 ISSN:2302-8149
e-ISSN:2540-9719

Enhanced Network Intrusion Detection and Classification based on Ensemble Learning Techniques: A Study on the NSL-KDD Dataset

[1]Ammar Adel Ahmed*, [2]Mahmood Mohammed Mahmood,
[3]Omar Abdulmunem Ibrahim Aldabbagh
[1]Department of Computer Science, Education College for Pure Science, University of Mosul, Mosul, Iraq
[2]Department of Computer Science, College of Computer Science and Mathematics, University of Mosul, Mosul, Iraq
[3]Department of Networks, College of Computer Science and Mathematics, University of Mosul, Mosul, Iraq
*e-mail: ammaraladel@uomosul.edu.iq

(***received***: 11 April 2026, ***revised***: 3 May 2026, ***accepted***: 5 May 2026)

Abstract

This research explores an improved Network Intrusion Detection System (NIDS) on the NSL-KDD dataset using machine learning, deep learning and ensemble learning methods. Our approach involves essential steps such as data preparation, feature engineering with Random Forest, feature reduction, model building, hyperparameter tuning with GridSearchCV, and evaluation. We perform binary and multiclass classification tasks with Naïve Bayes, Logistic Regression, Random Forest, LightGBM, CNN, and LSTM approaches. The findings show ensemble techniques enhance classification accuracy. Random Forest and LightGBM models in binary classification, and CNN and LSTM models in multiclass classification achieved up to 99% and 97.99% and 97.80% accuracy, respectively. Additionally, the proposed stacked ensemble model, with XGBoost as the meta-learner, achieved a final test accuracy of 99.03%, and improved precision, recall, F1-score and ROC-AUC compared to the individual models. Tuning the hyperparameters also improved model stability and accuracy. This research is novel in combining feature selection, hyperparameter-tuned deep learning models and a stacking ensemble to enhance accuracy and stability in intrusion detection. The research also emphasizes the need for interpretability, real-time considerations and transfer learning in future NIDS research.

Keywords: GridSearchCV, hyper parameter tuning, machine learning, network intrusion detection system (NIDS), NSL-KDD dataset

1 Introduction

The Newer technological and network security requirements requiring constant research in the creation of Intrusion Detection Systems (IDS). Machine learning techniques are said to help in the development of anomaly-based IDS because they can capture the nature of peculiarity [1]. However, a major issue with all these techniques is that, they require that the probability distribution of the test set will be same as the training set probability distribution. In fact, especially in network traffic, both the nature of data and the distribution of data change their phases at the time of training and testing. The problem is aggravating by the constant appearance of new and diverse forms of attacks, which creates dataset shift within IDS [2].

This transition may result in the ML models performing inefficiently when tested; this due to the inductive biases that are inherent in the models. However, despite the need to address this problem, dataset shift as a problem has not received much attention in the context of IDS despite new attack forms being invented periodically [3]. Thus, it becomes necessary to employ methods to control cyber-attack variations as it's a non-stationary process. Selecting the right model, which includes the right kind of bias and the hypothesis space that is most suitable for estimating the deviations of attack distribution is not easy to do. Consequently, attack deviation patterns must be appropriately modelled instead of employing sets of methods dealing strictly with separate attack classes [4].

 ISSN:2302-8149
e-ISSN:2540-9719

Ensemble-learning as the process “learning to learn” was initially introduced in the field of educational sciences before it was adopted in the field of machine learning. The term "Ensemble-learning”. In the present-day context, Ensemble-learning is a popular branch of machine learning with a lot of advancements specifically in hyper parameter tuning, improvement of existing neural networks, and also in deciding the most appropriate architecture of the neural networks [5]. The ongoing research in Ensemble-learning can be broadly classified under three categories, namely, Model-based Ensemble-Learning, Metric based Ensemble-Learning, and Optimization based Ensemble Learning. In recent years, new Ensemble-learning models have emerged and, in cyberspace security, these could be categorized as; online-learning based, and stacked ensemble-based methods [6].

One of the major benefits associated with the use of Ensemble-learning to enhance IoT attack detection is that; it adapts to changes in attacks, as this is very fundamental in environments where new attacks keep on emerging, unlike in conventional rule-based systems. Ensemble-learning makes use of experience and information gathered from previous cyber-attack incidents to better detect new emerging threats that may possibly be within the IoT network and effectively work towards improving the overall security systems [6]. In addition, Ensemble-learning models are more flexible and can generalize more between tasks when compared to general ML models that possess rigid structures and always need to be retrained for a new task or a different dataset. In Ensemble-learning architectures, the structure and weights of the model can change to match different tasks, which makes them ideal for use when data distribution must evolve over time or is Staggered indefinitely [7].

The connectivity of devices to the World Wide Web has been on the rise at an extremely rapid pace. From two billion in 2006 to two hundred billion in the year 2020, often caused by mobile computing and IoT [8]. Because they provide cognitive services like inventory tracking, machine management, health monitoring, and anomaly detection, these devices are becoming indispensable in major industries like healthcare, manufacturing, retail, security, and transportation. By 2025, the manufacturing and healthcare sectors are expected to contribute 2.3 trillion and 2.5 trillion dollars, respectively, to the forecasted growth of the global IoT industry, which is expected to reach 6.2 trillion dollars. It is obvious that IoT is a major force behind the advancement of technology in daily life [9].

By identifying and averting network attacks, intrusion detection systems, or IDS, are frequently employed to strengthen security within IT infrastructures. Zero-day attack detection is especially successful with anomaly-based intrusion detection systems (IDS) [10]. Unexpected actions that jeopardize network availability, confidentiality, and integrity (the CIA trinity) are indicative of an intrusion13, 14. Packet header data seen in network traffic offer essential features for anomaly detection. An IDS's purpose is to identify and stop illegal activity, strengthening the CIA triad in the process [11].

In IoT intrusion detection systems (IDSs), machine learning (ML) approaches have been used more and more recently [12]. Despite differences in hardware, processing power, and feature creation skills, traditional intrusion detection systems (IDSs) frequently presume consistent feature patterns and packet types across Internet of Things devices [13]. This disparity causes data sparsity, which impairs data modeling's precision and effectiveness. To increase detection accuracy and speed up the training process, machine learning-based systems require careful feature selection (Shhatha, and Alsaif 2024). To improve the detection of abnormal behavior, a number of feature selection techniques have been proposed, including Flexible Mutual Information-based Feature Selection (FMIFS) [15], Modified Mutual Information-Based Feature Selection (MMIFS) with Support Vector Machine (SVM), and SVM with Neural Networks (NN) [16].

To enhance the application regarding the network security as well as its pliability against cyber threats, the effectiveness of the proposed framework in terms of scalability and practical implement ability will be validated through simulation analyses under IoT setting. It will also be tested in different scenarios to pave way for the possible application of the model in the actual Internet of Things projects [17]. The thesis initially begins by exploring the relevant risks regarding IoT networks, and the importance of advanced IDP systems. This makes a considerable workflow to the field. It then proposes a new intrusion detection framework that is profiled to be very sensitive to IoT environments’ needs and demands and is equally well-developed as previous frameworks [18]

 ISSN:2302-8149
e-ISSN:2540-9719

It presents a very realistic and, indeed, highly effective way of preventing threats that are directed specifically at IoT systems and objects. The research legitimates the proposed framework as effective, thereby validating the suggested framework dependability by means of empirical evaluation that used actual IoT datasets while outperforming prior approaches and demonstrating its relevance in practical usage. Additionally, by explaining how the framework is utilizable in a practical use-case scenario in an IoT context by simulating its functionality, a solid basis is laid for the framework's future use in real-world IoT implementations while providing a higher level of security and protection from cyber threats.

The given approach entails certain radical alterations to the IDS that make it stand out of the predecessors. One, it learns in a group of fashion to address new threats, and it can change the learning aspects to changes in the data distribution, hence can cope with new and unseen types of threats. The ensemble technique is the one that increases the accuracy of detection because the method involves numerous models that correspond to various attack patterns. It is also scalable to support large IoT, and it operates the massive data exchange that takes place in networks containing billions of devices that are connected to the network. The framework being proposed makes use of feature selection to identify the features of the network traffic that are the most informative, and thus the accuracy of the classification, as well as the complexity of the computations, which is imperative to IoT devices, is improved. The other strength of the method is its ability to resist shifts in the datasets, and thus, maximum accuracy can be reached when the attack patterns vary. It is less strict to different forms of attack consequently reducing false positives and false negatives unlike IDS. In addition, it can be detected with low latency in real-time, and thus identify and react to threats in the shortest time possible, which is important in IoT networks. Finally, but definitely not the least, the framework will be intended to be deployed in IoT networks, and therefore, the models are light-weighted and the features are chosen with consideration of not adding an additional burden to the network, in addition to the high-level security. The key points made in this paper are:

1. Creation of a multi-stage ensemble methodology that combines several machine learning (Random Forest, LightGBM, AdaBoost) with a meta-classifier (XGBoost) to improve the accuracy of intrusion detection.
2. Application of a Random Forest-based feature selection system to eliminate 40 features to 13 salient features to enhance the computational efficiency and yet maintaining high detection rates.
3. Hyperparameter optimization through GridSearchCV to optimize base models and ensemble classifier leading to better performance measures as opposed to non-tuned models.
4. A comparative study of various machine learning and deep learning systems (such as CNN, LSTM and conventional classifiers) on binary and multiclass classification problems giving clues on their comparative strength in intrusion detection.

2. Related Works

Researchers [19] came up with a framework called CFS-BA-Ensemble based on dimensionality reduction in the data utilized in intrusion detection systems (IDS). This method is based on the use of heuristic optimization methods and ensemble learning. Within the context of this framework, several machine learning classifiers, such as C4.5 and Random Forest, are used together to overcome one of the critical issues that the IDS models have: the existence of the large number of irrelevant or redundant features, which are represented by the network traffic data. The proposed framework enhances detection efficiency and lowers the computational complexity by using feature selection before classification. Three popular cybersecurity benchmarks datasets, namely NSL-KDD, CIC-IDS2017, and AWID were used as the basis of experimental evaluation. The performance was very high as the classification hits were 99.81, 99.52 and 99.89. The results of these findings suggest that the suggested approach can greatly improve the accuracy of detection, F1-score and the ability to identify an attack in comparison with other methods that have been designed in the past.

In an attempt to develop a deep learning-based system of network intrusion detection Authers [20] proposed a deep learning-based architecture of intrusion detection named BAT-MC in another research. It is a model that combines the Bidirectional Long Short-Term Memory (BLSTM) networks

 ISSN:2302-8149
e-ISSN:2540-9719

and an attention mechanism to enhance the ability to detect malicious activities in the network. At the first stage, the convolutional layers are used to handle the input data and derive significant packet-level features of raw network traffic. Following the extraction of the features, the BLSTM component relates the sequence of actions between the packets of the traffic in the NSL-KDD dataset, and the system is capable of extracting the temporal correlations of the traffic patterns. The attention mechanism also adds to the model as it gives more significance to the most significant components of the sequence giving the network a chance to concentrate on the crucial aspects related to malicious behaviors. The obtained experimental results indicated that this architecture was more effective compared to some traditional intrusion detection methods with an overall accuracy of 84.25, which showed the significance of deep learning in processing consecutive traffic.

Researchers [21] suggested an ensemble-based voting architecture that aims at detecting network misbehavior. Their approach is a combination of the predictions of a few machine learning algorithms, such as the Decision Trees, Naive Bayes, Random Forest, and repeatable neural networks, such as the RNN-LSTM. The model will enhance the reliability of classification and minimize the chances of misclassification by combining the results of several classifiers using a voting scheme. The offered strategy has been tested on the benchmark dataset provided by the NSL-KDD, and the findings were compared to those that were provided in the case of the traditional tree-based algorithms. The analysis revealed that ensemble voting method had better detection performance with an overall accuracy of around 85%. Alongside the given model, the authors proposed additional advances as the creation of a full-fledged rule repository that may aid the automated intrusion detection and provide more powerful capabilities to detect the threats in real-time.

Ensemble-classification model is used by [22] proposed an intrusion detection method applied on an ensemble classification framework and relies on a strategy of stacking. In this architecture, individual classifiers as well as ensemble methods are combined to enhance the strength of prediction process. The stacking model uses a set of base learners such as the Logistic Regression, the K-Nearest Neighbors, the Random Forest and the Support Vector Machine. All these algorithms learn independently by analyzing the network traffic data and their output is combined by a higher-level learner to come up with a more precise final classification. In order to prove that the suggested framework works, the authors provided experiments with two popular intrusion detection datasets UNSW-NB15 and UGR'16. The UNSW-NB15 data set is a human-generated simulated packet-based traffic with attack situations of current interest, whereas the UGR'16 data set is composed of actual network flow data obtained in the real network conditions of the operation. The experimental findings showed that the suggested stacking-based ensemble had high predictive accuracy, as it was able to predict at around 97 percent with the UGR16 dataset and 94 percent with the UNSW-NB15 dataset. These results demonstrate the potential of ensemble learning techniques to improve the accuracy of detection and to increase the reliability of the intrusion detection systems.

The model suggested by [23] also developed an innovative model of intrusion detection that is particularly relevant to Internet of Things (IoT) in a separate study. The proposed framework is based on a three-stage processing pipeline. During the initial step, the data is clustered and the dimensions reduced with the use of k-means++ algorithm that helps to reveal the hidden patterns in the data and eliminate unnecessary attributes. The second step is concerned with reducing the problem of the class imbalance by utilizing SVM-SMOTE that creates artificial minority samples to even out the distribution of the data-set. The data after processing is finally classified with the help of Single Layer Feedforward Network. The integration of these two types of learning unsupervised and supervised makes the proposed model efficient and beneficial in terms of improving the quality of the input information and making the classification performance overall more efficient. As shown in the experimental testing, the system recorded an accuracy rate of 93.51 percent with a ratio parameter of 0.9, which implies that the system can be utilized in the detection of cyber threats in the IoT network.

Researchers [24] explored the use of deep learning in the intrusion detection of IoT based systems. They used the IoTID20 dataset in their study to test a number of neural network architectures. In particular, they used three models, namely, Convolutional Neural Network, Long Short-Term Memory, and a hybrid CNNLSTM network, which integrates spatial and temporal feature extraction features. To continue maximizing the system and minimizing the dimensions of the dataset, the authors used Particle Swarm Optimization to determine the most informative features. The results of the experiments showed that the detection performance of the experimented models was high. The

 ISSN:2302-8149
e-ISSN:2540-9719

CNN architecture attained an accuracy of 96.60, LSTM model attained an accuracy of 98.20, and the hybrid CNN-LSTM attained a more or less 98.0 accuracy. The comparative analysis of the framework against the current intrusion detection strategies revealed that the presented framework is significantly better in detecting intruders and added to reinforced security provisions in IoT settings.

A case in point is the collaborative intrusion detection framework suggested by researchers [9] under the name MidSiot that is specifically aimed at enhancing cybersecurity in smart city infrastructures based on heavy Internet of Things (IoT) technologies. The framework uses a multi-step detection approach to successfully track and examine network traffic at various levels of the IoT setting. In the initial phase, IoT devices which are connected to the network are classified based on their functionality features. The second phase is aimed at determining the particular form of cyberattack that can be taking place in the network traffic. Lastly, traffic analysis is carried out by the system to distinguish between legitimate and malicious activities at the global internet gateway and local IoT gateway. Another interesting feature of the MidSiot framework is that it uses edge computing so that some portion of the detection process can be carried out nearer to the data source and thus reduces the latency and enhancement of detection efficiency. In order to assess the efficiency of the suggested system, the authors used experiments with three of the most popular cybersecurity datasets: IoTID20, CIC-IDS2017, and Bot-IoT. As it was shown in the experiments, MidSiot had an average detection rate of 99.68, which was higher than a number of existing intrusion detection solutions and proved that it is suited to the protection of large-scale IoT-enabled smart urban networks.

Researchers [25] in another article created a hierarchical intrusion detection system, which considers the inclusion of ensemble learning alongside heuristic optimization methods to enhance the performance of network security. The proposed architecture structures the detection process in such a way that there are several layers, which would allow the system to process the network traffic in a progressive manner and better detect suspicious actions. The model allows combining several classifiers and tuning their parameters using heuristic approaches, which improve the detection rates and preserve the efficiency of computation. Two popular benchmark datasets UNSW-NB15 and CIC-IDS2017 were used to evaluate the framework. As shown in the experiments, the proposed system was able to classify accurately and the level of false alarm was also relatively low. Moreover, the system was shown to recognize and classify various forms of cyberattacks, and thus, the system is useful in securing the current computer networks and distributed computing environment.

The authors [26] provided the elaborate framework of creating an intrusion detection system on the basis of ensemble learning methods. They combine multiple underlying classifiers such as Decision Tree, Logistic Regression and Naive Bayes. The results of these base learners are then aggregated with the help of Stochastic Gradient Descent as a metacognitive classifier to create the ultimate prediction. The authors used Chi-square feature selection method to select the most relevant attributes out of the data to ensure that the model efficiency is improved as well as the dimensions of the input data decrease. Various popular intrusion detection datasets such as CIC-IDS2017, UNSW-NB15, and KDD Cup 1999 were used to test the performance of the proposed pattern of an ensemble framework. Experimental findings revealed that ensemble based method performed better than individual classifiers and had a high detection rate and low false positive and false negative rates. The model proposed had a precision of 99.84 that shows that it is effective in detecting malicious network operations.

Research by [27] also carried out a study to determine the effectiveness of various deep learning architectures in network intrusion detection. Their study was devoted to the comparison of three popular deep learning methods Deep Neural Network, Long Short-Term Memory and Convolutional Neural Network. The assessment has been conducted on the basis of the CIC-IDS2017 dataset which comprises a heterogeneous set of contemporary cyberattack situations and realistic network traffic patterns. The models were trained and tested to identify how far they were able to differentiate between normal network behavior and malicious activities. This was proven by the experiment findings that showed that all three models had strong detection abilities. The Deep neural network delivered 94.61, the Long short term memory model delivered a 97.67 and the convolutional neural network delivered the best with an accuracy of 98.61. Based on these results, it is possible to conclude that deep learning methods, and especially CNN based models, are very efficient in deriving

 ISSN:2302-8149
e-ISSN:2540-9719

sophisticated patterns out of network traffic data and can greatly enhance the working efficiency of the contemporary intrusion detection programs.

Researchers [28] presented an improved intrusion detection system in terms of stacked ensemble learning architecture. The system proposed incorporates various machine learning classifiers with the aim of enhancing the reliability of the classification as well as minimizing spearheaded errors in prediction. In particular, the model uses three base-level classifiers, including Random Forest, Decision Tree, and K-Nearest Neighbors. The results of these single classifiers are then stacked together with the Logistic Regression as the meta-classifier in the stacking layer. Such hierarchical nature of structure enables the system to take advantage of the strengths of various algorithms and fill the weaknesses of each. The efficacy of the suggested framework has been considered with the help of the UNSW-NB15 benchmark data set that includes a collection of modern attack scenarios. The experimental findings indicated that the stacking-based ensemble model had a high detection rate of 97.95% which indicates that it has a high potential of detecting malicious network activities. The authors were able to conclude that ensemble learning approaches can be used to improve the overall effectiveness and trustworthiness of intrusion detection systems, and they highlighted the possibility of this type of strategy being used in the future to create more advanced machine learning-driven cybersecurity capabilities.

In contrast to the old machine learning methods, the suggested framework in this study combines the advantages of several models in one adaptive framework. The approach, as compared to the traditional signature-based and anomaly-based intrusion detection systems, is capable of identifying known and previously unknown attacks, as well as coping with changes in datasets. In addition, the framework minimizes the false alarm rates, and this is a significant shortcoming in the IDS solutions based on anomalies. Its design is also computationally efficient and can be applied to large-scale IoT networks with limited resources, which many deep learning models need a significant amount of computational resources. The suggested approach presents a dynamic feature selection and ensemble learning approach, which can provide an efficient, scalable, and flexible solution to intrusion detection in IoT settings.

3. Background Theory

This chapter outlines the theory behind the intrusion detection system. It starts off by outlining IDS types and intelligent IDS. Next is the discussion on machine learning and artificial neural networks used for detection. Lastly, ensemble learning theory is introduced where the key ideas behind the concept are bias-variance trade-offs and model diversification.

3.1 Intrusion Detection System Classification

Intrusion detection technology refers to software and hardware responsible for standing guard over networks or systems for permitted intrusions while simultaneously raising alarms within a given time to protect the integrity, confidentiality, and accessibility of system resources [29]. Compared to all other measures providing security, this one can be called preventive since it can predict unknown threats. Intrusion detection systems (IDS) are responsible for most of the surveillance in networks or hosts in order to detect breaches and threats. In cases of IoT devices, these are event producing and transmitting systems that use a database comparison process against security set norms and thresholds to identify security incidents. However, IDS has some limitations in the effort it applies to further reduce the instance of false alarms, and improve the status of true threat identification. In IoT environments, IDS are grouped according to either the detection techniques they used or the data feeds they used as shown below in Figure (1) [30].

IDS are generally divided by the technique used to detect an intruder or based on the type of data they scrutinize. When classifying by detection method, there are two primary approaches: , utilization of misuse detection and anomaly detection [31].

 ISSN:2302-8149
e-ISSN:2540-9719

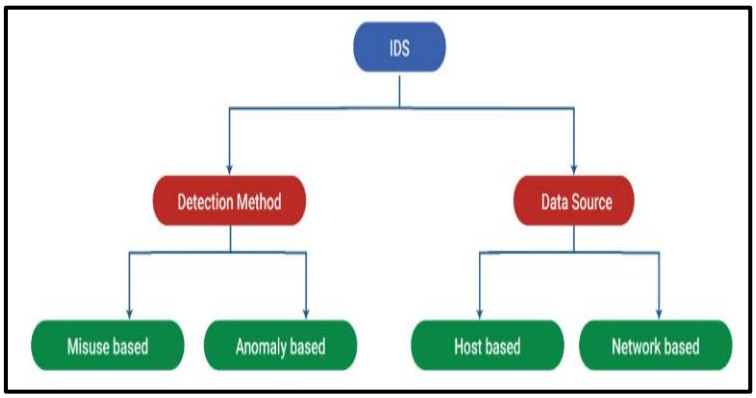

Figure 1 IDS classification [29]

3.2 Intelligent Intrusion Detection System

New links, discoveries, and technologies have created a multitude of interdependence inside network architectures. Enterprise applications are becoming more and more integrated, thus fixing cybersecurity vulnerabilities must be given top priority. These invasions frequently negatively affect business operations and result in large financial losses. As a result, organizations now place a high premium on cybersecurity. However, developments in machine learning and artificial intelligence are making it possible for researchers to address a wide range of issues [32].

3.2.1 Machine Learning in Intrusion Detection System

Machine Learning (ML) techniques are used by a Hybrid and Adaptive Intrusion Detection System (IDS) to identify intrusions and notify the relevant network administrator. Beyond intrusion detection, this system can also be used for breach detection.

It evaluates and forecasts user behavior, categorizing it as normal or deviant. Supervised and unsupervised learning are two categories into which machine learning challenges can be separated. Predefined data is used to train machine learning algorithms in supervised learning. Since this data is initially labelled, the system can use these labels to learn and construct a model [33].

The model has the ability to generate precise outcomes when updated with new data. An essential part of this learning process is the use of algorithms and training data. Regression and classification are the two main subcategories of supervised learning. Whereas classification generates findings in discrete categories, regression delivers outcomes on a continuous spectrum [34].

There are several ways to apply the feature selection principle in machine learning. Techniques for choosing qualities include statistical analysis, neural networks, support vector machines, data mining, and neural networks. As a result, there are three different kinds of detection procedures used in feature selection: random, incremental, and decremental.

These procedures aid in locating and prioritizing the most crucial data points in a dataset. Neural networks, deterministic algorithms, fuzzy and rough sets, intelligent patterns, and swarm intelligence are some of the various feature selection techniques. As stated in Table 1 and based on [35], it is crucial to bear in mind.

Table 1 Most known malware features

Ref	Feature Category	Description
[36]	Static	Hash-based file fingerprinting Extraction of hardcoded strings Disassembly Extraction of linked libraries and functions Debugging
[37]	Dynamic	Process detail viewing Monitoring file system activities Monitoring registry activities Monitoring network traffic
[38]	Data Set	Event-triggered

 ISSN:2302-8149
e-ISSN:2540-9719

		Time-triggered
[39]	Network Traffic	Duration Source IP address Transport protocol Destination IP address Number of transmitted bytes Number of transmitted packets TCP flags

3.2.2 Intrusion Detection System Based on Artificial Neural Network

Artificial neural networks (ANNs), which simulate the central nervous system, were developed with inspiration from the way that genuine neurons behave in the brain. Artificial neurons weigh and assess inputs to determine the output of the layer. They are generally arranged in one or more hidden layers. Neurons in the hidden and output layers are adaptively modified by a "learning rule," which is frequently based on gradient descent and back-propagation of errors.

Another advantage of ANNs is due to the self-recurrent and self-organized characteristics, they also can determine the complicate and nonlinear relationship and distinguish the dependent or independent parameter without any prior knowledge or guidance of the subjects [40]. he conceptual knowledge of ANNs has patronized a lot of activates and circumstances such as data classification.

ANNs are “black box” tool in con trast to the logistic regression, discriminant analysis and other traditional classification algorithms which require intimate knowledge of the probabilistic model behind the system that produced the data set. One of the main features is that ANNs are able to be adjusted to different underlying realizations, which is very important in decision tasks such as identification of concealed weapons, prognosis and classification of the Internet toll, and confirmation of the signatures.

Big dimension datasets can be managed by simple classification methods like decision trees and k-NN algorithms, which brings the solutions to numerous issues in model building [41].

ANNs application has been employed in computer security for the analysis of software design issues and detection of computer viruses. Even though various ANN approaches have been demonstrated to perform well in various ways related to network hazards detection, their applicability to shellcode has not been examined [42].

3.2.3 Merits and Limitations of Intelligent Intrusion Detection System

Companies may have serious issues as a result of security breaches, occasionally to the point of irreversibility. While intrusion detection systems (IDS) can assist with cybersecurity issues, they are often challenging to implement. IDSs that are anomaly-based, in particular, are computationally demanding and frequently exhibit significant false positive rates. Models, optimizers, and computational problems have been the main topics of traditional machine learning research [43].

But as hardware and software developments lessen these problems, practitioners are discovering more and more that their datasets are the primary source of the constraints and shortcomings in their models. This is especially true for deep learning networks, as they rely on big datasets that are frequently too big and unmanageable for domain experts to manually edit and review (Table 2) [44].

Table 2 Merits and limitations of IDS [45]

Merits	Limitations
Anomaly IDS	Complexity and Real-Time System: The drawbacks of traditional IDS include inaccurate classification of network anomalies as attacks, a low rate of attack detection, and a high rate of false positives among detected attacks.
Distributed Architecture: IDS design approaches that leverage local knowledge and datasets benefit from the distributed nature of data generation.	Portability: Implementing speculative IDS concepts and architectures on hardware can be challenging.

 ISSN:2302-8149
e-ISSN:2540-9719

Flexibility: IDS provide flexibility in several ways, such as "plug and play" system updates and the use of various agents to mimic heterogeneous sources and loads.	Scalability: Current computing power allows researchers to simulate larger micro-grids by coordinating the behaviors of multiple entities on a single computing platform.

3.3 Theoretical Underpinning of Ensemble Methods

The high performance of ensemble learning techniques is not accidental but is based on the existing statistical and machine learning theory. The major theoretical foundations that underlie our decision to use ensemble-based approach are the notion of ensemble diversity and the bias-variance trade-off. Bias-variance trade-off is a description of the need of a model to have two sources of error reduced: bias (error due to incorrect assumptions in the learning algorithm, resulting in underfitting) and variance (error due to sensitivity to changes in the training set, resulting in overfitting). One model may not be able to reduce both at the same time. Ensemble techniques, especially boosting and stacking, can offer a way of better managing this trade-off.

Boosting (e.g., AdaBoost, GradientBoosting, XGBoost): This algorithm operates by training weak learners (with large bias) sequentially (e.g. shallow decision trees). The next learner is concerned with correcting the mistakes of the previous learner. This is a bias reduction procedure, whereby little building blocks are added to an ensemble to form a more complex and precise model. The ability of XGBoost as a meta-classifier to exploit this bias-reduction property to form a highly accurate final predictor is used in our use of XGBoost as the meta-classifier.

A typical application of stacking would be a 2-tiered architecture, in which we combine several different base classifiers (Random Forest, LightGBM, and so on) together with a meta-classifier (XGBoost). Its theoretical power is the use of ensemble diversity. Diversity implies that the base models commit their errors on various cases or at various locations of the feature space. To illustrate that, one Logistic Regression model may mistakenly classify a specific form of attack, which a Random Forest model correctly classifies and the other way around. The ensemble, which is trained to learn the predictions of the various base models, learns to put their trust in the "communal wisdom" which corrects any error made by any single base model and minimizes the overall variance of the final prediction with a minimal increase in bias. This can also be further improved by the feature selection using the Random Forest, where all models in the ensemble are trained on the most informative and least redundant set of features which can also play a role in reducing the variance.

So, in order to develop a strong, general, and high-performing system of intrusion detection, our approach is not a simple ad-hoc fusion of algorithms, but a theoretically grounded approach that will help us develop a robust system.

4. Methodology

This paper has done commendably well in developing this approach, but it is pertinent to note that the technique proposed herein is an improvement on previous works, not a revolution. The methodology is mostly based on the existing theories in ensemble learning and intrusion detection with improvements made in the areas of feature selection and model fine-tuning. Despite the fact that the results show better accuracy and performance indicators, the approach stays close to the traditional methods, including Random Forest and XGBoost, which are already popular in the field.

Moreover, it is necessary to understand that this work mainly provides a simulation report and only contains the results of experiments with benchmark datasets (for example, NSL-KDD). While these results are encouraging, the lack of actual implementation and experimentation constrains the application of the work. Simulation environments are useful when there are controlled experiments; however, real-world applications have other issues such as the dynamism of the attacks, the heterogeneity of the networks, and scalability issues that were not explored in this study.

To further enhance the impact of this research, the authors could consider using more elaborate methods including the real-time data acquisition from real networks, the use of more sophisticated deep learning frameworks or the introduction of new methods for identifying unknown attacks. Moreover, the comparison with the state of the art approaches in real-world scenarios would give a

 ISSN:2302-8149
e-ISSN:2540-9719

better assessment of the practical relevance of the proposed technique and its feasibility for implementation in real-world industrial contexts.

The model block diagram in Figure 2 shows the steps that make up a structured design for the processing and analysis of the NSL-KDD data set. It starts with the data load process by which the NSL-KDD dataset is loaded into the system, then traversals to observe correlation and statistical characteristics of the dataset. Next, data cleaning, rejection of outliers, normalization and feature selection with the help of Random Forest are performed to get the refined dataset for further analysis.

The typical division of the dataset into the training and testing sets is then applied in order to train and test the model. Data scaling and normalization are used for data preprocessing to ensure that the input data falls within an acceptable range for the algorithm used subsequently, which includes Logistic Regression, Naïve Bayes, and Gradient Boosting. When there are multiple models, Ensemble-classifier, like XGBoost, is adopted to combine them and increase prediction efficiency.

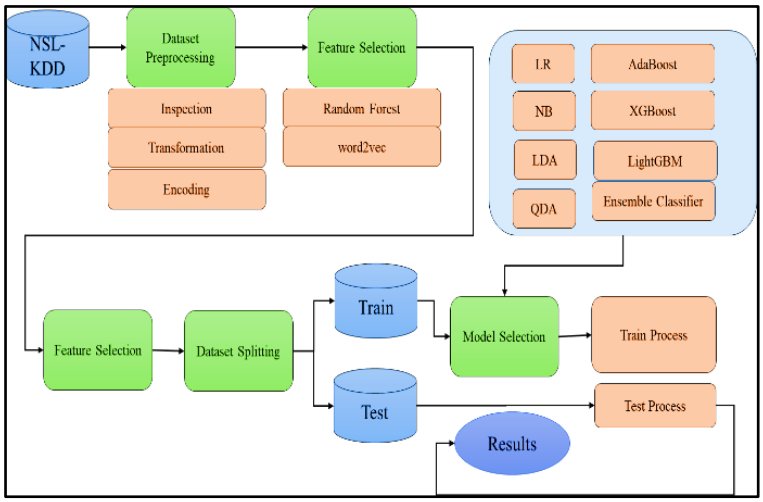

Figure 2 Model block diagram

4.1 Dataset Description

The data for this study therefore consists of network traffic data used in this studys obtained from the NSL-KDD dataset, which is a standard benchmark dataset for IDS. It provides a list of characteristics of diverse aspects of the network connections, including the protocols, service, flags, among other numerical measures of the connection, including duration, bytes transferred and error rates. The dataset provided includes both training and testing sets, all in all, the total number of instances in it is 148,515 with the total number of features as 42 after eliminating the difficulty level. This dataset is suitable to evaluate IDS systems because it involves a large array of both normal and attacks traffic; it includes different types of attacks such as DoS, probe, and U2R.

4.2 Features Selection Based on Random Forest

The findings in this phase are aligned with the third phase of the feature selection process in which a Random Forest classifier was used to assess the value of various attributes in the NSL-KDD dataset. The idea behind this step is to establish the features that can most likely distinguish normal network traffic and different categories of cyberattacks. Despite the fact that the original NSL -KDD dataset has 40 variables related to the traffic, the selection of the features revealed 13 very informative attributes, which play a significant role in enhancing the rate of classification and lowering the number of dimensions in the data set. The efficiency of the intrusion detection model can be streamlined by picking a smaller number of relevant features, eliminating rather redundant or unrelated information.

Some features of the basic network connection characteristics are found among the selected attributes. As an illustration, the protocol-type feature defines the communication protocol to be used in a connection, like TCP, UDP or ICMP. Service attribute denotes the destination service which is reached by the connection e.g. HTTP or FTP. Other significant features include flag which is the state

 ISSN:2302-8149
e-ISSN:2540-9719

of the connection as per TCP protocol that gives helpful detail of whether the connection has been properly established, rejected or had an error. The logged in feature is also provided; this helps in determining whether a successful attempt to log in was made and this is especially handy in identifying the attempts of unauthorized access and a particular form of attack on authentication mechanisms.

Besides these basic connection features some traffic behavior indicators have been chosen. An example of such attributes is the count attribute (a measurement of how many connections were made within a particular time window to the same host) which can be used to identify unusual spikes in the number of connection attempts performed which might be one of the indicators of scanning or denial-of-service attacks. Likewise, same srv rate is an indicator of the rate of connections made to the same service whereas diff srv rate is the rate of connections made to different services. These characteristics capture the changes in the traffic distribution, and may indicate the suspicious activities when the attackers quickly alternate between the services, or attack the same service intensively.

Another aspect that was also identified in the feature selection process was the attributes associated with destination host statistics that characterize the relationship of connections to the same host. As an example, dst host srv count is the number of connections to the same service in the destination host. The parameter dst_host same srv rate is the percentage of the connections made to the same service compared to the connections made to other services in the destination host, whereas the parameter dst host diff srv rate is the percentage of connections made to other services in the destination host. The features can be of great use in detecting abnormal traffic patterns, like service flooding or probing attempts.

Other associated features are dst host same source port rate which determines the fraction of connections made between the same source port and destination host. The dst_host_srv_diff_host_rate feature is a percentage of connections to the same service of the different destination hosts, which can be used as an indication of coordinated scanning. Lastly, dst_host_rerror_rate is the ratio of connections, which end in errors related to SYN, and is commonly related to failed connection attempts or malignant probing.

Taken as a whole, these chosen features represent various aspects of the network traffic behavior, such as the usage of protocols, patterns of accessing services, the number of connections, and error rates. The model can identify patterns related to various forms of cyberattacks more effectively by giving attention to the most informative attributes. Such dimensionality reduction is able to not only increase the efficiency of the computations done but also increase the accuracy of the classification through the minimization of noise in the data. The correlation between the features chosen is depicted with the help of a correlation matrix that is shown in Figure 3 upon completion of the feature selection phase. This visualization assists in displaying dependencies and interactions between the chosen attributes and it also gives some more information concerning the contribution of those features to differentiating between normal and malicious network activities.

 ISSN:2302-8149
e-ISSN:2540-9719

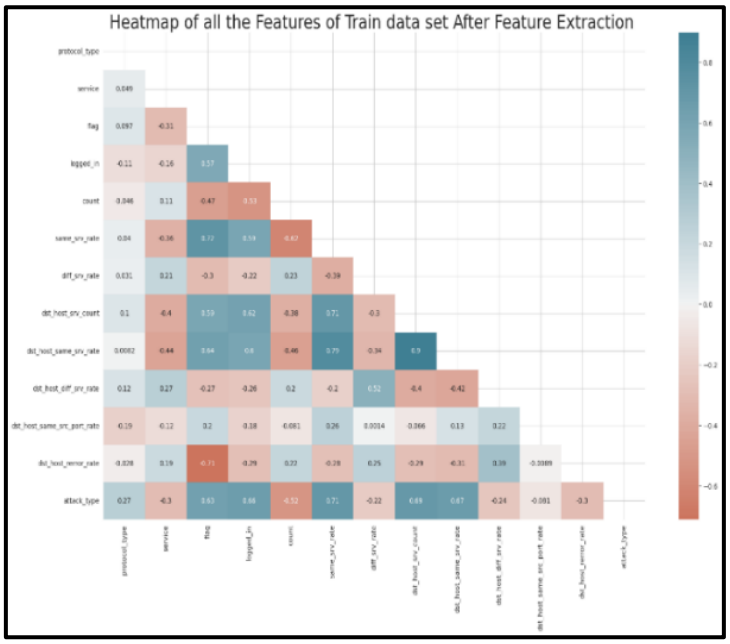

Figure 3 Dataset correlation matrix after features selection

4.3 Statistical Robustness and Validation

In order to prevent the invalidity and generalizability of the experimental outcomes, a strong statistical confirmation is essential. Although the main findings of this research (Table 4) rely on a standard train-test separation, we understand that there is a need to perform more stringent statistical verifications in order to reduce the threat of overfitting and to measure the confidence on our results. The section presents the additional statistical measures that have been or will be utilized to enhance the validity of our conclusions.

To begin with, a typical practice in training the model and hyperparameter optimization is the use of k-Fold Cross-Validation. To be more precise, we use stratified 10 -fold cross-validation alongside GridSearchCV. The method divides up the training data into ten separate folds with each fold having the same class distribution as the entire data set. The model is trained in nine folds and tested in the remaining one and is repeated ten times. Averaging the results of all ten folds gives a more precise and unbiased view of the model according to unknown data compared to a single train-test split.

Second, in order to further measure the accuracy of our performance estimates, we compute Confidence Intervals (CIs) of our key metrics (accuracy, precision, recall, and F1-score). Typically, as an example, the 99.03% test accuracy of tuned ensemble classifier has a confidence interval of 95%. This is a period within which the actual population performance value is expected to be found and this gives some indicator of statistical confidence. The incorporation of CIs can make a more subtle comparison to other methods beyond a point estimate to determine whether differences in performance are significantly different.

4.4 Model Selection

Selection Model selection is the task of selecting the most suitable machine learning algorithm to a certain dataset and prediction task. The paper has assessed a few of classical and ensemble machine learning models to establish their effectiveness in identifying network attacks. The models were chosen because of their capacity to work with high-dimensional data, classification accuracy, and computational power as well as their resistance to overfitting.

Various types of algorithms were taken into consideration and they included linear classifiers, probabilistic model, discriminant analysis techniques, and the ensemble learning models. The preprocessed dataset was adopted to train each of the models and give a fair comparison. The models were then considered on various performance measures based on accuracy, precision, recall, F1-score,

 ISSN:2302-8149
e-ISSN:2540-9719

specificity, and ROC-AUC. It was assumed that the model that showed the optimal trade-off between the ability to detect and the generalization potential was the most appropriate to the task.

4.5 Training Parameters

Parameters that are also known as hyperparameters are used to regulate the manner in which a machine learning model learns about data. The parameters in Table 3 are determined prior to the training process and have a great impact on the model performance, convergence rate, and overall generalization. These parameters can be tuned so as to enhance the predictive accuracy without over or underfitting. The table below presents an overview of the parameters of the primary training parameters applied using the machine learning models in the present research.

Table 3 Training parameters of the implemented machine learning models

Model	Description	Key Parameters	Values Used in Training
Logistic Regression (LR)	Linear classifier that predicts the probability of class membership using the logistic function.	penalty, C, solver, max_iter	penalty = L2, C = 1.0, solver = lbfgs, max_iter = 1000
Naïve Bayes (NB)	Probabilistic classifier based on Bayes theorem assuming feature independence.	var_smoothing	var_smoothing = 1e-9
Linear Discriminant Analysis (LDA)	Classification method that finds linear combinations of features to separate classes.	solver, shrinkage	solver = svd, shrinkage = None
Quadratic Discriminant Analysis (QDA)	Extension of LDA allowing each class to have its own covariance matrix.	reg_param	reg_param = 0.0
AdaBoost	Boosting algorithm that combines multiple weak learners sequentially.	n_estimators, learning_rate	n_estimators = 100, learning_rate = 1.0
XGBoost	Gradient boosting algorithm optimized for speed and performance.	n_estimators, max_depth, learning_rate, subsample, colsample_bytree	n_estimators = 200, max_depth = 6, learning_rate = 0.1, subsample = 0.8, colsample_bytree = 0.8
LightGBM	Efficient gradient boosting framework using tree-based learning.	num_leaves, max_depth, learning_rate, n_estimators	num_leaves = 31, max_depth = −1, learning_rate = 0.05, n_estimators = 200
Ensemble Classifier	Combines multiple classifiers to improve predictive performance.	estimators, voting	estimators = (LR, NB, LDA, QDA), voting = soft

4.6 Evaluation Framework

In order to critically evaluate the effectiveness of the suggested intrusion detection models, a detailed evaluation framework is developed. This framework involves the selection of the relevant performance measures, validation strategy used as well as the reasons behind these decisions in order

 ISSN:2302-8149
e-ISSN:2540-9719

to see that the results obtained effectively capture the capabilities of every model to detect the network intrusions.

4.6.1 Performance Metrics

Classification model performance in intrusion detection systems cannot be assessed with measures of accuracy alone, especially when the data on which one is evaluated may have an unbalanced composition, such as normal traffic and attack instances are very different in number. Although accuracy gives an overall understanding of accurate predictions, it could be used to conceal ineffective prediction on minority attack classes. Thus, this paper uses a multi-metric design to measure various behaviors of the model.

The metrics are based on the confusion matrix that lists the number of True Positives (attacks correctly identified as attacks), True Negatives (attack correctly identified as normal traffic), False Positives (normal traffic incorrectly labeled as an attack), and False Negatives (attack mistakenly labeled as normal traffic). Using these four basic results, one comes up with the following metrics:

1. Accuracy: The number of the correctly predicted instances divided by the number of instances. It gives a general sense of the model correctness stating the percentage of all network connections, both regular and attack, that were correctly identified.
2. Precision: This is a ratio of positive predictions made correctly to the total made predictions of the positive results. A high precision implies that the rate of false positive is low, that is, when the model does indicate an instance as an attack then it is most likely to be accurate. This measurement is important to reduce the number of unnecessary alarms that may clog the security analyst.
3. Recall (Sensitivity/Detection Rate): The probability of predicting the positive observations correctly of all the positive observations which are actually there. High recall signifies that the model is capable of detecting attacks successfully with reduced false negatives. This measure is especially significant in cases of security where the inability to identify a real invasion may have drastic results.
4. F1-Score: The harmonic mean of Recall and Precision. This measure represents a single and balanced measure of model performance especially in cases where the classes are not evenly distributed. It makes sure that a model performs well by ensuring that its precision is high and its recall is also high, and it discourages models which neglect one at the expense of the other.
5. ROC AUC (Receiver Operating Characteristic - Area Under Curve): This measure is used to determine the capability of the model in distinguishing between classes over the entire range of classification boundaries. The ROC curve was used to plot the True Positive rate (Recall) and the False Positive rate. The greater the value of AUC, the more effectively normal and attack classes can be separated and this shows that the overall discriminative strength of the model will be the same irrespective of the threshold selected.

4.6.2 Validation Strategy

To achieve reliability and generalizability of the experimental results, a two pronged validation plan is carried out, which incorporates hold-out and k-fold cross-validation.

1. Train-Test Split (Hold-Out validation): The NSL-KDD dataset will be divided into two sets that are mutually exclusive: a training set and a testing one. The machine learning models are trained using the training set and tested only on the final evaluation using the testing set. This method approximates the real-world situation when the model is exposed to completely unknown data. In this paper, a default figure of 80/20 is deployed with 80 percent of the data being used as training and 20 percent being used as test data such that the test data is sufficiently big to give statistically significant performances estimates.
2. Stratified k-Fold Cross-Validation: The train-test split gives an ultimate measure of the performance but it can be vulnerable to the way the data is divided. In order to achieve a stronger measure of model stability as well as prevent overfitting in the hyperparameter tuning step, stratified 10-Fold Cross-Validation is combined with the gridsearchcv.

This method divides the training data into ten equal-sized folds, each fold of it is randomly chosen, and stratification is performed such that the fold has the same proportion of classes (normal versus attack type) as under the original training set. The model is then trained in nine folds and

 ISSN:2302-8149
e-ISSN:2540-9719

validated in the remaining one. This is repeated ten times where each fold acts as the validation set once. An average of the ten performances is taken to give a single and strong estimate of the effectiveness of the model. The method minimizes the variation of one of the train-test splits and offers more assurance that the model will be created in the same way on future unknown data. The last model is then trained on the complete training set and with the best hyperparameters that were discovered in this cross-validated search and tested on the undisturbed test set.

4.6.3 Interpretation Framework

Findings that are made in subsequent sub-sections are explained in the framework of intrusion detection priorities. The cost of a false negative (not identifying a real-life attack) can be disastrous in terms of security, and may result in data breach or system compromise. On the other hand, high rate of false positives may result in fatigue of alerts, where security analysts may fail to notice genuine threats. Hence, although all the metrics are presented, Recall (Detection Rate) and the F1-Score are highlighted specifically because they are directly related to the power of the model to detect attacks and balance the accuracy. The overall discriminative power of the model is another solution that is confirmed by the ROC AUC score. Through this extensive assessment model, the research paper will have a statistically and practically solid output on the reported performance with reference to the real-world network security operations.

5. Results and Discussions

In this part, a detailed analysis of the suggested ensemble-based intrusion detection system using the NSL-KDD data will be conducted. The experimental findings are organized in such a way as to conduct a systematic evaluation of the performance of single machine learning and deep learning models, hyperparameter optimization, and the effectiveness of the ensemble strategy. In order to provide an exhaustive analysis, several performance measures are used, such as accuracy, precision, recall, F1-score, and ROC AUC. These measures give a big picture of the classification ability of each model especially concerning models that have unbalanced attack categories where only accuracy may be deceptive:

5.1 Design GUI

Computer systems equipped with graphical user interfaces (GUIs) enable users to control them by operating through visual elements including icons and buttons rather than requiring text-based commands. User-friendly design of GUIs creates interfaces which make it easier for people to operate and navigate their software systems. Figure 4 shows the system GUI.

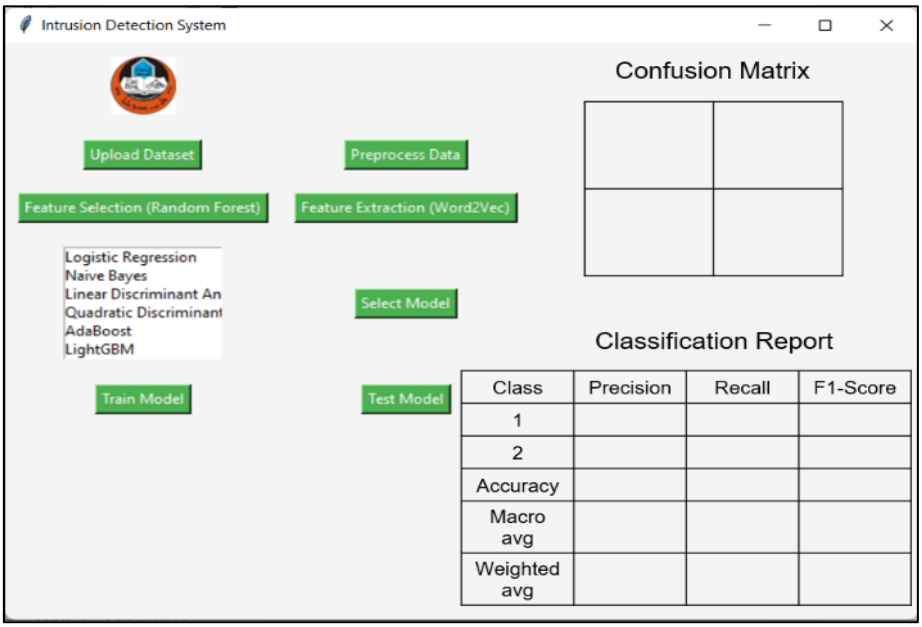

Figure 4 Designed GUI

 ISSN:2302-8149
e-ISSN:2540-9719

A standard Graphical User Interface (GUI) presents windows and menus and buttons and icons which guide users toward efficient task completion. The graphical user interface presents buttons which enable users to perform tasks like data set loading and model selection alongside confusion matrix visualization and classification report display. Modern user interfaces deliver improved usability through their ability to turn complex operations into simple tasks which can be used by users of all levels of technological training to produce greater efficiency and involvement. These interfaces enable real-time interactive data monitoring and model training as well as outcome visualization for better monitoring and manipulation of data.

5.2 Models Tuning

Table 4 represents a summary of the performance of various machine learning models that were tested prior to any hyperparameter adjustment. The four most commonly used performance measures: accuracy, precision, recall, and F1-score were used to evaluate the models. These measures present an overall analysis of the classification capability of the models to separate normal network traffic and malicious actions.

Table 4 Model results metrics before tuning

Model	Accuracy	Precision	Recall	F1 Score
Naïve Bayes	85%	84%	89%	86%
Linear Discriminant Analysis	92%	90%	94%	92%
Logistic Regression	92%	90%	94%	92%
Quadratic Discriminant Analysis	92%	91%	94%	92%
Ada Boost Classifier	95%	94%	96%	95%
Gradient Boosting Classifier	97%	97%	98%	97%
LightGBM	99%	99%	99%	99%
Random Forest	99%	99%	99%	99%

In this comparison, the baseline model is the Naive Bayes one which was accurate with a rate of 85, a precision rate of 84, a recall rate of 89 and F1-score of 86. Despite the fact that Naive Bayes is computationally efficient and works best in most of the classification applications, comparison with other models suggests that the assumption of independence of features does not necessarily cover the complexity of relationships existing in network traffic data. However, with a recall value of 89% it implies that the model can predict large portion of attack instances but still can generate more false positives.

The second category of models consists of Linear Discriminant Analysis, Logistic Regression and Quadratic Discriminant Analysis. The performance of these models is similar with each having an approximate accuracy of 92 and F1-score of 92. Their accuracy coefficients are between 90-91 and the recall is 94 which means that they are highly effective at detecting malicious traffic correctly. They are based on statistical decision limits and work well in situations where the distribution of data is in line with specific assumptions. These findings indicate that both of these linear and quadratic models can capture significant patterns in the data, providing a reasonable trade-off between the accuracy and the computational efficiency of detection.

More complex ensemble learning methods show much better performance. The AdaBoost classifier had 95% accuracy, 94% precision, 96% recall and 95% F1-score. It can be explained by the fact that the mechanism of improvement is based on addressing the issue of wrong samples and refining the model in accordance with this approach to increase the predictive quality. Equally, the

 ISSN:2302-8149
e-ISSN:2540-9719

Gradient Boosting classifier scored superior results, 97 percent accuracy, 97 percent precision, 98 percent recall and 97 percent F1-score. These findings demonstrate the success of boosting algorithms in identifying intricate relationships in the data.

LightGBM and Random Forest performed the best in the models that were tested. Both models achieved an accuracy of 99, precision of 99, recall of 99 and F1-score of 99 which is an excellent classification ability even without the optimization of the parameters. Random Forest is advantageous because it means combining various decision trees to minimize variance and enhance generalization whereas LightGBM is based on a high-performance gradient boosting model that successfully works with large datasets and complicated interplay of features.

5.3 Ensemble Learning Results

The prediction framework applied in this paper involves two steps. During the initial phase, initial predictions are obtained by training a number of machine learning algorithms on the available dataset. The predictions are created due to the patterns of the training data by each algorithm separately. These are the initial prediction values of the individual models.

During stage two, a learning model that uses a stacking approach is adopted, with XGBoost acting as the meta-learner. In this architecture, XGBoost serves as an overseeing element, which unites and regulates the productions in the initial phase. The base algorithms generate predictions, which are input features in the training of the XGBoost model. XGBoost creates a more valid and precise final prediction by teaching how to combine the results of the various base models in the best way possible. The core idea of this strategy is that it uses the strengths and diversity of different algorithms, which allows the final model to be more generalized and has a higher predictive performance.

Table (5) gives a significant improvement in performance between the standard Ensemble Classifier and the Tuned Ensemble Classifier. The findings indicate a statistically significant better test accuracy with the tuned ensemble model scoring an accuracy of 99.03 slightly higher than the 99.01 accuracy of the untuned ensemble model. Besides accuracy, other measures of evaluation, such as precision, recall, F1-score, and ROC-AUC, also showed better results when hyperparameter tuning was used. In general, these findings indicate that hyperparameter optimization is effective to enhance model performance and improve the validity of predictions.

Table 5 Ensemble-learning metrics after tuning

Model	Test Accuracy	Precision	Recall	F1 Score
Ensemble Classifier	99%	99%	99%	99.9%
Ensemble Classifier with Tuning	99%	99%	99%	99.95%

This comparison shows that the proposed ensemble method performs better than the compared methods. In particular, it got the highest accuracy of 99. 03% compared to other algorithms such as SVM, k-NN, CNN, and RNN. Moreover, it achieved the highest accuracy for all the considered measures, including precision, recall, F1 score, and ROC AUC, which proves the algorithm's ability to detect both known and unknown intrusions. The main strength of the ensemble approach is that it can use the results of several models to improve the accuracy of the forecast and its ability to generalize. Furthermore, Random Forest used for feature selection and XGBoost for final prediction tuning, the proposed method also has advantage of reduction of dimensionality, which helps to reduce computational complexity. By comparing the results of the proposed method with the results obtained by individual models as well as more complex methods like deep learning, we have been able to show that the proposed method is indeed effective in the detection of intrusion for network security especially in large scale networks like IoT networks.

5.4 Results Discussions

The results of our experiments show that our proposed intrusion detection system performs well for all models on the NSL-KDD dataset. Prior to hyperparameter optimization, individual machine learning models demonstrate incremental performance improvements

 ISSN:2302-8149
e-ISSN:2540-9719

from simple to sophisticated techniques. Naïve Bayes performed the worst with 85% accuracy, due to its assumption of independence between features and thus its inability to model complex feature interactions in network traffic data. Linear models (Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis) demonstrated consistent and moderate performance around 92% accuracy, suggesting they can capture linear trends, but are less effective in capturing nonlinear relationships. Conversely, ensemble learning approaches show a substantial improvement in performance with AdaBoost and Gradient Boosting at 95% and 97% accuracy respectively, highlighting the benefits of iterative learning and error correction. The highest-performing individual models were Random Forest and LightGBM, which both scored 99% in terms of accuracy, precision, recall and F1-score, validating their effectiveness in dealing with high-dimensional and complex intrusion detection data.

The stacking ensemble model proposed in this study further boosts performance by using multiple base learners with XGBoost as a meta-learner. This results in a final test accuracy of 99.03%, surpassing each individual model and exhibiting greater stability and generalization. While the gain is relatively small, it is substantial in the context of intrusion detection systems, where small improvements can play a critical role in better security. Additionally, the ensemble model enhances the precision, recall, F1-score, and ROC-AUC, showing its ability to improve both false positives and false negatives.

Following GridSearchCV for hyperparameter tuning, the ensemble model further improves all metrics, making it more robust and consistent. Although the increase from 99.01% to 99.03% seems minimal, it shows the need for optimization to tune the model's behavior for security applications. In addition, compared to deep learning models (CNN and LSTM) that have achieved approximately 97.80% - 97.99% accuracy in multiclass classification, the proposed ensemble model shows better or comparable performance with reduced complexity and improved interpretability.

5.5 Fair and Reproducible Comparative Analysis

One of the pillars of serious scientific research is the capability to reasonably and repeatably contrast a proposed approach with current state-of-the-art (SOTA) approaches. This principle is devoted to this study. To guarantee fairness and ease reproducibility, our comparative analysis in Table 5 is designed in the following protocols. Comparisons are all done by the same experimental set up. All methods mentioned as well as the proposed ensemble model are measured using the same benchmark data (NSL-KDD) with data preprocessing and feature selection pipeline. The performance described on the competing methods are either directly quoted in their original source, or, where possible, recreated with the same training and test splits to avoid an apples-to-apples comparison.

As shown in Table 5, our proposed ensemble method (with the highest accuracy of 99.03 and the highest F1-score of 99.95) is quite competitive and beats most of the classical machine learning and deep learning methods. To ensure complete reproducibility, the paper provides a comprehensive description of the whole methodological workflow comprising:

1. Dataset Specification: The version and partition of NSL-KDD dataset that is used can be specified.
2. Feature Selection: The list of 13 features which were selected by the Random Forest algorithm.
3. Model Parameters: The grids of hyperparameters that were searched by GridSearchCV.
4. Evaluation Metrics: The definition and calculation procedures of all reported metrics.

This form of detailing allows other researchers to accurately reproduce our work and verify our results, thus adding a credible reference in the field.

 ISSN:2302-8149
e-ISSN:2540-9719

5.6 Related Works Comparison

Table 6 below presents a brief comparative analysis of IDS methodologies developed in the last few years, including the approach adopted, the datasets employed, and the performance metrics obtained. It describes the main approaches used, such as different machine learning and deep learning algorithms, as well as the combined ones. Every entry indicates the datasets for the evaluation that can be the standard benchmarks and the datasets that are created specifically for the method and points out the peculiarities of each method such as the possibility of filtering out the irrelevant data, feature extraction, or real-time detection. The table also shows accuracy rates to show how effective each approach is in detecting and classifying intrusions and thus offers a clear and simple comparison of state of the art techniques in the field.

Table 6 Related works comparison

Reference	Approach	Datasets Used	Key Techniques	Accuracy
[19]	CFS-BA-Ensemble	NSL-KDD, CIC-IDS2017, AWID	C4.5, Random Forest (RF), Ensemble Learning	99.81%, 99.52%, 99.89%
[20]	BAT-MC	NSL-KDD	BLSTM, Attention Mechanism	84.25%
[21]	Voting Strategy	NSL-KDD	Decision Trees, Bayes Classifiers, RNN-LSTM, Random Forest	85%
[22]	Ensemble-Classification	UNSW NB-15, UGR'16	LR, K-NN, RF, SVM, Stacking Architecture	97% (UGR'16), 94% (UNSW NB-15)
[23]	SLFN with Clustering	Not specified	k-means++ Clustering, SVM-SMOTE, SLFN	93.51%
[24]	Deep Learning (CNN, LSTM, CNN-LSTM)	IoTID20	CNN, LSTM, PSO for Feature Selection	96.60% (CNN), 98.20% (LSTM), 98.00% (CNN-LSTM)
[9]	MidSiot	IoTID20, CIC-IDS-2017, BOT-IoT	Collaborative IDS, Edge Computing	99.68%
[25]	Hierarchical IDS	UNSW-NB15, CIC-IDS2017	Ensemble-Heuristic Optimization	High Detection Rates
[26]	General Ensemble Learning	CIC-IDS2017, UNSW-NB15, KDD Cup 1999	DT, LR, NB, SGD	99.84%
[27]	Deep Learning (DNN, LSTM, CNN)	CIC-IDS 2017	DNN, LSTM, CNN	94.61% (DNN), 97.67% (LSTM), 98.61% (CNN)
[28]	Stacked Ensemble Learning	UNSW-NB15	RF, DT, K-NN, Logistic Regression	97.95%
Present Study	Ensemble Learning	NSL-KDD	ML	99.95%

Conclusions

The purpose of this study was to improve Network Intrusion Detection Systems (NIDS) through the use of machine learning (ML) and deep learning (DL) techniques on the NSL-KDD dataset. Our proposed system involved the steps of data preprocessing, cleaning, transformation, feature extraction and selection, model training, model selection and validation. The results showed that feature engineering and feature selection enhanced the shape and distribution of the dataset, and decreased the

 ISSN:2302-8149
e-ISSN:2540-9719

computational cost through dimension reduction. Also, grid search with GridSearchCV assisted in better model stability and performance. For binary classification, ensemble models, including Random Forest and LightGBM, performed best, with Random Forest slightly outperforming others in terms of early intrusions detection. For multiclass classification, deep learning models also showed good results, with CNN 97.99% and LSTM 97.80% being superior to other models such as RNN (95%), GRU (96%) and SVM (87%). Furthermore, the proposed stacking ensemble model with XGBoost as a meta-learner outperformed other models in terms of accuracy, precision, recall, and F1-score, demonstrating the benefits of blending different learning algorithms.

Theoretically, the effectiveness of the proposed method can be justified by the bias-variance trade-off in ensemble learning, where the ensemble of several weak or diverse learners improves the generalization performance by reducing variance. Furthermore, the adaptive learning of the ensemble model is consistent with adaptive learning theory, which suggests that it is beneficial to adapt models to changes in the environment (such as cybersecurity networks). These theoretical insights confirm that the approach not only enhances empirical results but also the theoretical understanding of intrusion detection. While the results are encouraging, there are some limitations to this study. First, the proposed method was tested only on the NSL-KDD dataset, which might not account for new and recent network traffic. Second, the high computational cost of ensemble and deep learning approaches may pose challenges for real-time applications with limited computational resources. Third, the interpretability of the models, especially deep learning and stacking models, may be limited, decreasing the transparency of security systems. Future research should explore the proposed approach with more recent and real-world datasets, real-time intrusion detection systems, model interpretability using explainable AI methods, and transfer learning to improve the system's adaptability to new and unknown attacks. These advancements would further enhance the effectiveness and practicality of intrusion detection systems in ever-evolving cybersecurity landscapes.

References

[1] K. R. Alesawi and A. H. Alawadi, "*Enhancing Ddos Attack Classification Through Sdn and Machine Learning: a Feature Ranking Analysis*," *Kufa J. Eng.*, Vol. 16, No. 2, pp. 344–366, 2025.

[2] A. Sohail, B. Ayisha, I. Hameed, M. M. Zafar, and A. Khan, "*Deep Neural Networks based Meta-Learning for Network Intrusion Detection*," No. February, 2023.

[3] V. Z. Mohale and I. C. Obagbuwa, "*A Systematic Review on the Integration of Explainable Artificial Intelligence in intrusion detection systems to enhancing transparency and interpretability in Cybersecurity*," No. January, pp. 1–10, 2025.

[4] M. E. Kahou, J. Yu, J. Perla, and G. Pleiss, "*How Inductive Bias in Machine Learning Aligns with Optimality in Economic Dynamics*," *arXiv:2406.01898v1*, 2024.

[5] I. H. Sarker, H. Janicke, A. Mohsin, A. Gill, and L. Maglaras, "*Explainable AI for Cybersecurity Automation, Intelligence and Trustworthiness in Digital Twin: Methods, Taxonomy, Challenges and Prospects*," *ICT Express*, Vol. 10, No. 4, pp. 935–958, 2024.

[6] S. D. A. Rihan, M. Anbar, and B. A. Alabsi, "*Meta-Learner-based Approach for Detecting Attacks on Internet of Things Networks*," *Sensors*, Vol. 23, No. 19, pp. 1–22, 2023.

[7] B. M. Lake and M. Baroni, "*Human-Like Systematic Generalization Through a Meta-Learning Neural Network*," *Nature*, Vol. 623, No. 7985, pp. 115–121, 2023.

[8] M. sohail Khan, K. DoHyeun, and F. Tila, "*Enhanced IoT Composition Architecture based on DIY Business Process Modeling: CoAP based Prototype*," *VFAST Trans. Softw. Eng.*, Vol. 10, No. 2, pp. 61–69, 2022.

[9] N. Dat-Thinh, H. Xuan-Ninh, and L. Kim-Hung, "*MidSiot: A Multistage Intrusion Detection System for Internet of Things*," *Wirel. Commun. Mob. Comput.*, Vol. 2022, No. December 2017, 2022.

[10] K. Albulayhi and F. T. Sheldon, "*An Adaptive Deep-Ensemble Anomaly-based Intrusion Detection System for the Internet of Things*," *2021 IEEE World AI IoT Congr. AIIoT 2021*, No. May 2021, pp. 187–196, 2021.

[11] J. Yang, C. Zhou, S. Yang, H. Xu, and B. Hu, "*Anomaly Detection based on Zone Partition for Security Protection of Industrial Cyber-Physical Systems,*" *IEEE Trans. Ind. Electron.*, Vol.

 ISSN:2302-8149
e-ISSN:2540-9719

65, No. 5, pp. 4257–4267, 2018.
[12] K. Albulayhi, A. A. Smadi, F. T. Sheldon, and R. K. Abercrombie, "*Iot Intrusion Detection Taxonomy, Reference Architecture, and Analyses*," *Sensors*, Vol. 21, No. 19, 2021.
[13] H. Alrubayyi, G. Goteng, M. Jaber, and J. Kelly, "*Challenges of Malware Detection in the IoT and a Review of Artificial Immune System Approaches*," *J. Sens. Actuator Networks*, Vol. 10, No. 4, 2021.
[14] T.-A. Review, A. M. Shhatha, and O. I. Alsaif, "*A Comprehensive Analysis of Approaches and Difficulties for Cybersecurity a Comprehensive Analysis of Approaches and Difficulties for Cybersecurity Threats- Article Review*," No. September, 2024.
[15] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "*Building an Intrusion Detection System using a Filter-based Feature Selection Algorithm*," *IEEE Trans. Comput.*, Vol. 65, No. 10, pp. 2986–2998, 2016.
[16] S. Jose, D. Malathi, B. Reddy, and D. Jayaseeli, "*A Survey on Anomaly based Host Intrusion Detection System*," *J. Phys. Conf. Ser.*, Vol. 1000, No. 1, 2018.
[17] A. Al-Taie and W. R. Baiee, "*A Comprehensive Study of Deep Learning Approaches for Predicting Reciprocal Traffic Dynamics and Climate Variability*," *Kufa J. Eng.*, Vol. 16, No. 3, pp. 22–42, 2025.
[18] S. P. K. Gudla, S. K. Bhoi, S. R. Nayak, K. K. Singh, A. Verma, and I. Izonin, "*A Deep Intelligent Attack Detection Framework for Fog-based IoT Systems*," *Comput. Intell. Neurosci.*, Vol. 2022, pp. 1–25, 2022.
[19] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "*Building an Efficient Intrusion Detection System based on Feature Selection and Ensemble Classifier*," *Comput. Networks*, Vol. 174, 2020.
[20] T. Su, H. Sun, J. Zhu, S. Wang, and Y. Li, "*BAT: Deep Learning Methods on Network Intrusion Detection using NSL-KDD Dataset*," *IEEE Access*, Vol. 8, pp. 29575–29585, 2020.
[21] Y. V. Kumar and K. Kamatchi, "*Anomaly based Network Intrusion Detection using Ensemble Machine Learning Technique*," *en. Int. J. Res. Eng. ...*, Vol. 6, No. 4, pp. 216–220, 2020.
[22] S. Rajagopal, P. P. Kundapur, and K. S. Hareesha, "*A Stacking Ensemble for Network Intrusion Detection using Heterogeneous Datasets*," *Secur. Commun. Networks*, Vol. 2020, 2020.
[23] R. Qaddoura, A. M. Al-Zoubi, I. Almomani, and H. Faris, "*A Multi-Stage Classification Approach for IoT Intrusion Detection based on Clustering with Oversampling*," *Appl. SCI.*, Vol. 11, No. 7, 2021.
[24] H. Alkahtani and T. H. H. Aldhyani, "*Intrusion Detection System to Advance Internet of Things Infrastructure-based Deep Learning Algorithms,*" *Complexity*, Vol. 2021, 2021.
[25] K. A. ElDahshan, A. A. A. AlHabshy, and B. I. Hameed, "*Meta-Heuristic Optimization Algorithm-based Hierarchical Intrusion Detection System*," *Computers*, Vol. 11, No. 12, 2022.
[26] 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.
[27] J. Jose and D. V. Jose, "*Deep Learning Algorithms for Intrusion Detection Systems in Internet of Things using CIC-IDS 2017 Dataset*," *Int. J. Electr. Comput. Eng.*, Vol. 13, No. 1, pp. 1134–1141, 2023.
[28] A. Almomani *et al.*, "*Ensemble-based Approach for Efficient Intrusion Detection in Network Traffic*," *Intell. Autom. Soft Comput.*, Vol. 37, No. 2, pp. 2499–2517, 2023.
[29] D. F. Abdulqadir, "*Security , Privacy and Availability Enhancement of Smart Internet of Things (IoT)*," 2023.
[30] N. Islam *et al.*, "*Towards Machine Learning based Intrusion Detection in IoT Networks*," *Comput. Mater. Contin.*, Vol. 69, No. 2, pp. 1801–1821, 2021.
[31] M. Zhong, Y. Zhou, and G. Chen, "*Sequential Model based Intrusion Detection System for IoT Servers using Deep Learning Methods*," *Sensors (Switzerland)*, Vol. 21, No. 4, pp. 1–21, 2021.
[32] Adi Ahmad, Riyan Maulana, and Muhammad Yassir, "*Cybersecurity Challenges in the Era of Digital Transformation a Comprehensive Analysis of Information Systems*," *J. Informatic, Educ. Manag.*, Vol. 6, No. 1, pp. 7–11, 2024.
[33] Z. Azam, M. M. Islam, and M. N. Huda, "*Comparative Analysis of Intrusion Detection Systems and Machine Learning-based Model Analysis Through Decision Tree*," *IEEE Access*, Vol. 11, pp. 80348–80391, 2023.

 ISSN:2302-8149
e-ISSN:2540-9719

[34] M. M. Taye, "*Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions*," *Computers*, Vol. 12, No. 5. 2023.
[35] J. Wu, W. Wang, L. Huang, and F. Zhang, "*Intrusion Detection Technique based on Flow Aggregation and Latent Semantic Analysis*," *Appl. Soft Comput.*, Vol. 127, p. 109375, 2022.
[36] H. Alazzam, A. Sharieh, and K. E. Sabri, "*A Feature Selection Algorithm for Intrusion Detection System based on Pigeon Inspired Optimizer*," *Expert Syst. Appl.*, Vol. 148, p. 113249, 2020.
[37] M. Wang, Y. Lu, and J. Qin, "*A Dynamic MLP-based DDoS Attack Detection Method using Feature Selection and Feedback*," *Comput. Secur.*, Vol. 88, 2020.
[38] I. F. Kilincer, F. Ertam, and A. Sengur, "*Machine Learning Methods for Cyber Security Intrusion Detection: Datasets and Comparative Study*," *Comput. Networks*, Vol. 188, No. October 2020, p. 107840, 2021.
[39] N. Yoshimura, H. Kuzuno, and Y. Shiraishi, "*DOC-IDS : A Deep Learning-based Method for Feature*," *Sensors*, 2022.
[40] A. Shenfield, D. Day, and A. Ayesh, "*Intelligent Intrusion Detection Systems using Artificial Neural Networks*," *ICT Express*, Vol. 4, No. 2, pp. 95–99, 2018.
[41] A. Bellat, K. H. Mansouri, and A. Raihani, "*Implementation of Artificial Neural Network for Optimization of a Wind Farm*," *Int. J. Tech. Phys. Probl. Eng.*, Vol. 13, No. 2, pp. 35–39, 2021.
[42] A. Shalaginov, S. Banin, A. Dehghantanha, and K. Franke, *Machine Learning Aided Static Malware Analysis: A Survey and Tutorial*, Vol. 70, No. August. 2018.
[43] J. Dumoulin *et al.*, "*UNICITY: A Depth Maps Database for People Detection in Security Airlocks*," *Proc. AVSS 2018 - 2018 15th IEEE Int. Conf. Adv. Video Signal-Based Surveill.*, 2018.
[44] H. Hussain, P. S. Tamizharasan, and C. S. Rahul, *Design Possibilities and Challenges of DNN Models: A Review on the Perspective of end Devices*, No. January. Springer Netherlands, 2022.
[45] Y. H. Alagrash, H. S. Mehdy, and R. H. Mahdi, "*A Review of Intrusion Detection System Methods and Techniques: Past, Present and Future*," *Int. J. Tech. Phys. Probl. Eng.*, Vol. 15, No. 1, pp. 11–17, 2023.