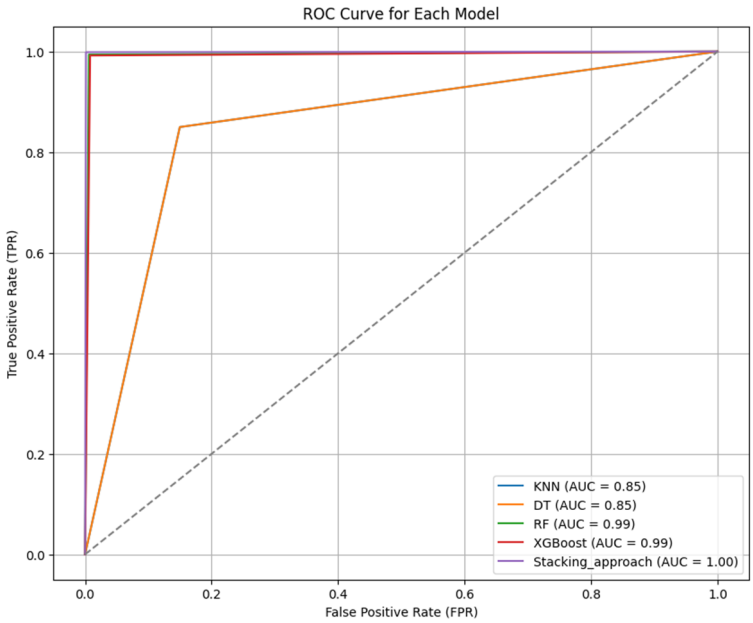


Figure 6. ROC curve for the NSL-KDD dataset.

4.3.2. CIC-IDS Dataset Results

Table 4 shows the binary confusion matrices for the individual ML models on the CIC-IDS2017 dataset. These ML models have as inputs the feature subset resulting from the CFS-embedded proposed techniques. From these matrices, other performance metrics can be concluded (see Table 5). Figure 7 shows the confusion matrix of applying the

suggested approach to the CIC-IDS2017 dataset. This figure shows that the proposed approach produces high model performance with an almost perfect classification of attack types: benign, DoS slowloris, DoS Hulk, and DoS GoldenEye. In addition, it produces high accuracy in detecting the attack type "Heartbleed", even if the number of its occurrences is very low, as illustrated in Figure 3b.

Table 4. Binary confusion matrix for KNN, RF, DT, and XGBoost.

			Predicted Label	
	Model	**True Label**	**Non-Attack**	**Attack**
True Label	**DT**	**Non-Attack**	209,704	109
		Attack	349	208,849
	RF	**Non-Attack**	209,803	10
		Attack	244	208,954
	KNN	**Non-Attack**	209,005	808
		Attack	916	208,282
	XGB	**Non-Attack**	209,111	702
		Attack	761	208,437

Table 5. Comparison between performance metrics of ML models and proposed stacking approach for the CIC-IDS2017 dataset.

(a) Results with the suggested FS technique				
ML model	Accuracy	Precision	Recall	F1 score
KNN	0.8646	0.8646	0.8618	0.8641
DT	0.9607	0.9601	0.9601	0.9606
RF	0.9949	0.9955	0.9942	0.9949
XGBoost	0.9965	0.9964	0.9966	0.9965
Stacking approach	0.9997	0.9992	0.9994	0.9993
(b) Results without any FS technique				
ML model	Accuracy	Precision	Recall	F1 score
KNN	0.7340	0.7345	0.7340	0.7347
DT	0.7431	0.7435	0.7434	0.7432
RF	0.7004	0.6980	0.7012	0.6996
XGBoost	0.8062	0.8060	0.8060	0.8060
Stacking approach	0.8521	0.8702	0.8253	0.8509
(c) Results with CFS				
ML model	Accuracy	Precision	Recall	F1 score
KNN	0.8647	0.8641	0.8647	0.8644
DT	0.8954	0.8959	0.8947	0.8953
RF	0.8772	0.8756	0.8791	0.8771
XGBoost	0.8969	0.8967	0.8967	0.8967
Stacking approach	0.928	0.929	0.924	0.927
(d) Results with embedded FS technique				
ML model	Accuracy	Precision	Recall	F1 score
KNN	0.888	0.869	0.8624	0.8466
DT	0.911	0.929	0.911	0.915
RF	0.929	0.929	0.924	0.927
XGBoost	0.941	0.932	0.944	0.937
Stacking approach	0.948	0.942	0.964	0.955

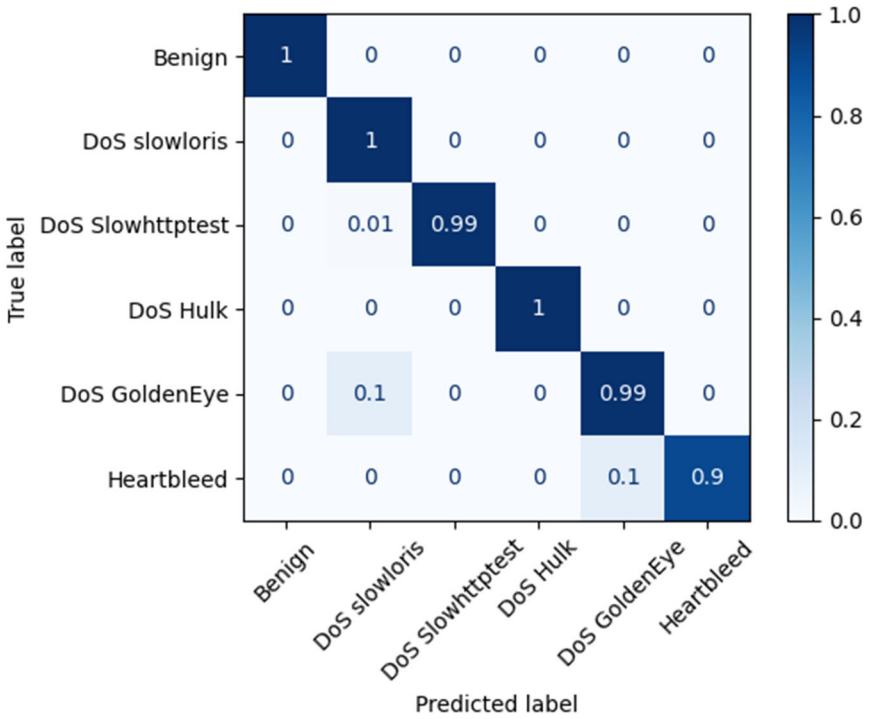


Figure 7. Multi-class CM for the proposed stacking learning approach on the CIC-IDS2017 dataset.

In Table 5, the performance of the proposed stacking method and the individual ML models with and without FS techniques is illustrated. In part (b), suggested stacking learning shows the highest performance metrics with an accuracy of 0.8521 and lower values of precision and recall. As shown in part (c), when the CFS is used for selecting the suitable feature subsets, the performance metrics become higher with maximum accuracy for the suggested approach (0.928). The individual ML models produce an accuracy score of 0.8969 for XGBoost, 0.8677 for the RF, 0.8647 for the KNN, and 0.8954 for DT. In part (d), with XGBoost-based embedded techniques, the stacking approach produces the highest performance with 0.948 accuracy, 0.942 precision, 0.964 recall, and 0.955 F1 score. XGBoost also shows enhancement with an accuracy of 0.941 and an F1 score of 0.937. The RF and DT models also achieve better results, with RF achieving 0.929 accuracy and DT at 0.911. The KNN accuracy of 0.928 and F1 score of 0.926 shows significant enhancement. The proposed approach in (a) produces the best performance in accuracy (0.9992), precision (0.9992), recall (0.9994), and F1 score (0.9993). XGBoost achieves close performance with an accuracy of 0.9965 and an F1 score of 0.9965. In addition, random forest (RF) and decision tree (DT) produce robust performance, with an accuracy of 0.9949 for RF and 0.9607 for DT. These results point out the effectiveness of using stacking techniques in these kinds of problems. Also, they show that combining the CFS and embedded technique for feature selection leads to an appropriate feature subset.

Figure 8 represents a comparison of the ROC curve for the different ML models and the stack approach for the CIC-IDS2017 dataset. The proposed stacking ensemble approach with RF and XGBoost reached an AUC of 1.00, which shows a perfect classification. The DT model comes behind with a close AUC of 0.96, while the KNN model shows the lowest performance with an AUC of 0.86. These results indicate that the proposed stacking techniques significantly enhance the classification accuracy for the CIC-IDS2017 dataset.

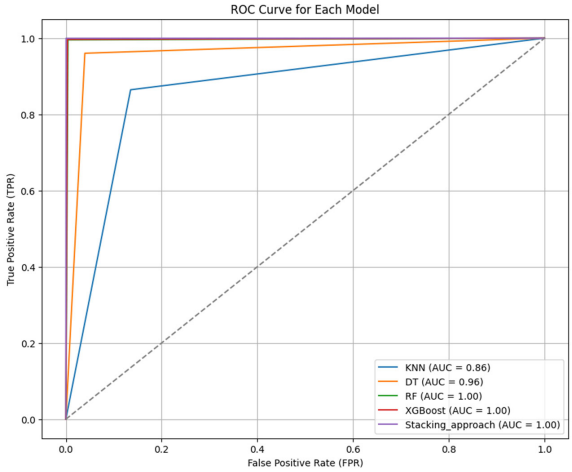


Figure 8. ROC curve for CIC-IDS 2017 dataset for RF, XGBoost, KNN, DT, and stacking proposed approach.

4.3.3. Comparison and Discussion

Tables 6 and 7 show a comparison of the proposed approach with previous research results on NSL-KDD and CIC-IDS2017 datasets.

Table 6. Comparison between the proposed approach and existing study for the NSL-KDD dataset.

Study	Model	Data Balancing	Feature Selection	Test Accuracy
[39]	MCA-LSTM	-	IG	94.12
[20]	LSTM	-	XGBoost	91.57
[40]	Voting approach	SMOTE	CFS-BA	99.8
[41]	LSTM	Extended Synthetic Sampling	PCA	99.65
[42]	BO-KNN-Bagging Ensemble	-	Extremely Randomized Trees	82.48
[43]	Stacking (RF, XGBoost, Extra Tree), LR as meta model	-	RFE	99.95
[44]	XGBoost	SMOTE	-	99.72
-	Proposed Approach(Stacking)	SMOTE-TOMEK	CFS with embedded FS based on RF	99.99

Table 7. Comparison between the proposed approach and existing studies for the CIC-IDS2017 dataset.

Study	Data Balancing	Dimension Reduction	Algorithm	Selected Features	Accuracy (%)
[45]	IGR+CR +ReF	PART	–	–	99.95 (Binary)
[46]	–	t-SNE	RF	–	99.78
[47]	–	EDFS	DT	–	98.80
[48]	IG+Ranking +Grouping	RF	-	–	99.86
[48]	IG+Ranking +Grouping	J48	-	–	99.87
[49]	–	–	DNN+ACO	–	98.25
[50]	SMOTE-ENN	-	Deep Learning (DL)-Based Ensemble Framework		99.6
[43]	-	RFE	Stacking (RF, XGBoost, Extra Tree), LR as meta model	-	99.99
[51]	SMOTE with space reduction based on feature correlation index	Correlation Features selection (Pearson, Spearman, Kendall)	Soft Voting (RF, Gradient Boost, Extra Tree, MLP)	-	97.3
[52]	SGM	–	CNN	–	99.85
Our Proposal	SMOTE-TOMEK	CFS with embedded techniques based on RF	Stacking	10	99.99

Table 6 shows a comparison of different IDS approaches. It illustrates the associated data balancing and the FS techniques. As illustrated in this table, the proposed approach outperforms the existing study with an accuracy of 99.99%. The voting approach proposed in [41] shows a closed performance with an accuracy of 99.8%. The voting approach relies on CFS and employs the Bat algorithm to identify the optimal feature subset. BA technique is a heuristic search that may examine the overall feature set. It is often based on local search tactics that may lead to local optima. Hence, it may miss superior feature subsets. In [43], the authors proposed an XGBoost approach that uses SMOTE for data balancing. This approach gives a robust performance with an accuracy of 99.72%. However, using SMOTE techniques may increase complexity and lead to an overfitting situation. Furthermore, the LSTM in [42] gives the highest performance with an accuracy of 99.65%. It is based on extended synthetic sampling for data balancing and PCA for feature selection. However, the increased computational cost is highly dependent on the data distribution and may increase the complexity and lead to an overfitting situation. As a conclusion, this table shows the effectiveness of using SMOTE-TOMEK for handling the data imbalanced situation and the efficacy of using embedded FS techniques with the CFS. The BO-KNN-Bagging [43] Ensemble employed a bagging ensemble approach combined with Bayesian Optimization (BO) to enhance feature selection. This method proved its robustness. However, it is required to enhance accuracy. The stacking (RF, XGBoost, Extra Tree) combined individual models but used LR as a meta-model [44]. The model uses the RFE feature selection process that assists in reducing the dimensionality and eliminating irrelevant features. It is a robust model with 99.95% accuracy. However, RFE necessitates training the model many times, each time with a reduced feature set. This process is computationally expensive, especially with huge datasets and complex models such as the NIDS case. In contrast, embedded methods typically require fewer iterations because they perform feature selection during the training of the model itself.

Table 7 shows a comparison of the proposed stacking method with existing studies on the CIC-IDS2017 dataset. Techniques for handling the data balancing challenge and FS issue are the basis for this comparison. This table shows that the proposed approach provides the highest accuracy with a value of 99.99%. Other studies like [44] and [47] achieve closed accuracy values (99.86% and 99.95%). In [44], t-Distributed Stochastic Neighbor Embedding (t-SNE) is used. It is highly dependent on the parameter and may lead to an overfitting situation. The IG+Ranking+Grouping is used in [47]. It is highly sensitive to feature redundancy and cannot handle complex iterations. In addition, the approach proposed in [49] leads to a high accuracy of 99.85%. It is based on the Stochastic Gradient Descent (SGM) technique that is not able to handle complex situations and is very sensitive to noise. In summary, this table proves the effectiveness of the proposed approach and that it is suitable for the NID problem. The soft voting approach that combines the outcomes of multiple classifiers (random forest (RF), Gradient Boosting, Extra Trees, and Multi-Layer Perceptron (MLP)) to deliver the final prediction achieved an accuracy of 97.3%. These results suggest that mixing these models under a voting system produces accurate results. However, soft voting simply averages the predictions without considering the underlying model strengths, which can reduce its performance in complex situations such as the NIDS case. The stacking technique exploits a meta-model to combine the predictions of base models more accurately. This process permits the ensemble model to learn optimal weights and predict the relationships between basic models.

5. Conclusions

The use of smart devices has increased significantly due to recent huge developments in many sectors. Thus, the number of network users has increased, and this leads to an

expansion in network size and databases stored in them. Hence, the network growth produced a rise in policy violations that had a significant impact on information security. With this huge growth, the need for a system that can detect inappropriate intrusion in the network becomes an essential need. In this paper, an ensemble learning approach for detecting network intrusion was proposed. Thus, an innovative feature selection technique that combines the embedded method with the correlation-based feature selection techniques was proposed. This advanced technique may improve model efficiency by decreasing computation time and enhancing detection accuracy. A hybrid strategy that includes data augmentation techniques, undersampling, and oversampling was used to address the issue of an imbalanced dataset. The suggested strategy got better at classifying by using an ensemble approach that combines the decisions of several classifiers into a single decision using the stacking technique. The NSL-KDD and CIC-ISD 2017 datasets are used to test the performance of the proposed approach. Performance techniques such as accuracy, F1 score, recall, and precision are used to evaluate the efficacy of this approach. For the NSL-KDD dataset, the proposed stacking learning strategy that blended the outputs of the multiple ML models produced high accuracy with a value equal to 0.9992. It also has a high precision (0.9982) and recall (0.9984). As a result, ensemble learning, such as stacking, has the potential to outperform individual learning methods and provide predictions that are both accurate and reliable. For the CIC-IDS2017 dataset, the suggested approach also offers a high accuracy of 99.99%. It outperformed the existing approaches for NIDS. These outstanding results are due to the use of CFS with an XGBoost-embedded-based technique to select the most suitable features from the overall feature set. In addition, SMOTE-TOMEK was used to resolve the imbalanced dataset issue. This technique reduces noise and improves decision boundaries. Furthermore, the tacking ensemble learning can combine multiple models' strengths to enhance predictive performance. It is also flexible in merging various ML models to capture different data patterns. The use of such techniques usually offers better and more robust solutions than individual models.

In future work, the proposed approach will be tested on more recent datasets to assess its generalization capabilities across different domains. Additionally, explainable AI (XAI) techniques will be integrated to provide better interpretability of the model's predictions and decision-making process. This will help improve the transparency of the model and enhance trust in its outputs for real-world applications.

Author Contributions: Conceptualization, G.N. and O.A.-K.; methodology, G.N. and O.A.-K.; software, G.N.; validation, G.N., O.A.-K. and M.N.; formal analysis, O.A.-K.; investigation, O.A.-K.; resources, M.N.; data curation, O.A.-K.; writing—original draft preparation, G.N.; writing—review and editing, O.A.-K. and M.N.; visualization, O.A.-K.; supervision, M.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new data were created in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ML	Machine Learning
FS	Feature Selection
FL	Federal Learning
RF	Random Forest

DT	Decision Tree
CFS	Correlation-based Feature Selection
SMOTE	Synthetic Minority Over-sampling Technique
CM	Confusion Matrix
LR	Logistic Regression
KNN	K nearest Neighbors
XGBoost	extreme Gradient Boosting
SGM	Stochastic Gradient Descent
t-SNE	t-Distributed Stochastic Neighbor Embedding
IG	Information Gain
BA	Bat Algorithm
CNN	Convolutional Neural Network
EDFS	Ensemble Data Fusion Strategy
LSTM	Long Short-Term Memory
PCA	Principal Component Analysis
MCA-LSTM	Multivariate Conditional Autoencoder with Long Short-Term Memory
NDAE	Nonsymmetric Deep AutoEncoder
ENN	Edited Nearest Neighbor
RFE	Recursive Feature Elimination
MI	Mutual Inclusion
LFS	Lasso Feature Selection
AUC	Area under the curve
ROC	Receiver Operating Characteristic

References

1. Coker, J. Top 10 Cyber-Attacks of 2023. Available online: https://www.infosecurity-magazine.com/news-features/top-cyber-attacks-2023/ (accessed on 3 February 2025).
2. Governance, I. List of Data Breaches and Cyber Attacks in 2023—8,214,886,660 Records Breached. Available online: https://www.itgovernance.co.uk/blog/list-of-data-breaches-and-cyber-attacks-in-2023? (accessed on 3 February 2025).
3. Yang, Y.; Zheng, K.; Wu, B.; Yang, Y.; Wang, X. Network Intrusion Detection Based on Supervised Adversarial Variational Auto-Encoder with Regularization. *IEEE Access* **2020**, *8*, 42169–42184. [CrossRef]
4. Rahim, K.; Nasir, Z.U.I.; Ikram, N.; Qureshi, H.K. Integrating contextual intelligence with mixture of experts for signature and anomaly-based intrusion detection in CPS security. *Neural Comput. Appl.* **2025**, *7*, 1–7. [CrossRef]
5. Zhang, A.; Zhao, Y.; Zhou, C.; Zhang, T. ResACAG: A graph neural network based intrusion detection. *Comput. Electr. Eng.* **2025**, *122*, 109956. [CrossRef]
6. Chen, F.; Ye, Z.; Wang, C.; Yan, L.; Wang, R. A Feature Selection Approach for Network Intrusion Detection Based on Tree-Seed Algorithm and K-Nearest Neighbor. In Proceedings of the 2018 IEEE 4th International Symposium on Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS), Lviv, Ukraine, 20–21 September 2018; pp. 68–72. [CrossRef]
7. Ahmad, I.; Basheri, M.; Iqbal, M.J.; Rahim, A. Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection. *IEEE Access* **2018**, *6*, 33789–33795. [CrossRef]
8. Zhang, C.; Jia, D.; Wang, L.; Wang, W.; Liu, F.; Yang, A. Comparative research on network intrusion detection methods based on machine learning. *Comput. Secur.* **2022**, *121*, 102861. [CrossRef]
9. Magán-Carrión, R.; Urda, D.; Díaz-Cano, I.; Dorronsoro, B. Towards a Reliable Comparison and Evaluation of Network Intrusion Detection Systems Based on Machine Learning Approaches. *Appl. Sci.* **2020**, *10*, 1775. [CrossRef]
10. Wang, S.; Yao, X. Multiclass Imbalance Problems: Analysis and Potential Solutions. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2012**, *42*, 1119–1130. [CrossRef]
11. Mbow, M.; Koide, H.; Sakurai, K. An Intrusion Detection System for Imbalanced Dataset Based on Deep Learning. In Proceedings of the 2021 Ninth International Symposium on Computing and Networking (CANDAR), Matsue, Japan, 23–26 November 2021; pp. 38–47. [CrossRef]
12. Rao, Y.N.; Babu, K.S. An Imbalanced Generative Adversarial Network-Based Approach for Network Intrusion Detection in an Imbalanced Dataset. *Sensors* **2023**, *23*, 550. [CrossRef]
13. Alkasassbeh, M.; Almseidin, M. Machine Learning Methods for Network Intrusion Detection. *arXiv* **2018**, arXiv:1809.02610. [CrossRef]

14. Shone, N.; Ngoc, T.N.; Phai, V.D.; Shi, Q. A Deep Learning Approach to Network Intrusion Detection. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 41–50. [CrossRef]
15. Alavizadeh, H.; Alavizadeh, H.; Jang-Jaccard, J. Deep Q-Learning Based Reinforcement Learning Approach for Network Intrusion Detection. *Computers* **2022**, *11*, 41. [CrossRef]
16. Tang, Z.; Hu, H.; Xu, C. A federated learning method for network intrusion detection. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6812. [CrossRef]
17. Ansari, M.S.; Bartoš, V.; Lee, B. GRU-based deep learning approach for network intrusion alert prediction. *Future Gener. Comput. Syst.* **2022**, *128*, 235–247. [CrossRef]
18. Talukder, M.A.; Islam, M.M.; Uddin, M.A.; Hasan, K.F.; Sharmin, S.; Alyami, S.A.; Moni, M.A. Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction. *J. Big Data* **2024**, *11*, 33. [CrossRef]
19. Turukmane, A.V.; Devendiran, R. M-MultiSVM: An efficient feature selection assisted network intrusion detection system using machine learning. *Comput. Secur.* **2024**, *137*, 103587. [CrossRef]
20. Altulaihan, E.; Almaiah, M.A.; Aljughaiman, A. Anomaly Detection IDS for Detecting DoS Attacks in IoT Networks Based on Machine Learning Algorithms. *Sensors* **2024**, *24*, 713. [CrossRef]
21. Farfoura, M.E.; Mashal, I.; Alkhatib, A.; Batyha, R.M. A lightweight machine learning methods for malware classification. *Clust. Comput.* **2025**, *28*, 1. [CrossRef]
22. Zhou, W.; Xia, C.; Wang, T.; Liang, X.; Lin, W.; Li, X.; Zhang, S. HIDIM: A novel framework of network intrusion detection for hierarchical dependency and class imbalance. *Comput. Secur.* **2025**, *148*, 104155. [CrossRef]
23. Olanrewaju-George, B.; Pranggono, B. Federated learning-based intrusion detection system for the internet of things using unsupervised and supervised deep learning models. *Cyber Secur. Appl.* **2025**, *3*, 100068. [CrossRef]
24. Rodríguez, P.; Bautista, M.A.; Gonzàlez, J.; Escalera, S. Beyond one-hot encoding: Lower dimensional target embedding. *Image Vis. Comput.* **2018**, *75*, 21–31. [CrossRef]
25. Hajisalem, V.; Babaie, S. A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection. *Comput. Netw.* **2018**, *136*, 37–50. [CrossRef]
26. Hota, H.S.; Shrivas, A.K. Decision Tree Techniques Applied on NSL-KDD Data and Its Comparison with Various Feature Selection Techniques. In *Advanced Computing, Networking and Informatics–Volume 1*; Kumar Kundu, M., Mohapatra, D.P., Konar, A., Chakraborty, A., Eds.; Smart Innovation, Systems and Technologies; Springer International Publishing: Cham, Switzerland, 2014; Volume 27, pp. 205–211. [CrossRef]
27. Singh, S.; Singh, A.K. Web-Spam Features Selection Using CFS-PSO. *Procedia Comput. Sci.* **2018**, *125*, 568–575. [CrossRef]
28. Chandra, W.; Suprihatin, B.; Resti, Y. Median-KNN Regressor-SMOTE-Tomek Links for Handling Missing and Imbalanced Data in Air Quality Prediction. *Symmetry* **2023**, *15*, 887. [CrossRef]
29. Wei, J.; Chu, X.; Sun, X.Y.; Xu, K.; Deng, H.X.; Chen, J.; Wei, Z.; Lei, M. Machine learning in materials science. *InfoMat* **2019**, *1*, 338–358. [CrossRef]
30. Almotairi, A.; Atawneh, S.; Khashan, O.A.; Khafajah, N.M. Enhancing intrusion detection in IoT networks using machine learning-based feature selection and ensemble models. *Syst. Sci. Control Eng.* **2024**, *12*, 2321381. [CrossRef]
31. Azam, Z.; Islam, M.M.; Huda, M.N. Comparative Analysis of Intrusion Detection Systems and Machine Learning-Based Model Analysis Through Decision Tree. *IEEE Access* **2023**, *11*, 80348–80391. [CrossRef]
32. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
33. Feng, Q.; Liu, J.; Gong, J. UAV Remote Sensing for Urban Vegetation Mapping Using Random Forest and Texture Analysis. *Remote Sens.* **2015**, *7*, 1074–1094. [CrossRef]
34. Brownlee, J. A Gentle Introduction to XGBoost for Applied Machine Learning. Available online: https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/ (accessed on 3 February 2025).
35. Rincy, T.N.; Gupta, R. Ensemble Learning Techniques and its Efficiency in Machine Learning: A Survey. In Proceedings of the 2nd International Conference on Data, Engineering and Applications (IDEA), Bhopal, India, 1–6 February 2020. [CrossRef]
36. Aldwairi, T.; Perera, D.; Novotny, M.A. An evaluation of the performance of Restricted Boltzmann Machines as a model for anomaly network intrusion detection. *Comput. Netw.* **2018**, *144*, 111–119. [CrossRef]
37. Shi, Z.; Li, J.; Wu, C.; Li, J. DeepWindow: An Efficient Method for Online Network Traffic Anomaly Detection. In Proceedings of the 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Zhangjiajie, China, 10–12 August 2019; pp. 2403–2408. [CrossRef]
38. Nasserddine, G.; El Arid, A.A. Decision-Making Systems. In *Encyclopedia of Data Science and Machine Learning*; Wang, J., Ed.; IGI Global: Hershey, PA, USA, 2022; pp. 1391–1407. [CrossRef]
39. Dong, R.; Li, X.; Zhang, Q.; Yuan, H. Network intrusion detection model based on multivariate correlation analysis—Long short-time memory network. *IET Inf. Secur.* **2020**, *14*, 166–174. [CrossRef]

40. Zhou, Y.; Cheng, G.; Jiang, S.; Dai, M. Building an efficient intrusion detection system based on feature selection and ensemble classifier. *Comput. Netw.* **2020**, *174*, 107247. [CrossRef]
41. Devendiran, R.; Turukmane, A.V. Dugat-LSTM: Deep learning based network intrusion detection system using chaotic optimization strategy. *Expert Syst. Appl.* **2024**, *245*, 123027. [CrossRef]
42. Hossain, M.A.; Islam, M.S. Ensuring network security with a robust intrusion detection system using ensemble-based machine learning. *Array* **2023**, *19*, 100306. [CrossRef]
43. Urmi, W.F.; Uddin, M.N.; Uddin, M.A.; Talukder, M.A.; Hasan, M.R.; Paul, S.; Chanda, M.; Ayoade, J.; Khraisat, A.; Hossen, R.; et al. A stacked ensemble approach to detect cyber attacks based on feature selection techniques. *Int. J. Cogn. Comput. Eng.* **2024**, *5*, 316–331. [CrossRef]
44. Thana-Aksaneekorn, C.; Kosolsombat, S.; Luangwiriya, T. Machine Learning Classification for Intrusion Detection Systems Using the NSL-KDD Dataset. In Proceedings of the 2024 IEEE International Conference on Cybernetics and Innovations (ICCI), Chonburi, Thailand, 1–6 March 2024. [CrossRef]
45. Kshirsagar, D.; Kumar, S. An efficient feature reduction method for the detection of DoS attack. *ICT Express* **2021**, *7*, 371–375. [CrossRef]
46. Hammad, M.; Hewahi, N.; Elmedany, W. T-SNERF: A novel high accuracy machine learning approach for Intrusion Detection Systems. *IET Inf. Secur.* **2021**, *15*, 178–190. [CrossRef]
47. Guezzaz, A.; Benkirane, S.; Azrour, M.; Khurram, S. A Reliable Network Intrusion Detection Approach Using Decision Tree with Enhanced Data Quality. *Secur. Commun. Netw.* **2021**, *2021*, 1–8. [CrossRef]
48. Kurniabudi; Stiawan, D.; Darmawijoyo; Idris, M.Y.B.; Bamhdi, A.M.; Budiarto, R. CICIDS-2017 Dataset Feature Analysis with Information Gain for Anomaly Detection. *IEEE Access* **2020**, *8*, 132911–132921. [CrossRef]
49. Bhardwaj, A.; Mangat, V.; Vig, R. Hybrid Deep Neural Architecture for Detection of DDoS Attacks in Cloud Computing. In *Intelligent Systems, Technologies and Applications*; Paprzycki, M., Thampi, S.M., Mitra, S., Trajkovic, L., El-Alfy, E.-S.M., Eds.; Advances in Intelligent Systems and Computing; Springer: Singapore, 2021; Volume 1353, pp. 71–86. [CrossRef]
50. Sayem, I.M.; Sayed, M.I.; Saha, S.; Haque, A. ENIDS: A Deep Learning-Based Ensemble Framework for Network Intrusion Detection Systems. *IEEE Trans. Netw. Serv. Manag.* **2024**, *21*, 5809–5825. [CrossRef]
51. Khan, M.A.; Iqbal, N.; Imran; Jamil, H.; Kim, D.-H. An optimized ensemble prediction model using AutoML based on soft voting classifier for network intrusion detection. *J. Netw. Comput. Appl.* **2023**, *212*, 103560. [CrossRef]
52. Zhang, H.; Huang, L.; Wu, C.Q.; Li, Z. An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset. *Comput. Netw.* **2020**, *177*, 107315. [CrossRef]