Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.065031

ARTICLE

Addressing Modern Cybersecurity Challenges: A Hybrid Machine Learning and Deep Learning Approach for Network Intrusion Detection

Khadija Bouzaachane[1,*], El Mahdi El Guarmah[2], Abdullah M. Alnajim[3] and Sheroz Khan[4]

[1]Department of Computer Sciences, Faculty of Sciences and Technology, L2IS, Cadi Ayyad University, Marrakech, 40000, Morocco
[2]Mathematics and Informatics Departement, Royal Air School of Aeronautics, L2IS, Marrakech, 40000, Morocco
[3]Department of Information Technology, College of Computer, Qassim University, Buraydah, 51452, Saudi Arabia
[4]Department of Electrical Engineering, College of Engineering and Information Technology, Onaizah Colleges, Onaizah 56447, Saudi Arabia

*Corresponding Author: Khadija Bouzaachane. Email: k.bouzaachane@uca.ac.ma or kbouzaachane@gmail.com

Received: 01 March 2025; Accepted: 19 May 2025; Published: 03 July 2025

ABSTRACT: The rapid increase in the number of Internet of Things (IoT) devices, coupled with a rise in sophisticated cyberattacks, demands robust intrusion detection systems. This study presents a holistic, intelligent intrusion detection system. It uses a combined method that integrates machine learning (ML) and deep learning (DL) techniques to improve the protection of contemporary information technology (IT) systems. Unlike traditional signature-based or single-model methods, this system integrates the strengths of ensemble learning for binary classification and deep learning for multi-class classification. This combination provides a more nuanced and adaptable defense. The research utilizes the NF-UQ-NIDS-v2 dataset, a recent, comprehensive benchmark for evaluating network intrusion detection systems (NIDS). Our methodological framework employs advanced artificial intelligence techniques. Specifically, we use ensemble learning algorithms (Random Forest, Gradient Boosting, AdaBoost, and XGBoost) for binary classification. Deep learning architectures are also employed to address the complexities of multi-class classification, allowing for fine-grained identification of intrusion types. To mitigate class imbalance, a common problem in multi-class intrusion detection that biases model performance, we use oversampling and data augmentation. These techniques ensure equitable class representation. The results demonstrate the efficacy of the proposed hybrid ML-DL system. It achieves significant improvements in intrusion detection accuracy and reliability. This research contributes substantively to cybersecurity by providing a more robust and adaptable intrusion detection solution.

KEYWORDS: Network intrusion detection systems (NIDS); NF-UQ-NIDS-v2 dataset; ensemble learning; decision tree; K-means; smote; deep learning

1 Introduction

The use of networks, as one of the most exciting developments across diverse domains of applications including smart-grids, has been resulting into gigantic volume of data. The smart-grids are designed as a result of information technology coupled with electrical engineering to ensure optimal utilization of assets to address shortcomings of existing electrical grids [1,2]. The use of mobile services has increased tremendously, making a substantial number of devices and equipment monitored and managed remotely in compliant manner, using specially designed applications. Consequently, this has resulted in cybersecurity risks at levels whose severity is proportional to the volume of data [3,4]. Attackers can exploit software shortcomings to

gain access to networks and systems, exploiting the continuous flow of network traffic by undermining ML-based detectors. These types of attacks can lead to data theft for the purpose of financial fraud, causing significant financial and reputation damage.

To address the changing nature of cybersecurity threats, Network Intrusion Detection Systems (NIDS) have proven to be area of intense research focus as a vital element of the defensive strategies being developed. The utility of traditional signature-based NIDS lies in their ability to identify known cyber threats; nevertheless, they lack the required adaptability to counter the constantly evolving strategies of cybercriminals, who primarily target healthcare companies, ports, airports, water, oil, and energy utility companies [5]. Apart from putting in place directives, policies, and regulations to better secure power grids, specific security measures are emerging in the domains of capacity building and technical aspects of security demanding careful authentication of the huge amount of data. There is thus a growing interest in more advanced techniques, including those based on Artificial Intelligence, as a means to mitigate the issues presented by equivalently intricate cyberattacks and thus to protect confidential information from cybersecurity crimes.

Computational intelligence methods are powerful tools with intelligent architectural platforms to recognize network traffic patterns and identify irregularities that indicate malicious activity. These techniques can utilize valid and comprehensive datasets, allowing them to discover novel attack vectors that are capable to adapt to evolving threats. However, the effectiveness of NIDS depends on the relevance and quality of the learning dataset. Legacy datasets often lack representation of contemporary attack signatures and community traffic characteristics. This hampers the models' ability to generalize and perform optimally in real-world scenarios [6].

Prior research has extensively investigated machine learning (ML) for Network Intrusion Detection Systems (NIDS), utilizing algorithms like neural networks, decision trees, SVM, ensemble learning, ..., etc., [7,8]. Despite these efforts, significant limitations persist. Many existing approaches are trained on obsolete or unrepresentative datasets, hindering their ability to detect novel threats [9]. Furthermore, high false positive/negative rates often plague these systems in real-world deployments. Crucially, computational and real-time performance constraints, essential for practical application, are frequently neglected. This work directly confronts these deficiencies by introducing a robust, scalable, and adaptive NIDS designed to meet the demands of modern, dynamic network environments.

The NF-UQ-NIDS-v2 dataset [8–10] was selected based on its structural advantages and superior representativeness compared to conventional datasets such as UNSW-NB15 and CICIDS2017. This dataset synthesizes heterogeneous network flows from four distinct sources (UNSW-NB15, BoT-IoT, CSE-CIC-IDS2018, and ToN-IoT), standardized into a uniform NetFlow format. Such standardization facilitates cohesive modeling of contemporary network environments and attack vectors. The resulting comprehensive corpus (75,987,976 flows) offers enhanced granularity, comprising 33.12% benign traffic and 66.88% diverse attack patterns, including IoT/cloud hybrid scenarios absent in single-source datasets. Unlike UNSW-NB15, which has been subject to criticism regarding temporal homogeneity and laboratory artifacts, NF-UQ-NIDS-v2 retains essential metadata while eliminating redundant elements (e.g., IP addresses, ports) through systematic preprocessing protocols. The dataset's modular structure enables comparative performance analysis across different network topologies through flow origin indicators. These characteristics address the methodological requirements for research focused on feature optimization and cross-network generalization capabilities, where traditional datasets exhibit coverage limitations and obsolete attack signatures.

1.1 Contribution of the Study

Our contribution to the field of intrusion detection can be summarized as follows, with a visual representation provided in Fig. 1.

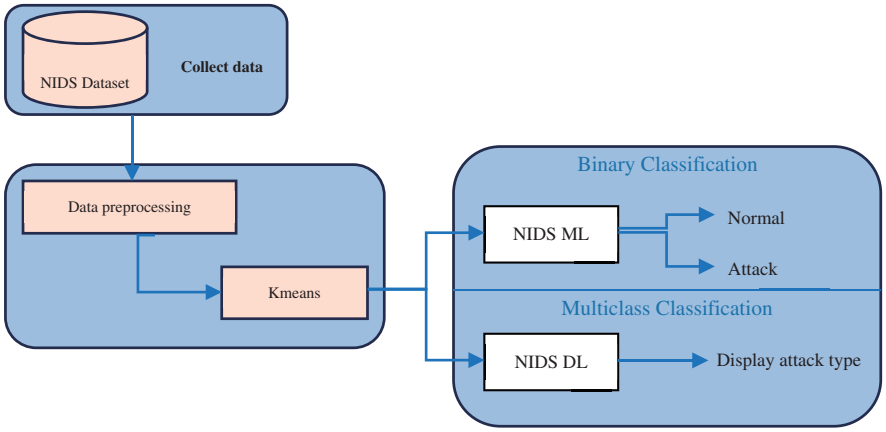

Figure 1: The proposed NIDS security farmwork

- Data Pre-processing and Analysis: A meticulous exploratory data analysis (EDA) of the NF-UQ-NIDS-v2 dataset was performed, employing the K-means clustering algorithm for dimensionality reduction and feature extraction. This step aimed to improve the dataset's quality and determine the most relevant attributes, providing the scientific community with high-performance intrusion detection data.
- Binary Classification: We undertook a detailed examination of several prominent machine learning (ML) algorithms, namely Support Vector Machine (SVM), Decision Trees (DT), Adaboost, XGBoost, Random Forest (RF), Gradient Boosting (GB), and Logistic Regression (LR), to assess their effectiveness in binary classification for distinguishing between normal and anomalous network traffic.
- Multi-class Classification: To further refine the classification of intrusion types, a diverse set of deep learning (DL) architectures, including Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), Deep Belief Network (DBN), Deep Neural Network (DNN), and Autoencoder, have been utilized to improve the accuracy and robustness of multi-class intrusion classification.

By integrating both Machine Learning (ML) and Deep Learning (DL), this combined strategy provides specific benefits depending on the type of detection required. We leverage a traditional ML model, known for its efficiency and reliability with well-defined features, for the crucial first step: binary classification. This allows for rapid and robust initial filtering, effectively distinguishing general attack traffic from normal network activity. However, identifying the specific type of attack requires a deeper analysis of complex patterns, a task where DL models excel. Therefore, we employ a DL component specifically for multi-class classification. Its ability to automatically learn intricate relationships within the network data enables more granular identification of various attack categories once an initial potential threat is flagged. This synergistic architecture allows each technique to operate in its area of strength—ML for efficient broad detection and DL for detailed threat identification—leading to a more comprehensive and effective NIDS than could be achieved using either method alone for both tasks.

Moreover, the proposed hybrid NIDS holds significant potential for real-world deployment in critical sectors. Environments such as healthcare and transportation are increasingly reliant on interconnected systems, making them prime targets for cyberattacks where disruptions can have severe consequences. Our

system's architecture, combining the efficiency of ML for initial threat detection with the nuanced classification capabilities of DL for identifying specific attack types, is well-suited to these demanding scenarios. For instance, in a healthcare setting, it could monitor network traffic for anomalies that might indicate attempts to access sensitive patient data or disrupt critical medical equipment. Similarly, in transportation networks, it could help secure communication channels and control systems against intrusions that could compromise safety or operational integrity. The ability to provide both rapid alerts and detailed attack information makes this approach a practical step towards enhancing cybersecurity resilience in vital operational environments.

1.2 Structure of Paper

The organization of the remaining sections of this paper is detailed below: Section 2 reviewing the relevant literature on Network Intrusion Detection Systems (NIDS) developed using machine and deep learning techniques, focusing on research published between 2020 and 2024. Section 3 outlines our methodology, including a detailed description of the evaluation metrics employed and provides a comprehensive exploration of the NF-UQ-NIDS-v2 dataset. Section 4 presents a discussion of the experimental results presented. In conclusion, Section 5 provides a summary of the main results and suggests avenues for future investigation.

2 Literature Review

A plethora of recent studies have investigated computational intelligence algorithms to enhance the intrusion detection rates. The following sub-sections delve into research conducted between 2020 and 2024. The significance of these studies is discussed based on the specific type of solution proposed. Table 1 presents a concise synthesis of the relevant literature.

2.1 NIDS with Machine Learning

Automatic learning has become a very powerful tool for developing effective NIDS models to detect network security intrusions. In [11], the authors aim to classify intrusions using Random Forest (RF), Linear Discriminant Analysis (LDA), Classification and Regression Trees (CART). The effectiveness of this work has been measured on the dataset KDD-CUP and compared with recent developments in the field.

The authors in [12–15] have used ten different popular Machine Learning (ML) algorithms, including decision tree (DT), random forest (RF), Support Vector Machine (SVM), K-means, and Expectation-Maximization (EM) for efficient network security and computer anomaly-based intrusion detection system (AIDS). These algorithms have been tested using the most recent highly imbalanced multiclass dataset CICIDS2017 that simulates real network attacks. The K-Nearest Neighbor (KNN)-AIDS, DT-AIDS and NB-AIDS models performed better and demonstrated greater ability to detect web attacks.

The authors in [16] have conducted a comprehensive review of the literature utilizing the CSE-CIC-IDS2018, UNSW-NB15, ISCX-2012, NSL-KDD, and CIDDS-001 datasets for developing Intrusion Detection Systems (IDSs). This work, carefully investigated how well standard machine learning methods—specifically Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Decision Trees (DT)—performed when applied to aforementioned datasets. The results obtained offer useful insights for creating intelligent intrusion detection systems (IDS) that use machine learning techniques.

The model presented in the study [17] utilizes the UNSW-NB15 dataset that has not only achieved superior accuracy compared to previous research works, but also has effectively identified all attack categories. To enhance the performance of classifiers, the research addresses the problem of class imbalance by using

SMOTE. Subsequently, the classification techniques of Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), K-Nearest Neighbors (KNN) and Artificial Neural Network (ANN) have been employed.

In addition, recent major works [18–21] aimed to investigate the impact of data balancing and feature selection on ML-based network intrusion detections. These studies have applied SMOTE or ADASYN to the IDS dataset to deal with the imbalanced dataset before applying ML algorithms.

2.2 NIDS with Deep Learning

Deep learning algorithms have shown unrivaled potential for Network Intrusion Detection Systems (NIDS). For instance, the authors in [22] have compared Recurrent Neural Net-works (RNN), Convolutional Neural Networks (CNN), Restricted Boltzmann Machines (RBM), Deep Neural Networks (DNN), Deep Belief Networks (DBN) and Deep Auto-encoders (DA) by using the CSE-CIC-IDS2018 dataset and the Bot-IoT. Their results indicated strong performance across these models, with accuracy exceeding 90%. The authors in [23] tackled network intrusion detection with imbalanced data. They used GANs to generate more data, then applied a combined CNN and BiLSTM model for detection. Testing on the CIC-IDS 2017 dataset, their proposed GAN-CNN-BiLSTM approach showed significantly better accuracy and efficiency compared to traditional methods (like Random Forest, Decision Tree) and other models (SVM, DBN, CNN, BiLSTM).

The paper [24] introduced a hybrid model using both CNNs and a Transformer encoder. After applying under-sampling and oversampling to balance the data, the CNN extracted spatial features while the Transformer captured temporal dependencies. This combined approach aimed to improve detection accuracy and minimize errors (FPs/FNs). Experiments on the NSL-KDD and CICIDS2017 benchmarks demonstrated superior performance, with higher accuracy and fewer false positives compared to existing leading models.

In their work, the authors in [25] presented an intrusion detection system (IDS) for Internet of Things (IoT) networks using deep learning with Long Short-Term Memory (LSTM). To improve the detection of rare attacks despite imbalanced data, the SMOTE over-sampling technique is integrated. The authors chose SMOTE for its simplicity and proven effectiveness in this field, compared to more complex alternatives like GANs. Evaluated on the CICIDS2017, NSL-KDD, and UNSW-NB15 datasets, this combined LSTM-SMOTE model showed superior performance compared to existing methods. This work offers a practical and adaptable approach to enhance the security of IoT networks.

The research in [26] introduced a new two-stage deep learning model for intrusion detection that integrates Long Short-Term Memory (LSTM) and Autoencoders (AE). Recognizing the need for better defenses against constantly evolving cyber threats, the authors evaluated their LSTM-AE approach using the CICIDS2017 and CSE-CIC-IDS2018 datasets. The study also considers other deep learning methods like DNNs and CNNs as points of comparison. Experimental results indicate that the proposed hybrid LSTM-AE model is effective at detecting attacks in modern network settings.

2.3 NF-UQ-NIDS-V2 Dataset

The NF-UQ-NIDS-v2 dataset has been constructed purposely for training and evaluating Network Intrusion Detection Systems (NIDS) [27]. The dataset is publicly available for research purposes, allowing researchers to develop and test new intrusion detection techniques. A brief overview of the research methodology used to investigate the NF-UQ-NIDS-V2 dataset has been provided. The authors in [28] suggested a hybrid model combining an Autoencoder (AE) and a Multi-Layer Perceptron (MLP) for DDoS attack detection. The AE first extracts important features from network traffic, and then the MLP uses this information to classify threats. This AE-MLP approach aims for better accuracy and fewer false alarms.

During tests on the recent NF-UQ-NIDS-V2 dataset, the model's performance yielded an accuracy rate of 99.98%. The results suggest that this hybrid method performs better than several comparable techniques. In [29], researchers introduced a new intrusion detection system that uses a Deep Neural Network. This system performed exceptionally well on the NF-UQ-NIDS-V2 dataset, a standard benchmark, achieving a reported accuracy exceeding 98%. The authors in [30] assessed an Extra Trees ensemble classifier using the merged NF-UQ-NIDS data. The classifier achieved 97.25% accuracy for binary classification and 70.81% accuracy for multiclass classification.

These studies have demonstrated that NIDS remain an active area of research, along with the growing adoption of computational intelligence techniques, such as Random Forest, Decision Trees, Support Vector Machines, K-Nearest Neighbor and Convolutional Neural Networks, Recurrent Neural Networks, Long Short-Term Memory networks, etc. They have emphasized the importance of addressing data imbalance and reducing reliance on labelled data for practical implementation reasons. Furthermore, because it is newer and more balanced than previous datasets, the NF-UQ-NIDS-v2 dataset has been recognized as an invaluable tool for training and evaluating NIDS models.

Table 1: Summary of the intrusion detection system

Ref.	Dataset used	Models	Classification type	Best accuracy
[11]	KDD'99CUP	RF	Multi-class	99.81%
[12]	CICIDS2017	ANN, DT, k-NN, RF, SVM, CNN	Multi-class	99.52%
[13]	CICIDS2017	Random forest (RF), linear support vector machine (LSVM), Gaussian Naive Bayes (GNB), and logistic regression (LG)	Multi-class	99% for RF
[14]	CICIDS2017	Random Forest (RF), Bayes Net (BN), Random Tree (RT), Naive Bayes (NB), J48, and Feature Selection	Multi-class	99.87%
[15]	CICIDS2017 UNSW-NB15, ICS	LR, GNB, KNN, DT, AdaB, RF, CNN, CNN-LSTM, LSTM, GRU, RNN, DNN	Multi-class	99% accuracy, precision, recall, and F-score for RF and CICIDS-2017
[16]	CSE-CIC-IDS2018, UNSW-NB15, ISCX-2012, NSL-KDD, CIDDS-001	SVM, K-Nearest Neighbors (KNN), and Decision Trees (DT),	Multi-class	99.92% DT (CSE-CIC-IDS 2018) 99.92% DT for NSL-KDD 100% ISCX 2012 100% DT CIDDS-001 99.84%. DT UNSW-NB15
[17]	UNSW-NB15	RF, DT, LR, KNN and ANN, SMOTE	Multi-class	95% RF
[18]	NSL-KDD, UNSW-NB15	DT, RF and KNN.	Multi-class	82.35% RF

(Continued)

Table 1 (continued)

Ref.	Dataset used	Models	Classification type	Best accuracy
[22]	CSE-CIC-IDS2018, Bot-IoT	DNNs RNNs CNNs RBMs DBNs DBMs DA	Multi-class	Over 90% for all models
[23]	CIC-IDS 2017	GAN-CNN-BiLSTM	Multi-class	96.32%
[24]	NSL-KDD CICIDS2017	CNN + Transformer	Multi-class	99.22 (NSL-KDD) 99.77 CICIDS2017)
[25]	CICIDS2017, NSL-KDD, UNSW-NB15	SMOTE + LSTM	Binary	99.34% (CICIDS2017) 99.75% (NSL-KDD) 98.31% (UNSW-NB15)
[26]	CICIDS-2017, CSE-CICIDS-2018	LSTM-AE	Multi-class	99.99% (CICIDS-2017) 99.10% (CSE-CICIDS-2018)
[28]	NF-UQ-NIDS-V2	AE-MLP	DDOS classification	99.98%
[29]	NF-UQ-NIDS-V2	GNNs	Multi-class	Over 98%
[30]	NF-UQ-NIDS	ExtraTrees	Binary Multi-class	97.25% 70.81%

3 Methodology

3.1 Data Gathring

A comparative study by the authors in [31] established a framework for assessing the quality of modern IDS datasets, encompassing dimensions such as variety, adequacy, balanced representation, attack diversity, protocol diversity and labeling accuracy. The results obtained indicated that the NF-UQ-NIDS-v2 dataset is the most comprehensive and of the highest quality among those evaluated, suggesting its adoption as the primary dataset for IDS development to create more robust, accurate and generalizable models against various threats.

The NF-UQ-NIDS-V2 dataset comprises network traffic data for intrusion detection from diverse network setups and attack settings, integrating four datasets: NF-UNSW-NB15, NF-BoT-IoT, NF-TON-IoT, and CSE-CIC-IDS2018. The NetFlow-based NF-UNSW-NB15 dataset is labeled with attack categories and further classified into nine subcategories: Benign, Fuzzers, Analysis, Backdoor, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms. The NF-BoT-IoT dataset, an IoT NetFlow-based dataset, is derived from the BoT-IoT dataset, with features extracted from pcap files and flows labeled with five attack categories: Benign, Reconnaissance, DDoS, DoS, Theft. The NF-ToN-IoT dataset, another NetFlow-based IoT dataset, utilizes ToN-IoT pcaps and includes categories like Benign, Backdoor, DoS, DDoS, Injection, MITM, Password, Ransomware, Scanning, and XSS. The NF-CSE-CICIDS2018 dataset, a NetFlow-based dataset, was created using the original CSE-CIC-IDS2018 pcap files with a distribution of Benign, BruteForce, Bot, DoS, DDoS, Infiltration, and Web Attacks. The merged datasets form a comprehensive NIDS dataset emulating diverse attack scenarios and network flows, totaling 75,987,976 logs categorized as benign (33.12%) and malicious (66.88%), with specific attack category distribution detailed in Table 2.

Table 2: Distribution of attack categories

Class	Count
Worms	164
Shellcode	1427
Analysis	2299
Theft	2431
Ransomware	3425
MITM	7723
Generic	16,560
Backdoor	18,978
Fuzzers	22,310
Exploits	31,551
Infiltration	116,361
Brute force	123,982
Bot	143,097
Injection	684,897
Password	1,153,323
XSS	2,455,020
Reconnaissance	2,633,778
Scanning	3,781,419
Dos	17,875,585
DDoS	21,748,351
Benign	25,165,295

3.2 Data Preparation

To manage computational demands of the large NF-UQ-NIDS-v2 dataset, each constituent dataset (NF-UNSW-NB15-v2, NF-BoTIoT-v2, NF-ToN-IoT-v2, NF-CSE-CIC-IDS2018-v2) was processed independently. Attack classes were categorized as unique or shared, with shared classes consolidated as in Table 3.

Table 3: Sample of unique and common attack classes within the NF-UQ-NIDS-v2 dataset, for benign and DDos class, categorized by their source datasets

Class	Original dataset	Number of instances	Total number of instances
Benign	NF-UNSW-NB15-v2	2,295,222	25,165,295
	NF-BoT-IoT-v2	135,037	
	NF-ToN-IoT-v2	6,099,469	
	NF-CSE-CIC-IDS2018-v2	16,635,567	
DDoS	NF-BoT-IoT-v2	18,331,847	21,748,351
	NF-ToN-IoT-v2	2,026,234	
	NF-CSE-CIC-IDS2018-v2	1,390,270	

3.3 Class Exploration

To address class overlap issues mentioned in [32], the "Exploit" class was excluded due to diverse attack instances. Furthermore, the "Password" class, encompassing both brute-force and network sniffing attacks [33], was also removed due to heterogeneity, especially the inclusion of network sniffing attacks not represented elsewhere and differing significantly from brute-force methods. Web attacks, exploiting application layer vulnerabilities, were categorized under "Web Attack," including Injection, XSS, Infiltration, Fuzzers, Generic, and Analysis, based on OWASP's top ten critical web vulnerabilities [34].

3.4 Instance Sampling

To expedite model training, the dataset initially underwent Mini-Batch K-means clustering, a computationally efficient variant of the K-means algorithm, to procure a representative sample for each class. This algorithm distinguishes itself by processing data in smaller, manageable subsets. Specifically, only a fraction of the complete dataset is loaded into Random Access Memory (RAM) during each iteration, thereby substantially diminishing memory demands. This methodological choice not only accelerates the algorithm's execution time but also enhances the efficient allocation of computational resources. Furthermore, cluster sampling was employed to select instances from individual CSV files, each containing homogeneous class instances. This probabilistic sampling technique entailed the aggregation of original instances into clusters based on their Euclidean distances, followed by the proportional extraction of instances from each cluster.

In practical implementation, CSV files corresponding to identical classes were segregated into distinct folders. Subsequently, these files were concatenated and ingested into a 'dataset' DataFrame structure. For illustrative purposes, the Denial of Service (DoS) class, which comprises 17,875,585 instances, is visually represented in Fig. 2. Following this, the MiniBatchKMeans algorithm, sourced from the scikit-learn library, was implemented to discern cluster centroids and subsequently group the instances.

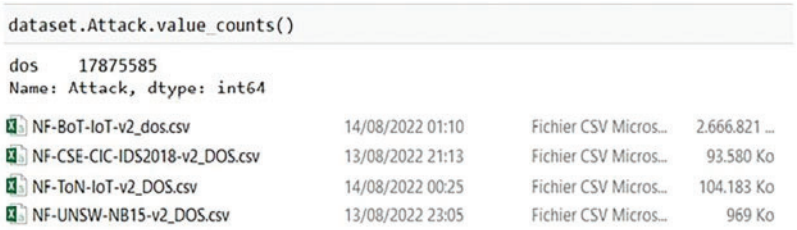

Figure 2: List of DOS class instances with folder structure overview

The parameter 'n-clusters', representing the number of clusters, constitutes a hyperparameter that can be optimized through diverse algorithmic approaches. However, for datasets exceeding 20,000 instances, a cluster count of 1000 was determined to be sufficiently effective in enhancing computational efficiency within this study. The 'random-state' parameter, which dictates the initialization of cluster centroids, was empirically set to a value of 0.3.

The DOS dataset was subsequently sampled to yield approximately 2500 instances. The K-means clustering process, coupled with the sampling operation, was iteratively applied to each file containing instances of the same class.

Notably, the number of clusters was dynamically adjusted based on dataset size: reduced to 500 for instances below 20,000, to 250 for instances below 10,000, and further reduced to 125 for instances below

5000. Subsequent to the sampling operation, a more balanced class distribution was observed, as evidenced by the following diagrams, Fig. 3.

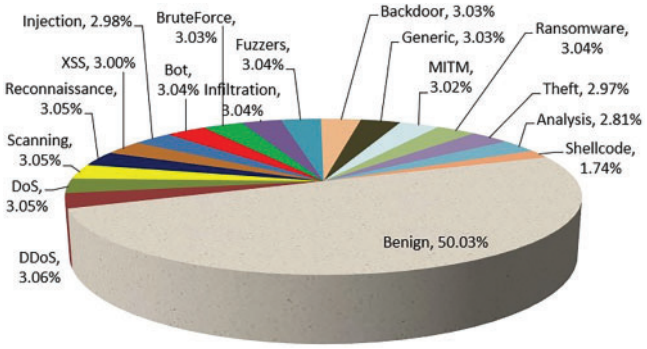

Figure 3: K-means clustering-based sampling to create novel classes

3.5 Normalization

The subsequent step is to normalize the features after the sampling process. This conversion is required to address the significant differences in scale among the various feature attributes. By normalizing these ranges to a common interval, we ensure that all attributes contribute equally to the classification process. We have chosen to normalize the minimum and maximum of the instances using the Eq. (1):

$$x\text{scaled} = (\mathrm{x} - \mathrm{x}_{min}) / (\mathrm{x}_{max} - \mathrm{x}_{min}) \tag{1}$$

where: x is the original feature value, x_{min} and x_{max} are the minimum and maximum values of the feature. This normalization technique entails scaling of each attribute value to a range between 0 and 1, effectively creating a uniform scaling across all attributes.

3.6 Statistical Learning

This research has explored intrusion detection (ID) using a two-prong strategy. First, we have focused on binary classification, employing a set of well-regarded machine learning models: Random Forest, Logistic Regression, AdaBoost, Gradient Boosting, XGBoost, Support Vector Machine, and Decision Trees. Next, we have addressed the more complex challenge of multi-class intrusion detection (ID). To capture the evolving nature of network traffic, we have utilized powerful models such as LSTM, DNN, GRU, Autoencoder and DBN. Before delving into multi-class analysis, we have prepared the data. To address the uneven distribution of different types of intrusions, we have employed the Synthetic Minority Oversampling Technique (SMOTE) to balance the dataset [19].

3.7 Benchmarking

The quantitative metrics are essential for a thorough evaluation of the algorithms. These metrics provide objective measures of an algorithm's effectiveness. The NF-UQ-NIDS-v2 dataset is used Within this investigation to assess the performance of NIDSs, employing a combination of metrics.

3.7.1 Confusion Matrix

The confusion matrix allows for a complete evaluation of an IDS's ability to accurately classify network traffic as either attack or normal, resulting in four potential outcomes [35], as presented in Table 4.

Table 4: Confusion matrix for predicted vs. actual

		Predicted class	
		Normal	**Attack**
Actual class	Normal	True negative (TN)	False positive (FP)
	Attack	False negative (FN)	True positive (TP)

3.7.2 Precision

Precision, as expressed in Eq. (2), measures the proportion of actual positive cases correctly identified by the model out of all cases the model predicted as positive.

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FP}} \tag{2}$$

3.7.3 Recall

Recall is defined by the following Eq. (3):

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

It calculates the percentage of actual positive cases that the model accurately predicted as positive.

3.7.4 Accuracy

Accuracy, a metric for evaluating performance, is calculated as the proportion of correct predictions, considering both true positives (TP) and true negatives (TN). See Eq. (4) for the specific formula.

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TN} + \boldsymbol{TP}}{(\boldsymbol{FN} + \boldsymbol{FP} + \boldsymbol{TP} + \boldsymbol{TN})} \tag{4}$$

3.7.5 F1-Score

The F1-score seeks to establish an equilibrium between Precision and Recall, as illustrated by the Eq. (5).

$$\text{F1} - \text{score} = \frac{2 * (Precision * Recall)}{(Precision + Recall)} \tag{5}$$

3.7.6 Training and Prediction Time

This research evaluated two key performance indicators: training time and prediction time. Training time encompasses the duration required for the model to learn from the provided training data. This includes the time spent on data processing, hyperparameter tuning, and the subsequent model building phase. Conversely, prediction time measures the length of time needed for the trained model to generate predictions for novel, previously unseen data instances.

Prediction time is of paramount importance for real-time intrusion detection systems, where swift response times are crucial for effectively mitigating potential threats.

3.7.7 Macro-Averaged AUC-ROC

Macro-Averaged AUC-ROC provides a measure of the model's performance across all classes, where each class contributes equally to the final score, without considering class size or relative importance.

3.7.8 Micro-Averaged AUC-ROC

Micro-Averaged AUC-ROC represents the overall performance of the model across all classes, considering the frequency of each class. It gives more weight to larger classes.

3.7.9 Computational Cost

Computational cost in deep learning quantifies the necessary computing resources—primarily processing operations and memory—for a model's training and inference phases. Factors like the model's size (parameter count), training time, and prediction speed directly indicate this cost.

4 Results

To assess accurately the performance of the proposed models, it is imperative to employ instances that were not utilized during the training process. This requires the division of our dataset of 81,951 into two distinct subsets: a training set of 80% of the overall dataset and a test set of 20% of the overall dataset. The size of the training data is 65,560 and the size of the testing data is 16,391. Training of our proposed models has been implemented on a workstation with the following specifications: Intel(R) Core(TM) i7-6600U, CPU@2.60 GHz (4 CPUs), 16Go in RAM memory, and we used for this purpose Python programming language and the Scikit-learn, keras, Numpy, and pandas libraries.

4.1 Binary Classification

This section evaluates several machine learning algorithms for binary attack classification. The models tested include Logistic Regression (LR), Support Vector Machines (SVM), Decision Trees (DT), Random Forests (RF), AdaBoost, Gradient Boosted Trees (GBT), and XGBoost. The dataset used for this task features a balanced distribution between classes, as shown in Fig. 4. Before training, the dataset was thoroughly shuffled to help ensure unbiased model learning and improve the models' ability to generalize to new data. This shuffling was performed using the shuffle function available in the Keras library.

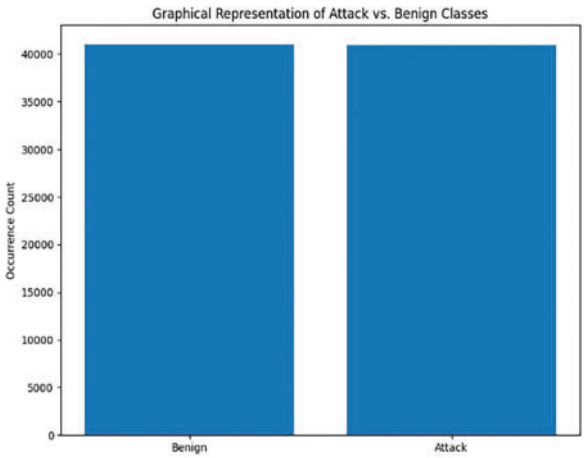

Figure 4: Graphical representation of attack vs. benign classes

We trained each model using fixed hyperparameters. For the Support Vector Machine (SVM), the regularization parameter was set to 50 to balance the classification margin and errors. We used the default settings for the Logistic Regression (LR) and Decision Tree (DT) models. For Random Forest (RF), the number of trees was set to 11. For the boosting algorithms, the number of boosting stages was set to 100 for Gradient Boosting (GBT), 100 for AdaBoost, and 240 for XGBoost. As detailed in Table 5, the training process resulted in the following accuracies: 0.99 for SVM, 0.86 for LR, 1.0 for DT, AdaBoost, and RF, and 0.99 for GBT and XGBoost.

Table 5: Summary of training process

Models	Training time (s)	Accuracy
SVM	1.44427	0.99
LR	0.00417	0.86
DT	1.49690	1.0
RF	0.01253	1.0
GB	0.29613	0.99
Adaboost	0.28366	1.0
XGboost	0.01257	0.99

The performance metrics summarized in Table 6 clearly show that Logistic Regression (LR) performed significantly worse than the other models, achieving only 85% across accuracy, precision, recall, and F1-score. More critically, LR resulted in a very high number of false negatives (1129 instances), meaning it frequently misclassified actual attacks as benign traffic. This level of performance is inadequate for a reliable intrusion detection system.

Table 6: Summary of results of testing process

	LR	SVM	DT	RF	Adaboost	GBT	XGB
Time of prediction (s)	0.010	6.025	0.0109	0.031	0.848	0.08	0.045
Accuracy	0.85	0.998	0.999	1.0	0.999	0.999	0.9999
Precision	0.858	0.998	0.999	1.0	0.999	0.999	0.999
Recall	0.858	0.998	0.999	1.0	0.999	0.999	0.999
F1-score	0.858	0.998	0.999	1.0	0.999	0.999	0.999
FN count	1129	12	3	0	1	0	0

In contrast, all other evaluated models demonstrated excellent performance. SVM, DT, AdaBoost, GBT, and XGB all achieved accuracy, precision, recall, and F1-scores of 99.8% or 99.9%. While highly effective, these models still produced a small number of false negatives (ranging from 0 for RF to 3 for DT). Random Forest (RF) stood out as the top performer, achieving perfect scores across all metrics: 100% accuracy, precision, recall, and F1-score. Crucially, RF resulted in zero false negatives, meaning it correctly identified every single attack instance in the test set, Fig. 5.

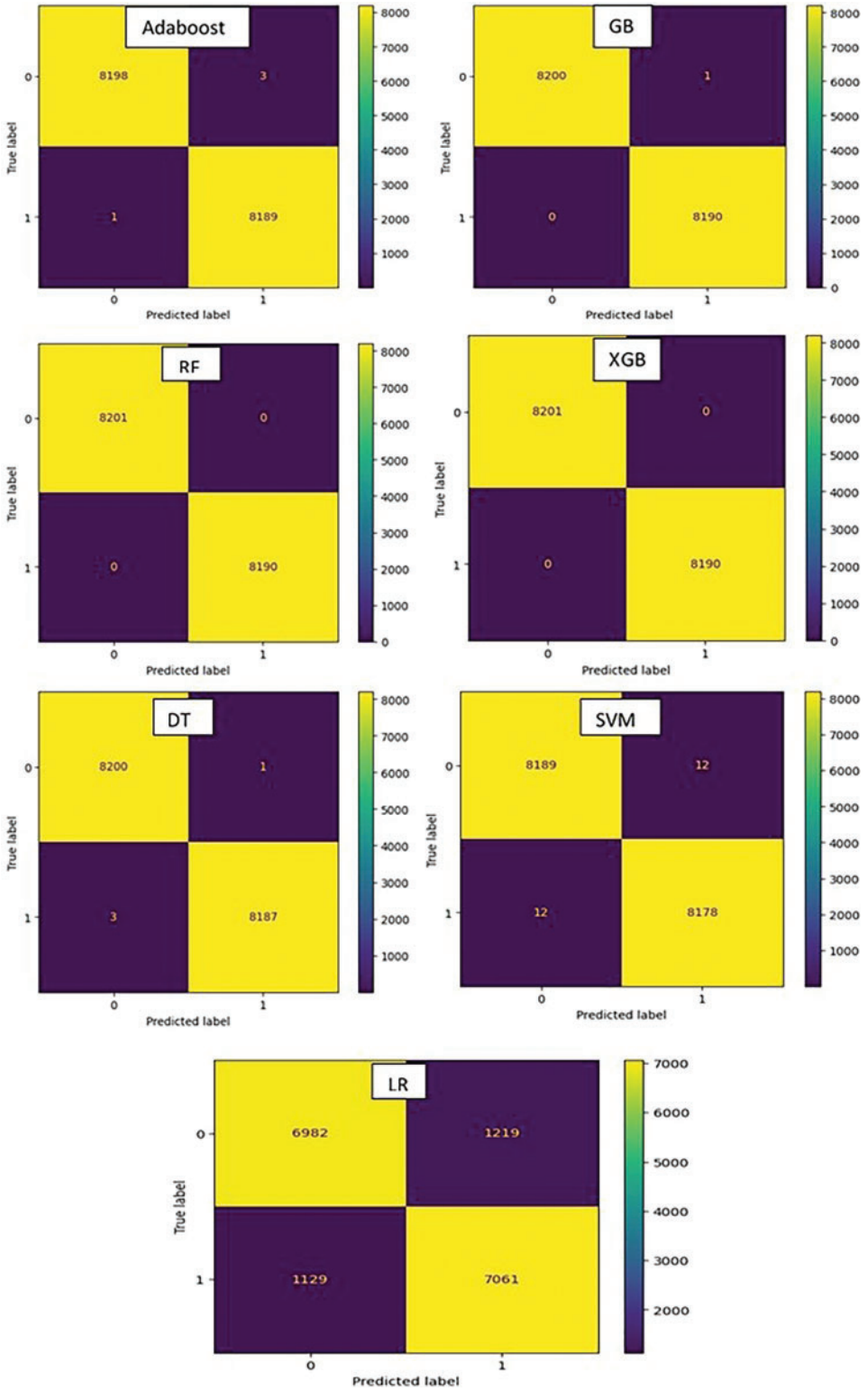

Figure 5: Confusion matrix

When considering computational efficiency (Tables 5 and 6), LR offered the fastest training time (0.00417 s), but its poor accuracy makes this irrelevant. SVM was notably slow to train (1.44427 s). RF exhibited moderate training (0.01253 s) and prediction times (0.031 s), which are practical for real-world applications where high accuracy and reliability are paramount.

4.2 Multi-Class Classification

This section evaluated five different deep learning models—LSTM, GRU, DNN, DBN, and Autoencoder—for the complex task of classifying network traffic into 13 distinct categories, including various attack types and benign traffic.

A key challenge was the significant imbalance in the dataset, with the 'Shellcode' attack class being heavily underrepresented (Table 7). To ensure fair and robust model training, we applied the SMOTE technique to synthetically generate more 'Shellcode' examples, creating a more balanced distribution for the models to learn from (as shown in Fig. 6).

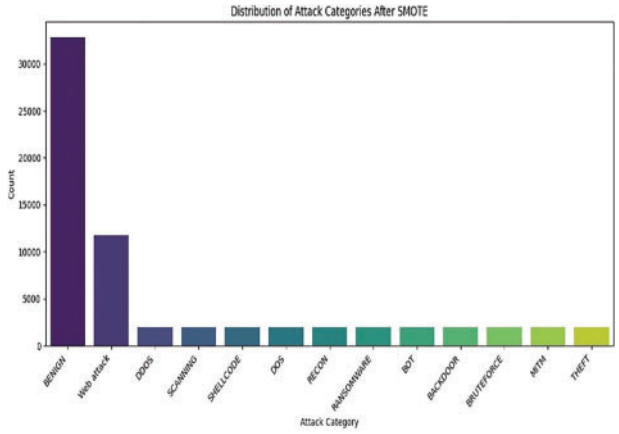

Figure 6: Class distribution after SMOTE application

Table 7: Distribution of attack categories

Class	Count
Benign	41,002
Web attacks	14,667
DDos	2506
Reconnaissance	2498
DOS	2497
Ransomware	2493
Bot	2489
Brute force	2485
Backdoor	2485
Scanning	2500
MITM	2471
Theft	2431
Shellcode	1427

During training, we monitored both training and validation accuracy to guide the learning process for each model (Fig. 7). In the final testing phase, the Deep Belief Network (DBN) model clearly stood out. It achieved the highest test accuracy at 99% (Fig. 8a), significantly outperforming the other four models evaluated.

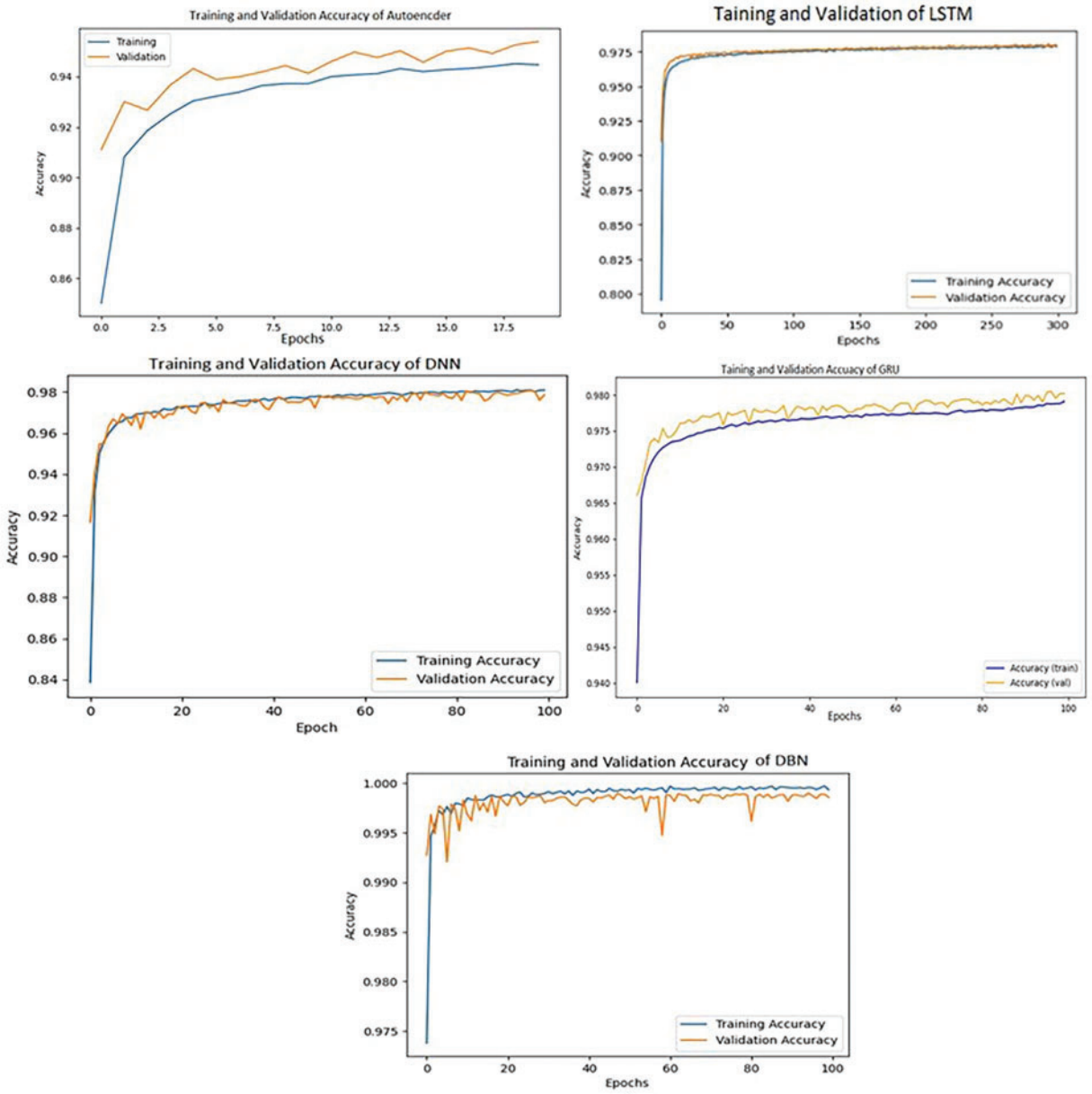

Figure 7: Training and validation accuracy of DL models

Furthermore, The Receiver Operating Characteristic (ROC) curve analysis provided further strong evidence of the DBN's effectiveness (Fig. 8b). Both the micro-averaged and macro-averaged Area Under the Curve (AUC) scores were nearly perfect at 0.9996. This highlights the DBN's outstanding ability to reliably distinguish between different attack types and normal traffic with very high sensitivity and specificity.

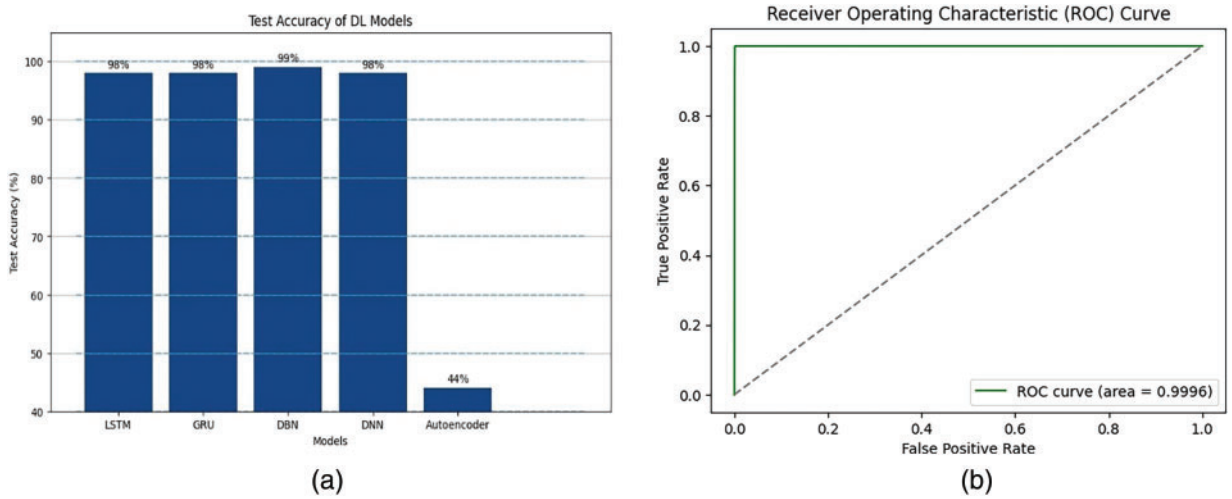

Figure 8: (a) DL model performance: test accuracy comparison; (b) ROC curve analysis of DBN model for attack type

We assessed the computational characteristics of each model (summarized in Table 8). This revealed significant differences:

- The Autoencoder model was the fastest to train (134 s) and offered instant predictions (1 s), but had the highest number of parameters (32,503), indicating greater model complexity.
- The Deep Belief Network (DBN), after its pre-training phase, resulted in a final classifier with the fewest parameters (10,765). It also achieved very fast predictions (1 s), with a moderate training time (527 s).
- Recurrent models, LSTM and GRU, showed considerable complexity (28,237 and 21,389 parameters) and required the longest training times (587 s and 540 s, respectively), though their prediction speeds remained quick (3–4 s).
- The standard Deep Neural Network (DNN) had moderate complexity (14,605 parameters) and fast predictions (2 s), but unexpectedly took the longest to train in our setup (660 s).

Table 8: Performance comparison of deep learning models for multi-class attack classification

DL models	GRU	LSTM	DBN	DNN	Autoencoder
Training time (s)	540	587	527	660	134
Time of prediction (s)	4	1	3	2	1
Computational cost (parameters)	21,389	28,237	10,765	14,605	32,503

5 Discussion and Conclusions

This study aimed to identify an effective machine learning model for Network Intrusion Detection Systems (NIDS). After evaluating several algorithms on NF-UQ-NIDS-v2 dataset, we selected a hybrid approach combining Random Forest (RF) and Deep Belief Network (DBN). The results showed that RF performs very well on the initial task of binary classification (separating normal from attack traffic), achieving high accuracy "100%". This efficient filtering step means only suspected malicious traffic is passed to the DBN model. The DBN specializes in the more complex multiclass classification needed to determine the specific type of attack. This two-stage architecture leverages the RF's speed and accuracy for initial filtering, and the DBN's capability to model complex relationships for detailed attack categorization.

Despite its potential, the hybrid RF-DBN method involves inherent limitations that must be carefully addressed for effective real-world deployment. Error propagation stands out as the most critical limitation. Specifically, if the initial Random Forest (RF) incorrectly flags benign traffic as an attack (a False Positive), it triggers unnecessary processing by the Deep Belief Network (DBN). This can lead to false alerts if the DBN fails to correct the initial misclassification. To mitigate this, one practical approach is to train the DBN specifically on the False Positives generated by the RF, enabling it to potentially rectify errors from the first stage.

Furthermore, the issue of interpretability arises. While Random Forests are already less transparent than simple decision trees, DBNs are notoriously opaque "black boxes." Consequently, understanding precisely why the system classified a particular network flow as a specific type of attack becomes difficult. This lack of clarity complicates the analysis of security alerts and can hinder trust in the system's decisions. To address this challenge, tools like SHAP or LIME can be employed. These techniques provide local explanations for the model's outputs by identifying the features that most influenced a given prediction.

Another significant concern relates to the system's ability to generalize to new, previously unseen attacks (Zero-Day threats). Because the model is trained only on known attack types present in the dataset, it may struggle to correctly identify entirely novel threats or significant variations of existing attacks that were not part of its training experience. Several strategies can help address this:

- Firstly, implementing a process for regularly retraining both models using recent network data that includes newly identified threats is crucial.
- Secondly, supplementing the RF-DBN architecture with an unsupervised model, such as an Autoencoder, can be beneficial. Such models excel at detecting highly unusual behavior, even if it doesn't match any known attack signature, effectively acting as a safety net.
- Finally, continuous monitoring of key performance indicators, like detection rates and false alert frequencies, is essential. Any degradation in performance should be investigated promptly, as it might signal the emergence of new, undetected threats.

Although we used SMOTE to address the dataset's initial class imbalance, we need to consider how the model would perform with *severely* unbalanced data. In real-world situations, some critical attacks are extremely rare, creating much greater imbalance than addressed in this study. This extreme rarity poses a challenge: the initial Random Forest stage might fail to detect these infrequent attacks, and even if they reach the Deep Belief Network, learning their patterns effectively can be difficult, despite oversampling. Consequently, the system's ability to detect these rare but important threats could be reduced. To handle such scenarios better, future work could explore more advanced strategies beyond standard SMOTE. Options include adaptive sampling techniques (like ADASYN), cost-sensitive learning (which gives more weight to errors on rare classes), or ensemble methods specifically designed for imbalanced data. Evaluating the model under these demanding, highly imbalanced conditions is crucial for ensuring its reliability in real-world network environments

In conclusion, while the hybrid RF-DBN model demonstrates significant potential for intrusion detection due to its specialized two-stage structure, its successful deployment hinges on acknowledging and proactively managing its inherent limitations. Implementing practical solutions, such as those suggested for error propagation, interpretability, and zero-day detection, will be vital for building a robust, reliable NIDS capable of adapting to the constantly evolving threat landscape.

Acknowledgement: The authors would like to express their gratitude to all the editors and anonymous reviewers for their insightful comments and suggestions. They also wish to thank the College of Computer, Qassim University in Saudi Arabia, L2IS laboratory at Cadi Ayyad University in Marrakech and the College of Engineering and Information

Technology at Onaizah Colleges in Saudi Arabia for their support. The researchers would like to thank Onaizah College of Engineering and Information Technology, Onaizah Colleges, for funding the publication of this project.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Khadija Bouzaachane, Abdullah M. Alnajim; draft manuscript preparation: Khadija Bouzaachane, El Mahdi El Guarmah; funding acquisition and supervision: Khadija Bouzaachane, El Mahdi El Guarmah, Abdullah M. Alnajim, Sheroz Khan; review: Khadija Bouzaachane, El Mahdi El Guarmah, Abdullah M. Alnajim, Sheroz Khan. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The authors confirm that the data supporting the findings of this study are available within the article in Section 2.3, in this link: https://staff.itee.uq.edu.au/marius/NIDS_datasets/ [27] (accessed on 18 May 2025).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Radoglou Grammatikis P, Sarigiannidis P, Efstathopoulos G, Panaousis E. *ARIES*: a novel multivariate intrusion detection system for smart grid. Sensors. 2020;20(18):5305. doi:10.3390/s20185305.
2. Yan Y, Qian Y, Sharif H, Tipper D. A survey on cyber security for smart grid communications. IEEE Commun Surv Tutor. 2012;14(4):998–1010. doi:10.1109/SURV.2012.010912.00035.
3. AlHaddad U, Basuhail A, Khemakhem M, Eassa FE, Jambi K. Ensemble model based on hybrid deep learning for intrusion detection in smart grid networks. Sensors. 2023;23(17):7464. doi:10.3390/s23177464.
4. Kavre M, Gadekar A, Gadhade Y. Internet of things (IoT): a survey. In: 2019 IEEE Pune Section International Conference (PuneCon); 2019 Dec 18–20; Pune, India. p. 1–6. doi:10.1109/punecon46936.2019.9105831.
5. Zhang C, Patras P, Haddadi H. Deep learning in mobile and wireless networking: a survey. IEEE Commun Surv Tutor. 2019;21(3):2224–87. doi:10.1109/COMST.2019.2904897.
6. Abdullahi M, Baashar Y, Alhussian H, Alwadain A, Aziz N, Capretz LF, et al. Detecting cybersecurity attacks in Internet of Things using artificial intelligence methods: a systematic literature review. Electronics. 2022;11(2):198. doi:10.3390/electronics11020198.
7. Al-Riyami S, Lisitsa A, Coenen F. Cross-datasets evaluation of machine learning models for intrusion detection systems. In: Proceedings of Sixth International Congress on Information and Communication Technology. Singapore: Springer; 2021. p. 815–28. doi:10.1007/978-981-16-2102-4_73.
8. Apruzzese G, Pajola L, Conti M. The cross-evaluation of machine learning-based network intrusion detection systems. IEEE Trans Netw Serv Manag. 2022;19(4):5152–69. doi:10.1109/tnsm.2022.3157344.
9. Vourganas IJ, Michala AL. Applications of machine learning in cyber security: a review. J Cybersecur Priv. 2024;4(4):972–92. doi:10.3390/jcp4040045.
10. Luay M, Layeghy S, Hosseininoorbin S, Sarhan M, Moustafa N, Portmann M. Temporal analysis of NetFlow datasets for network intrusion detection systems. arXiv:2503.04404. 2025.
11. Saranya T, Sridevi S, Deisy C, Chung TD, Khan MKAA. Performance analysis of machine learning algorithms in intrusion detection system: a review. Procedia Comput Sci. 2020;171(4):1251–60. doi:10.1016/j.procs.2020.04.133.
12. Maseer ZK, Yusof R, Bahaman N, Mostafa SA, Foozy CFM. Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset. IEEE Access. 2021;9:22351–70. doi:10.1109/access.2021.3056614.
13. Al Lail M, Garcia A, Olivo S. Machine learning for network intrusion detection—a comparative study. Future Internet. 2023;15(7):243. doi:10.3390/fi15070243.

14. Kurniabudi K, Stiawan D, Idris MYB, Bamhdi AM, Budiarto R. CICIDS-2017 dataset feature analysis with information gain for anomaly detection. IEEE Access. 2020;8:132911–21. doi:10.1109/access.2020.3009843.
15. Elmrabit N, Zhou F, Li F, Zhou H. Evaluation of machine learning algorithms for anomaly detection. In: 2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security); 2020 Jun 15–19; Dublin, Ireland. p. 1–8. doi:10.1109/cybersecurity49315.2020.9138871.
16. Kilincer IF, Ertam F, Sengur A. Machine learning methods for cyber security intrusion detection: datasets and comparative study. Comput Netw. 2021;188:107840. doi:10.1016/j.comnet.2021.107840.
17. Ahmed HA, Hameed A, Bawany NZ. Network intrusion detection using oversampling technique and machine learning algorithms. PeerJ Comput Sci. 2022;8(1):e820. doi:10.7717/peerj-cs.820.
18. Barkah AS, Selamat SR, Abidin ZZ, Wahyudi R. Impact of data balancing and feature selection on machine learning-based network intrusion detection. JOIV: Int J Inform Visualization. 2023;7(1):241–8. doi:10.30630/joiv.7.1.1041.
19. Jiao X, Li J. An effective intrusion detection model for class-imbalanced learning based on SMOTE and attention mechanism. In: 2021 18th International Conference on Privacy, Security and Trust (PST); 2021 Dec 13–15; Auckland, New Zealand. p. 1–6. doi:10.1109/pst52912.2021.9647756.
20. Ahsan R, Shi W, Corriveau JP. Network intrusion detection using machine learning approaches: addressing data imbalance. IET Cyber Phys Syst Theory Appl. 2022;7(1):30–9. doi:10.1049/cps2.12013.
21. Karatas G, Demir O, Sahingoz OK. Increasing the performance of machine learning-based IDSs on an imbalanced and up-to-date dataset. IEEE Access. 2020;8:32150–62. doi:10.1109/access.2020.2973219.
22. Ferrag MA, Maglaras L, Moschoyiannis S, Janicke H. Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study. J Inf Secur Appl. 2020;50(1):102419. doi:10.1016/j.jisa.2019.102419.
23. Li S, Li Q, Li M. A method for network intrusion detection based on GAN-CNN-BiLSTM. Int J Adv Comput Sci Appl. 2023;14(5):507–15. doi:10.14569/ijacsa.2023.0140554.
24. Chen H, You GR, Shiue YR. Hybrid intrusion detection system based on data resampling and deep learning. Int J Adv Comput Sci Appl. 2024;15(2):121–35. doi:10.14569/ijacsa.2024.0150214.
25. Sayegh HR, Dong W, Al-madani AM. Enhanced intrusion detection with LSTM-based model, feature selection, and SMOTE for imbalanced data. Appl Sci. 2024;14(2):479. doi:10.3390/app14020479.
26. Hnamte V, Nhung-Nguyen H, Hussain J, Hwa-Kim Y. A novel two-stage deep learning model for network intrusion detection: lSTM-AE. IEEE Access. 2023;11:37131–48. doi:10.1109/access.2023.3266979.
27. Sarhan M, Layeghy S, Portmann M. Towards a standard feature set for network intrusion detection system datasets. Mob Netw Appl. 2022;103(1):108379. doi:10.1007/s11036-021-01843-0.
28. Adeniyi O, Sadiq AS, Pillai P, Aljaidi M, Kaiwartya O. Securing mobile edge computing using hybrid deep learning method. Computers. 2024;13(1):25. doi:10.3390/computers13010025.
29. Gu Z, Lopez DT, Alrahis L, Sinanoglu O. Always be pre-training: representation learning for network intrusion detection with GNNs. arXiv:2402.18986v1. 2024.
30. Sarhan M, Layeghy S, Moustafa N, Portmann M. NetFlow datasets for machine learning-based network intrusion detection systems. In: Big data technologies and applications. Cham, Switzerland: Springer International Publishing; 2021. p. 117–35. doi: 10.1007/978-3-030-72802-1_9.
31. Pavlov A, Voloshina N. Dataset selection for attacker group identification methods. In: 2021 30th Conference of Open Innovations Association FRUCT; 2021 Oct 27–29; Oulu, Finland. p. 171–6.
32. Zoghi Z, Serpen G. UNSW-NB15 computer security dataset: analysis through visualization. Secur Priv. 2024;7(1):e331. doi:10.1002/spy2.331.
33. Derbyshire R, Green B, Prince D, Mauthe A, Hutchison D. An analysis of cyber security attack taxonomies. In: 2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW); 2018 Apr 23–27; London, UK. p. 153–61. doi:10.1109/EuroSPW.2018.00028.
34. Ben Fredj O, Cheikhrouhou O, Krichen M, Hamam H, Derhab A. An OWASP top ten driven survey on web application protection methods. In: Risks and security of internet and systems. Cham, Switzerland: Springer International Publishing; 2021. p. 235–52. doi: 10.1007/978-3-030-68887-5_14.
35. Gulshan K. Evaluation metrics for intrusion detection systems—a study. Int J Comput Sci Mobile Appl. 2014;2(11):11–7.