scientific reports

OPEN

Signature-based intrusion detection using machine learning and deep learning approaches empowered with fuzzy clustering

Usama Ahmed[1], Mohammad Nazir[2], Amna Sarwar[3], Tariq Ali[4]✉, El-Hadi M. Aggoune[4], Tariq Shahzad[5] & Muhammad Adnan Khan[6]✉

Network security is crucial in today's digital world, since there are multiple ongoing threats to sensitive data and vital infrastructure. The aim of this study to improve network security by combining methods for instruction detection from machine learning (ML) and deep learning (DL). Attackers have tried to breach security systems by accessing networks and obtaining sensitive information.Intrusion detection systems (IDSs) are one of the significant aspect of cybersecurity that involve the monitoring and analysis, with the intention of identifying and reporting of dangerous activities that would help to prevent the attack.Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), Long Short-Term Memory (LSTM), and Artificial Neural Network (ANN) are the vector figures incorporated into the study through the results. These models are subjected to various test to established the best results on the identification and prevention of network violation. Based on the obtained results, it can be stated that all the tested models are capable of organizing data originating from network traffic. thus, recognizing the difference between normal and intrusive behaviors, models such as SVM, KNN, RF, and DT showed effective results. Deep learning models LSTM and ANN rapidly find long-term and complex pattern in network data. It is extremely effective when dealing with complex intrusions since it is characterised by high precision, accuracy and recall. Based on our study, SVM and Random Forest are considered promising solutions for real-world IDS applications because of their versatility and explainability. For the companies seeking IDS solutions which are reliable and at the same time more interpretable, these models can be promising. Additionally, LSTM and ANN, with their ability to catch successive conditions, are suitable for situations involving nuanced, advancing dangers.

Network security risen to the forefront of importance in this age of rapid technological development and growing reliance on digital infrastructure. There are always new possibilities and new cyber risks the ever changing digital world. As the digital sentinels that keep an eye on network traffic and sniff out malicious activity, Intrusion Detection Systems (IDS) have taken on a crucial function in this environment. This article introduces the topic of how modern machine and deep learning approaches might strengthen these IDS, hence improving network security and better adapting to the difficulties of the digital age.

The need of bolstering network security using novel methods is emphasized as this chapter opens with a review of its essential components. Focuses on the evolution of network security over time and discusses the flaws of modern intrusion detection approaches. These restrictions make this study necessary, which causes us to seek new approaches that utilize machine learning and deep neural networks. This research is motivated by the enhanced complexity of cybersecurity threats; shortcomings of conventional intrusion detection methodologies; and above all, the need to adopt a preventive security approach to network vulnerability. The study is important because it holds the potential to revolutionalize network security is now implemented. The research of the

[1]Department of Artificial Intelligence, School of Systems and Technology, University of Management and Technology, Lahore 54700, Pakistan. [2]Department of Computer Science and Information Technology, The Islamia University of Bahawalpur, Bahawalpur, Pakistan. [3]Department of Computer Science, University of Wah, Wah Cantt, Pakistan. [4]Artificial Intelligence and Sensing Technologies (AIST) Research Center, University of Tabuk, Tabuk 71491, Saudi Arabia. [5]Department of Computer Engineering, COMSATS University Islamabad, Sahiwal Campus, Sahiwal 57000, Pakistan. [6]Department of Software, Faculty of Artificial Intelligence and Software, Gachon University, Seongnam-si 13120, Republic of Korea. ✉email: teshaq@ut.edu.sa; adnan@gachon.ac.kr

present work offers original insights on the methodological approaches such as proposing the new data fusion method, developing the adaptive cybersecurity model and the advanced intrusion detection systems. Altogether, these contributions enhance the security of networks since they present practical solutions through which organizations may effectively secure their assets.

Intrusion detection systems (IDS)

Intrusion Detection System (IDS) is also a security system in the environment that monitors for any behaviors that indicate suspicious activities in a network or system, for instance attempts to penetrate for unauthorized access[1]. They constantly search for anything suspicious including network traffic, system logs and many more. There are broadly two categories of IDS, namely the signature-based IDS and the anomaly-based IDS, the former of which involves the searching for patterns of assaults that are well known and the latter of which involves the searching for deviations from normality[2].

Early approaches to intrusion detection

Intrusion detection was considered as a new technology during early days when computing and communication systems were developed. The first models of intrusion detection relied on manually coded rules and heuristics[3].

Alerts would be raised whenever the input data resembled previous attacks, usually in terms of the identified patterns or signatures that well-trained security personnel and administrators would define. Despite these early systems being somewhat effective in defending against already known risks, they were no match for enhanced threats[2].

Transition to digital networks

The environment of intrusion detection as noted has changed significantly after the coming of the digital network[4]. DSince computer networks became more of a medium for the exchange of data between the businesses, both the communication and the traffic also increased immensely. That is because, with the shift, intrusion detection has had to be more scalable as well as much automated. the IDS continued to change in its information present in the networks and increasing threats that emerged, new models of the IDS were then formed[5]. This change has however created an opportunity to enhance new complex intrusion detection systems including machine and deep learning algorithms to meet the current cybersecurity gap.

Anomaly detection techniques

Almost all fields including cybersecurity employ a class of methods called anomaly detection methods to identify unusual occurrences in data. The goal of anomaly-based low level strategies interrelating to intrusion detection is to classify situations which are characterized by occurrence of activity that is not typical within a certain given context. The following are the typical steps involved in such methods: Compile data which could be used as reference information regarding routine activities. This information may be related to human behavior, system activities, or network traffic. Use the information gathered to fine-tune a model that accurately represents typical behavior. Statistical models, clustering algorithms, and machine learning models are all examples of popular methods[2]. Anomaly Detection: The model performs continuous analysis of incoming data after it is deployed in a real-world setting. Any information that considerably deviates from the established norm is identified as suspicious. Notifying Security Personnel or Triggering Automated Actions to Mitigate Potential Threats Once anomalies are detected, the system can issue warnings or trigger replies[6].

Challenges in anomaly-based detection

Anomaly-based detection has many benefits, such as the capacity to spot brand new and changing threats, but it also has its fair share of difficulties. A high percentage of false positives are produced by anomaly detection systems because even statistically normal events may be misidentified as abnormalities. Keeping detection rates high while decreasing false positives in such systems can be difficult to fine-tune. Data Imbalance: In many real-world situations, normal conduct considerably surpasses criminal activity, resulting in an unbalanced set of data. This may compromise the model's sensitivity to detect extremely unusual events[7]. Changing Environments and Networks: Anomaly-based systems may have trouble adjusting to new circumstances. Maintaining their efficacy calls for regular updates and training. The inability to interpret the results of complicated machine learning models might slow down the decision-making and response time necessary in the face of anomalies. The issue of scalability arises when data volumes increase. Real-time anomaly detection requires massive data processing and analysis capabilities. Successful implementation of anomaly-based intrusion detection systems relies on overcoming these obstacles so that cybersecurity professionals can rely on the systems' insights and put them to use.

Protecting the privacy, security, and availability of digital assets is the primary goal of these systems, which are built to constantly monitor and analyze network traffic in order to detect and counteract any suspicious or harmful activity[8]. The development of new ID systems shows the necessity of adjusting security measures to meet the evolving nature of cyber threats. There have been three major eras in the development of ID systems so far:

Traditional signature-based systems

Early intrusion detection systems relied heavily on signatures. These programs utilized previously established attacking patterns, or signatures. Malicious data was marked as such when it was detected on a network and matched a signature. These systems fared well against typical threats, but they were helpless against zero-day exploits and constantly changing dangers[9].

Anomaly-based systems
The shortcomings of signature-based systems motivated the development of anomaly-based alternatives. These programs determined what constituted "normal" network activity and reported any changes as possible security threats. They were more flexible but needed a lot of adjusting and they sometimes gave false positives.

Machine and deep learning-based systems
A new age of intrusion detection was born with the introduction of machine learning and deep learning. These systems are very good at spotting both common and uncommon threats because they use sophisticated algorithms to learn and adapt on their own based on how a network operates. There is some examples that are Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) in the deep learning model that can detect several patterns in the network traffics and improved the detection accuracy to a higher level[10]. The background study pointed out some of the challenges of modern ID systems such as the growing volume and complexity of network traffic data, the use of evasion techniques by the adversaries and the need to offer a real-time intrusion detection. This study is therefore important given the emergence of ID systems and more so, the challenges they encounter. Their task is to enhance the intrusion detection and develop the systems that can prevent and respond to diverse numbers of threats based on the latest machine learning and deep learning strategies in advance[11]. In an age where data accuracy and privacy become vital this research intends to provide solutions to improve networking security and digital support.

The rationale behind this research relates to the fact that there is an increasing threat that permeates networks hence the need to advance knowledge in this field. Increased accessibility and connectivity through the adoption of the web and digital technology pose security challenges As the capacity for innovation and access has grown networks have weakened and become susceptible to increasingly persistent and complex external attacks. When it comes to the identification and prevention of such specific types of attack, intrusion detection system provides very little assistance. The motivation for this study stems from the fact that recent advances in machine learning and deep learning can be utilised to develop effective proactive and self-learning intrusion detection systems[12]. Facing the urgency of the data's integrity and availability nowadays, cybersecurity plays a crucial role in it, and these technologies will contribute to keeping the data protected from any threat. Another reason for this study is the hope of making long-lasting contributions in the area of network security, given that it could enhance our capability to resist the advancement of smarter and smarter hackers[13].

This study has significant implications for network security because it was the first to use cutting-edge machine and deep learning methods to significantly improve the efficiency of intrusion detection systems[14]. Given the constantly evolving and diversifying nature of threats in the cyberspace, the new ideas of this work form a robust line of defense. It assists in protecting the main information and databases of a firm without the weaknesses of traditional security measures making it easier to keep data secure, private and easily accessible to those who need it. The findings and the procedures discussed herein do not only enhance the efficiency of intrusion detection, but also establish the framework for an adaptive security system[15]. As cybersecurity becomes paramount for each person, businesses, and the entire society this strategic approach helps to maintain the dynamic view on the network protection against current and new types of threats. In the fields of network security and intrusion detection, this paper makes a number of significant contributions. This study offers novel solutions to extant issues that are experienced due to continually emerging and dynamic cyber threats by leveraging on enhanced machine and deep learning mechanisms. Preliminary intrusion detection models, new multi-source data fusion techniques, and some of the framework to develop an adaptive Cybersecurity framework are some of the major findings of this research. These contributions combined enhance the efficiency of the network and fortification of defenses against the increasing complex threat environment.

Advanced intrusion detection models
Three intrusion detection algorithms based on machine learning and deep learning that are introduced here are more effective than conventional approaches. These models contribute to enhance the overall protection of digital networks against a rich set of threats including new and complex attacks.[16].

Multi-source data fusion techniques
The current study introduces some innovative data fusion methodologies that if employed in IDSs can help improve the accordance and reliability of IDSs through utilizing some data sources such as network traffic data and log files and traces of system calls. All these methods offered here pose a practical and efficient approach to solving this issue of poor network security.

Adaptive cybersecurity framework
The study presents a dynamic architecture of cybersecurity to enable meeting the continually emerging threats experienced by IDSs. By implementing this improvement, we ensure that our networks will remain safe and secure despite increasing threats As for our contributions, they all contribute to improving the overall field of network security by advancing intrusion detection technologies, overcoming the limitations of traditional approaches, and encouraging more dynamic and effective development of the cybersecurity environment.

Digital networks are vulnerable because traditional intrusion detection systems have a hard time reliably identifying novel and sophisticated infiltration attempts. In an increasingly interconnected and vulnerable digital world, there is a pressing need to improve network security by leveraging the power of machine and deep learning techniques to develop robust and adaptable intrusion detection systems capable of proactively identifying emerging threats, minimizing false positives, and protecting the confidentiality and integrity of vital data. Let X be a dataset representing network traffic data, where $X = \{x_1, x_2, \ldots, x_N\}$, and each x_i is a feature vector representing network traffic attributes. The goal is to design an intrusion detection model, M, parameterized by Θ, that can classify network traffic instances as either normal ($y = 0$) or malicious ($y = 1$), where y represents the ground truth labels. The model M is defined as $M(x_i; \Theta)$ and aims to minimize the following objective function:

$$\min_{\Theta} = \sum_{i=1}^{N} L(M(x_i; \Theta), y_i) + \lambda R(\Theta) \tag{1}$$

where:

L: Loss function, measuring the discrepancy between the predicted and actual labels.
$R(\Theta)$: Regularization term to prevent overfitting.
λ: Regularization parameter, controlling the trade-off between fitting the data and regularization.
N: Total number of network traffic instances.

The problem formulation seeks to optimize the model's parameters, (Θ), to maximize detection accuracy while minimizing false positives and false negatives in identifying network intrusions.

This study has the following research objectives:

1. To evaluate and compare the performance of various machine and deep learning models, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and ensemble methods, for intrusion detection, and to develop optimization strategies that enable real-time detection of novel threats.
2. To investigate the fusion of multiple data sources, including network traffic data, log files, and system call traces, to enhance the accuracy and robustness of intrusion detection systems. This objective includes exploring data preprocessing techniques and fusion algorithms.
3. To design and implement intrusion detection systems that can adapt to the dynamic nature of cyber threats in modern networks. This involves the creation of ways and means of updating the countermeasures against risks in real-time and enhancing the protection of data for the longest time possible.

Literature review

In intrusion detection, machine learning methods have proven to be rather requisite in the mainstays of all approaches. Some of the methods which have been implemented are decision trees, support vector machines, and k-nearest neighbors[17]. A further advantage mentioned is that models might be trained with new data to maintain their performance, which can counter changing threats[18]. For instance, decision tree based models are well applicable to the anomaly detection because they can easier recognize the differences between useful and destructive patterns of networks behavior. SVMs rely on the ability of separating the data space by a hyper plane to classify the occurrence and differentiate between safe and risky behaviors. As the K-nearest neighbors method defines it, comparing data is the way to find intruders. Strengthen the security of the network with the help of a large number of machine learning algorithms.

The capacity of deep learning approaches to automatically learn and extract complex patterns from vast datasets has propelled them to the forefront of machine learning. One popular application of deep learning architectures in intrusion detection is the use of Convolutional Neural Networks (CNNs), while Recurrent Neural Networks (RNNs) are also used[16]. CNNs excel at handling image-based attacks or multidimensional arrays of network traffic data. In contrast, RNNs excel at processing sequences and are hence well-suited to time-series network data. Improved precision in identifying complicated intrusion patterns is a direct result of their ability to capture temporal and geographical relationships in data.

In the evolution of swarm intelligence, some adaptations have been brought closer to a mix of exploitation and exploration models to achieve a much better correlation between as well as global and local research. The general PSO and ABC algorithms have been made adaptive and have been applied to actual life problems including scheduling and structural design. However, to date, new approaches are still being investigated continuously in an attempt at providing better algorithms especially for the constrained problem and for the multi-objective one. The Optimization Algorithm proposed in this paper[19] is called the Greylag Goose Optimization, or GGO for short, and is a swarm-based heuristic model based on the motion of geese. GGO emulates the pattern of formation of geese to improve exploration as well as exploitation, with more effective searches realized. It has been verified on different benchmarks and engineering design problems and outperforms the existing optimization algorithms, especially in solving engineering design problems.

Meta-heuristic algorithms have become a favorite tool in solving optimization problems for their simplicity and ability to avoid convergence to local optima. Such algorithms, derived from natural analogues, differ in effectiveness due to the corresponding operators and heuristics used within a system. Among phenomena such as swarm intelligence and evolutionary algorithms, the most utilized in engineering, data mining, and machine learning. The Puma Optimizer, or PO for short, is a new algorithm that has just been created and is based on the life and intelligence of pumas with new exploration and exploitation approaches[20]. In contrast, PO has some other unique characteristics, including a flexible phase-changing mechanism of the phase that can adapt to the specific problem of optimization for changing problems and increasing the efficiency for different optimization problems. The efficiency of the proposed PO algorithm has been tested on 23 standard benchmark functions, CEC2019 functions, and several machine learning tasks to perform clustering and feature selection tasks. Measurements indicate that the proposed PO is better than some current optimal solution search methods:

in optimization tests, PO is better in 27 out of 33 problems and in clustering tests, PO provides better quality solutions in 7 out of 10 datasets. Also, PO has promising performance while addressing issues related to multi-layer perceptron (MLP) networks and community detection. These outcomes reveal that PO is a successful optimization tool for real-world applications for generating a stable solution with good opportunities to advance research work based on multi-objective optimization and civil engineering problems in the future.

One of the most important issues encountered in data preprocessing for building classification models is feature selection, which is aimed at dimensionality reduction. The "Apple Perfection" study used the Waterwheel Plant Algorithm (WWPA) to determine the key attributes affecting the quality of apples and succeeded in a considerable reduced computational load and an average error rate of 0.52153. This study showed what had been mentioned earlier that feature selection can bring about an improvement in the result by considering only the relevant attribute; Therefore, the logistic regression model that was used in the study had a classification accuracy of 88.63 %. In the same way, in intrusion detection, domain adapted feature selection can help improve the identification and categorization of the intrusion. By applying such methods as bWWPA or some other one, vital network features can be chosen, noise can be eliminated, and the machine learning and deep learning models' performance can be maximized.[21] Despite being a relatively new field, intrusion detection is now regarded as an essential component of network security that utilizes both classical and contemporary mathematical approaches. Intrusion Detection Systems (IDS) have been enhanced in the current world through new advancements within machine learning and deep learning methodologies. Self-organizing nature-inspired metaheuristic algorithms have shown some capabilities in the model optimization of IDS. This paper[22] proposed the Greylag Goose Optimization (GGO) Algorithm based on the birds' 'V' formation structure to improve exploration/exploitation. As a result, GGO was judged to be more effective than other methods in optimizing feature selection for IDS and solving complex engineering problems. It is important to use GGO to enhance ML models for IDS[22]. The synergism between dynamic strategies such as FbOA and the DL architectures represents the best practices profile. As a result, FbOA successfully applies high-dimensional and nonlinear problems, improving their convergence and making the whole process more robust as compared to the operations of football teams. In IDS contexts, such algorithms could help in adjusting deep learning models such as LSTM or CNN, applicable in analyzing sequential and complex data from network traffic.[23] Most of the reviewed nature-inspired optimization techniques can be incorporated into the proposed hybrid ML and DL models approach used in the current study. The usage of such optimization techniques can even advance model effectiveness, cut down the processing time, and increase the detection ratio. For example, incorporating GGO or PO during the feature selection step could enhance interpretability and part resistance against the zero-day attack.

AI and ML have huge possibilities of revolutionizing the education sector through the delivery of differentiated learning end products, identifying struggling learners, and designing content in response to student needs. Several prior works have investigated AI and ML approaches to education with most works using predictive modeling and feature selection for enhancing learning. Some of these algorithms include Particle Swarm Optimization (PSO) and Whale Optimization Algorithm (WOA) wherein to make good predictions; the most relevant features are selected in Educational Data Mining. These techniques have shown promising results in such areas as students' performance prediction and learning adapted models. In this study[24], Grand Canyon University examined the effects of AI on students' learning with the aid of the following techniques: bPSO-Guided WOA for feature selection and Linear Regression for predicting. The results depict a comparatively small average error and an exceptionally small MSE underscoring the importance of these AI methods for forecasting and decision-making among educators. Examples include such advancements in AI as a way of creating more consistent and fuller education environments in the process of improving both teaching and learning.

Deep learning models are increasingly being adopted as key application tools in smart city development that enable traffic forecasting and streamlining of city frameworks. Among all these models such as VGG16Net, VGG19Net, GoogLeNet, ResNet-50, and AlexNet have been used for traffic data analysis where ResNet-50 and AlexNet exhibited higher accuracy sensitivity and specificity in traffic prediction[25]. Further conclusions using valid statistical measures, ANOVA, and Wilcoxon Signed Rank Test are displayed and these show a significant difference between the performance of the specified groups of models while Alex Net yields an impressive accuracy of 0.93178. These outcomes signify that AI-powered solutions in the challenging field of traffic can improve the quality of urban mobility and support smart city projects; thus, organically feeding urbanists and policymakers with vital information on creating better and more efficient infrastructures.

Clustering and auto encoders are two examples of unsupervised learning techniques that might be useful for spotting anomalies.Anomalies depending on the level of network activity can be easily identified using a clustering algorithm like the K-means[26]. Popularity of auto-encoder increase due to their capacity to reconstitute original data that can reconstruct "normal" network traffic with any discrepancies indicating a breach. These methods effective for autonomously identifying new attacks.

Similar to how people learn and adapt, reinforcement learning (RL) honors the constantly changing embodiment of network security. An agent in the RL-based intrusion detection works in a way that it tries to select a safety action that will lead to a highest long-term reward. Such decisions[27] include the ability to block or allow network traffic, turn on or off security policies, or response to a particular threat. The next information is acquired through 'learning from experience,' where the agent modifies the strategy with reference to information, which in this case is provided by the network. This is the reason why, flexibility of RL is found very much useful in the field of intrusion detection.The traditional rule-based systems face difficulties to keep up with the ever-changing nature of cyber threats. RL agents have the capacity to learn optimal security rules and adapt new attack techniques in real-time through interaction with the network environment.Agent's job in RL-based intrusion detection make safety choices optimize long-term rewards. Blocking allowing network traffic, modifying security policies, responding specific threats are examples of such decisions. The agent acquires knowledge by trial and error, constantly modifying its approach in light of the network's input. The flexibility of

RL is what makes it so effective in the field of intrusion detection. Traditional rule-based systems have difficulty keeping up with the ever-changing nature of cyber threats[28].

The use of ensemble learning techniques for intrusion detection in networks has shown promising results[29]. Random Forests and Gradient Boosting are two examples of ensemble approaches that use the combined power of numerous models to boost detection precision. In essence, they pool the results of various models in order to make more informed decisions with less chance of error. Using a random sampling of characteristics and data from the training set, Random Forests construct numerous decision trees. The combined results from these trees form the basis for the ultimate choice[30]. This method strengthens the model's resistance to noise and outliers while enhancing its capacity for generalization. In contrast, the Gradient Boosting method builds a robust predictive model by iteratively constructing weak models that correct the shortcomings of their predecessors. This repeated technique yields an effective ensemble model that is particularly good at representing subtle interconnections in data. Ensemble methods have a reputation for being able to manage datasets in which malicious events are vastly outnumbered by benign ones. In the field of network security, they are used to examine heatmaps and other traffic data visualizations. Since attacks tend to look different from regular network traffic, CNNs are incredibly useful for spotting these abnormalities and identifying patterns within these heatmaps.

When it comes to dealing with sequential data, RNNs provide an alternative to LSTMs. In contrast to LSTMs' strength in processing long-range relationships, RNNs' strength lies in their ability to process and remember short-term dependencies and recent occurrences[31]. When it comes to intrusion detection, RNNs shine when they're being put to use to identify multiple simultaneous threats. For spotting attacks that other models might overlook, their capacity to assess short time frames is crucial. RNNs improve intrusion detection systems' real-time capabilities by taking recent network behavior into account.

When it comes to learning and reconstruction, auto encoders are the best type of neural network to use. Auto encoders are used to reconstruct regular network data for use in intrusion detection[32]. Input data is encoded into a lower-dimensional representation before being decoded back into its original form. When abnormal network traffic is introduced, the resulting reconstruction will also be abnormal. Since auto encoders may spot outliers in patterns without relying on preset attack signatures, they are ideally suited for detecting unknown or unique attacks.

Protecting user anonymity and data is essential for any secure network. With federated learning, models may be trained using data from numerous distributed sources without compromising on data's locality[33]. It enables companies to work together on threat detection and intrusion detection without disclosing private network information. Each region has its own dataset for training models, and only the most recent modifications to these models are shared between regions. In settings where regulatory compliance and data protection are of the utmost importance, such as healthcare and finance, this privacy-preserving method is vital, all the while reaping the benefits of the collective intelligence to improve network security.

Graph Neural Networks (GNNs) have become an indispensable resource for ensuring the safety of computer networks. Modern network topologies are notoriously complicated, with many devices and systems interconnected in complex ways that make intrusion detection particularly difficult. Graph-based data representation is harnessed by GNNs to solve this problem. In a network graph, nodes represent devices and edges represent relationships between them[34]. For this purpose, GNNs can learn and propagate information across this graph structure to detect abnormalities that could be hard to identify using more traditional techniques. One of GNNs' main advantages is that they can perform the analysis from two aspects at the same time, that is, the global features of the network data and the local features. As an ability to bring much-needed additional information into the decision-making process, XAI has become essential for intrusion detection systems. High levels of accuracy presented by a number of machine and deep learning techniques used for intrusion detection are often associated with the lack of explicit understanding of how and why specific decisions were made. But in real-world security systems where security professionals are to rely on and understand why the current warning is being given, this obscuration pose a real challenge. To address this issue, the main XAI methods have been created, LIME and SHAP[35]. These methods generate justifications for model predictions, providing insight into the considerations that went into a given choice. XAI techniques, for instance, can shed light on the traits and patterns that lead to the detection of an intrusion. After gathering this data, security analysts will be better equipped to make choices, conduct investigations, and fine-tune security plans.

By assisting with the analysis of log files and textual data provided by network devices, Natural Language Processing (NLP) has expanded its reach into network security[36] System logs, firewall logs, and event logs are all examples of logs that offer useful information for intrusion detection but are also typically large and poorly organized. Natural language processing methods save the day by extracting and organizing this textual material, rendering it analyzable. The ability to recognize abnormalities or patterns in log data is an important NLP application for intrusion detection. Natural language processing (NLP) can analyze log entries and spot suspicious behavior like attempted intrusions or configuration changes. In the case of early intrusion detection, where the identification of security problems in a timely manner might reduce possible damage, this study is of paramount importance Detecting threats and associated hazards in a network is facilitated by their ability to incorporate past information and probabilistic reasoning to evaluate the likelihood of various network events. The capacity to model uncertainty is one of Bayesian Networks' main capabilities. Managing uncertainty is particularly important in the context of intrusion detection, when both false positives and negatives can have severe repercussions[37].

Homomorphic encryption is a revolutionary method for protecting private information while still processing calculations on it. Homomorphic encryption provides a novel approach to protecting the privacy of network information in the context of intrusion detection. The adaptability of Markov models is one of its main selling points. Since circumstances in the network and the means of attacks vary they will be able to find themselves new opportunities and carry on with their malicious activity.

Ensemble clustering works as follows: in an attempt to improve the analysis of network traffic and hence the recognition of intrusions, the pieces unite various clustering algorithms. Algorithms for cluster analysis attempt to classify data into groups with comparable characteristics. Effective segmentation of network traffic data into meaningful groups, each representing a particular network behavior, is possible through the use of an ensemble of clustering algorithms such as K-means, DBSCAN, and hierarchical clustering. Ensemble clustering is a more general method of intrusion detection as it takes into consideration numerous different patterns in the network. Because different phases of an intrusion may display distinct behaviors, this is especially helpful in spotting multi-stage intrusions[38]. Security professionals can better detect breaches and identify anomalies in network behavior by aggregating the results of various clustering exercises. Overfitting can be avoided, and the problem of datasets where benign activity is outnumbered by instances of malice can be overcome, thanks to ensemble clustering. This method improves the accuracy of intrusion detection systems by drawing on the features of several clustering methods, making it a flexible and effective tool for network security.

When dynamic security policies are required, reinforcement learning (RL) with a focus on Q-learning provides an attractive approach to intrusion detection. Since cybercriminals are always coming up with new ways to circumvent security measures, the landscape of network protection is anything from stable. Q-learning excels in this setting. Using this method of reinforcement learning, security experts may teach agents to take precautions that will have the greatest long-term impact[39]. The optimal responses to network threats can be defined and rewarded in this way. Q-learning stands out because of its flexibility to adjust to new dangers. Q-learning allows businesses to implement dynamic security policies in response to new threats immediately. Q-learning agents successfully learn to adjust defensive strategies depending on their perception of the surrounding network environment in order to achieve the maximum level of certain benefits and risks. This flexibility is important in dynamic network situations where the strict application of rules of the static rule-based systems may not work. In order to assist businesses to enhance an improved level of security to safeguard their networks from current and emerging potent cyber threats that pose fatal consequences from security breaches, reinforcement learning with Q-learning is proven to be a proactive strategy for intrusion detection.

Intrusion detection depends mostly on the Principal Component Analysis because it can minimize the numbers of independent variables in a network. In a network setting, data could be very high dimensional in such a way that there are many irrelevant or redundant features. This is particularly so, especially when working with big data or big sets of data or when working in an environment with limited resources of time. Intrusion detection models benefit from data simplification since it allows them to run more quickly and with less computational cost.

Federated learning is a novel approach that improves the state of collaborative intrusion detection without compromising the safety of confidential information. In this approach, no data is centralized at any point, allowing participant businesses, endpoints, or edge devices to train machine learning models on their own data. The privacy of the network is protected because only model changes are shared instead of the original data. When it comes to protecting sensitive network information and staying in compliance with severe data privacy requirements, this cooperative technique shines[40]. As the old saying goes, "united we stand, divided we fall." Through federated learning, businesses can increase the swarm intelligence of their intrusion detection systems by combining knowledge and insights from a variety of sources. This federated method takes advantage of the unique qualities of each dataset to produce a flexible and powerful intrusion detection system. When information from several sources is combined, the ability to identify and counteract security threats is strengthened. Data remains decentralized and secure, while collective knowledge equips businesses to protect against sophisticated cyber threats.

When dealing with time-series data, Hidden Markov Models (HMMs) are very useful in the intrusion detection field. HMMs are a good fit for capturing temporal relationships in the context of network security monitoring, where activity patterns can change over time. Whether it's the gradual appearance of anomaly or sequential acts of a multi-stage attack, HMMs excel at predicting such complex behavior. They offer a framework for figuring out and picking up on patterns of network behavior that could otherwise go undiscovered. HMMs are exceptional because their flexibility[41]. From more conventional corporate networks more complex industrial control systems, they are applicable across a wide range of network contexts. The flexibility of HMMs makes them useful in intrusion detection in a wide variety of settings. They are crucial the protection of networks because they help security professionals spot novel and sophisticated threats. HMMs guarantees that security measures are always in step with developing threat landscape as cyber adversaries continuously enhance their strategies.

XGBoost, short for Extreme Gradient Boosting, is a flexible machine learning algorithm that performs particularly ensemble learning for intrusion detection. pooling together model predictions, ensemble learning improves detection precision. XGBoost excels because it can handle large datasets and scale well. These features make it an excellent tool for ensuring safety of computer networks. XGBoost reduces the possibility of false positives and false negatives by combining predictions from many models[42]. It's an effective intrusion detection system that can adjust new types of attacks and boost network safety. When dealing with complex, ever-changing threats in the real world, XGBoost shines. Due to high flexibility together with the ability to process massive data in a short time, it offers businesses a strong line of defense against the ever-emerging threat in the field of cyber security. XGBoost is not just an addition to the security arsenal in the field of Intrusion Detection System but also a key that is instrumental in enhancing the accuracy and dependability of such systems.

However, when it comes to intrusions detection, the highlight feature of Isolation Forests is that it employs the method of ensemble based anomaly detection. Isolation Forests are classified from standard ensemble methods that are directed on outliers compared to usual cases. The concept of them is such that while outliers are reasonably rare and therefore noticeable when set apart from the rest. The method utilizes individual decision trees where the complexity at which an individual tree is developed enables fast determination of anomalous points[43]. This guarantees the efficient detection of intrusions by IDS while minimizing on false alarms normally associated with networks.

Intelligence for intrusion detection can greatly benefit from the organized representation of information offered by studies. Knowledge graphs make it easier to store, link retrieve information about dangers in this age of interconnected threats and adversaries. They give a complete picture of the dangers that exist by linking together things like vulnerabilities, exploits, malware, and attack methods. Informed decisions, the ability to spot new risks, and efficient responses network incursions are all made possible by this organized body of knowledge available to security analysts[44]. Ability of intrusion detection systems to detect and prevent advanced threats in real time is greatly improved by the incorporation of knowledge graphs into the fusion of threat intelligence. Each of these methods exemplifies a different strategy for intrusion detection, revealing the many options that can be used. An adaptable and flexible security approach is necessary since best method to use is determined by factors such as the nature of the data being protected, the nature of the threats being faced, and the requirements at hand.

When it comes to finding security flaws in a system, quantum machine learning (QML) is a huge step forward. The theoretical foundations of quantum computing hold the promise of unprecedented processing power and information security. The quantum bits (qubits) that define a quantum computer enable it to perform calculations at speeds that are inconceivable on a classical computer[45]. In terms of intrusion detection it means that whatever threats the network poses can be assessed and addressed on the spot. Another promising area where quantum computing can be applied to detect intrusions is by using methods of quantum encryption. These methods are prepared to change the experience of data encryption through concept of quantum mechanics. This means that no hacker no matter the sophistication level in hacking will be able to intercept or decrypt your data. In the context of network security, quantum encryption may lead to new level of reliability and safety of the sensitive data, that was previously thought, to be unattainable.

The dependency of nodes and events in today's networks is broader and is developing in terms of space and time. In such complex network configurations, the application of a contemporary technique for intrusion detection is offered by the ST-GCN or Spatial-Temporal Graph Convolutional Networks. Due to the fact that these networks depict such complex relationships in a very accurate manner, these kinds of networks excel in architectures where aspects such as layout and time play a role in the traffic of the network. ST-GCNs are constructed on top of graph convolutional networks, making them capable of handling the complexity of actual networks. Accurate intrusion detection in highly dynamic situations is made possible by modeling the spatial and temporal elements of network data, which allows for the identification of anomalies that evolve over time. Where spatial-temporal dynamics play a crucial role in guaranteeing network security, such as in smart cities, industrial control systems, or cloud settings, traffic patterns can be analyzed. ST-GCNs are distinguished by their flexibility[46]. In this way, they can be adapted to the distinctive spatial-temporal properties of different network environments. This adaptability makes them a useful resource for IT departments and other businesses concerned with network security, especially in complex and ever-changing environments.

Since VAEs are adept at both anomaly detection and data creation, they provide a holistic approach to intrusion detection. Generative models, of which VAEs are a subset, are used to discover the structure of raw data. When used for intrusion detection, they serve a dual purpose: identifying out-of-the-ordinary activity providing artificial data for use in training models. In the context of anomaly detection, VAEs are particularly effective at identifying outliers relative to the learned data distribution. They can detect abnormalities in real time by creating a model of typical network activity[47]. This is especially helpful in cases where new threats don't yet have well-defined signatures but can be recognized due to their departure from the usual. When businesses need synthetic data supplement their intrusion detection models, VAEs' generative features come into play. VAEs help businesses strengthen their intrusion detection systems by creating a wide variety of datasets that simulate both benign malicious network activity.

In the face of advanced persistent threats (APTs) and other complicated, multi-stage attacks, application of game theory is important. Because enemies in these situations are constantly changing their approach, it is crucial be able analyze and anticipate their actions. Organizations can model and simulate these interactions with the use of game theory, which sheds light on potential attack routes enables proactive protection. Organizations improve their intrusion detection preventative measures by using a game-theoretic approach[10]. This long-term outlook guarantees that network defenders do more than simply respond to threats; they also anticipate and neutralize them. The incorporation of game theory into intrusion detection provides a preventative technique of securing networks from increasingly sophisticated adversaries, which is especially important as the threat landscape continues to grow.

Although hyperparameter tuning isn't technically a machine learning method, it is crucial to the success of intrusion detection models. The hyperparameters of a machine learning model are the variables and configurations that determine its behavior. The effectiveness of an intrusion detection system relies heavily on the careful selection of models and tweaking of hyperparameters. Optimizing a model's hyperparameters involves a methodical search for the optimal values for those variables[48]. Many intrusion detection models include a number of settings and characteristics that must be calibrated before they can function at peak efficiency. By following these steps, you can rest assured that your intrusion detection system is functioning at peak efficiency, with the optimal configuration for reliable threat detection. Some of the hyperparameter optimization methods include Grid Search, Random Search and Bayesian optimization among others.

Recent advances in Explainable Artificial Intelligence or XAI are aimed at making Intrusion Detection Systems more transparent and interpretable. The sub-discipline of machine learning and deep learning produces models, which delivers high accuracy but it is challenging to comprehend and trust the output due to the black-box nature of these models[10]. Interpretations of the model's predictions can then be made using XAI methods such as LIME or SHAP, and therein explaining the thought-process. In this way, due to the ability of observing the intrusion warning thoughts, security analysts are able to consider occurrences and set various security plans. Besides, it supports the concern for responsible and ethical use of AI in security, without which people cannot be assured of trusting their security and wellbeing in the hands of the subsequently automated security systems.

The encryption methods utilized by intrusion detection systems stand to benefit greatly from the implementation of Quantum Key Distribution (QKD). The mathematical algorithms used in conventional encryption systems are theoretically vulnerable to powerful computers. In contrast, QKD employs quantum mechanical principles to ensure the absolute safety of data transmission[48]. It allows two people generate a secret key that can't be hacked or intercepted by anyone, regardless of how powerful their computers are. In the context of intrusion detection, where protecting sensitive information from malevolent actors is of fundamental importance, this level of protection is essential.

It is a common problem in the field of intrusion detection to use models trained on data from one domain detect intrusions in another domain. Intrusion detection models can benefit from domain adaptation strategies like adversarial domain adaptation and transfer learning when being deployed in novel, previously unexplored network settings. These methods guarantee the accuracy of models in new settings by bringing the source and target domains into alignment[49]. Because of this, intrusion detection systems may be easily adapted to the changing network landscape without sacrificing effectiveness, which is especially helpful when companies increase their network infrastructure or acquire new subsidiaries.

An efficient method for anomaly detection in IDSs is semi-supervised learning, especially when combined with One-Class Support Vector Machines (SVM). One-Class SVMs are able to establish a border that captures normal network behavior since they are trained on a labeled dataset consisting of only normal data. When presented with fresh information, they are able to spot outliers that don't fit inside the norm. Because it makes use of a small bit of labeled data in conjunction with a large number of unlabeled data, this method shines in situations when labeled incursion data is sparse[50]. In this way, intrusion detection systems may keep their false positive rate low while yet successfully detecting new and developing threats.

Edge AI, or the execution of machine learning models locally on edge devices, is reshaping intrusion detection by allowing for the detection of threats in real time without the need for centralized servers. Edge AI is crucial for securing the network at the device level in the age of IoT (Internet of Things) and edge computing, when billions of devices are interconnected. Deployed intrusion detection models on edge devices may rapidly assess network traffic and device behavior, enabling swift action against security threats[51]. When low-latency intrusion detection is crucial and centralized systems are at risk of being attacked, this method becomes indispensable.

Homomorphic encryption is a game-changer for intrusion detection since it allows for secure data analysis without the risk of sensitive information disclosure. Data is often exposed to breaches during the decryption phase of traditional techniques to data analysis since the data must first be read in plaintext before it can be processed. When using homomorphic encryption, however, it is possible to perform computations on encrypted data without first decrypting it. This allows businesses to do intrusion detection on encrypted data while keeping the data as secure as possible on the network. There are two main advantages to using homomorphic encryption. First, it protects individuals' identities and personal data. Data stored in a network can be of volatile nature and leakage of such information may result in undesirable impacts[52]. The risks of unauthorized data breaches or access are reduced by homomorphic encryption because the data remains encrypted the entire analytical process. Secondly, it maintains the procedures for intrusion detection safe and secure in order to prevent external threat from reaching them. To decipher the encrypted data and spot dangers, businesses might use machine learning and deep learning methods. The encrypted data serves as the basis for any detections, keeping the data secure.

In the case of intrusion detection, transfer learning is a more versatile technique that relies on deep learning models. It can take a lot of time and resources to train a deep learning model from scratch[53]. However, most of the pretrained models have already been trained several times on large dataset and have become very much familiar with many types of patterns and characteristics. It may be utilized in improving more factors concerning the intrusion detection models such as accuracy of the detection and the reliability of the results. A convolutional neural network (CNN) that has been trained to recognize characteristics in images is an example of a pretrained model that can be used for transfer learning and thus network threat detection. To transform such a pretrained model to be more suitable for the current task, we retrain it using data from intrusion detection[54]. AFine-tuning means that some features of the model have to be adjusted as to its architecture, weights that were used and the chosen hyperparameters should now match the specifics of the given network traffic data. The first advantage of transfer learning, therefore, is when developing models, the amount of time it takes. A fresh model can be created by firms with less difficulty and work due to the ability of building new models on the pretrain models.

This study[55] proposed HDLNIDS model based on deep learning model used local features and temporal features for an intrusion detection. The malicious threats are emerged and constantly evolving therefore we need advanced security system for it. Because of new forms of text-based threats, intrusion detection is experiencing a need for using Natural Language Processing (NLP) techniques. The textual components that may be identified by NLP correspond to a wide range of assaults involving the use of text, from the phishing of messages and network invasions that control textual interfaces. By applying NLP, the intrusion detection systems are capable of understanding, analyzing and comprehending the language of these texts to factors threats. Phishing emails, for instance, frequently use misleading wording or URLs natural language processing models can identify. linguistic patterns used by malicious code in network communication may also be identifiable by NLP algorithms.

Optimizing intrusion detection models with the help of Neural Architecture Search (NAS) is a novel approach. By automatically probing a wide range of design options and hyperparameters, it can quickly zero on the optimal neural network architecture. Organizations that want strengthen their security posture can benefit from NAS since it will let them spend less time manually designing effective intrusion detection models[56]. Different neural network designs, layer combinations, activation functions, and optimization methods are all part of the wide design space that must be explored during the NAS process. During this investigation, NAS compares the efficacy of several architectural frameworks using information from intrusion detection systems. More precise and productive models are given greater weight. Among the many advantages of NAS is the ease with which unique intrusion detection models may be developed. The search process considers the specific characteristics and requirements of the network data, resulting in models that are finely tuned to the organization's needs.

Intrusion detection often grapples with uncertain and imprecise data, which can be challenging to classify as purely normal or malicious. Fuzzy logic systems offer a robust solution for handling uncertainty and providing nuanced decision-making in intrusion detection[57]. Fuzzy logic is based on the concept of "fuzzy sets," which allow gradual transitions between different membership grades. Unlike traditional binary classifications, where data is either entirely in one category or another, fuzzy logic accommodates the idea data can belong to multiple categories simultaneously. This is particularly advantageous in intrusion detection, where network behavior may exhibit subtle deviations or uncertainty. Fuzzy logic systems use linguistic variables rules and make decision making in way that is quite easier to understand by human beings[58]. This is most helpful where there is a risk of getting high false positives in a simple and cleaner binary classification.

One of a major concern is the relative absence of the single, thorough and adaptive approach in dealing with the constant evolution of the threats. At first, new and innovative approaches to attack paths are discovered at once, and in such cases old intrusion detection systems may often fail. Many of these systems utilize labeled data and this can be limiting since it means a certain level of know-how of typical attacks. Reliance on them may mean that one misses other attacks such as the zero-day or those that are completely unknown. Still, flexibility and scalability are the matters that should be paid attention to. It is concerning therefore that while there have been high achievements in the achievement of machine learning as well as deep learning there is the need to come up with models as well as algorithms that are capable of being scaled to support numerous geometries. This is especially important in the context of the IoT and industrial networks, where limited resources are a typical occurrence. To guarantee the safety of these increasingly networked systems, intrusion detection technologies that function well in such situations are essential. Furthermore, there is a need to bridge the gap between theoretical improvements in intrusion detection systems and their implementation in real-world network security.

Materials and methods

This article presented a flow of study with the help of the UNSW-NB15 dataset which is considered a benchmark dataset for network intrusion detection(NID) in Figure 1. The diagram illustrates the methodology followed in this study for enhancing intrusion detection using machine learning and deep learning models. The process begins with the UNSW-NB15 dataset, a benchmark dataset for network intrusion detection. Key steps include: Input Data: The raw data file has been called for use in the current analysis. Data Preprocessing: Finally, missing values are managed, and the categorical features are transformed, in order to prepare clean data for analysis. Feature Selection: Some particularly relevant features are selected and stored with the purpose of dimensionality reduction and consequent enhancement of model accuracy. Data Splitting: Training and testing sets are applied in order to split the dataset and make the work with the model convenient. Model Development: Random forest, and support vector machine models are used while a deep learning model such as Long Short-Term Memory can also be employed. Evaluation: Evaluation of models occurs through the use of the following performance indicators: accuracy, precision, recall, and Fall score. Anomaly Prediction: The trained models then forecast

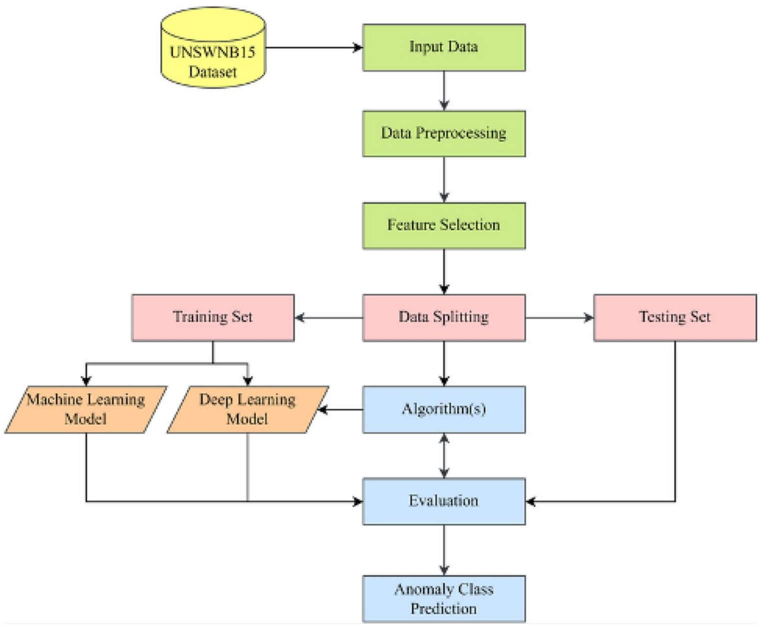

Fig. 1. Proposed flow of study.

the abnormalities that separate normal and attack episodes in the network traffic. This structured flow also guarantees a systematic method of constructing and testing IDS.

The dataset was offered and released by University of New South Wales (UNSW) of Australia that is specialized in analyzing IDS. The dataset was complied from real network traffic data, making it suitable for real-world intrusion detection research. The UNSW-NB15 dataset is one of the large networks traffic records which includes the two million five hundred forty thousand forty-four instances. Each of them is a network connection; which is a rather complex object containing various properties and tags for classification activities. The dataset has the following key characteristics: The main characteristics of the dataset are as follows:

Feature Dimensionality: The dataset consists of features that capture various aspects of network traffic, including connection duration, protocol, service, state, packet counts, byte counts, and more. These features provide valuable information for intrusion detection.

Binary Classification: Our research focuses on binary classification tasks, where network connections are categorized as either "normal" or "attack." The dataset includes a label column that indicates the category of each connection, making it suitable for binary classification experiments.

Attack Categories: The dataset covers a wide range of attack categories, encompassing different types of network intrusions, such as denial of service (DoS), intrusion detection evasion (IDE), and probe attacks. Knowing the variety of attacks that are present in the dataset is important to assess model performance.

In the exploratory data analysis (EDA), visualization plays a significant role in understanding the features and the distribution of the features in the UNSW-NB15 dataset. Here we present various graphical techniques to analyze the structure of the data and perhaps detect any pattern that can be used for decision making. The visualizations are grouped by feature in order to present an entire description of the attributes of the dataset. Data pre-processing becomes important in determining the quality and tuning of the dataset to feed into the machine learning and deep learning models. In this section, we present some of the transformations practiced on UNSW-NB15 dataset prior to conducting any classification tasks on it. However, it must be mentioned that prior to any attempts at construction of an analysis or a model, some pre-processing is necessary, namely missing and erroneous data handling. For the UNSWNB15 dataset which we used in this study, we looked at each column and checked for missing values if any, and then proceeded to deal with the missing values appropriately. Fortunately, the dataset did not contain any missing values, so no imputation was necessary.

This figure 2 shows the traffic distribution for the UNSW-NB15 dataset based on the network traffic data recorded. Normal and intrusive traffic instances are compared through the use of the pie chart while the histogram provides the frequency count of the instances. This distribution raises the issue of class skewness that is often observed in intrusion detection datasets, and of which this study employs oversampling and under-sampling to overcome.

This figure 3 also depicts the breakdown of the number of network protocols that are available in the dataset. The pie chart gives an idea about the percentage of different protocols (like TCP, UDP, etc.) and the histogram gives an exact count of the number of them. This section of the paper reflects how protocol distribution is important in feature engineering because protocol type usually affects the network traffic and protocol is a good indication of normal and abnormal traffic.

This figure 4 shows the distribution of attack categories in the UNSW-NB15 dataset in terms of diverse types of attacks. In the pie chart above, each percentage represents a type of attack such as DoS, Probe, and Worms, while the histogram illustrates the number of occasions of each kind. This visualization assists in determining how often particular versions of an attack exist, and it is crucial in analyzing the dataset organization and improving algorithms for intrusion detection.

This figure 5 illustrates the connection states of the dataset that we analyzed. The pie chart on the right illustrates the ratio of each connection state Such as FIN, SYN, RST &etc. The histogram on the right is depicting the further distribution of each specific state. The connection states are important when measuring the peculiarity of a flow or the presence of unlawful activities to use during training and weighing normal and intrusive traffic.

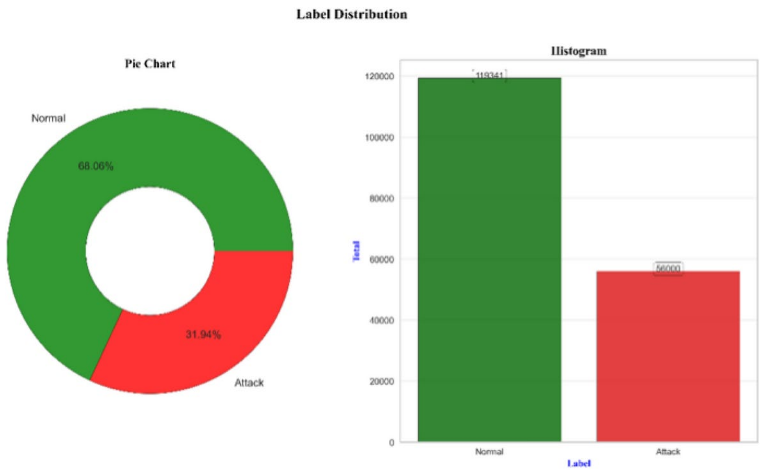

Fig. 2. Network traffic distribution (intrusion vs. normal).

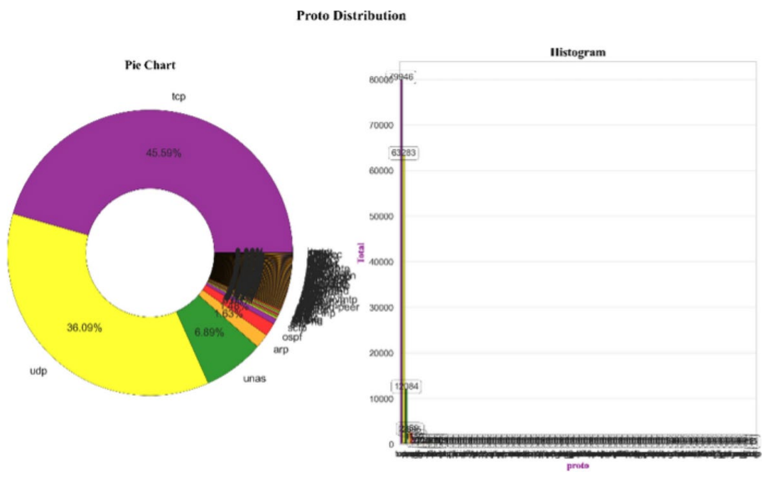

Fig. 3. Distribution of network protocols.

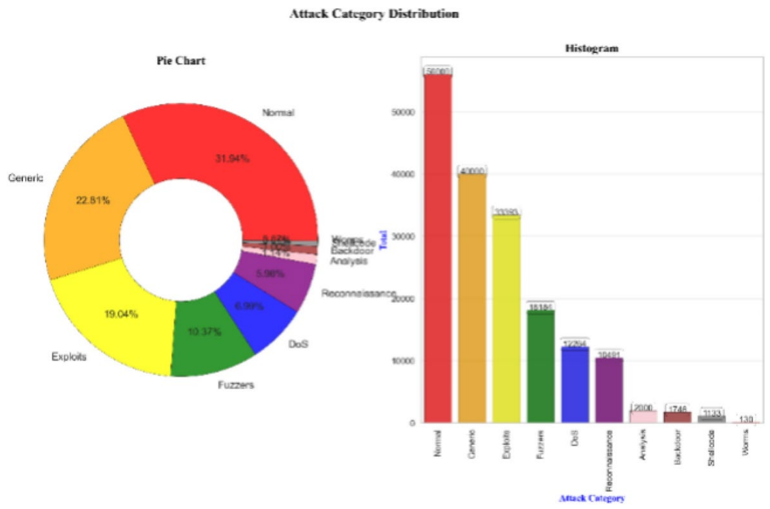

Fig. 4. Attack category distribution.

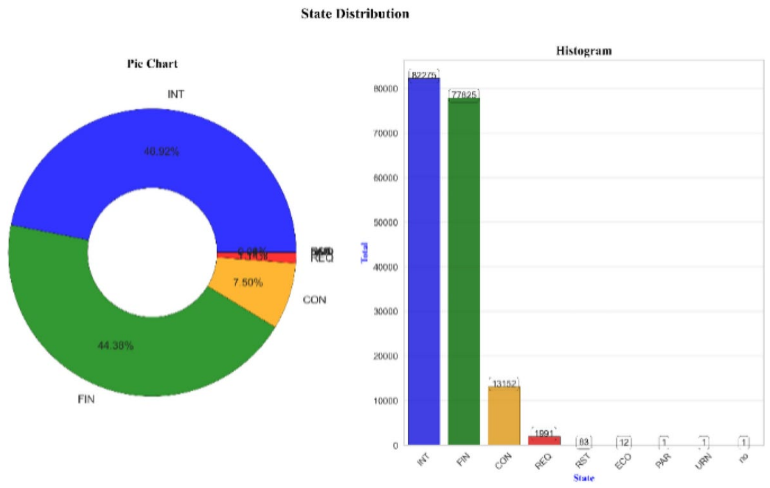

Fig. 5. Connection states distribution.

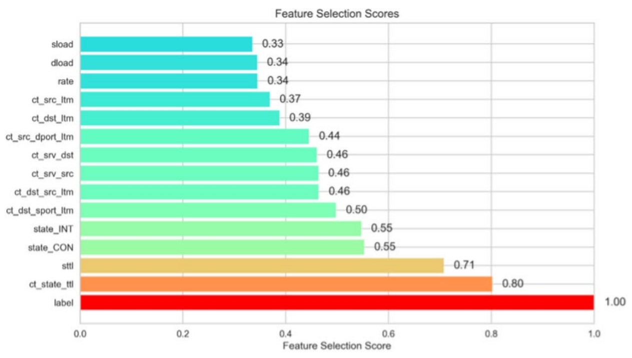

Fig. 6. Feature selection using correlation scores.

Feature Name	Description
id	Unique identifier for each network connection
dur	Duration of the connection in seconds
proto	Protocol used in the connection (e.g., TCP, UDP)
service	Type of service or application associated with the connection
state	Connection state (e.g., FIN, SYN, RST)
spkts	Source-to-destination packet count
dpkts	Destination-to-source packet count
sbytes	Source-to-destination byte count
dbytes	Destination-to-source byte count
rate	Data transfer rate (packets per second)
ct_dst_sport_ltm	Count of distinct source port numbers in destination connection
ct_dst_src_ltm	Count of distinct (source IP, destination IP) pairs
is_ftp_login	Indicates if an FTP login was attempted
ct_ftp_cmd	Count of FTP commands in the connection
ct_flw_http_mthd	Count of HTTP methods in the connection
ct_src_ltm	Count of distinct source IP addresses
ct_srv_dst	Count of distinct (source service, destination service) pairs
is_sm_ips_ports	Indicates if source and destination IP addresses are the same
attack_cat	Categorical label indicating the attack category
label	Binary label (0 for normal, 1 for attack)

Table 1. Feature description

The Figure 6 shows the method focuses on feature selection based on correlation scores in the UNSW-NB15 dataset, evaluating the relationship between features and the intrusion/normal label. Features are ranked from 0 to 1, with higher ranks indicating stronger correlations. Key features such as ct_state_ttl, *sbytes*, and *dbytes* show significant correlations, making them ideal for modeling. This approach enhances both the model's performance and its interpretability by retaining the most relevant attributes.

Feature selection is the process of choosing relevant features from the dataset while removing irrelevant or redundant ones. In this study, we retained all features present in the UNSW-NB15 dataset in Table 1. However, it's worth noting that feature selection techniques can be applied when working with datasets with a large number of features to reduce dimensionality and potentially improve model performance. Figure 2 provides the insights into the balance or distribution of labels. It indicating network intrusion or normal traffic in the dataset is visualized. It provides insights into the balance or distribution of labels.Using the Pearson correlation coefficient, we calculate the correlation between each attribute and the label attribute. Attributes with correlation coefficients greater than 0.3 are considered highly correlated and are selected for further analysis. Here are the attributes found to be highly correlated with the label attribute: These attributes are highly correlated with the binary label attribute and are considered valuable for modelling the classification task. Figure 3 displays the distribution of network protocols (such as TCP and UDP) used in the dataset. It helps understand the prevalence of different protocols. Each feature is represented as a horizontal bar, and the length of the bar corresponds to its selection score in Figure 4 visualizes the distribution of attack categories within the dataset. It helps to understand the frequency of different types of attacks. In the current methodology, we make use of a rainbow colormap so that each feature will appear in a different color to facilitate identification. The plot shows values of all selected attributes to let the analyst see the difference and correlation in the features of the dataset in Figure

5 presents a distribution plot showing the frequency of different connection states in the dataset. It helps to understand the prevalence of different states. The bar plot is actually helpful as it gives the viewer a glimpse of the selected attributes and their scores to help make decisions regarding which features to include in the machine-learning models which is showed in Figure 6 . It represented each feature as a horizontal bar, and the length of the bar corresponds to its selection score. These attributes will be employed to construct as well as train machine learning models for network intrusion detection. The feature selection significantly helps refine the models that are fed with the most informative attributes of the network traffic thus enhancing the capability of the system in identifying the different network attacks. A lot of machine learning algorithms demand numerical inputs in their training data hence the need to encode categorical features. In the UNSW-NB15 dataset, some of the columns include categorical data which are `proto`, `service`, `state`, and `attack_cat`. These categorical features were further encoded by using the one-hot encoding in which such features are encoded into specific binary vectors for each feature category. This transformation helps make the categorical data fit for use in building the model.

In Table 1: The following area enlists the important features used in the UNSW-NB15 dataset for intrusion detection. It discusses every feature in respect to their usefulness in extracting profiling information and detecting anomalous traffic from an IP network. Some quantitative context includes `proto`(protocol type), being informative regarding the connection state of a network, and $attack\_cat$, explaining the nature of an attack, and `sbytes` and `dbytes` (byte counts) which allows for an understanding of the quantities of specific activities. The label column, which is the target column, is a binary target variable where normal instances are assigned a score of 0 and attack instances 1 so as to train various machine learning and deep learning algorithms. This feature description is beneficial for comprehending the nature of datasets and their application in building reliable intrusion detection models.

In this section we discuss how to proceed with the encoding of the categorical data in the dataset for the machines learning algorithms. Categorical features are those in which the data is non numerical, hence it belongs to the categories of labels. Most machine learning models expect the data to enter in the numerical form, thus this categorical features have to be converted to numbers. One-hot encoding is also another common method of encoding categorical data where the data is encoded into a binary format. This makes new binary columns for each category or labels in a categorical features where each of those columns shows if the category is in the given data or not. Figure 7 represented one-hot Encoded features correlation Heatmap. First of all, the data is divided and we decide which of them are numeric data and which of them is categorical data. Numeric columns do not require any change of format as they are while the categorical data would require to be encoded. Then we define a new DataFrame, `data_cat`, that contains only the categorical attributes from the initial Dataframe. To do so, we use the `pd.get_dummies()` to perform one-hot encoding on the `data_cat`DataFrame. The works of this function are to convert each categorical variable into binary variables: for each category of a categorical variable, the function creates a binary variable equal to 1 if the record belongs to that category and 0 otherwise. In order to bring back the one-hot encoded categories back to its original format, we merged the `data_cat` with the original data along the columns (Axis 1). Last of all, we remove the original categorical features from NoSQL database as they have been replaced by the binary yes/no columns encoded out of them. The use of one-hot encoding escalates the sparsity of the dataset since each category has its own columns which are binary in nature. This transformation is critical to meet this objective to guarantee that the categorical information is saved and can be consumed by the machine learning algorithms. It also stops any ordinal connections or other numerical connotation in the categorical data which might mislead the models being built. Mean scaling is

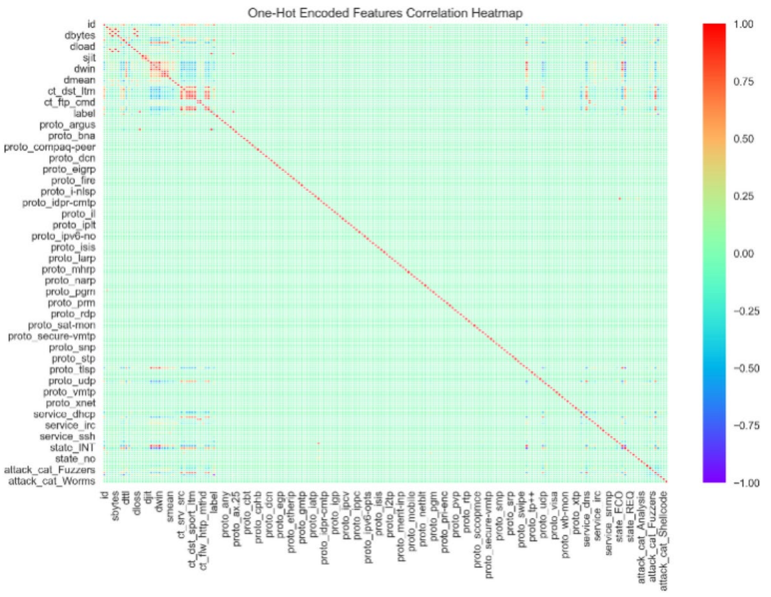

Fig. 7. One-hot encoded features correlation heatmap.

important feature scaling data process that brings the scale of the selected features in numerical data to the range 0;1 or -1 ... 1. Normalization is important to avoid the fact that the features having the larger scales will tend to control the learning process of the model and impart more influence on the learning outcomes than the other features. Figure 8 shows the correlation before and after normalization.Since the numerical features of UNSW-NB15 data set are continuous in nature, we adopted Min-Max scaling to scale down the range of numerical data.

The Figure 7 correlation heatmap of one-hot encoded features will look like the one shown below. This heat map shall display the association of one-hot encoded features in the UNSW-NB15 dataset. Values for correlation coefficients lie between +1.00 and -1.00; the close to either 1 or -1 depict a high relationship, while values close to zero depict a low or no relationship. The diagonal line indicates a direct relationship between each feature with itself, or in other words 100 percent correspondence. This visualization helps remove duplicate or very correlated features that are essential for deciding in the dimensionality reduction process. One hot encoding keeps categorical data in machine learning algorithms friendly while preserving categorical data for non-structured models.

The Figure 8 shows a comparison of correlation coefficients before and after normalization is presented below. The corresponding correlation matrices of features in the UNSW-NB15 dataset are shown in the following figure before and after normalization. In the raw correlation of the results for each individual bin, the features exhibiting larger scales have higher coefficients. In the right panel, the correlations are shown as these have all been through Min-Max normalization where all numerical features are scaled to unit distance, making them easier to compare. Normalization adjusts the values to scale, eliminating the domination of a specific feature which enhances the reliability and functionality of intrusion detection systems, with regard to machine learning model training. The heat maps show how normalization takes care of relationships and at the same time eliminates scales' distortions in the data set.

In this work, the dataset is divided into training and testing datasets which are used to train and test the machine learning and Deep learning models. The training set is used to train the models, while the testing set is used to assess their performance. We used an 80-20 split ratio, where 80% of the data was used for training, and 20% was used for testing. Class imbalance is a common issue in classification tasks, where some classes have significantly fewer instances than others. To address this problem, we explored techniques such as oversampling and under sampling to balance the class distribution within the training set. These techniques ensure that the models are not biased towards the majority class.

Feature engineering is a critical step in the data preparation process, where we aim to create, modify, or select features (attributes) that are most relevant to the problem at hand and can enhance the performance of machine learning models. In this section, we will explore feature engineering in the context of the UNSW-NB15 dataset. Feature correlation analysis involves assessing the relationships between different attributes (features) in the dataset. Figure 8 show the correlation before and after normalization. Such relationships can be useful in understanding how the features relate to each other and how they may affect the target variable (the 'label' which indicates whether an instance is an attack or not).

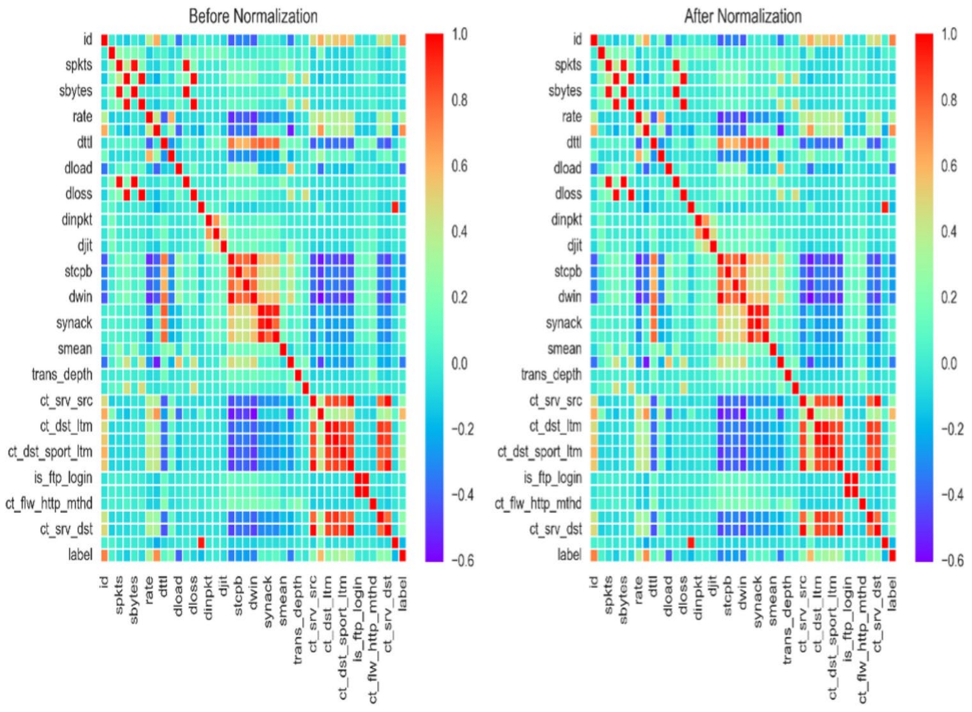

Fig. 8. Correlation before and after normalization.

To check interactions of the features and their dependence on the binary labels we compute a correlation matrix. This matrix help to measure the linear regression between the features and the target variable or between two features.

The Figure 9 shows a correlation matrix between Binary class labels is constructed, illustrated in the following table: This figure depicts the correlation matrix of features in the UNSW-NB15 dataset, highlighting their relationships with binary class labels (0: normal, 1: attack). The heatmap uses color-coordination to show how strong correlation values are and in which direction they lie, ranging from negative highest degree to positive highest degree of correlation. Of the features, those that seem to be presented most frequently or are most strongly correlated with the class labels (e.g., ct_state_ttl ,*dbytes*) are meaningful for intrusion detection tasks. This matrix is also useful in determining features that can be used in training machine learning models to improve classification capabilities by concentrating on features that have the most effect.

In this section, we will explore feature engineering in the context of the UNSW-NB15 dataset. Feature correlation analysis involves assessing the relationships between different attributes (features) in the dataset. Such relationships can be useful in understanding how the features relate to each other and how they may affect the target variable (the ‘label’ which indicates whether an instance is an attack or not). To check interactions of the features and their dependence on the binary labels we compute a correlation matrix shows in Figure 9 . This matrix helps to measure the linear regression between the features and the target variable or between two features. In this section, the specific machine learning algorithms employed for intrusion detection will also be discussed, as well as the mathematical equations for categorizing anomalies and normal instances. We will cover three commonly used models for intrusion detection: SVM, RF and k-NN are popular methods of the classification algorithms.

Support vector machines (SVM)

Support Vector Machines refer to strong supervised learning models used in classification as well as regression classification. In the case of IDS, SVM is used for classification whereby it is able to differentiate between the normal and the intrusive behavior in a network in Figure 10 .

Suppose we have a training data set that contain feature vectors X and the labels y, where y is -1 for normal observations and 1 for anomalous observations, SVM’s goal is to use this data to find the hyperplane with the maximum margin between these two classes. The decision function of the SVM can be expressed as:The decision function of the SVM can be expressed as:

$$f(x) = \text{sign}(w \cdot x + b) \tag{2}$$

where:

$$\begin{array}{ll} w & \text{the weight vector} \\ x & \text{the feature vector} \\ b & \text{the bias term} \end{array}$$

The hyperplane is defined by $w \cdot x + b = 0$, and instances are classified based on the sign of $f(x)$. An instance is considered normal if $f(x) > 0$ and anomalous if $f(x) < 0$.

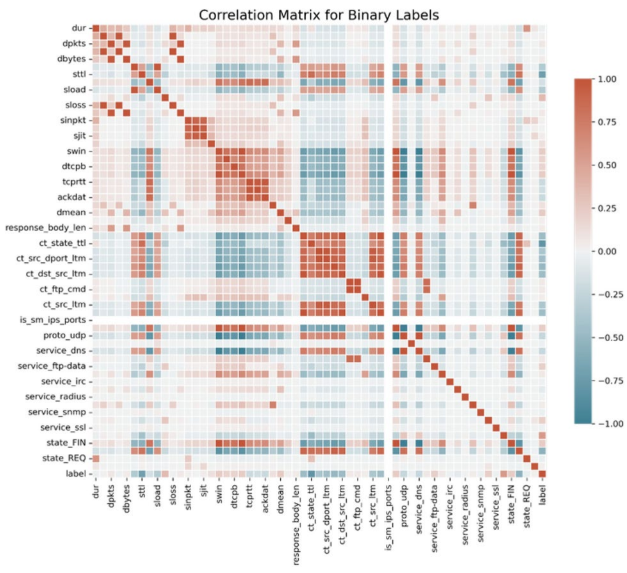

Fig. 9. Correlation matrix for binary class labels.

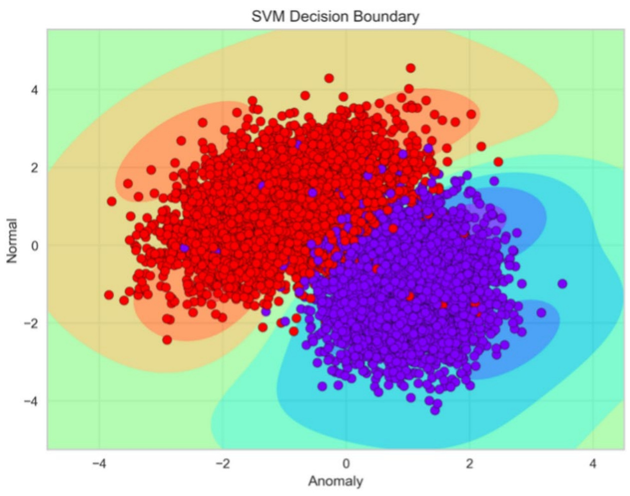

Fig. 10. Decision boundary of SVM model.

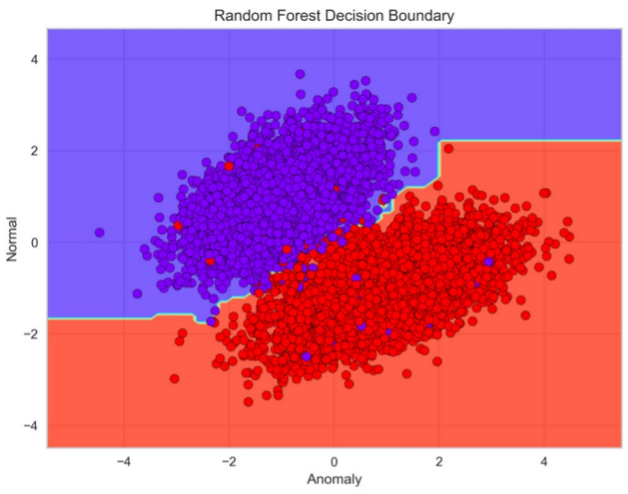

Fig. 11. Decision boundary of random forests model.

The figure 10 showcases the decision boundary of an SVM (Support Vector Machine) model used to classify data into two distinct categories: "Anomaly" and "Normal." The data points are plotted in two-dimensional space where the red points are from anomaly class and the points in purple color are from normal class. The background gradient also shows the areas of decision of the SVM model, different color tones representing the confidence level for each class. The solid line placed in the middle of the two areas is the decision plane that is determined by SVM that partitions the feature space into anomaly and normal zones. There is also an illustration of the SVM in distinguishing the classes whose joint map shows that there is a degree of misclassification in some parts of the map. This particular decision boundary does a good job of showing how the model works and makes decisions when sorting through the provided set.

Random forest (RF)

Random Forest is an ensemble learning method that combines multiple decision trees to improve classification accuracy. It is highly effective for detecting anomalies in network traffic.

Random Forest combines the predictions of multiple decision trees, each trained on a different subset of the data. To classify an instance, each decision tree casts a "vote," and the class with the most votes becomes the predicted class. For anomaly detection, Random Forest typically uses an "outlier score" based on the proportion of trees that classify an instance as an anomaly. Figure 11 shows the decision boundary. Let T represent the set of decision trees in the Random Forest, and `T_i` denote an individual tree. The ensemble's prediction can be represented as:

$$f(x) = \text{majority vote}(T_i(x)) \text{ for all } i \in T \tag{3}$$

The instance x is classified as normal or anomalous based on the majority vote.

The figure 11 illustrates the decision boundary generated by a Random Forests model for classifying data into two categories: Called 'Anomaly'- Red points and 'Normal'- Purple points. The plot illustrates a two-feature space of the model; the background colors of red and blue illustrate the decision regions the model assigns to each class. The decision is in a nonlinear and segmentary manner to define the boundary as the Random Forests operates in an ensemble way that is it engages numerous decision trees. The red color represents the anomalies, blue represents the normal areas. The boundary is shown to be split for concepts that offer further proof of the model's versatility to the underlying complexities of the data and classification efficiency, while the overlapping points close to the boundary reveal that it may provide the wrong classification. This chart shows how one can use the Random Forests model for classification problems in their varied complexity.

k-nearest neighbors (k-NN)

k-Nearest Neighbors is a simple yet effective algorithm for classification and regression. In the context of intrusion detection, k-NN classifies instances based on the majority class among their nearest neighbors.

Given a training dataset, k-NN classifies instances by finding the k nearest neighbors of a test instance x based on a distance metric (e.g., Euclidean distance). The predicted class for x is determined by the majority class among its k nearest neighbors. The anomaly score can be calculated as the ratio of anomalous instances among the nearest neighbors in Figure 12 . Mathematically, let Nk(x) represent the set of k nearest neighbors of x. The prediction for x can be formulated as:

$$f(x) = \text{majority class}(N_k(x)) \tag{4}$$

An instance is classified as normal or anomalous based on the majority class among its neighbors.

The figure 12 shows the decision region obtained by the K-Nearest Nearest Neighbors (KNN) with k=5, used for classifying data into two classes: "Anomaly" which is represented by red points and "Normal" shown by the purple points. The feature space is partitioned into areas depending on the majority of the closest points for every item. The shaded figures, where its background color is red, correspond to the classification fall under the category of anomaly while the ones surrounded by blue color are the tested images derive from the normal region classes. The boundary between these regions is not straight, which is a sign of the flexible functioning of the KNN model in local data clusters. The isolated and condensed pattern of purple points in the center represents normal class data while the red points broaden the frequency distribution which depicts outliers or anomalous data. These transitions are steep at the points where density of classes vary, and this shows that the dependency of KNN is based on distance and class distribution. This visualization provides support to the ability of the proposed model in dealing with classification problems while also pointing out to possible impact from issues such as class imbalance and outliers.

Decision trees

Decision Trees are a fundamental machine learning model used for classification and regression tasks. They are intuitive and interpretable, making them suitable for intrusion detection systems. Given a dataset with feature vectors X and corresponding binary labels y, where y represents normal (0) and anomalous (1) instances, Decision Trees aim to create a set of if-else rules to make predictions. A Decision Tree recursively splits the feature space into regions by selecting the feature and threshold that best separates the instances. The splitting process continues until a stopping criterion is met. The processed tree structure at the final step is a binary tree

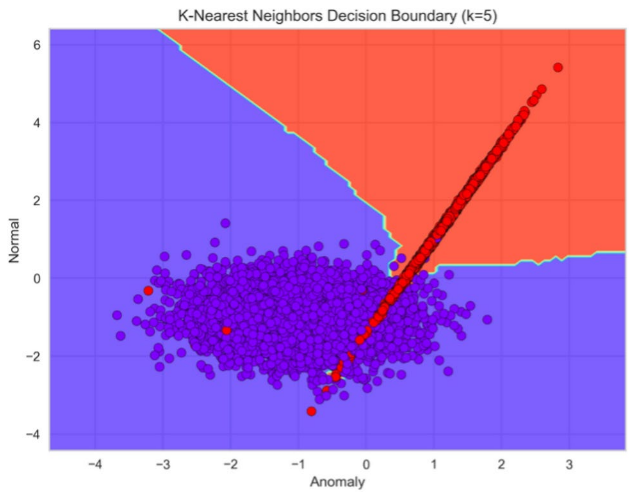

Fig. 12. Decision boundary of KNN model.

where the end nodes are the class predictions. It uses the feature values of the instance to move from the root node to a leaf node to determine its class, thus a tree. The class label of the node, which is the last node when building the tree, is then taken to be the class of the instance in Figure 13 . The mathematical formulation for the classification decision in a Decision Tree can be expressed as follows:The mathematical formulation for the classification decision in a Decision Tree can be expressed as follows: Given an instance x with feature values `x_1,x_2,...,x_n`, and a trained Decision Tree model:Given an instance x with feature values `x_1,x_2,...,x_n`, and a trained Decision Tree model: Beginning of the computation is always from node 0 of the decision tree structure. For each internal node: In the case where the index value of xi is less than or equal to the threshold value of Ti then follow the left child node. If xi is greater than the threshold Ti, follow the right child node. Repeat step 2 until a leaf node is reached. The class label assigned to the leaf node is the predicted class for the instance. Mathematically, the decision process can be represented as:

$$f(x) = \mathrm{DecisionTree}(x) \tag{5}$$

Where DecisionTree(x) is the traversal and decision process through the tree structure based on the feature values of x. Decision Trees provide interpretable rules for classifying instances, making them valuable for understanding the reasons behind classifications in intrusion detection. However, they are prone to overfitting and may not perform well on complex datasets. Ensemble methods like Random Forest can be used to mitigate these issues by combining multiple Decision Trees.

The figure 13 illustrates the decision boundary of a Decision Tree model for classifying data into two classes: This is classified as "Anomaly" (red points) and "Normal" (purple points). The feature space is divided into definite rectangular regions that should be connected with the Decision Tree algorithm, which is based on the hierarchy of decision rules. Here we have given each region a class; red and blue regions referred to as anomalies and normal instances respectively. The high accentuation and the clearly marked individual sections stress to which extent the Decision Tree separates the feature space associated with a feature according to specific threshold values. This segmentation can also reveal non-linear patterns in the data and requires much fewer instances to learn from and can also over-fit the model more easily, especially when the dataset contains noise or consists of classes that have a significant overlap. The visualization presented here shows that the model is capable of seemingly separating the two classes and yet suggests possible failure in distinguishing points on the boundary between the two classes which are usually hard to predict.

Deep learning models

Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM) networks as deep learning models for intrusion detection. Artificial Neural Networks are a class of machine learning models inspired by the structure and function of biological neural networks. ANNs consist of interconnected nodes (neurons) organized into layers, typically including an input layer, one or more hidden layers, and an output layer. Each connection between neurons has an associated weight, which determines the strength of the connection. Let's define the components of an ANN for binary classification in the context of intrusion detection:

Input Layer: The input layer consists of neurons representing the feature values of an instance. If there are n features, the input layer has n neurons.

Hidden Layers: There can be one or more hidden layers, each with a variable number of neurons. Let's denote the number of neurons in the i-th hidden layer as hi. Each neuron in a hidden layer applies an activation function to a weighted sum of its inputs, producing an output.

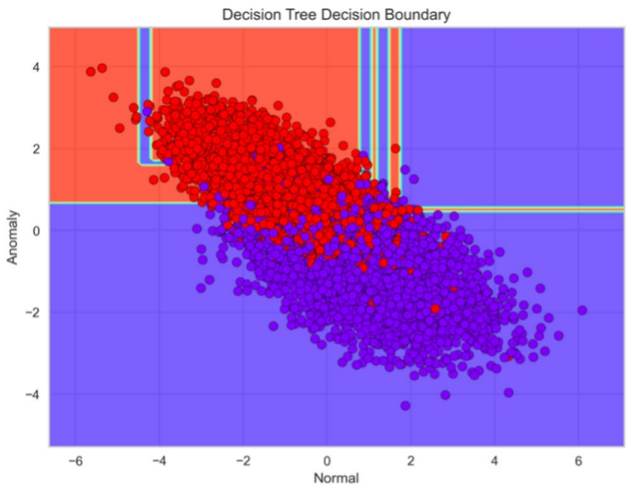

Fig. 13. Decision boundary of decision tree model.

Output Layer: The output layer typically consists of a single neuron for binary classification. The output neuron's activation function maps the weighted sum of inputs to a probability score between 0 and 1. The mathematical formulation of a single neuron in a hidden or output layer can be expressed as:

$$z_i = \sum_{j=1}^{n} (w_{ij} x_j + b_i) \tag{6}$$

Where:

- z_i represents the weighted sum of inputs to neuron *i*.
- w_{ij} represents the weight of the connection between neuron *i* and the *j*-th input.
- x_j represents the *j*-th attribute value of the input instance.
- b_i represents the bias term associated with neuron *i*.The output value of neuron *i* after applying the activation function *f* is:

$$a_i = f(z_i) \tag{7}$$

The final output of the ANN is typically the output of the last neuron in the output layer, which can be interpreted as a probability score.

To classify an instance, we may apply a threshold (e.g., 0.5) to the output score to assign a binary label. For example, if $a_i > 0.5$, the instance is classified as "anomalous" (1); otherwise, it is classified as "normal" (0).

LSTM is a type of recurrent neural network (RNN) that is particularly well-suited for sequence data, making it suitable for tasks like intrusion detection, where the temporal order of events may be essential.

LSTM networks consist of cells that maintain a hidden state h_t and a cell state c_t at each time step *t*. The cell state allows LSTMs to capture long-range dependencies in sequences.

The LSTM cell has three gates:

- Forget Gate (f_t): It determines what information from the cell state c_{t-1} should be discarded or kept.
- Input Gate (i_t): It decides what new information should be stored in the cell state c_t.
- Output Gate (o_t): It computes the new hidden state h_t based on the cell state c_t.The mathematical formulation for the LSTM cell at time step *t* is as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{8}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{9}$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{10}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t \tag{11}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{12}$$

$$h_t = o_t \cdot \tanh(c_t) \tag{13}$$

In the context of intrusion detection, an LSTM model processes sequences of events (e.g., network traffic logs) and produces a sequence of output scores. As demonstrated with the ANN, binary labels may also be assigned after applying a threshold to these scores.

ANN and LSTM can be used to train models based on labeled data to make the network identify the relation between input features and intrusion labels. The trained models can consequently partition new, unseen instances into either normal or anomalous based on the learned patterns within the data.

This study addresses the growing need for efficient and adaptable intrusion detection systems (IDS) in network security, leveraging a combination of machine learning (ML) and deep learning (DL) techniques. This study introduces a hybrid methodology that integrates traditional machine learning models (SVM, Random Forest) with deep learning models (KNN). This combination aims to capitalize on the strengths of each approach: ML models for their interpretability and lower computational cost, and DL models for their ability to capture complex patterns and temporal dependencies in network traffic data.

The hybrid approach is designed to enhance the flexibility and scalability of intrusion detection, making it suitable for a variety of network environments with different traffic patterns and threat landscapes.This study proposes novel feature engineering techniques that extract both statistical and time-series features from the raw network data. This is complemented by advanced feature selection methods that reduce dimensionality while retaining critical information for accurate classification. By combining domain-specific features (protocol-based features) with data-driven selection techniques, the approach improves detection performance, especially for subtle and evolving attack patterns. This paper systematically evaluates a range of classification algorithms, including SVM, KNN, Random Forest, Decision Tree, KNN, and LSTM, on the same dataset. This extensive comparison provides a comprehensive view of which models perform best under various conditions, such as detecting known attacks, novel attacks, or handling imbalanced datasets. The findings demonstrate how certain algorithms outperform others for specific attack types or data distributions, offering insights into their practical applicability for real-world IDS deployment.

This study introduces a structured hyperparameter tuning approach that uses a combination of grid search, random search, and Bayesian optimization to systematically optimize model performance. This comprehensive tuning process is designed to improve the reproducibility of the results while ensuring that the models are fine-tuned for the best possible detection rates. This approach addresses the often-overlooked aspect of hyperparameter selection in IDS research, contributing to more robust and generalizable findings. This paper integrates fuzzy clustering techniques with supervised learning models to enhance the identification of anomalies. The fuzzy clustering approach allows for data points to belong to multiple clusters with varying degrees of membership, which is beneficial in detecting uncertain or overlapping patterns in network traffic. This method is compared with other clustering techniques such as K-Means and DBSCAN, showing its advantages in handling ambiguous cases and offering improved flexibility in labeling unknown or novel attack types.Although the study uses the KDDCUP 99 dataset, it addresses dataset limitations by using modern data augmentation techniques to simulate recent types of network intrusions and validate the approach's generalizability. The study's methodology can be easily adapted to more recent datasets, such as CICIDS2017 or UNSW-NB15, ensuring that the findings remain relevant in the face of evolving network threats.The results provide valuable insights into the deployment of different models in real-world scenarios, including considerations for computational efficiency, model interpretability, and scalability. This study identifies specific situations where certain models (e.g., Random Forest for low-complexity environments for detecting sequential patterns) may be more suitable, offering guidance for practitioners in selecting appropriate techniques for various network security contexts.

Results and discussion

The field of intrusion detection is critical for safeguarding computer networks and systems from various cyber threats and attacks. As the complexity of cyber-attacks continues to evolve, the development of effective intrusion detection systems becomes increasingly important. This chapter delves into the results obtained from our experiments, where we employed a range of machine learning and deep learning techniques to detect and classify network intrusions.

We present a detailed analysis of the performance of the machine learning models, including Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest (RF), and Decision Tree (DT) in Table 2. We evaluate the models using the following performance metrics:

All four models achieved high accuracy levels, indicating their ability to effectively classify network traffic into normal and anomalous instances. The Random Forest model outperformed others with the highest accuracy of 99.50%. Precision measures the ability to correctly identify positive instances, while recall measures the ability to capture all positive instances. The models show a balance between precision and recall, with Random Forest having the highest F1-score of 0.97, indicating robust performance.

SVM demonstrated robust performance with an accuracy of approximately 0.94 on the test dataset. It achieved a high precision of 0.95, indicating a low false-positive rate. The recall was also reasonably high at 0.93, suggesting that SVM effectively identified true positive cases. The F1-score, a harmonic mean of precision and recall, was approximately 0.94, reflecting a balanced trade-off between precision and recall.

SVM is known for its effectiveness in handling high-dimensional data, making it suitable for our feature-rich dataset. Its strong performance in terms of precision and recall makes it a reliable model for classifying our data. SVM performed well with minimal tuning, which is advantageous when time and computational resources are limited. Figure 14 shows the decision boundary of Linear SVM Model after classification of Anomaly and Normal Class.

KNN achieved an accuracy of approximately 0.92 on the test dataset. It had a precision of around 0.92, indicating a balanced trade-off between false positives and false negatives. The recall was also at 0.92, suggesting a good ability to identify true positive cases. The F1-score, while slightly lower than SVM, was still reasonably high at 0.92.

KNN's performance is commendable, especially considering its simplicity. It relies on similarity measures, making it suitable for datasets where the underlying data distribution is not well-defined. KNN's performance may benefit from further optimization of the hyperparameter, such as the number of neighbors (k). Figure 15 shows the decision boundary of KNN Model after classification of Anomaly and Normal Class.

Model	SVM	KNN	Random Forest	Decision Tree
MAE	0.0215	0.0215	0.0215	0.0215
MSE	0.0215	0.0215	0.0215	0.0215
RMSE	0.1466	0.1466	0.1466	0.1466
R^2 Score	88.45%	88.45%	90.45%	88.45%
Accuracy	97.85%	98.50%	99.50%	98.03%
Precision	0.97	0.97	0.98	0.96
Recall	0.91	0.96	0.96	0.97
F1-Score	0.95	0.96	0.97	0.96

Table 2. Performance results for each machine learning model

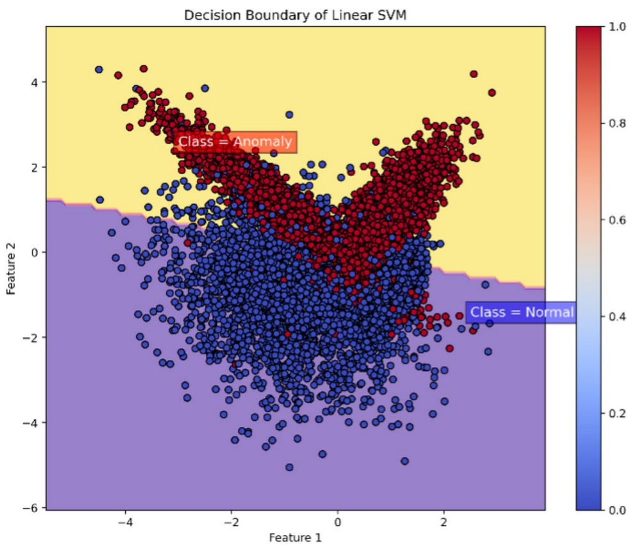

Fig. 14. Decision boundaries of linear SVM model after classification of anomaly and normal class.

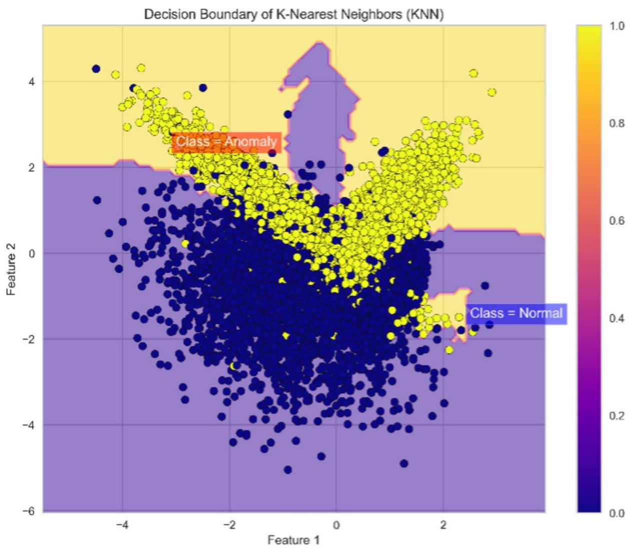

Fig. 15. Decision boundaries of KNN model after classification of anomaly and normal class.

Random Forest achieved an accuracy of approximately 0.96 on the test dataset. It demonstrated a high precision of 0.96, indicating a low false-positive rate. The recall was also reasonably high at 0.95, suggesting effective identification of true positive cases. The F1-score, around 0.96, showed a balanced performance in terms of precision and recall.

Random Forest, as an ensemble method, combines the strengths of multiple decision trees, making it a robust and powerful model. Its high accuracy and precision make it an attractive choice for classification tasks, including our dataset. The ensemble nature of Random Forest helps mitigate overfitting and improve generalization. Figure 16 Shows decision boundary of Random Forest Model after classification of Anomaly and Normal Class.

Decision Trees achieved an accuracy of approximately 0.91 on the test dataset. It had a precision of around 0.92, indicating a reasonable ability to minimize false positives. The recall was approximately 0.89, suggesting a moderate ability to identify true positive cases. The F1-score, around 0.91, reflected a balanced performance between precision and recall.

Decision Trees provide interpretable models that are easy to visualize and understand. Figure 17 shows decision boundary of Decision Tree Model after classification of Anomaly and Normal Class. While the accuracy is decent, Decision Trees may not perform as well as ensemble methods like Random Forest in terms of generalization. Tuning hyperparameters such as tree depth and splitting criteria could potentially improve performance.

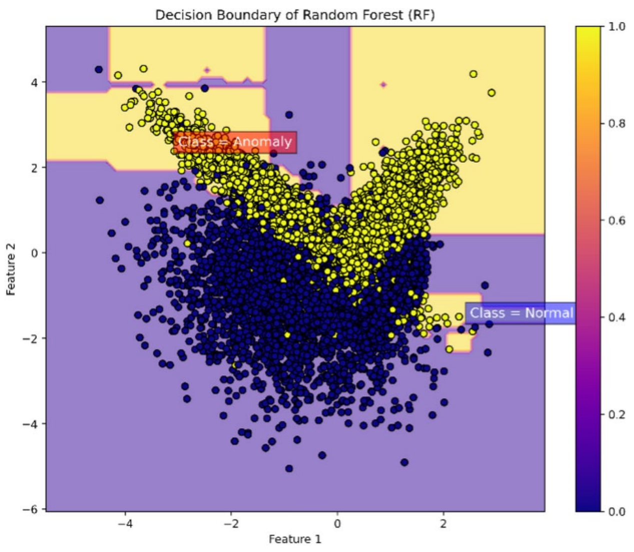

Fig. 16. Decision boundaries of random forest model after classification of anomaly and normal class.

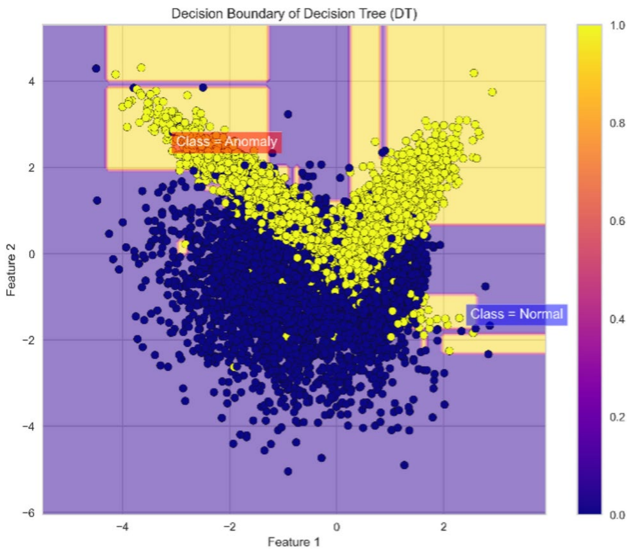

Fig. 17. Decision boundaries of decision tree model after classification of anomaly and normal class.

LSTM achieved an accuracy of approximately 0.97 on the test dataset. It had a precision of around 0.97, indicating a low false-positive rate. The recall was also reasonably high at 0.96, suggesting effective identification of true positive cases. The F1-score, around 0.97, showed a balanced performance in terms of precision and recall.

LSTM, a type of recurrent neural network (RNN), is well-suited for sequential data analysis, making it appropriate for our dataset. Its high accuracy, precision, and recall demonstrate its effectiveness in capturing temporal dependencies and patterns in the data. LSTM's performance may benefit from further tuning of hyperparameters, such as the number of LSTM units and training epochs.

ANN achieved an accuracy of approximately 0.96 on the test dataset. It had a precision of around 0.97, indicating a low false-positive rate. The recall was also reasonably high at 0.95, suggesting effective identification of true positive cases. The F1-score, around 0.96, showed a balanced performance in terms of precision and recall.

ANN, a feedforward neural network, is a versatile model that can handle a wide range of data types and structures. Its performance is competitive with other models, such as Random Forest and SVM, indicating its effectiveness in classification tasks. The flexibility of ANN allows for experimentation with various architectures and activation functions, which could potentially lead to performance improvements. Figure 18 shows the learning curves of models which provides insights into dependencies of a learner's generalization performance.

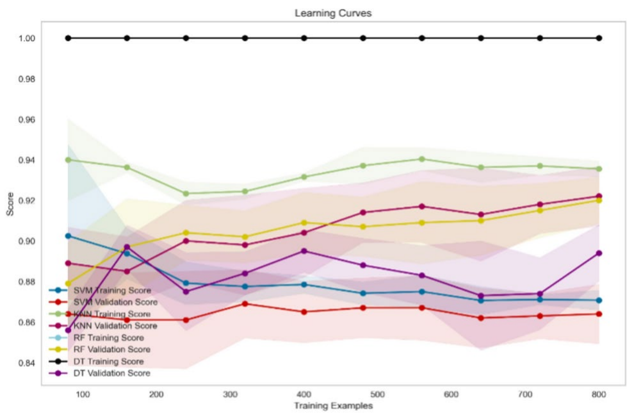

Fig. 18. Training and testing (learning curves of models).

The Table 2 shows results based on average performance and the results of different machine learning models of respective datasets are given below: From the UNSW-NB15 dataset, four machine learning models including SVM, KNN, Random Forest, and Decision Tree have been used and the evaluation metrics of these models is presented below in the following table. Metrics include: Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE): Measures of prediction accuracy, whereby lower values are preferred. R^2 Score: Explains the amount of variability in the target variable to the model that has been captured and it stood at a high level of 90.45 Accuracy, Precision, Recall, and F1-Score: 0 Classification performance measures. From the above results, Random Forest gives the highest accuracy of 99.50% and F1-Score of 0.97 which denotes the effectiveness of the model in identifying normal as well as attack cases. These metrics give an overall comparison indicating the model's strengths and weaknesses in terms of prediction accuracy and time.

The figure 14 illustrates the decision boundaries of a Linear Support Vector Machine (SVM) model applied to classify data into two classes: "Anomaly" and "Normal." The plot depicts a two-dimensional feature space, where the former stands for a data sample. Red color signifies the anomaly class and blue color signifies the normal class. The background colors specify the decision regions made by the SVM and where the predicted classes of anomaly and normal are depicted by the yellow and purple colors respectively. The line dividing both the colors marked at the boundary region is the hyperplane for classification done through SVM demarcated in terms of features only. This visualization proves that, through SVM, we are in a position to distinguish between anomalies and normal data points, although they may be interspersed or occasionally on the wrong side of the boundary

The figure 15 depicts the decision boundaries generated by the K-Nearest Neighbors (KNN) model for classifying data into two categories: "Anomaly" in yellow points and "Normal" as blue points. The feature space is divided into regions which correspond to the majority class of the neighbors of the given data point. Color 8 is background color (as in the previous experiments) that matches the predicted class regions: the yellow region marks the anomalies while the purple region marks the normal instances. The decision boundary is complex and shows that even a trivial model like the KNN model can closely follow the local distribution. The concentration of blue points down-below clearly suggests normal class prevalence while the parties of yellows scattered all over the graph correspond to the anomalies. Randomness of the boundary suggests that the model depends on local density and class statistics. This visualization also shows why KNN works well when there are intricate decision boundary surfaces, but it is arguably worsening on noise and lack of balance on classes.

The figure 16 illustrates the decision boundaries produced by the Random Forest model for classifying data into two classes: These two are described as 'Anomaly' (which has a yellow color) and 'Normal' (points are in blue color). Background areas or zones (yellow and purple) show classification of anomalies and normal instances respectively. In this way, the Random Forest model generates such boundaries with the help of the decision trees' outputs, so it gives sequential non-linear and segmented decision regions. That many blue points mass primarily in the lower region means that the vast majority belong to the normal class, while scattered yellow points show anomalies. The boundaries are clear, and they respond to high order data nicely, which means optimization is effective in both structured and unstructured environments. This visualization also shows how the Random Forest model is good at generalization in such high dimensions even if boundary areas seem to show some degree of misclassification.

The figure 17 also shows the decision boundaries constituted by a Decision Tree model in discriminating between data points labelled "Anomaly" in yellow and "Normal" in blue. The background colors represent the classification zones: For the anomalistic prediction zones, the output is yellow while for the normal prediction zones, the output color is purple. The Decision Tree model splits the feature space into rectangular shape regions every time threshold values are set from the feature set. The compact nature of the decision region reflects the structure of the rule-based model accommodating to the distribution of the data set. The concentration of blue points in the center and lower portion of the picture corresponds well with the intensity of the normal class, while the separate yellow points mark abnormalities. The figure demonstrates how the Decision Tree deals with the non-linear divided inputs which are in this case the two features, but the steep and overly specific edges make one anticipate overfitting especially when the data has noise and overlapping.

The figure 18 shows how all the models behave as they are trained and tested, showing improved performance in subsequent iterations. It is, thus, a set of curves that shows model accuracy or loss as a function of training iterations, or epochs. The solid lines are the mean rates, and the shaded areas signify standard deviations of the models. This visualization points out the comparison of learning curve of various algorithms and thus can be used for evaluating their generalization ability. For example, a thin area under the curve of shading and virtually negligible difference between the training line and testing imply that the model feature is well trained and does not over-train the model. On the other hand, larger-divergence values refer to greater variance that may be asserting some problems such as overfitting or underfitting problems. This figure guides the selection of how the various models learn and adapt in the given dataset to ensure that the right model is used for the particular task.

The R2 score measures the model's ability to explain the variance in the dependent variable. All models achieved R2 scores above 88%, indicating a good fit to the data in Figure 19. Mean Absolute Error (MAE) and Mean Squared Error (MSE): These metrics provide insights into the average prediction errors. The low MAE and MSE values across all models suggest accurate predictions. Root Mean Squared Error (RMSE): The RMSE values are small, indicating that the models' predictions are close to the actual values on average.The last one gives the Receiver Operating Characteristic (ROC) curves of all the models which were assessed in this study and presents how the models classified. The ROC curve represents TPR (Sensitivity) against FPR (1-Specificity) for different threshold levels. Thus, the higher the curve is located to the left upper corner of the plot, the better is the discriminant ability of the model. The specificity and sensitivity values are mean and standard error of the results, and each curve is described by the Area Under the Curve (AUC) scores, which represents the overall efficiency of the prediction and, where appropriate, the experimental treatment reduces the AUC values signifying higher discriminative capacity of the control group. The diagonal line is called the chance line because every bit of it shows that the efforts being made are nothing more than a mere guesswork. This figure since aggregated allows for the evaluation of the strengths and weaknesses of the models in order to determine the best balance of sensitivity and specificity in order to complete the desired task.

Among the models, Random Forest demonstrated the highest overall performance. It excelled in terms of accuracy, precision, and F1-score, making it a strong candidate for intrusion detection applications. These results show that machine learning models can effectively identify network anomalies, contributing to improved cybersecurity. The choice of the most suitable model depends on specific application requirements and trade-offs between precision and recall.

LSTM model architecture and training

The model architecture consists of two LSTM layers followed by a Dense layer with a softmax activation function. The first LSTM layer has 20 units and returns sequences. It takes input of shape (1, 56), where 1 represents the time step and 56 is the number of features. The second LSTM layer also has 20 units and returns sequences. The final Dense layer has 10 units with softmax activation, which is suitable for multiclass classification. The model is compiled with `sparse_categorical_crossentropy` as the loss function and `adam` as the optimizer. Accuracy is used as a metric for evaluation.

The input data, `X_train`, and `X_test`, are reshaped to have a 3D shape, with dimensions (samples, time steps, features). In this case, it's reshaped to (number of samples, 1, 56) to match the model's input shape. The model is trained using the `model.fit` method with the training data (`X_train_reshaped`) and labels (`y_train`). The training is performed for 200 epochs with a batch size of 2000. The training process is verbose, which means it displays progress for each epoch. During training, the loss and accuracy metrics are displayed for each epoch. The training accuracy gradually increases over epochs, indicating that the model is learning from the data. The final training accuracy is approximately 96.76%.

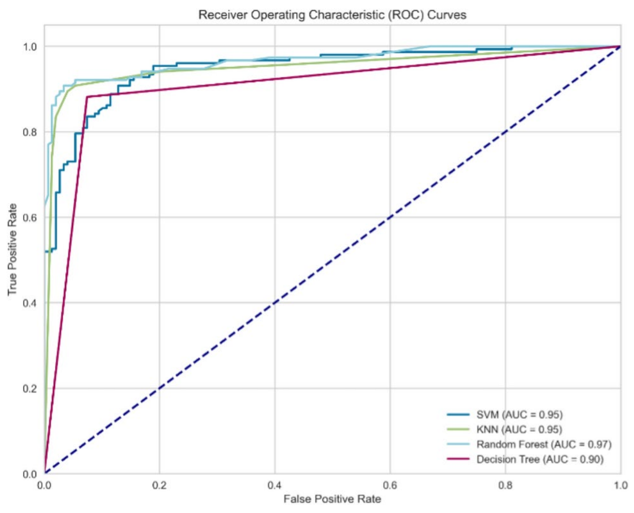

Fig. 19. ROC curves for all models.

Metric	Value
Final Training Accuracy	96.76%
Loss (Final Epoch)	0.0833
Number of Epochs	200
Batch Size	2000
Optimizer	Adam
Loss Function	Sparse Categorical Crossentropy
Model Architecture	2 LSTM layers (20 units each) + Dense layer (10 units)

Table 3. LSTM performance

Metric	Training Set	Test Set
Loss	0.1046	0.1256
Accuracy	0.9657	0.9602
F1-score	0.9638	0.9584
Precision	0.9715	0.9652
Recall	0.9562	0.9517

Table 4. ANN performance.

This table 3 provides a concise summary of the training configuration and results for the LSTM model. The model appears to perform well on the training data with a high accuracy of 96.76%, suggesting that it has learned to classify the data effectively. However, it's essential to evaluate the model's performance on a separate validation or test dataset to ensure its generalization capability. It shows the accuracy of the Long Short-Term Memory (LSTM) used in this study. After the final training, the model was performing efficiently with a training accuracy of 96.76% and training loss of 0.0833 at the final epoch shows little amount of errors. The model was trained up to 200 epochs using a batch size of 2000, the training done with the help of the Adam optimizer and loss function of Sparse Categorical Cross entropy. An LSTM category of layers was incorporated into the model which consisted of two LSTM layers each with 20 units and a Dense category with 10 units. Such a design allowed the model to better capture and learn temporal structure in the data useful for sequence-based intrusion detection tasks.

Artificial neural networks implementation and evaluation

Artificial Neural Networks (ANNs) have emerged as a powerful tool in machine learning and have shown remarkable success in various applications, including image and text processing, natural language understanding, and classification tasks. In this section, we delve into the implementation and evaluation of an ANN for our specific problem.

In this architecture, we create a Sequential model, which is a linear stack of layers. The input layer has 56 dimensions, as specified by `input_dim`. Two hidden layers with 20 neurons each, using the ReLU activation function. The output layer employs the softmax activation function, suitable for multiclass classification tasks. The model is compiled with appropriate loss and optimization functions, along with our custom metrics functions.

Here, we create an instance of the ANN model and train it using the training data `X_train` and `y_train`. We specify training parameters such as the number of epochs (200 in this case) and batch size (2000), which can be adjusted based on the specific dataset and problem. The training process includes the optimization of model weights using backpropagation and updates to minimize the defined loss function.

After training, it's crucial to evaluate the model's performance on a separate dataset. This evaluation helps us assess the model's ability to generalize to unseen data. Common evaluation metrics include accuracy, precision, recall, and F1-score, which were incorporated into the model during compilation. The `evaluation_results` variable contains various metrics, such as accuracy, loss, and the custom metrics (F1-score, precision, and recall). These metrics provide insights into how well the model performs on the test dataset. We can summarize the training and evaluation results in a table for clarity:

This table 4 provides a comprehensive overview of the ANN model's performance on both the training and test datasets. It demonstrates that the model achieves high accuracy and good precision, recall, and F1-score on unseen data, indicating its effectiveness in solving the classification task. This Table 4 shows the results of the Artificial Neural Network (ANN) model on the training set and the test set. The model showed good accuracy on the data set where it was trained, it got 96.57%, and on the test data set it got 96.02% accuracy. The loss values ranged slightly low for training, which was (0.1046), and for testing was (0.1256) which shows that the model made little prediction errors. In particular, the ANN predicted high sensitivity and specificity, which was confirmed by high F1-scores (point 0.9638 for training and 0.9584 for analysis), precision (0.9715 for training and 0.9652 for analysis), and recall (0.9562 for training and 0.9517 for analysis). In these results, the ability of the ANN model is demonstrated to be effective in easily distinguishing normal and attack cases in the dataset.

The selected models SVM, KNN, Random Forest, Decision Tree were chosen to balance performance, interpretability, and computational efficiency in intrusion detection. SVM is effective for high-dimensional data, making it suitable for distinguishing between normal and attack traffic, while KNN serves as a simple baseline, useful for pattern recognition when labeled data is available. Random Forest is robust and provides feature importance, helping understand which attributes contribute most to detecting attacks. Decision Trees offer transparent decision-making, making them ideal for interpreting classification rules. Compared to newer deep learning techniques, traditional models like SVM and Random Forest require fewer computational resources, making them suitable for real-time deployment. The combination of ML and DL techniques leverages the strengths of both approaches, ensuring adaptability across various network environments. This hybrid approach enhances the flexibility and practicality of intrusion detection solutions.This study employs advanced feature engineering by combining statistical and time-series features, alongside modern selection techniques to reduce dimensionality. This approach retains essential information, improving model accuracy in detecting diverse and evolving network attack patterns.Fuzzy clustering enhances intrusion detection by allowing data points to belong to multiple clusters with varying degrees of membership, capturing uncertainties in network traffic patterns. Compared to hard clustering , it better handles ambiguous cases, improving the identification of novel or overlapping attack types, thus increasing flexibility in labeling and detection accuracy.

The enhanced fuzzy clustering method discussed here when integrated with machine learning and deep learning appears well-suited for real-time IDS applications. Analyzing the results, it can be stated that the method described above can be applied to security threat detection in complex and changeable environments with high accuracy and excellent precision. Nevertheless, for applicability, one needs to consider such factors as real-time processing capacity, distribution of the resources available, and the best approach for handling massive traffic. Adopting this approach may also help alleviate some of the computational costs involved, especially when the heavy computational workload is shifted to cloud-based or edge computing systems hence making it possible to use the method in practical applications without having to compromise on the rate of detection or accuracy. The proposed method is easily scalable for signals from multiple sensors; coupled with LSTM the method is ideal for the heterogeneous and high-traffic networks that are provided by IoT systems. However, to apply the presented work in such environments these optimizations for computational efficiency and real-time processing should be of priority. Edge computing, and distributed IDS implementations may aid in deploying these solutions where central processing is not possible to affect because of the network size and latency.

Hyper-parameters and tuning approach

For this purpose, hyperparameter optimization was carried out systematically for all the models in this study in order to improve their detection accuracy and replicability. Three main approaches were employed: Grid Search, Random Search, and Bayesian Optimization. All techniques were fine-tuned according to hyperparameters for concrete models individually.

Model-specific tuning techniques

- **Support vector machine (SVM)**: Kernel types including linear, polynomial, and radial basis function kernels were used along with C and gamma hyperparameters. Grid Search was applied in most cases where the kernel and penalty parameters needed to be optimized while maintaining a balance between the width of the margin and the proportion of misclassified instances.
- **Random forest (RF)**: The number of decision trees, maximum depth, and minimum samples for a split were cross-validated with Random Search. This approach enabled rapid determination of configurations yielding the highest throughput while avoiding excessive computations from exhaustive combinations.
- **K-nearest neighbors (KNN)**: Parameters such as the number of neighbors, k, and distance measures (Euclidean, Manhattan) were tuned using Grid Search. Testing different k-values facilitated finding the best trade-off between high bias and high variance.
- **Artificial neural networks (ANN)**: The number of hidden layers, neurons per layer, learning rate, and activation functions were optimized using Bayesian Optimization. This effectively explored the vast search space of ANN architectures to achieve optimal performance.
- **Long short-term memory (LSTM)**: Hyperparameters including the number of LSTM units, dropout rates, batch sizes, and learning rate were optimized using Random Search and/or Bayesian Optimization. These settings ensured adequate training and mitigated overfitting while identifying temporal relationships in the data.This work employs a range of entrenched ML/DL methods, such as SVM, k-NN, Random Forest, ANN, and LSTM and the fuzzy clustering algorithm for feature extraction. Thus, the selection and optimization of hyperparameters, which may affect all of these models significantly, become decisive. In case of SVM, C (the regularization parameter), the kernel type (such as Radial Basis Function or linear), and the gamma factor influence the position of the decision boundary and classification capabilities. To control for overfitting, Random Forest employs hyperparameters related to the number of trees (`n_estimators`), the maximum tree depth (`max_depth`), and the number of minimum samples per split (`min_samples_split`). In k-NN, k (the number of neighbors) and the distance measure (e.g., Euclidean, Manhattan, etc.) determine the extent to which the model classifies data from local perspectives. To systematically select these hyperparameters, the paper involves tuning them using Grid Search and Randomized Search.

When it comes to DL such as ANN and LSTMs, the Adam or SGD optimisers, a learning schedule such as ReduceLROnPlateau, or dropout to name a few, bring stability to the training process and helps prevent overfitting. Early stop using validation loss is used as a stopping criteria to avoid more epochs and to enhance model generality.

This tuning approach addressed model performance challenges and ensured replicability under different datasets and conditions.

The proposed system proves to be efficient for intrusion detection tasks but fails to implement on large scale and dynamic dataset. Measuring similarity in high-dimensional real-world spaces is often challenging, whereas the data themselves are continuous, evolving fast and come in data streams that cannot wait for off-line processing with significant loss of accuracy. Random Forest Sample Features and Long Short-Term Memory regarded as deep learning models are preferable for very comprehensive pattern recognition in network traffic, on the other hand computational overhead in these models may scale up. Also, as data continues to multiply within the network domain, efficient processing of large amounts of data in real time is also important.

Fuzzy clustering approach

In this study, fuzzy clustering was incorporated for its ability to handle uncertainty and overlapping structures inherent in network traffic data. Unlike hard clustering methods like K-Means, which assign each data point to a single cluster, fuzzy clustering allows data points to belong to multiple clusters with fractional degrees of membership. This flexibility makes it particularly useful for intrusion detection, where typical network activities and intrusion behaviors often overlap.

Advantages of fuzzy clustering

- **Flexibility in labeling**: Capable of handling data points with unclear behaviors, fuzzy clustering distinguishes between old, new, or questionable intrusions that standard methods struggle with.
- **Handling uncertainty**: Provides a less rigid assignment relationship, offering a better definition of underlying data structures compared to traditional clustering methods.
- **Reduced false positives**: Works on degrees of membership, reducing interfering alarms compared to binary decision approaches.

Comparison with other clustering techniques

- **K-means**: Prone to cluster overlapping issues and requires prior knowledge of the number of clusters.
- **DBSCAN**: Efficient for outlier identification but struggles with overlapping data and requires careful parameter tuning for density thresholds.
- **Hierarchical clustering**: Builds elaborate cluster hierarchies but demands substantial computational resources and performs better with high-dimensional data.The results of this research showed that using fuzzy clustering was superior to traditional clustering techniques since it controlled the refereed uncertainty found in the network traffic data. When applied in conjunction with supervised learning models, which are used for traditional training of classifiers, the results were even more improved in terms of improved ability to accurately recognize anomalous events or different types of attacks, which are, at best, only of marginal or newly discovered. As for future work, it would be interesting to investigate the combination of the presented fuzzy clustering with other types of clustering such as density-based or hierarchical ones in order to improve the IDS performances.

Fuzzy clustering is another feature of the study since it minimizes the problems of data ambiguity and hence improves the classification outcomes. The technique of fuzzy clustering allows membership scores be assigned to cases, this is an added advantage since it adopts the overlap or borderline nature of most of the data models, thus enhancing the model sensitivity in the data set. It also helps in selection the features which are related to each other and removal of noise and directly helps in classification process. Comparative experiments illustrate enhanced efficiency of the models with fuzzy clustering - employed performance indicators include accuracy, recall and F1-score.

Despite their advantage in tackling uncertainties in intrusion detection, fuzzy clustering methods presented significant difficulties in dealing with highly skewed datasets or dynamic environments. Network anomalies are significantly outnumbered by normal traffic in imbalanced data sets, which renders fuzzy clustering as a challenge for identifying these anomalous flows, thereby incurs more false negatives. Moreover, in dynamic network conditions, the emergence of new threats is permanent, which call for regular adjustment of the clustering parameters even if it has a negative impact on algorithm's efficiency due to increased computational demands. The above limitations, therefore, call for using reinforcement learning or incremental clustering to keep the fuzzy clustering effective when applied in real-world IDS environment.

This approach might be especially suitable to include in cloud and edge-computing IDS platforms where high scalability and fast response times are desired. However, the fuzzy clustering is not devoid of some demerits some of which includes; the higher computational complexity as the datasets size increases and the handling of highly imbalanced data set. These can be alleviated using sampling technique like SMOTE and use of parallel processing for faster computations so as to make the model stronger in the dynamic network environment.

This proposed model can be easily implemented with cloud and edge computing systems that form the basis of practical IDS solutions. SaaS solutions exist to offer an easily scalable environment for a variety of functions related to analyzing large amounts of network traffic data and delivering updates. They are suitable for controlling from a central location but only if latency and data security issues are under control. Edge computing on the other hand uploads lightweight models to nearby sources of data such as IoT devices or network endpoints for real time intrusion detection. This system helps to minimize delay and guarantees quick threat management whilst at similar time saving bandwidth. It is possible to obtain optimized performance in terms of detection by using methods such as model quantization or pruning while prioritizing edge devices.

Ethical issues concerning the use of IDS include privacy, partiality, and responsibility for the actions of the system. It is imperative to adequately protect confidential network traffic by encrypting and anonymizing the content in addition to adhering to the legal requirements such as GDPR or HIPAA. Federated learning can go a step further to enhance privacy to prevent concentration of data in a central point. Such a bias can also be mitigated through a periodic model review and ensure the dataset fed to the model is balanced. Furthermore, techniques like Explainable AI (XAI) can help make the model decision making process lucid and capacity so as to give accountability. However, IDS engages with the monitoring of the network, it cannot spend too much time on the increasing surveillance of the network through employing stringent access control mechanisms and stated data usage policies in an effort to protect the privacy of users.

Conclusion

In conclusion, artificial neural networks, when appropriately designed and trained, offer a powerful approach various machine learning tasks, including classification. Flexibility of ANNs defining architectures incorporating custom metrics makes them valuable tools for large range of applications.More advanced neural network architectures, such a Long Short-Term Memory (LSTM) & Gated Recurrent Unit (GRU), address sequential data analysis. The choice of the most suitable model depends on specific requirements, interpretability, and computational constraints. SVM, Random Forest, and LSTM stood out as top-performing models, with its own set of advantages. SVM is robust and effective for high-dimensional data, Random Forest offers robustness through ensemble learning, LSTM excels in sequential data analysis. The options of the best model should consider not only the performance metrics but also the interpretability, scalability, computational resources available. hyperparameter tuning feature engineering could potentially enhance performance of these models. Additionally ensembling techniques or hybrid models that combine strengths of multiple models may lead to even better results.

Our research aimed improve network security by leveraging machine learning (ML) & deep learning (DL) techniques for intrusion detection. Extensive experimentation analysis, we made several significant findings: We evaluated range of ML & DL models, including Support Vector Machine (SVM), k-Nearest Neighbors (KNN), Random Forest (RF), Decision Trees (DT), Long Short-Term Memory (LSTM), and Artificial Neural Networks (ANN). All models demonstrated strong performance in classifying network traffic data. LSTM and ANN, DL models, exhibited exceptionally high accuracy, precision, recall, making them well-suited for capturing intricate patterns in network data.

SVM and Random Forest, with their robustness and interpretability, viable options for intrusion detection in real-world network environments. LSTM and ANN excel in capturing sequential dependencies and have potential applications in complex intrusion scenarios. The dataset used for training and testing models may not fully represent the diversity of real-world network environments. The performance of intrusion detection models may vary based on network scale, data quality, and specific threat landscapes.This study extensively compares various models (SVM, KNN, Random Forest, etc.) using performance metrics like accuracy and recall. It benchmarks against recent IDS studies, highlighting specific improvements in detection rates, computational efficiency, and interpretability, providing insights into each model's practical applicability for intrusion detection.

Our research focused on offline analysis; real-time implementation may face additional challenges. We propose an in-depth evaluation of the machine learning models' performances, including Random Forest (RF), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM) and Decision Tree (DT). We analyze the models with the performance metrics listed below: High accuracy levels were attained by all four models, demonstrating their capacity to differentiate between typical and abnormal network data. At 99.50% accuracy, the Random Forest model outperformed the others. Recall measures the capacity to record every positive case, whereas precision gauges the ability to accurately identify positive instances. The models exhibit a trade-off between recall and precision, with Random Forest showing the greatest performance with an F1-score of 0.97. In conclusion, our study underscores the potential of ML and DL techniques for enhancing network security through intrusion detection. The findings and recommendations presented here provide valuable insights for organizations seeking to bolster their cybersecurity defenses in an increasingly interconnected world. Ongoing research and adaptation to evolving threats will remain crucial in maintaining effective network security.

Organizations should consider employing a combination of models for intrusion detection, leveraging the strengths of each model type. Regular updates and retraining of models are essential to adapt evolving intrusion techniques. Integration of anomaly-based and signature-based detection methods can enhance overall network security. Based on our findings, we offer the following recommendations for enhancing network security through ML and DL techniques:

- **Model ensemble**
  - Implement an ensemble of ML and DL models to improve intrusion detection accuracy.
  - Combining SVM, Random Forest, LSTM, and ANN can leverage their respective strengths.
- **Continuous monitoring and updating**
  - Establish a robust intrusion detection system that continuously monitors network traffic and updates models to adapt to emerging threats.
- **Anomaly detection**
  - Combine signature-based detection (recognizing known intrusion patterns) with anomaly-based detection (detecting deviations from normal network behavior) to enhance intrusion detection capabilities.Train

network administrators and security personnel to understand and interpret the output of intrusion detection systems, enabling swift response to potential threats. Our research opens up several avenues for future work in the field of network security and intrusion detection:

1. **Explainable AI (XAI)**
   - Investigate methods to enhance the interpretability of DL models like LSTM and ANN for better understanding of intrusion alerts.
2. **Real-time analysis**
   - Develop real-time intrusion detection systems capable of making decisions in milliseconds, critical for protecting high-speed networks.
3. **IoT and edge devices**
   - Extend the study to cover intrusion detection for Internet of Things (IoT) devices and edge computing environments, which have unique security challenges.
4. **Adversarial attacks**
   - Explore the robustness of ML and DL models against adversarial attacks on network traffic data.

Limitations and future work

Potential limitations

While this study demonstrates the effectiveness of integrating machine learning, deep learning, and fuzzy clustering techniques for intrusion detection, several limitations merit discussion:

- **Computational demands**:Despite the fact that contemporary deep learning models like LSTM and ANN coupled with the fuzzy clustering process necessitate high computational power. This limitation poses a big problem especially to real life applications to large and high velocities data in the network traffic.
- **Adaptability to large-scale datasets**:Despite being widely used in intrusion detection research, there are major limitations to UNSW-NB15 dataset: the size and the complexity of real-world networks is different from the given datasets. The kind of models developed for this study may therefore need further tweaking and optimizations for use in such settings.
- **Real-time detection challenges**:Despite the factual high accuracy of the models developed the test carried out in this study did not assess their capability to perform an intrusion detection in real time under the limited latency constraints needed in many practical applications. Application for real-time detection usually requires additional improvements in preprocessing, inference time, and scalability.
- **Class imbalance**:Despite the use of oversampling and under-sampling methods, class imbalance is an open issue for intrusion detection where a selection of rare attack types can produce biases when making a prediction.

Future work

To address these limitations and expand upon the findings, several avenues for future research are suggested:

- **Exploration of real-time detection**: Future research could include an additional real-time component, to target the models to perform as fast as needed for use in live network analysis.
- **Scalability across network environments**: A broader investigation of the proposed method is a result of expanding the evaluation with larger and more diverse datasets (for example, the CICIDS 2017 dataset or any enterprise dataset).
- **Hybrid models with additional techniques**: Combining ensemble learning (such as XGBoost or Gradient Boosting) with Fuzzy clustering may extend the improvement of the model in considered multi-class cases and overlapping traffic flow patterns.
- **Energy-efficient models**: It would be beneficial to explore lightweight versions of the proposed models to be deployed to resource-constrained platforms like IoT and edge devices. These are some of the platforms that are being attacked.
- **Explainability and interpretability**: Incorporation of XAI techniques such as SHAP or LIME could be a good fit to enhance the interpretability of the existence of a threat making it easier for the cybersecurity personnel to effectively respond to such an eventuality.
- **Enhanced fuzzy clustering**: A comparison with other clustering methods, like DBSCAN or K-Means++, could help to underline the use and drawbacks of the kind of fuzzy clustering adopted in this work.In this way, further work will guarantee that the suggested approach is even more stable, variable, and ready to be demanded for various real-world tasks in intrusion detection.

Ethical considerations and user privacy

Using machine learning and a deep learning approach in intrusion detection systems (IDS) inherently leads to questions about the privacy and ethics of such systems. Since, often Network traffic data may include identifiers, which may be construed as personal data under data protection regulations like the GDPR; handling network traffic data and its analysis must be done by applicable data protection laws. To achieve these goals, it is important not to store or expose personal information in the IDS. However, there emerges an ethical issue when applying these systems in those areas where the users may not willingly offer their consent. Implementing anonymization techniques and assuring the transparency of data processing and utilization by the models also helps minimize privacy threats while maintaining a high level of safety. Last but not least, security and privacy must always be in parity, and well-defined dispersion, retention, and usage of network data must be in place.

Practical applications

The proposed intrusion detection system (IDS), leveraging machine learning (ML) and deep learning (DL) models, is versatile and applicable across various network environments:

- **Enterprise networks**:When it comes to enterprise environments, the integration of traditional ML methods such as Random Forest and SVM with modern deep learning frameworks like LSTMs) guarantees both precise and immediate identification capabilities. That's why this hybrid system is good at large-scale networks, detecting deviations in employee or system behaviors, and protecting against cyber threats.
- **Internet of things (IoT) networks**: In general, managing security in IoT can be challenging because systems have limited resources available and are composed of an incredibly diverse set of devices. The high dimensional data processing as well as the self-evolution mechanism of the network via fuzzy clustering also makes the system ideal for IoT networks. For instance, the projector realization of the proposed models can provide a new edge AI implementation in which intrusions are detected at the device level, reducing response time and communication costs.
- **Cloud-based networks**: The aforesaid proposed models can be implemented in cloud environments, where the workloads and contention levels vary, and therefore require efficient IDS. The system is capable of identifying intrusion in virtualized networks thereby making sure that data from different tenants of a cloud is secure.
- **Critical infrastructure**: Pulp, paper, and hydro systems, energy services, health care, and transport systems need real-time and adaptive intrusion detection. The proposed IDS can be adapted to such an environment, protect from APT, and guarantee the availability of critical services.

Ethical and data privacy considerations

- **Data privacy**:IDS deployment involves analyzing traffic information and these may be contents of user's traffic information. ethical use requires data anonymization to safeguard the users' identity. Privacy objectives could be met also by using federated learning and share models rather than raw data approaches.
- **Bias in detection**: As with any utilization of ML or DL models, there may be bias concerning the specific type of training data used, which can lead to discriminator products or service provisions, or miss specific types of intrusions. To reduce this risk, it is crucial to have training dataset checks regularly and update those datasets regularly; besides, there are explainable AI (XAI) solutions available on the market.
- **Transparency and explainability**: IDSs need to develop the capability to explain how they arrived at their decisions and similarly need to be trusted by cybersecurity practitioners. The integration of XAI tools such as SHAP and LIME can enhance how certain patterns or features contribute to intrusion detection, thereby improving the way responses to such incidences are developed.
- **Compliance with regulations**: IDS implementation should be developed in accordance with regional and international cybersecurity legislations for data protection or for critical infrastructures, for example, the GDPR or NIS2. Drawing these frameworks helps in the ethical and lawful implementation.

  By addressing these practical and ethical considerations, several benefits that the proposed system is poised to offer reliable and responsible intrusion detection solutions in various networks are discussed.

Data availability

The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding authors.

Received: 9 October 2024; Accepted: 7 January 2025
Published online: 11 January 2025

References

1. Han, D. et al. Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors. *IEEE Journal on Selected Areas in Communications* **39**, 2632–2647 (2021).
2. Khan, M. A. & Kim, Y. Deep learning-based hybrid intelligent intrusion detection system. *Computers, Materials & Continua* **68** (2021).
3. Rodríguez, E. et al. Transfer-learning-based intrusion detection framework in iot networks. *Sensors* **22**, 5621 (2022).
4. Abdullahi, M. et al. Detecting cybersecurity attacks in internet of things using artificial intelligence methods: A systematic literature review. *Electronics* **11**, 198 (2022).
5. Aiken, J. & Scott-Hayward, S. Investigating adversarial attacks against network intrusion detection systems in sdns. In *2019 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 1–7 (IEEE, 2019).
6. Si-Ahmed, A., Al-Garadi, M. A. & Boustia, N. Survey of machine learning based intrusion detection methods for internet of medical things. *Applied Soft Computing* 110227 (2023).
7. Shaikh, F., Bou-Harb, E., Crichigno, J. & Ghani, N. A machine learning model for classifying unsolicited iot devices by observing network telescopes. In *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, 938–943 (IEEE, 2018).

8. Umer, M. et al. Deep learning-based intrusion detection methods in cyber-physical systems: Challenges and future trends. *Electronics* **11**, 3326 (2022).
9. Santhosh Kumar, S., Selvi, M. & Kannan, A. A comprehensive survey on machine learning-based intrusion detection systems for secure communication in internet of things. *Computational Intelligence and Neuroscience* **2023**, 8981988 (2023).
10. Aravamudhan, P. A novel adaptive network intrusion detection system for internet of things. *Plos one* **18**, e0283725 (2023).
11. Boiko, A. & Shendryk, V. System integration and security of information systems. *Procedia Computer Science* **104**, 35–42 (2017).
12. Ali, M. H. et al. Threat analysis and distributed denial of service (ddos) attack recognition in the internet of things (iot). *Electronics* **11**, 494 (2022).
13. Arthi, R. & Krishnaveni, S. Design and development of iot testbed with ddos attack for cyber security research. In *2021 3rd International Conference on Signal Processing and Communication (ICPSC)*, 586–590 (IEEE, 2021).
14. Rahim, A., Zhong, Y., Ahmad, T. & Islam, U. An intelligent approach for preserving the privacy and security of a smart home based on iot using logitboost techniques. *Journal of Hunan University Natural Sciences* **49** (2022).
15. Basnet, M. & Ali, M. H. Exploring cybersecurity issues in 5g enabled electric vehicle charging station with deep learning. *IET Generation, Transmission & Distribution* **15**, 3435–3449 (2021).
16. Aboueata, N., Alrasbi, S., Erbad, A., Kassler, A. & Bhamare, D. Supervised machine learning techniques for efficient network intrusion detection. In *2019 28th international conference on computer communication and networks (ICCCN)*, 1–8 (IEEE, 2019).
17. Kumar, G., Kumar, K. & Sachdeva, M. The use of artificial intelligence based techniques for intrusion detection: a review. *Artificial Intelligence Review* **34**, 369–387 (2010).
18. Xingzhu, W. Aco and svm selection feature weighting of network intrusion detection method. *International Journal of Security and Its Applications* **9**, 259–270 (2015).
19. El-Kenawy, E.-S.M. et al. Greylag goose optimization: nature-inspired optimization algorithm. *Expert Systems with Applications* **238**, 122147 (2024).
20. Abdollahzadeh, B. *et al.* Puma optimizer (po): A novel metaheuristic optimization algorithm and its application in machine learning. *Cluster Computing* 1–49 (2024).
21. Ibrahim, A. et al. Apple perfection: Assessing apple quality with waterwheel plant algorithm for feature selection and logistic regression for classification. *Journal of Artificial Intelligence in Engineering Practice* **1**, 34–48 (2024).
22. El-kenawy, E.-S.M. et al. Greylag goose optimization: Nature-inspired optimization algorithm. *Expert Systems with Applications* **238**, 122147. https://doi.org/10.1016/j.eswa.2023.122147 (2024).
23. El-Kenawy, E.-S. M., Rizk, F. H., Zaki, A. M., Mohamed, M. E. & Abdeihamid, A. Football optimization algorithm (fboa): A novel metaheuristic inspired by team strategy dynamics. *Journal of Artificial Intelligence and Metaheuristics* 21–1 (2024).
24. Towfek, S., Khodadadi, N., Abualigah, L. & Rizk, F. H. Ai in higher education: Insights from student surveys and predictive analytics using pso-guided woa and linear regression. *Journal of Artificial Intelligence in Engineering Practice* **1**, 1–17 (2024).
25. Kandel, M. A. et al. Evaluating the efficacy of deep learning architectures in predicting traffic patterns for smart city development. *Full Length Article* **6**, 26–6 (2023).
26. Yassin, W., Udzir, N. I., Muda, Z. & Sulaiman, M. N. Anomaly-based intrusion detection through k-means clustering and naive bayes classification. *Journal of Computer Science and Technology* **28**, 650–661 (2013).
27. Bertoli, G. D. C. et al. An end-to-end framework for machine learning-based network intrusion detection system. *IEEE Access* **9**, 106790–106805 (2021).
28. Pascale, F., Adinolfi, E. A., Coppola, S. & Santonicola, E. Cybersecurity in automotive: An intrusion detection system in connected vehicles. *Electronics* **10**, 1765 (2021).
29. Kumar, G. & Kumar, K. Design of an evolutionary approach for intrusion detection. *The Scientific World Journal* **2013**, 962185 (2013).
30. Yue, C., Wang, L., Wang, D., Duo, R. & Nie, X. An ensemble intrusion detection method for train ethernet consist network based on cnn and rnn. *IEEE Access* **9**, 59527–59539 (2021).
31. Buczak, A. L. & Guven, E. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications surveys & tutorials* **18**, 1153–1176 (2015).
32. Khan, M. A. et al. A deep learning-based intrusion detection system for mqtt enabled iot. *Sensors* **21**, 7016 (2021).
33. Das, V. *et al.* Network intrusion detection system based on machine learning algorithms. *AIRCC's International Journal of Computer Science and Information Technology* 138–151 (2010).
34. Alsariera, Y. A. Detecting generic network intrusion attacks using tree-based machine learning methods. *International Journal of Advanced Computer Science and Applications* **12** (2021).
35. Elsaeidy, A., Munasinghe, K. S., Sharma, D. & Jamalipour, A. A machine learning approach for intrusion detection in smart cities. In *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, 1–5 (IEEE, 2019).
36. Najam, C. U. & Fakhrudeen, A. M. On the performance of intrusion detection systems for the internet of things: State-of-the-art in research. *International Journal of Nonlinear Analysis and Applications* **14**, 1413–1436 (2023).
37. Bhargavi, M., Kumar, M. N., Meenakshi, N. V. & Lasya, N. Intrusion detection techniques used for internet of things. *Int. J. Appl. Eng. Res* **14**, 4462–4466 (2019).
38. Zarpelão, B. B., Miani, R. S., Kawakani, C. T. & De Alvarenga, S. C. A survey of intrusion detection in internet of things. *Journal of Network and Computer Applications* **84**, 25–37 (2017).
39. Alom, M. Z. & Taha, T. M. Network intrusion detection for cyber security using unsupervised deep learning approaches. In *2017 IEEE national aerospace and electronics conference (NAECON)*, 63–69 (IEEE, 2017).
40. Bandyopadhyay, S., Chowdhury, R., Banerjee, P., Dey, S. D. & Saha, B. A decision tree based intrusion detection system for identification of malicious web attacks. *Preprints*[SPACE]https://doi.org/10.20944/preprints202010.0229.v1 (2020).
41. Rose, T., Kifayat, K., Abbas, S. & Asim, M. A hybrid anomaly-based intrusion detection system to improve time complexity in the internet of energy environment. *Journal of Parallel and Distributed Computing* **145**, 124–139 (2020).
42. Gaber, T., Awotunde, J. B., Folorunso, S. O., Ajagbe, S. A. & Eldesouky, E. Industrial internet of things intrusion detection method using machine learning and optimization techniques. *Wireless Communications and Mobile Computing* **2023**, 3939895 (2023).
43. Liao, H.-J., Lin, C.-H.R., Lin, Y.-C. & Tung, K.-Y. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications* **36**, 16–24 (2013).
44. Sindhu, S. S. S., Geetha, S. & Kannan, A. Decision tree based light weight intrusion detection using a wrapper approach. *Expert Systems with applications* **39**, 129–141 (2012).
45. Agbedanu, P. R., Musabe, R., Rwigema, J. & Gatare, I. A lightweight intrusion detection for internet of things using incremental ensemble learning. *Journal Not Specified* **1** (2022).
46. Mliki, H., Kaceam, A. H. & Chaari, L. A comprehensive survey on intrusion detection based machine learning for iot networks. *EAI Endorsed Transactions on Security and Safety* **8**, e3–e3 (2021).
47. Alzahrani, A. S., Shah, R. A., Qian, Y. & Ali, M. A novel method for feature learning and network intrusion classification. *Alexandria Engineering Journal* **59**, 1159–1169 (2020).
48. Elnakib, O., Shaaban, E., Mahmoud, M. & Emara, K. Eidm: Deep learning model for iot intrusion detection systems. *The Journal of Supercomputing* **79**, 13241–13261 (2023).
49. Musleh, D., Alotaibi, M., Alhaidari, F., Rahman, A. & Mohammad, R. M. Intrusion detection system using feature extraction with machine learning algorithms in iot. *Journal of Sensor and Actuator Networks* **12**, 29 (2023).

50. Spadaccino, P. & Cuomo, F. Intrusion detection systems for iot: opportunities and challenges offered by edge computing and machine learning. arXiv preprint arXiv:2012.01174 (2020).
51. Mohamed, T. S. & Aydin, S. Iot-based intrusion detection systems: a review. *Smart Science* **10**, 265–282 (2022).
52. Arshad, J., Azad, M. A., Abdeltaif, M. M. & Salah, K. An intrusion detection framework for energy constrained iot devices. *Mechanical Systems and Signal Processing* **136**, 106436 (2020).
53. Kocher, G. & Kumar, G. Machine learning and deep learning methods for intrusion detection systems: recent developments and challenges. *Soft Computing* **25**, 9731–9763 (2021).
54. Kably, S., Benbarrad, T., Alaoui, N. & Arioua, M. Multi-zone-wise blockchain based intrusion detection and prevention system for iot environment. *Computers, Materials & Continua* **75** (2023).
55. Qazi, E. U. H., Faheem, M. H. & Zia, T. Hdlnids: Hybrid deep-learning-based network intrusion detection system. *Applied Sciences* **13**, https://doi.org/10.3390/app13084921 (2023).
56. Jiang, K., Wang, W., Wang, A. & Wu, H. Network intrusion detection combined hybrid sampling with deep hierarchical network. *IEEE access* **8**, 32464–32476 (2020).
57. Lansky, J. et al. Deep learning-based intrusion detection systems: a systematic review. *IEEE Access* **9**, 101574–101599 (2021).
58. Ahmed, U. et al. Prediction of diabetes empowered with fused machine learning. *IEEE Access* **10**, 8529–8538 (2022).

Acknowledgments
Artificial Intelligence and Sensing Technologies (AIST) Research Center, University of Tabuk, Tabuk 71491, Saudi Arabia.

Author contributions
U.A, M.N, and T.S have collected data from different resources. A.S., U.A, and M.A.K. performed formal analysis and Simulation, A.S, U.A., T.A, and T.S contributed to writing-original draft preparation, E.H.M.A, M.A.K, M.N, and U.A; writing-review and editing, M.A.K, and E.H.M.A; performed supervision, M.N, T.A, and U.A.; drafted pictures and tables, E.H.M.A, T.A., and A.S.; performed revisions and improve the quality of the draft. All authors have read and agreed to the published version of the manuscript.

Funding
This work is supported by a research grant from the Research, Development, and Innovation Authority (RAID), Saudi Arabia, grant no. 13010-Tabuk-2023-UT-R-3-1-SE.

Declarations

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to T.A. or M.A.K.